



ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

Τομέας Σημάτων, Ελέγχου και Ρομποτικής
Εργαστήριο Όρασης Υπολογιστών, Επικοινωνίας Λόγου και Επεξεργασίας Σημάτων

Οπτική Αναγνώριση Συναισθήματος
με Βάση το Σημασιολογικό Περιεχόμενο
και Χρήση Βαθιών Νευρωνικών Δικτύων

Context-Based Visual Emotion Recognition
Using Deep Neural Networks

Διπλωματική Εργασία
Ιωάννης Πίκουλης

Επιβλέποντες: Πέτρος Μαραγκός
Καθηγητής Ε.Μ.Π.

Παναγιώτης Π. Φιλντίσης
Υποψήφιος Διδάκτορας Ε.Μ.Π.

Αθήνα, Ιούνιος 2021

[This page is left intentionally blank.]



ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
Τομέας Σημάτων, Ελέγχου και Ρομποτικής
Εργαστήριο Όρασης Υπολογιστών, Επικοινωνίας Λόγου
και Επεξεργασίας Σημάτων

Οπτική Αναγνώριση Συναισθήματος
με Βάση το Σημασιολογικό Περιεχόμενο
και Χρήση Βαθιών Νευρωνικών Δικτύων

Context-Based Visual Emotion Recognition
Using Deep Neural Networks

Διπλωματική Εργασία
Ιωάννης Πίκουλης

Επιβλέποντες: Πέτρος Μαραγκός
Καθηγητής Ε.Μ.Π.
Παναγιώτης Π. Φιλντίσης
Υποψήφιος Διδάκτορας Ε.Μ.Π.

Εγκρίθηκε από τη τριμελή εξεταστική επιτροπή. Ημερομηνία εξέτασης: 24/06/2021

(Υπογραφή)

.....
Πέτρος Μαραγκός
Καθηγητής
Ε.Μ.Π.

(Υπογραφή)

.....
Κωνσταντίνος Τζαφέστας
Αναπληρωτής Καθηγητής
Ε.Μ.Π.

(Υπογραφή)

.....
Γεράσιμος Ποταμιάνος
Αναπληρωτής Καθηγητής
Παν/μιο Θεσσαλίας

Αθήνα, Ιούνιος 2021.

(Υπογραφή)

.....
Ιωάννης Πίκουλης

Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Πνευματική ιδιοκτησία © Ιωάννης Πίκουλης, 2021. Με επιφύλαξη παντός δικαιώματος.

Απαγορεύεται η αντιγραφή, αποθήκευση και διανομή της παρούσας εργασίας, εξ' ολοκλήρου ή τμήματος αυτής, για εμπορικό σκοπό. Επιτρέπεται η ανατύπωση, αποθήκευση και διανομή για σκοπό μη κερδοσκοπικό, εκπαιδευτικής ή ερευνητικής φύσεως υπό την προϋπόθεση να αναφέρεται η πηγή προέλευσης και να διατηρείται το παρόν μήνυμα. Ερωτήματα που αφορούν τη χρήση της εργασίας για κερδοσκοπικό σκοπό πρέπει να απευθύνονται προς τον συγγραφέα.

Οι απόψεις και τα συμπεράσματα που περιέχονται σε αυτό ο έγγραφο εκφράζουν τον συγγραφέα και δεν πρέπει να ερμηνευθεί ότι αντιπροσωπεύουν τις επίσημες θέσεις του Εθνικού Μετσόβιου Πολυτεχνείου.

Copyright © Ioannis Pikoulis, 2021. All rights reserved.

It is prohibited to copy, store and distribute this work, in whole or in part, for commercial purposes. Reproduction, storage and distribution for a non-profit, educational or research nature are permitted, provided the source of origin is indicated and the present message maintained. Questions about the use of the work for profit should be directed to the author.

The views and conclusions contained in this document are those of the author and should not be construed as representing the official positions of the National Technical University of Athens.

Περίληψη

Η οπτική αναγνώριση συναισθήματος αποτελεί ένα μείζων θέμα του διεπιστημονικού τομέα της Όρασης Υπολογιστών το οποίο συσχετίζεται με την αποκωδικοποίηση και προσδιορισμό των ανθρώπινων συναισθημάτων τόσο σε κατηγορικό (διακριτό) όσο και σε διαστατικό (συνεχές) επίπεδο, όπως αυτά απεικονίζονται σε εικόνες ή ακολουθίες από βίντεο. Μια ανασκόπηση της σχετικής βιβλιογραφίας αποδεικνύει πως η πλειοψηφία των έως τώρα προσπαθειών έχουν περιοριστεί κυρίως στην ανάλυση των εκφράσεων του προσώπου ενώ ορισμένες μελέτες έχουν είτε ενσωματώσει πληροφορίες σχετικές με την ανθρώπινη πόζα είτε έχουν επιχειρήσει να πραγματοποιήσουν οπτική αναγνώριση συναισθήματος εξ' ολοκλήρου βασισμένες σε χειρονομίες και κινήσεις του σώματος. Ενώ μερικές από αυτές τις μεθόδους αποδίδουν αξιοπρεπώς σε ελεγχόμενα περιβάλλοντα, αποτυγχάνουν να ερμηνεύσουν πραγματικά, καθημερινά σενάρια όπου μη προβλέψιμες κοινωνικές συνθήκες και καταστάσεις μπορούν να καταστήσουν μία ή και πολλαπλές από τις προαναφερόμενες πηγές συναισθηματικής πληροφορίας, μη προσβάσιμες. Αντ' αυτού, στοιχεία από μελέτες συσχετιζόμενες με τη ψυχολογία υποστηρίζουν πως το οπτικό-σημασιολογικό περιεχόμενο πέραν των εκφράσεων του προσώπου και της πόζας του σώματος, παρέχει σημαντικές πληροφορίες για την αντίληψη των ανθρώπινων συναισθημάτων.

Κατά τη παρούσα εργασία, στόχο μας αποτελεί η ενίσχυση των ιδεών περί αναγνώρισης συναισθήματος βασισμένη στο οπτικό-σημασιολογικό περιεχόμενο. Για αυτό το σκοπό, πραγματοποιούμε εκτενή πειράματα σε δύο πρόσφατα σχηματισμένες και δύσκολες βάσεις δεδομένων, την EMOTions In Context (EMOTIC) και την Body Language Dataset (BoLD), αντιμετωπίζοντας κατά αντιστοιχία τόσο τη στατική (βασισμένη σε εικόνες) όσο και δυναμική (βασισμένη σε βίντεο) έκδοση του εν λόγω προβλήματος. Πιο συγκεκριμένα:

- Επεκτείνουμε ήδη πετυχημένες και ευρέως διαδεδομένες αρχιτεκτονικές νευρωνικών δικτύων ενσωματώνοντας πολλαπλές ροές πληροφορίας που ουσιαστικά κωδικοποιούν στοιχεία του ανθρώπινου σώματος και προσώπου, στοιχεία σημασιολογικού περιεχομένου καθώς και στοιχεία συσχετιζόμενα με το περιφερειακό εικονιζόμενο περιβάλλον, ενισχύοντας κατά αυτό το τρόπο τα μοντέλα μας και την ικανότητα τους να αντιλαμβάνονται εν γένει τα ανθρώπινα συναισθήματα.
- Εισάγουμε τις πιθανότητες κατηγοριοποίησης σκηνης και σκηνικών ιδιοτήτων ως επιπρόσθετα στοιχεία κατά τη διαδικασία αναγνώρισης συναισθήματος τα οποία λειτουργούν συμπληρωματικά ως προς τις υπόλοιπες πηγές συναισθηματικής πληροφορίας. Απ' όσο γνωρίζουμε, είμαστε οι πρώτοι που εφαρμόζουμε την εν λόγω μέθοδο.
- Εκμεταλλευόμαστε τις συσχετίσεις μεταξύ των διακριτών συναισθηματικών κατηγοριών, όπως αυτές εμφανίζονται εντός των εκάστοτε βάσεων δεδομένων, μέσω χρήσης Γραφο-Συνελικτικών Δικτύων και προσθήκης ενός ενσωματωμένου όρου σφάλματος, εμπνευσμένο από το τομέα της μετρικής μάθησης και βασισμένο σε ενσωματώσεις λέξεων μοντέλου GloVe.
- Πετύχαμε συγκρίσιμα αποτελέσματα αναγνώρισης στη βάση EMOTIC καθώς και ξεπεράσαμε τα καλύτερα δημοσιευμένα αποτελέσματα στη βάση BoLD.

Ένα μεγάλο τμήμα των συνεισφορών μας υποβλήθηκε υπό μορφή άρθρου [86] προς δημοσίευση στο 16th IEEE International Conference on Automatic Face and Gesture Recognition (FG), με συγγραφείς τους Ιωάννη Πίκουλη, Παναγιώτη Παρασκευά Φιλντίση και Πέτρο Μαραγκό.

Λέξεις Κλειδιά - αναγνώριση συναισθήματος, βαθιά νευρωνικά δίκτυα, σώμα, πρόσωπο, πόζα, οπτικό-σημασιολογικό περιεχόμενο, CNN, GCN, TSN, ST-GCN, σύνολο δικτύων

Abstract

Visual emotion recognition constitutes a major subject in the interdisciplinary field of Computer Vision which is associated with the process of identifying human emotion on categorical (discrete) and/or dimensional (continuous) level, as it is being depicted in still images or video sequences. A review of related literature reveals that the majority of past efforts in visual emotion recognition have been mostly limited to the analysis of facial expressions, while some studies have either incorporated information relative to body pose or have attempted to perform emotion recognition solely on the basis of body movements and gestures. While some of these approaches perform well in controlled environments, they fail to interpret real-world scenarios where unpredictable social settings can render one or multiple of the aforementioned sources of affective information inaccessible. However, evidence from psychology related studies suggest that visual context, in addition to facial expression and body pose, provides important information to the perception of people’s emotions.

In this work, we aim at reinforcing the concept of context-based visual emotion recognition. To this end, we conduct extensive experiments on two newly assembled and challenging databases, i.e. the EMOTions In Context (EMOTIC) and Body Language Dataset (BoLD), tackling both the image-based and video-based versions of the problem. More specifically we:

- Extend already successful baseline architectures by incorporating multiple input streams that encode bodily, facial, contextual as well as scene related features, thus enhancing our models’ understanding of visual context and emotion in general.
- Directly infuse scene classification scores and attributes as additional features in the emotion recognition process that function in a complementary manner with respect to all other sources of affective information. To the best of our knowledge, our approach is the first to do so.
- Exploit categorical emotion label dependencies, that reside within the datasets, through the usage of Graph Convolutional Networks (GCN) and the addition of metric-learning inspired loss that is based on GloVe word embeddings.
- Achieve competitive results on EMOTIC and significant improvements over the state-of-the-art techniques with relation to BoLD.

A big portion of our contributions [86] was submitted to the 16th IEEE International Conference on Automatic Face and Gesture Recognition (FG), with the authors being Ioannis Pikoulis, Panagiotis Paraskevas Filntisis and Petros Maragos.

Keywords - emotion recognition, deep neural networks, body, face, pose, visual-semantic context, CNN, GCN, TSN, ST-GCN, network ensemble

Ευχαριστίες

Αρχικά θα ήθελα να ευχαριστήσω τον κύριο Πέτρο Μαραγκό, όχι μόνο για την ευκαιρία που μου έδωσε ώστε να εκπονήσω την εν λόγω διπλωματική εργασία αλλά και για το γεγονός πως με τη παρουσία του ως καθηγητής καθώς και το έργο του ως ερευνητής, μου κίνησε το ενδιαφέρον και με ενέμπνευσε να ασχοληθώ ουσιαστικά με τους τομείς της Όρασης Υπολογιστών και της Μηχανικής Μάθησης.

Εν συνεχεία, θα ήθελα ισάξια να ευχαριστήσω τον κύριο Παναγιώτη Π. Φιλντίση ο οποίος συνεπίβλεψε τη διπλωματική μου εργασία. Αισθάνομαι ευγνώμων που είχα την ευκαιρία να συναναστραφώ με ένα τόσο αξιόλογο άτομο, τόσο σε ακαδημαϊκό όσο και σε διαπροσωπικό επίπεδο. Η καθοδήγηση που μου προσέφερε από τη πρώτη κιόλας στιγμή αποτέλεσε αδιαμφισβήτητα καθοριστικότερο παράγοντα για την ομαλή και επιτυχή εκπόνηση της εργασίας μου.

Τέλος θα ήθελα να ευχαριστήσω τους γονείς μου, καθώς ήταν αυτοί που πρώτοι μου έδειξαν το δρόμο της συστηματικής μελέτης και με δίδαξαν πως «τα αγαθά κόποις κτώνται».

Contents

List of Figures	viii
List of Tables	xiii
Notation	xvii
Εκτεταμένη Περίληψη στα Ελληνικά	xviii
1 Introduction	1
1.1 Models of Emotion	1
1.1.1 Categorical Models	1
1.1.2 Dimensional Models	2
1.1.3 Componential Models	3
1.2 Sources of Visual Affective Information	4
1.2.1 Facial Features	5
1.2.2 Body Features	6
1.2.3 Contextual Features	8
1.3 Thesis Outline	10
2 Deep Learning	11
2.1 Underlying Concepts	11
2.1.1 Artificial Intelligence	11
2.1.2 Machine Learning	11
2.1.3 Representation Learning	12
2.2 Predecessors of Modern ANNs	12
2.2.1 The Artificial Neuron	12
2.2.2 The Perceptron	13
2.2.3 The Neocognitron	14
2.3 Modern Deep Neural Network Practices	16
2.3.1 The Multilayer Perceptron	16
2.3.2 Network Training and Loss Functions	19
2.3.3 Gradient Descent Optimization	22
2.3.4 Backpropagation	25
2.3.5 Regularization	27
2.3.6 Convolutional Neural Networks	35
2.3.7 Graph Convolutional Networks	44
2.3.8 Recurrent Neural Networks	45

3	Introduction to Visual Emotion Recognition	53
3.1	Databases	53
3.1.1	Facial Expression Databases	53
3.1.2	Gesture-Based Bimodal Databases	54
3.1.3	Context-Oriented Databases	55
3.2	Hand-Crafted Features	56
3.2.1	Bimodal Visual Emotion Recognition using Body/Face	56
3.2.2	Texture Features	60
3.3	Deep Learning Approach to Visual Emotion Recognition	63
3.3.1	Data Preprocessing	63
3.3.2	Feature Extraction	65
4	Image-Based Visual Emotion Recognition in Context	67
4.1	Related Work	67
4.1.1	Baseline Model	67
4.1.2	EmotiCon	69
4.1.3	Leveraging Categorical Label Dependencies	73
4.2	Experimental Results	76
4.2.1	Ablation Studies	76
4.2.2	Cross-Study Performance Comparison	92
5	Video-Based Visual Emotion Recognition in Context	94
5.1	Related Work	95
5.1.1	Two-Stream Convolutional Networks	95
5.1.2	Temporal Segment Networks	96
5.1.3	Spatial-Temporal Graph Convolutional Networks	98
5.1.4	Laban Movement Analysis	101
5.1.5	Baseline Model	103
5.2	Experimental Results	103
5.2.1	Ablation Studies	103
5.2.2	Proposed Method	112
6	Conclusion and Future Work	115
6.1	Conclusion	115
6.2	Future Work	116
	Bibliography	117

List of Figures

A.1	Κατηγορικός τρόπος περιγραφής των ανθρώπινων συναισθημάτων βασισμένος στα έξι καθολικά συναισθήματα: χαρά, λύπη, φόβο, οργή, έκπληξη και απέχθεια. Πηγή: [77].	xix
A.2	Ο συνεχής τρόπος περιγραφής των συναισθημάτων ως σημεία σε ένα τρισδιάστατο χώρο Valence-Arousal-Dominance (VAD). Πηγή: [5].	xx
A.3	Η συστατική προσέγγιση περιγραφής συναισθημάτων βασισμένη στο χρωματικό τροχό του Plutchik. Πηγή [87].	xxi
A.4	Παραδείγματα Μοναδιαίων Δράσεων όπως εμφανίζονται μεμονωμένα ή σε συνδυασμούς κατά τη διάρκεια καθημερινών κοινωνικών αλληλεπιδράσεων. Πηγή: [80].	xxii
A.5	Η γλώσσα του σώματος περιλαμβάνει διαφορετικούς μη λεκτικούς περιγραφητές όπως τις εκφράσεις του προσώπου, τη στάση σώματος, χειρονομίες και κινήσεις των ματιών. Αυτοί είναι σημαντικοί δείκτες της συναισθηματικής και γνωστικής εσωτερικής κατάστασης ενός ατόμου και αποτελούν σημαντικές πηγές πληροφοριών σε σχέση με την συναισθηματική πληροφορική. Πηγή: [78].	xxii
A.6	Παράδειγμα που αναδεικνύει τη χρησιμότητα του σημασιολογικό περιεχομένου για την αντίληψη συναισθημάτων. Για την εικόνα στα αριστερά, θα υποθέταμε ότι η εικονιζόμενη γυναίκα βρίσκεται σε κατάσταση πόνου. Ωστόσο, με την εισαγωγή σημασιολογικού περιεχομένου στην δεξιά εικόνα, γίνεται εμφανές ότι η γυναίκα είναι αθλήτρια, πανηγυρίζει και βρίσκεται σε κατάσταση έκστασης. Πηγή: [8].	xxiii
B.1	Στιγμιότυπα από τις βάσεις δεδομένων AffectNet (αριστερά), FABO (μέση) και EMOTIC (δεξιά).	xxv
B.2	Περιγραφή υψηλού επιπέδου των επιμέρους σταδίων επεξεργασίας του υπολογιστικού συστήματος OpenPose. Πηγή: [15].	xxviii
B.3	Υψηλού επιπέδου απεικόνιση της εσωτερικής δομής του δικτύου MTCNN. Πηγή: [112].	xxviii
B.4	Το μακροπρόθεσμο επαναληπτικό συνελκτικό νευρωνικό δίκτυο (Long-term Recurrent Convolutional Networks, LRCN). Το LRCN επεξεργάζεται ακολουθίες μεταβλητού μήκους από frame (αριστερά) μέσω ενός CNN (μεσαία αριστερά), οι έξοδοι του οποίου τροφοδοτούν μία στοίβα απο επαναληπτικά ακολουθιακά μοντέλα (μεσαία δεξιά) τα οποία εν τέλει παράγουν μεταβλητού μήκους διανύσματα εξόδου (δεξιά). Πηγή: [29].	xxix
Γ.1	Δομή του baseline μοντέλου, για αναγνώριση συναισθημάτων σε κατηγορικό και συνεχές επίπεδο, στη βάση δεδομένων EMOTIC. Πηγή: [58].	xxx
Γ.2	Επισκόπηση της αρχιτεκτονικής του μοντέλου EmotiCon. Πηγή: [71].	xxxii

Δ.1	Πλήρες σχηματικό διάγραμμα του προτεινόμενου μοντέλου για αναγνώριση συναισθημάτων στη βάση δεδομένων BoLD. Τα συνενωμένα διανύσματα χαρακτηριστικών απεικονίζονται με κυανό χρώμα, τα πλήρως συνδεδεμένα στρώματα απεικονίζονται με πορτοκαλί χρώμα και οι ενσωματώσεις GloVe απεικονίζονται με πράσινο χρώμα, μαζί με τη διάστασή τους ή τον αριθμό των κρυφών μονάδων. Το ST-GCN μοντέλο παράγει εγγενώς προβλέψεις σε επίπεδο βίντεο, ενώ στην περίπτωση των TSN-RGB και TSN-Flow, αυτό απαιτεί προηγουμένως, την εφαρμογή μια συνάρτησης διατμηματικής συμφωνίας σχετικά με τις αντίστοιχες προβλέψεις σε επίπεδο snippet (26 πιθανότητες ταξινόμησης για διακριτά συναισθήματα, 3 συνεχείς τιμές για τις διαστάσεις VAD). Οι τελικές προβλέψεις λαμβάνονται μέσω εφαρμογής σταθμισμένου μέσου.	xxxv
1.1	The categorical way of describing affect on the basis of the universal set of six emotions, i.e. happiness, sadness, fear, anger, surprise and disgust. Source: [77].	2
1.2	The continuous way of describing affect in the form of a point in 3D Valence-Arousal-Dominance (VAD) space. Source: [5].	3
1.3	The componential way of describing affect on the basis Plutchik’s wheel of emotions. Source: [87].	4
1.4	Examples of detected Action Units (AUs) as they appear individually or in combinations during everyday social interactions. Source: [80].	5
1.5	Body language includes different types of nonverbal indicators such as facial expressions, body posture, gestures and eye movements. These are important markers of the emotional and cognitive inner state of a person and constitute significant sources of information relative to the task of human affect computing. Source: [78].	7
1.6	Example that illustrates the significance of contextual information for affective computing. For the image on the left, one would assume that the depicted woman is in a state of pain. However, with the introduction of context in the image on the right, it becomes evident that the woman is an athlete, she is celebrating and she is feeling ecstatic. Source: [8].	8
2.1	Illustration of a biological neuron and its myelinated axon are depicted, with signals flowing from inputs at dendrites to outputs at axon terminals. Source: https://en.wikipedia.org/wiki/Artificial_neural_network	13
2.2	Illustration of the McCulloch-Pitts artificial neuron model where a weighted sum of the input signals is passed through a unit step function, resulting in the corresponding output. Source: [95].	13
2.3	Schematic diagram illustrating the interconnections between layers in the neocognitron. Source: [39].	15
2.4	Illustration showing the input interconnections to the cells within a single cell-plane. Source: [39].	16
2.5	Schematic diagram of a multi-layer perceptron (MLP) with one hidden layer layer, p input units, L hidden units, q output units, with $\psi(\cdot)$ as the hidden unit activation functions and $\phi(\cdot)$ as the output unit activation functions. . .	18
2.6	Illustration of a standard 2D Convolutional Neural Network (CNN), consisting of a series of convolutional and pooling layers, responsible for feature extraction, followed by fully-connected (FC) layers which perform either classification or regression, depending on the task. Adapted from https://www.andreaperlato.com/aipost/cnn-and-softmax/	35

2.7	Illustration of a residual connection, the building block of residual learning. Source: [48].	42
2.8	A 5-layer dense block with a growth rate of $k = 4$. Each layer takes all preceding feature-maps as input. Source: [53].	43
2.9	Illustration of a graph convolutional network (GCN) with two hidden layers and ReLU activation functions. Source: https://github.com/kipf/pygcn	45
2.10	It is a RNN example: the left recursive description for RNNs, and the right is the corresponding extended RNN model in a time sequential manner. Source: [19].	46
2.11	Structure example of a Long Short-Term Memory (LSTM) block. Adapted from [44].	51
3.1	Sample images from the FABO database separately recorded by the (left) face and (right) body cameras. Source: [46].	57
3.2	A measure of QoM using SMIs (the shadow along the arms and the body) and the tracking of the hand. Source: [18].	58
3.3	The temporal evolution of an expression of “Happiness”. Source: [21].	58
3.4	Hand tracking by skin color-based tracker (left); position of hands using skin color tracking and position of head using an Active Shape Model (ASM) (middle); extraction of motion areas of hand and head regions (right). Source: [21].	59
3.5	Illustration of the basic Local Binary Pattern (LBP) operator. Source: [1].	61
3.6	Illustration of the facial expression representation procedure using LBP-TOP descriptors. Localization of facial features in each block volume, extraction of LBP features in three orthogonal planes. Features are concatenated for each block volume in order to represent shape and appearance. Source: [115].	61
3.7	OpenPose processing pipeline. The entire image constitutes the input for a two-branch CNN to jointly predict confidence maps for body part detection, shown in (b), and part affinity fields for parts association, shown in (c). The parsing step performs a set of bipartite matchings to associate body parts candidates (d). Finally full body poses are assembled for all people in the image (e). Source: [15].	64
3.8	Illustration of the MTCNN inner structure. Architectures of P-Net, R-Net, and O-Net, where “MP” means max pooling and “Conv” means convolution. The step size in convolution and pooling is 1 and 2, respectively. Source: [112].	65
3.9	2D and 3D convolution operations. a) Applying 2D convolution on a video volume (multiple frames as multiple channels) also results in an image. b) Applying 3D convolution on a video volume results in another volume, preserving temporal information of the input signal. Adapted from [101].	66
3.10	The Long-term Recurrent Convolutional Networks (LRCN). The LRCN processes the (possibly) variable-length visual input (left) with a CNN (middle-left), whose outputs are fed into a stack of recurrent sequence models (LSTMs, middle-right), which finally produce a variable-length prediction (right). Both the CNN and LSTM weights are shared across time, resulting in a representation that scales to arbitrarily long sequences. Source: [29].	66
4.1	Baseline end-to-end model for emotion recognition in context. The model consists of two feature extraction modules and a fusion network for jointly estimating the discrete categories and the continuous dimensions. Source: [59].	68
4.2	Overview of the Attention Branch Network (ABN). Source: [38].	70

4.3	Overview of the EmotiCon model architecture. Source: [71].	72
4.4	Overview of the ML-GCN model architecture. Source: [22].	74
4.5	CNN localization using CAMs. The first column depicts the input images without the application of context masking, the second and third column depict the CAMs from the context and body branches, overlaid on the original input image and the corresponding agent body crop.	78
4.6	Application of context masking. The first column depicts the input images after having masked the primary agents, the second and third column depicts the CAMs from the context and body branches, overlaid on the masked input image and the corresponding agent body crop.	79
4.7	Examples of estimated depth maps for 3 random images of the EMOTIC train set. The top row illustrates the original RGB images while the bottom row illustrates the corresponding grayscale depth maps as produced by the MegaDepth [63] pre-trained model.	84
4.8	<i>Mean average precision</i> (mAP, %) and <i>average absolute error</i> (AAE) scores for various values of λ_{att} , as obtained on the EMOTIC test set after incorporating the ABN mechanism in the context feature encoding branch.	86
4.9	<i>Mean average precision</i> (mAP, %) scores for various combinations of the parameters τ and p as obtained on the EMOTIC test set after incorporating the ML-GCN module in the the body feature encoding network.	89
4.10	<i>Mean average precision</i> (mAP, %) and <i>average absolute error</i> (AAE) scores for various values of λ_{emb} , as obtained on the EMOTIC test set after adding the categorical label embedding loss \mathcal{L}_{emb} to the standard ML-GCN model.	91
5.1	Overview of the two-stream convolutional architecture for action recognition in videos. Source: [94].	95
5.2	Overview of the TSN architecture. One input video is divided into K segments and a short snippet is randomly selected from each segment. The class scores of different snippets are fused by an the segmental consensus function to yield segmental consensus, which is a video-level prediction. Predictions from all modalities are then fused to produce the final predictions. Source: [104].	96
5.3	The spatial temporal graph of a skeleton sequence. Blue dots denote the body joints. The intra-body edges between body joints are defined based on the natural connections in human bodies. The inter-frame edges connect the same joints between consecutive frames. Joint coordinates are used as inputs to the ST-GCN. Source: [108].	99
5.4	Partitioning strategies for constructing convolution operations. From left to right: (a) An example frame of input skeleton. Body joints are drawn with blue dots. The receptive fields of a filter with $D = 1$ are drawn with red dashed circles. (b) Uniform labeling partitioning strategy, where all nodes in a neighborhood has the same label (green). (c) Distance partitioning. The two subsets are the root node itself with distance 0 (green) and other neighboring points with distance 1 (blue). (d) Spatial partitioning. The nodes are labeled according to their distances to the skeleton gravity center (black cross) compared with that of the root node (green). Centripetal nodes have shorter distances (blue), while centrifugal nodes have longer distances (yellow) than the root node. Source: [108].	100

5.5	Top-5 predicted scene categories, top-9 predicted attributes, ground truth and predicted (regressed) emotion categories (VAD values) as well as <i>Jaccard similarity coefficient</i> (JC) on samples that have been randomly selected from the BoLD validation set. All predictions are made at video level.	106
5.6	Input examples for the <i>body</i> (first column), <i>context</i> (second column) and <i>face</i> (third column) streams, as obtained from the same single snippet of a video sequence. The first row corresponds to <i>RGB</i> , the second row corresponds to the vertical component of <i>Optical Flow</i> and the third row corresponds to the <i>RGB Difference</i> modality.	110
5.7	Complete schematic diagram of the proposed network ensemble, featuring an ST-GCN module and three TSN input streams (<i>body</i> , <i>context</i> , <i>face</i>) for both the <i>RGB</i> and <i>Optical Flow</i> modalities, plus a <i>scene & attribute</i> related stream, especially for the <i>RGB</i> modality. Concatenated feature vectors are depicted in cyan, fully-connected layers are depicted in orange and GloVe word embeddings are depicted in green, along with their dimensionality or number of hidden units. The ST-GCN inherently produces video-level predictions, while in the case of the TSN- <i>RGB</i> and TSN- <i>Flow</i> , this requires the prior application of segmental consensus upon the corresponding snippet-level predictions (26 confidence scores for discrete emotions, 3 regressed values for VAD dimensions). Final predictions are obtained through late score fusion.	113
5.8	<i>Average precision</i> (AP) scores per emotion category, as obtained on the BoLD validation set, using our proposed network ensemble.	114
5.9	<i>Coefficient of determination</i> (R^2) per emotional dimension, as obtained on the BoLD validation set, using our proposed network ensemble.	114
5.10	<i>Jaccard similarity coefficient</i> (JC) per sample in the BoLD validation set, sorted in descending order.	114
5.11	<i>Absolute error</i> (AE) across the VAD dimensions, per sample in the BoLD validation set, sorted in ascending order.	114

List of Tables

Γ.1	Σύγκριση απόδοσης ως προς το <i>average precision</i> (AP, %), για κάθε κατηγορικό συνάισθημα, μεταξύ της προτεινόμενης μεθόδου μας και άλλων δημοσιευμένων μοντέλων, χρησιμοποιώντας το σύνολο δειγμάτων αξιολόγησης της βάσης δεδομένων EMOTIC.	xxxii
Γ.2	Σύγκριση απόδοσης ως προς το <i>average absolute error</i> (AAE), για κάθε συναισθηματική διάσταση, μεταξύ της προτεινόμενης μεθόδου μας και άλλων δημοσιευμένων μοντέλων, χρησιμοποιώντας το σύνολο δειγμάτων αξιολόγησης της βάσης δεδομένων EMOTIC.	xxxiii
Δ.1	Ποσοτικά αποτελέσματα στα σύνολα επικύρωσης (validation) και αξιολόγησης (test) της βάσης δεδομένων BoLD αναφορικά με την προτεινόμενη υλοποίηση μας και άλλα δημοσιευμένα μοντέλα στην σχετικής βιβλιογραφίας. Οι μετρικές απόδοσης εκφράζονται στο διάστημα [0,1].	xxxvi
2.1	Common activation functions	18
4.1	<i>Average precision</i> (AP, %), per emotional category, of the baseline model [59], as obtained on the EMOTIC test set.	69
4.2	<i>Average absolute error</i> (AAE), per emotional dimension, of the baseline model [59], as obtained on the EMOTIC test set.	69
4.3	<i>Average precision</i> (AP, %), per emotional category, of EmotiCon [71], as obtained on the EMOTIC test set and performance comparison with baseline model [59].	73
4.4	Effect of including the context modality on <i>average precision</i> (AP, %), per emotional category, as obtained on the EMOTIC test set.	77
4.5	Effect of including the context modality on <i>average absolute error</i> (AAE), per continuous emotional dimension, as obtained on the EMOTIC test set.	77
4.6	Effect of including the face modality on <i>average precision</i> (AP, %), per emotional category, as obtained on the EMOTIC test set.	80
4.7	Effect of including the face modality on <i>average absolute error</i> (AAE), per emotional dimension, as obtained on the EMOTIC test set.	80
4.8	Effect of including the pose modality on <i>average precision</i> (AP, %), per emotional category, as obtained on the EMOTIC test set. “GCN” and “Conv1D” denote the pose feature extraction method that has been used.	81
4.9	Effect of including the pose modality on <i>average absolute error</i> (AAE), per emotional dimension, as obtained on the EMOTIC test set. “GCN” and “Conv1D” denote the pose feature extraction method that has been used.	81

4.10	Effect of including scene scores and attributes on <i>average precision</i> (AP, %), per emotional category, as obtained on the EMOTIC test set. “+Scenes” column corresponds to the model after incorporating only the scene scores, “+Attributes” column corresponds to the model after incorporating both scene scores and attributes.	83
4.11	Effect of including the the scene scores and attributes on <i>average absolute error</i> (AAE), per emotional dimension, as obtained on the EMOTIC test set. “+Scenes” column corresponds to the model after incorporating only the scene scores, “+Attributes” column corresponds to the model after incorporating both scene scores and attributes.	83
4.12	Effect of including socio-dynamic contextual features on <i>average precision</i> (AP, %), per emotional category, as obtained on the EMOTIC test set. Two CNN backbones are evaluated for depth feature extraction, namely the baseline model [71] as well as a ResNet-18 pre-trained on ImageNet.	85
4.13	Effect of including the socio-dynamic contextual features on <i>average absolute error</i> (AAE), per emotional dimension, as obtained on the EMOTIC test set. Two CNN backbones are evaluated for depth feature extraction, namely the baseline model as well as a ResNet-18 pre-trained on ImageNet.	85
4.14	Performance comparison in terms of <i>average precision</i> (AP, %), per emotional category, of different network configurations after the incorporation of the Attention Branch Network (ABN) in the context feature encoding branch (“C-ABN”) and body branch (“BC-ABN”). Results are obtained on the EMOTIC test set, using $\lambda_{\text{att}} = 0.5$	87
4.15	Performance comparison in terms of <i>average absolute error</i> (AAE), per emotional dimension, of different network configurations after the incorporation of the Attention Branch Network (ABN) in the context feature encoding branch (“C-ABN”) as well as in the body branch (“BC-ABN”). Results are obtained on the EMOTIC test set, using $\lambda_{\text{att}} = 0.5$	87
4.16	Effect of utilizing deeper CNN backbones for feature extraction in performance, measured in terms of <i>mean average precision</i> (mAP, %) and <i>mean average absolute error</i> (mAAE), as obtained on the EMOTIC test set.	88
4.17	Performance comparison in terms of <i>average precision</i> (AP, %), per emotional category, of different network configurations after the incorporation of the ML-GCN module in our previously best model, as obtained on the EMOTIC test set. The selected parameter settings are $\tau = 0.4$ and $p = 0.25$	89
4.18	Performance comparison in terms of <i>average absolute error</i> (AAE), per emotional dimension, of different network configurations after the incorporation of the ML-GCN module in our previously best model, as obtained on the EMOTIC test set. The selected parameter settings are $\tau = 0.4$ and $p = 0.25$	89
4.19	Performance comparison in terms of <i>average precision</i> (AP, %), per emotional category, of different network configurations after the addition of the embedding loss \mathcal{L}_{emb} on top of our ML-GCN model, as obtained on the EMOTIC test set. Results correspond to $\lambda_{\text{emb}} = 0.4$	90
4.20	Performance comparison in terms of <i>average absolute error</i> (AAE), per emotional dimension, of different network configurations after the addition of the embedding loss \mathcal{L}_{emb} on top of our ML-GCN model, as obtained on the EMOTIC test set. Results correspond to $\lambda_{\text{emb}} = 0.4$	91
4.21	Performance comparison in terms of <i>average precision</i> (AP, %), per emotional category, of our best model with other publicized works relative to categorical emotion recognition, on the basis of the EMOTIC test set.	92

4.22	Performance comparison in terms of <i>average absolute error</i> (AAE), per emotional dimension, of our best model with other publicized works relative to continuous emotion recognition, on the basis of the EMOTIC test set.	93
5.1	Laban Movement Analysis (LMA) features (f_i : categories; dist.: distance; accel.: acceleration).	102
5.2	Baseline performance comparison among various network ensemble configurations, on the BoLD test set, as achieved by Luo et al. [67]. Performance metrics are presented in the range [0,1].	103
5.3	Performance comparison of various TSN-RGB model configurations on the BoLD validation set. The second column describes the various <i>RGB</i> input streams that have been included, namely the <i>body</i> , <i>context</i> , <i>face</i> , the Places365 scene-specific classification scores and SUN attribute scores. The third column specifies whether the categorical label embedding loss \mathcal{L}_{emb} has been incorporated in the training phase, while the fourth column specifies whether the <i>Partial BN</i> scheme has been utilized. Performance metrics are presented in the range [0,1].	105
5.4	Performance comparison of different TSN-RGB configurations on the BoLD validation set, relative to the number of segments K . All configurations utilize average pooling as the segmental consensus function. Performance metrics are presented in the range [0,1].	106
5.5	Performance comparison of different TSN-RGB configurations on the BoLD validation set, relative to the type of segmental consensus function that has been utilized, with $K = 7$ segments for both the training and validation phases. Performance metrics are presented in the range [0,1].	107
5.6	Performance comparison of various TSN-Flow-BC model configurations, on the BoLD validation set, relative to the datasets used for pre-training each one of the input streams. Performance metrics are presented in the range [0,1].	108
5.7	Performance comparison of various TSN-Flow-BF model configurations, on the BoLD validation set, relative to the datasets used for pre-training each one of the input streams. Performance metrics are presented in the range [0,1].	108
5.8	Performance comparison of various TSN-Flow model configurations, on the BoLD validation set. The second column describes the various <i>Optical Flow</i> input streams that have been included, namely the <i>body</i> , <i>context</i> and <i>face</i> . The third column specifies whether the categorical label embedding loss \mathcal{L}_{emb} has been incorporated in the training phase, while the fourth column specifies whether the <i>Partial BN</i> scheme is being utilized. Performance metrics are presented in the range [0,1].	109
5.9	Performance comparison of various TSN-RGBDiff model configurations, on the BoLD validation set. The second column describes the various <i>RGB Difference</i> input streams that have been included, namely the <i>body</i> , <i>context</i> and <i>face</i> . Neither categorical label embedding loss \mathcal{L}_{emb} nor <i>Partial BN</i> has been utilized. Performance metrics are presented in the range [0,1].	109
5.10	Performance comparison of various ST-GCN model configurations, on the BoLD validation and test sets. The third column specifies the joint labeling strategy that has been utilized during the construction of the graph adjacency matrix. Performance metrics are presented in the range [0,1].	111

5.11	Performance comparison of various LMA-based methods on the BoLD validation set. The first column specifies the LMA feature representation, i.e., raw sequences or descriptive statistics. Performance metrics are presented in the range [0,1].	112
5.12	Performance comparison of various network ensembles on the BoLD validation set, utilizing a late fusion scheme. The second column specifies the score fusion method, i.e., maximum, simple or weighted averaging. Performance metrics are presented in the range [0,1].	113
5.13	Quantitative results on the BoLD test set regarding our proposed network ensembles and other published works. Performance metrics are presented in the range [0,1].	114

Notation

\mathbb{R}	The set of real numbers
\mathbb{Z}	The set of integer numbers
$\{0, 1, \dots, n\}$	The set of all integers between 0 and n
$\ \cdot\ _1$	Absolute-value (L^1) norm
$\ \cdot\ _2$	Euclidean (L^2) norm. Equivalent to $\ \cdot\ $
$[\cdot]$	Iverson bracket
$\lfloor \cdot \rfloor$	Floor function
m	A scalar (real or integer)
\mathbf{m}	A row vector
\mathbf{m}^\top	Transpose of row vector \mathbf{m}
\mathbf{M}	A multidimensional tensor
M_{ij}	The element at position (i, j) of a 2D matrix \mathbf{M}
M^{ij}	Equivalent to M_{ij}
\mathbf{M}^\top	Transpose of 2D matrix \mathbf{M}
$\text{diag}(\mathbf{M})$	A square, diagonal matrix with diagonal entries given by \mathbf{M}
\mathbb{I}_n	$n \times n$ identity matrix
\mathbb{I}	Identity matrix with dimensionality implied by context
δ_{ij}	Kronecker delta with indices i and j
∇	Gradient vector
∇^2	Hessian matrix
\odot	Element-wise (Hadamard) product
\otimes	Cross-correlation operator
$*$	Convolution operator
$\ln x$	Natural logarithm of x
$\sigma(\cdot)$	Element-wise sigmoid operator
$\text{softmax}(\cdot)$	Row-wise softmax operator
$\text{sgn}(\cdot)$	Element-wise signum operator
$f : \mathbb{A} \rightarrow \mathbb{B}$	A function f with domain \mathbb{A} and range \mathbb{B}
$f \circ g$	Composition of function f with g
$x \sim D$	Random variable x has distribution D
$\mathcal{N}(x \mu, \sigma^2)$	Gaussian distribution over x with mean μ and variance σ^2
$P(A)$	Probability of event A
$p(x)$	Probability distribution over x
$\mathbb{E}(x)$	Expected value of random variable x
$\text{Var}(x)$	Variance of random variable x

Εκτεταμένη Περίληψη στα Ελληνικά

A Θεωρητικό Υπόβαθρο

Η ερμηνεία, αντίληψη και αναγνώριση των ανθρώπινων συναισθημάτων έχει αποτελέσει θέμα ενδελεχών μελετών και αναλύσεων σε διάφορους επιστημονικούς κλάδους όπως η βιολογία, η ψυχολογία, η κοινωνιολογία, η νευρολογία καθώς και η επιστήμη των υπολογιστών. Ενώ οι προαναφερθείσες γνωστικές επιστήμες επικεντρώνονται στην εξαγωγή των διαθέσιμων συναισθηματικών πληροφοριών, τα πεδία της Όρασης Υπολογιστών και της Μηχανικής Μάθησης στοχεύουν στην αυτοματοποίηση της διαδικασίας αναγνώρισης μέσω της ανάπτυξης νέων τεχνικών και αλγορίθμων που μπορούν να παράγουν αποτελεσματικές και ισχυρές κωδικοποιήσεις τέτοιων πληροφοριών. Η αυτόματη αναγνώριση συναισθήματος χρήζει τεράστιας πρακτικής σημασίας καθώς διαθέτει εκτεταμένες εφαρμογές σε περιβάλλοντα που περιλαμβάνουν συνεργασία ανθρώπου-ρομπότ, κοινωνική ρομποτική, ιατρική περίθαλψη, παρακολούθηση ψυχιατρικών ασθενών, παρακολούθηση κόπωσης οδηγών και πολλά άλλα σενάρια αλληλεπίδρασης ανθρώπου-υπολογιστή.

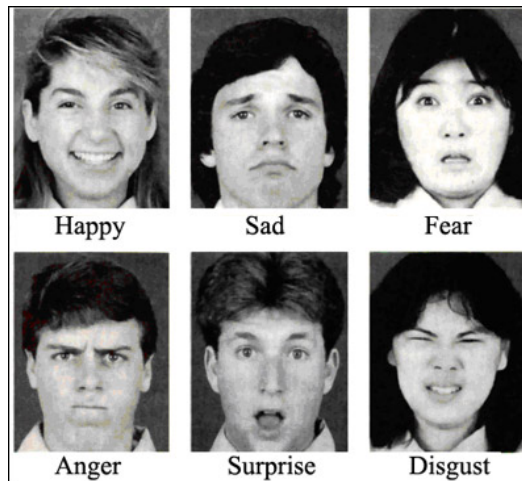
A.1 Μοντέλα Συναισθήματος

Ως πρώτο βήμα προς την κατανόηση των εννοιών της αντίληψης, ερμηνείας και αναγνώρισης των συναισθημάτων, πρέπει να τοποθετήσουμε τις θεωρητικές βάσεις στα πλαίσια των οποίων μοντελοποιούνται οι συναισθηματικές καταστάσεις. Η εύρεση ενός βέλτιστου τρόπου συναισθηματικής μοντελοποίησης έχει αποτελέσει θέμα εκτενών συζητήσεων και έχουν προταθεί πολλές προοπτικές επί του ζητήματος. Τα πιο ευρέως διαδεδομένα μοντέλα για την συναισθηματική μοντελοποίηση μπορούν να ταξινομηθούν σε τρεις κύριες κατηγορίες: κατηγορικά (categorical), διαστατικά (dimensional) και συστατικά (componential). Στις επόμενες παραγράφους θα προσπαθήσουμε να παρουσιάσουμε τα κύρια χαρακτηριστικά, τα πλεονεκτήματα και τα μειονεκτήματα κάθε είδους μοντέλου.

Κατηγορικά Μοντέλα

Τα κατηγοριακά μοντέλα χρησιμοποιούνται για την ταξινόμηση των συναισθημάτων σε διακριτές κατηγορίες που μπορούν να αναγνωριστούν και να περιγραφούν εύκολα στην καθημερινή γλώσσα. Η σημαντική εξέλιξη στα κατηγορικά συναισθηματικά μοντέλα αποδίδεται στο έργο των Ekman και Friesen [33], [34] και των υποκείμενων παραδοχών τους σχετικά με την καθολικότητα ενός συνόλου έξι βασικών συναισθημάτων, την ευτυχία, θλίψη, φόβο, οργή, απέχθεια και έκπληξη. Απεικονίσεις των προαναφερθέντων συναισθημάτων παρουσιάζονται στο σχήμα A.1. Για να αποδείξουν την υπόθεση περί καθολικότητας, οι Ekman και Friesen αρχικά πραγματοποίησαν πειράματα κατά τα οποία παρουσίασαν φωτογραφίες που απεικόνιζαν πρόσωπα σε άτομα με διαφορετικά πολιτισμικά υπόβαθρα με σκοπό να ελέγξουν εάν οι συμμετέχοντες θα ταξινομούσαν

τις εικονιζόμενες και αφηγούμενες καταστάσεις στις ίδιες συναισθηματικές κατηγορίες, παρά τις πολιτισμικές τους διαφορές. Τα άτομα που συμμετείχαν στα πειράματα ανήκαν σε πολλές διαφορετικές κοινωνικές ομάδες, συμπεριλαμβανομένων φοιτητών από τις Ηνωμένες Πολιτείες, την Αργεντινή, τη Βραζιλία, τη Χιλή και την Ιαπωνία, καθώς και κατοίκους από δύο αναπτυσσόμενες χώρες (το Σαντόνγκ του Βόρνεο και το Μέτωπο της Νέας Γουινέας) που όμως είχαν εκτεταμένη επαφή με το δυτικό πολιτισμό. Τα αρχικά αποτελέσματα έδειξαν ότι τα συναισθήματα γίνονται αντιληπτά ομοιόμορφα από όλα τα άτομα ανεξαρτήτως πολιτιστικού υπόβαθρου, ενισχύοντας την υπόθεση περί καθολικότητας στην αντίληψη συναισθημάτων.



Σχήμα Α.1: Κατηγορικός τρόπος περιγραφής των ανθρώπινων συναισθημάτων βασισμένος στα έξι καθολικά συναισθήματα: χαρά, λύπη, φόβο, οργή, έκπληξη και απέχθεια. Πηγή: [77].

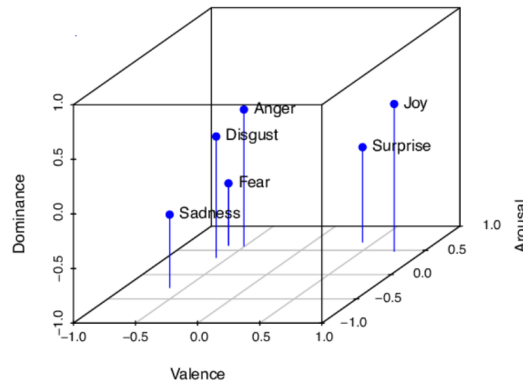
Το γεγονός πως όλοι οι συμμετέχοντες είχαν έρθει σε επαφή με διάφορες εκφάνσεις του δυτικού πολιτισμού καθώς και τον τρόπο με τον οποίο οι δυτικές κοινωνίες αντιλαμβάνονται και απεικονίζουν τις συναισθηματικές εκφράσεις του προσώπου, οδήγησε αναπόφευκτα στην αμφισβήτηση της εγκυρότητας των αντίστοιχων συμπεσμάτων. Κατά αυτό το τρόπο, τα πειράματα επαναλήφθηκαν χρησιμοποιώντας άτομα από τη Foie γλωσσικο-πολιτιστική ομάδα της Νέας Γουινέας και τα οποία επιλέχθηκαν υπό τις προϋποθέσεις ότι δεν είχαν επισκεφτεί ποτέ οικισμό του δυτικού κόσμου και δεν είχαν δει, ούτε και συναστραφεί με Καυκάσιο άτομο. Τα πειραματικά αποτελέσματα έδειξαν ότι οι κάτοικοι της εν λόγω αναπτυσσόμενης χώρας, όλων των ηλικιών και των δύο φύλων, πέτυχαν συγκρίσιμα αποτελέσματα με άτομα που ανήκαν σε αναπτυγμένες δυτικές κοινωνίες, υποστηρίζοντας την αρχική υπόθεση των συγγραφέων περί καθολικότητας των έξι προαναφερθέντων συναισθημάτων.

Τα κατηγορικά μοντέλα, λόγω της απλότητας τους, σε συνδυασμό με την υπόθεση περί οικουμενικότητας των συναισθημάτων υπήρξαν αναμφίβολα το πρωταρχικό εργαλείο σε έρευνες σχετικές με την αναγνώριση συναισθήματος και τη συναισθηματική πληροφορική.

Διαστατικά Μοντέλα

Μετά τα κατηγορικά μοντέλα, τα διαστατικά μοντέλα αποτελούν τη δεύτερη πιο δημοφιλή και ευρέως διαδεδομένη μέθοδο περιγραφής συναισθημάτων. Σε ένα διαστατικό μοντέλο, μια συναισθηματική κατάσταση αντιπροσωπεύεται ως ένα σημείο σε ένα συνεχές που εκτείνεται από ένα σύνολο ανεξάρτητων διαστάσεων. Αρκετά διαστατικά μοντέλα έχουν αναπτυχθεί με την πάροδο των ετών μέσω έρευνας που διεξήχθη στον τομέα της ψυχολογίας. Ωστόσο, το μοντέλο Pleasure/Valence-Arousal-Dominance (PAD/VAD) είναι το πιο συχνά χρησιμοποιούμενο κατά την συναισθηματική πληροφορική.

Το προαναφερθέν σύστημα αναπτύχθηκε από τους Mehrabian και Russell [70], [91]. Εκτεταμένες μελέτες και πειράματα υποστηρίζουν ότι οι τρεις διαστάσεις του Pleasure/Valence-Arousal-Dominance είναι ανεξάρτητες, καθώς οποιαδήποτε τιμή κατά μήκος μιας διάστασης μπορεί να λαμβάνεται ταυτόχρονα με κάθε τιμή σε οποιαδήποτε από τις άλλες δύο διαστάσεις. Επιπλέον, οι τρεις διαστάσεις είναι διπολικές. Μια σύλληψη του τρισδιάστατου χώρου VAD παρουσιάζεται στην εικόνα A.2.



Σχήμα A.2: Ο συνεχής τρόπος περιγραφής των συναισθημάτων ως σημεία σε ένα τρισδιάστατο χώρο Valence-Arousal-Dominance (VAD). Πηγή: [5].

Η συνεχής φύση των διαστατικών συναισθηματικών συστημάτων συμβάλει στη ανάδειξη πιο ολοκληρωμένων και πλούσιων αναπαραστάσεων για σύνθετες συναισθηματικές καταστάσεις. Ο πλούτος του συνεχούς χώρου είναι πιο δύσκολο να χρησιμοποιηθεί σε συστήματα αυτόματης αναγνώρισης εξαιτίας του γεγονότος ότι μπορεί να είναι δύσκολο να συσχετιστεί μια διανυσματική περιγραφή ενός συναισθήματος με μια συναισθηματική συμπεριφορά προσώπου ή κίνηση του σώματος.

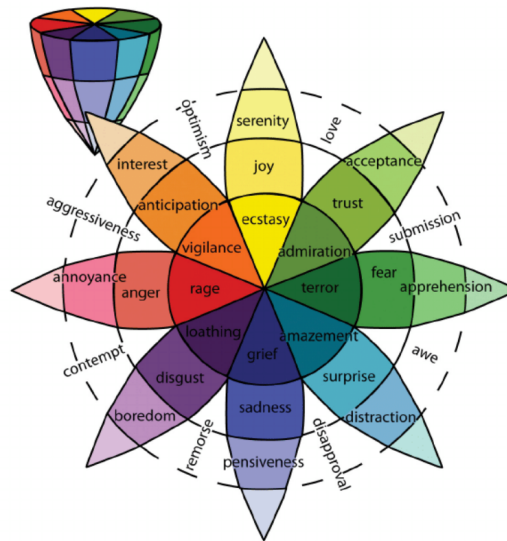
Συστατικά Μοντέλα

Τα συστατικά μοντέλα βρίσκονται ενδιάμεσα στα κατηγορικά και τα διαστατικά, όσον αφορά τη συναισθηματική διακριτική τους ικανότητα. Τα συστατικά μοντέλα περιγράφουν τα συναισθήματα με ιεραρχικό τρόπο, σύμφωνα με τον οποίο, τα συναισθήματα που ανήκουν σε ανώτερα στρώματα μπορούν να αποσυντεθούν σε ένα σύνολο πιο πρωτογενών συναισθημάτων που ανήκουν στα ακριβώς προηγούμενα στρώματα. Το πιο αξιοσημείωτο παράδειγμα ενός συστατικού συναισθηματικού μοντέλου είναι αυτό που εισήγαγε ο Plutchik [87] και ο οποίος αντιλήφθηκε τα πρωτογενή συναισθήματα με τρόπο ανάλογο με έναν τροχό χρώματος, τοποθετώντας παρόμοια συναισθήματα κοντά και συμπληρωματικά χρώματα σε απόσταση 180 μοιρών μεταξύ τους. Μια εικόνα του εκτεταμένου κυκλικού μοντέλου του Plutchik παρουσιάζεται στο σχήμα A.3.

Αυτοί οι τύποι μοντέλων χρησιμοποιούνται σπάνια στο πλαίσιο της συναισθηματικής πληροφορικής και της βιβλιογραφίας που σχετίζεται με την αυτόματη αναγνώριση συναισθημάτων, σε σύγκριση με τα προαναφερθέντα, κατηγορικά και διαστατικά μοντέλα.

A.2 Πηγές Συναισθηματικής Πληροφορίας

Το θέμα της τρέχουσας διατριβής σχετίζεται με την αυτόματη οπτική αναγνώριση συναισθημάτων σε εικόνες και βίντεο. Όσον αφορά την οπτική αναγνώριση συναισθημάτων, οι κύριες πηγές συναισθηματικών πληροφοριών εντοπίζονται στο πρόσωπο και το σώμα του εκάστοτε ατόμου. Πιο πρόσφατες μελέτες [8], [106] θεωρούν το περιφερειακό περιβάλλον και εν γένει την εικονιζόμενη σκηνή ως ισάξιες πηγές συναισθηματικής πληροφορίας. Το τελευταίο αναφέρονται συνήθως ως σημασιολογικά χαρακτηριστικά σχετιζόμενα με το περιβάλλον που απεικονίζεται σε



Σχήμα A.3: Η συστατική προσέγγιση περιγραφής συναισθημάτων βασισμένη στο χρωματικό τροχό του Plutchik. Πηγή [87].

μια εικόνα ή βίντεο. Ως πρώτο βήμα προς την κατανόηση του προβλήματος, θα πρέπει να αιτιολογήσουμε τη χρήση και τη σημασία καθεμιάς εκ των προαναφερθέντων πηγών συναισθηματικής πληροφορίας.

Στοιχεία Προσώπου

Το πρόσωπο θεωρείται συχνά ως το παράθυρο της ανθρώπινης ψυχής. Τα χαρακτηριστικά του προσώπου αποτελούν τη κύρια πηγή συναισθηματικών πληροφοριών, ενώ τα συστήματα αναγνώρισης συναισθημάτων που βασίζονται αποκλειστικά στο πρόσωπο, στοχεύουν στον εντοπισμό των κινήσεων του προσώπου καθώς και την αποκωδικοποίηση των συναισθημάτων που μεταδίδονται από τις κινήσεις αυτές. Τα προαναφερθέντα συστήματα αναγνώρισης βασίζονται κυρίως στο Σύστημα Κωδικοποίησης Κινήσεων του Προσώπου (Facial Action Coding System, FACS). Το FACS δημοσιεύτηκε από τους Ekman και Friesen [32] και αποτελεί ένα καθολικό υπολογιστικό σύστημα με βάση το οποίο είναι δυνατό να διακριθούν και να μελετηθούν όλες οι μεμονωμένες ορατές συμπεριφορές του προσώπου. Πιο συγκεκριμένα, το FACS εισάγει ένα πεπερασμένο σύνολο συστατικών της μυϊκής κίνησης του ανθρώπινου προσώπου, που ονομάζονται Μοναδιαίες Δράσεις (Action Units, AUs), καθώς και ένα σύνολο από Περιγραφητές Δράσης (Action Descriptors, ADs). Παραδείγματα διαφόρων ανιχνευμένων AUs απεικονίζονται στο σχήμα A.4. Τα κριτήρια για την παρατήρηση και την κωδικοποίηση κάθε Μοναδιαίας Δράσης και Περιγραφητή Δράσης περιγράφονται στο εγχειρίδιο Κωδικοποίησης Κινήσεων του Προσώπου (FAC), χρησιμοποιώντας το οποίο, οι άνθρωποι σχολιαστές μπορούν να κωδικοποιήσουν σχεδόν κάθε ανατομικά πιθανή συμπεριφορά του προσώπου.

Στοιχεία Σώματος

Οι κινήσεις του σώματος αποτελούν ιδιαίτερα σημαντικό μέσο αντίληψης και αναγνώρισης συναισθημάτων, ειδικά σε καταστάσεις κατά τις οποίες πρέπει να ανταλλαχθούν συναισθηματικές πληροφορίες σε μεγάλες αποστάσεις κατά τις οποίες οι εκφράσεις και τα χαρακτηριστικά του προσώπου δεν είναι πλέον ορατά. Επιπλέον, οι εκφράσεις του προσώπου ελέγχονται πιο εύκολα, ενώ η γλώσσα του σώματος αποκαλύπτει συχνά τα αληθινά μας συναισθήματα. Για παράδειγμα, σε αγχώδη κοινωνικά περιβάλλοντα, οι άνθρωποι τείνουν να διατηρούν ένα ίσιο πρόσωπο ή ακόμη



Σχήμα Α.4: Παραδείγματα Μοναδιαίων Δράσεων όπως εμφανίζονται μεμονωμένα ή σε συνδυασμούς κατά τη διάρκεια καθημερινών κοινωνικών αλληλεπιδράσεων. Πηγή: [80].

και να χαμογελούν, ώστε να κρύβουν τη νευρική τους και το άγχος τους, όπως αποκαλύπτεται από τρέμουλο και ιδρώτα στα χέρια, αμήχανες χειρονομίες και κούνημα των ποδιών. Επομένως, μπορούμε να πούμε ότι η γλώσσα του σώματος και τα ανθρώπινα συναισθήματα είναι αλληλεξαρτώμενα, ενώ το σχήμα Α.5 απεικονίζει διάφορες περιπτώσεις που επιβεβαιώνουν την εν λόγω άποψη.

Τα χέρια είναι πιθανώς η δεύτερη πλουσιότερη πηγή συναισθηματικών πληροφοριών μετά το πρόσωπο, όπως αναφέρεται στα [74], [83]. Στοιχεία όπως η ειλικρίνεια και η εντιμότητα μπορούν ενδεχομένως να προσδιοριστούν από τις θέσεις των χεριών ενός ατόμου. Πιο συγκεκριμένα, εάν ένα άτομο είναι ειλικρινές, πιθανότατα θα έχουν τα χέρια τους στραμμένα προς τον συνομιλητή, ενώ αν είναι ανυπόμονα, πιθανότατα θα κρύψουν τα χέρια τους πίσω από την πλάτη τους. Επιπλέον, η άσκηση χειρονομιών με ανοιχτά χέρια κατά τη διάρκεια μιας συνομιλίας δίνει συχνά την εντύπωση ενός πιο αξιόπιστου ατόμου.

Η θέση του κεφαλιού μπορεί να μεταφέρει πολύτιμες πληροφορίες για τη συναισθηματική κατάσταση ενός ατόμου. Λέγεται [84] ότι οι άνθρωποι τείνουν να μιλούν περισσότερο όταν ο ακροατής δείχνει κατανόηση και τους ενθαρρύνει με νεύματα, ενώ ο ρυθμός του νεύματος είναι ενδεικτικός της υπομονής ή έλλειψης αυτής. Επιπλέον, η ανύψωση του πηγουνιού, είναι πιθανό σημάδι ανωτερότητας ή αλαζονείας και η έκθεση του λαιμού θεωρείται ως ένδειξη υποταγής.

Ο κορμός του σώματος, μεμονωμένα, δεν είναι ικανός να μεταφέρει τόσες πληροφορίες όσο



Σχήμα Α.5: Η γλώσσα του σώματος περιλαμβάνει διαφορετικούς μη λεκτικούς περιγραφητές όπως τις εκφράσεις του προσώπου, τη στάση σώματος, χειρονομίες και κινήσεις των ματιών. Αυτοί είναι σημαντικοί δείκτες της συναισθηματικής και γνωστικής εσωτερικής κατάστασης ενός ατόμου και αποτελούν σημαντικές πηγές πληροφοριών σε σχέση με την συναισθηματική πληροφορική. Πηγή: [78].

τα προαναφερθέντα μέλη του σώματος, αλλά μπορεί να γίνει ενδεικτικό στοιχείο της συναισθηματικής κατάστασης κάποιου σε σχέση με το υπόλοιπο σώμα. Για παράδειγμα, η γωνία του κορμού σε σχέση με το σώμα μπορεί δυνητικά να αποκαλύψει πληροφορίες σχετικά με τη στάση ενός ατόμου σε μια συνομιλία, καθώς η εμπρόσθια τοποθέτηση του κορμού μπορεί να θεωρηθεί ως ένδειξη επιθετικότητας, ενώ μια ελαφρά γωνία του κορμού δείχνει αυτοπεποίθηση ή έλλειψη επιθετικότητας. Επιπλέον, μια ελαφριά κλίση του σώματος προς τα εμπρός σε συνδυασμό με ένα νεύμα ή χαμόγελο αποτελούν τυπική ένδειξη περιέργειας.

Σημαιολογικά Στοιχεία

Σε καταστάσεις της καθημερινής ζωής, όταν παρατηρούμε ένα άτομο, μπορούμε πιθανώς να εκτιμήσουμε πολλές πληροφορίες σχετικά με τη συναισθηματική τους κατάσταση παρά την έλλειψη πρόσθετων ειδικών πληροφοριών για αυτά. Αναλύοντας μια ευρύτερη άποψη της κατάστασης, αντί να επικεντρωνόμαστε στο εκάστοτε άτομο, μπορούμε ενδεχομένως να συλλέξουμε συναισθηματικές πληροφορίες που δεν μπορούν να γίνουν αντιληπτές δεδομένης της έλλειψης σηματολογικού περιεχομένου. Αυτή η ιδέα μπορεί να γίνει αντιληπτή μέσω ενός παραδείγματος στο σχήμα A.6, όπου παρουσιάζεται τόσο μια κοντινή όσο και μια ευρύτερη όψη μιας αθλήτριας. Η παραπάνω ιδέα ισχύει ιδιαίτερα σε καταστάσεις κατά τις οποίες το πρόσωπο είναι εν μέρει ή πλήρως μη ορατό λόγω κακών συνθηκών φωτισμού ή εμποδίων, ενώ το εκάστοτε άτομο μπορεί να λαμβάνει ασυνήθιστες πόζες, εμποδίζοντας την εξαγωγή πληροφοριών από τις προαναφερθείσες πηγές. Συνοπτικά, ο Dudzik και άλλοι [31] επισημαίνουν δύο κύριες πηγές σηματολογικού περιεχομένου που χρησιμοποιούνται για την ερμηνεία συναισθηματικών συμπεριφορών, το *αντιληπτό πλαίσιο κωδικοποίησης* καθώς και τη *γνώση και εμπειρία του διορόντα*.



Σχήμα A.6: Παράδειγμα που αναδεικνύει τη χρησιμότητα του σηματολογικού περιεχομένου για την αντίληψη συναισθημάτων. Για την εικόνα στα αριστερά, θα υποθέταμε ότι η εικονιζόμενη γυναίκα βρίσκεται σε κατάσταση πόνου. Ωστόσο, με την εισαγωγή σηματολογικού περιεχομένου στην δεξιά εικόνα, γίνεται εμφανές ότι η γυναίκα είναι αθλήτρια, πανηγυρίζει και βρίσκεται σε κατάσταση έκστασης. Πηγή: [8].

Το *αντιληπτό πλαίσιο κωδικοποίησης* περιλαμβάνει παράγοντες που γίνονται αντιληπτοί στα πλαίσια των εικονιζόμενων συναισθηματικών συμπεριφορών και οι οποίοι θεωρούνται ότι έχουν επηρεάσει δυνητικά την κωδικοποίηση μιας συναισθηματικής κατάστασης. Οι Wieser και Brosch [106] δηλώνουν ότι χαρακτηριστικά που αντιστοιχούν σε αυτήν την κατηγορία αφορούν δημογραφικά στοιχεία σχετικά με το είδος των συναισθηματικών συμπεριφορών καθώς και τις καταστάσεις ή σκηνές στις οποίες αυτές ενσωματώνονται. Ειδικότερα, το εικονιζόμενο περιφερειακό περιβάλλον και η σκηνή σε μια εικόνα ή βίντεο δύναται να σχετίζεται στενά με τα συναισθήματα των ατόμων που είναι παρόντα. Οι Barrett και Kensinger [9] αναφέρουν ότι τα δομικά χαρακτηριστικά του προσώπου, όταν τα βλέπουμε μεμονωμένα, συχνά αποδεικνύονται ανεπαρκή για την ορθή αντίληψη συναισθημάτων. Επιπλέον, εμπειρικά ευρήματα υποδηλώνουν ότι η κατηγοριοποίηση των εκφράσεων του προσώπου επιταχύνεται στη θέα σκηνών που είναι

συναφείς με τα εκαστοτε εικονιζόμενα συναισθήματα [88], ενώ η προβολή τόσο θετικών όσο και αρνητικών περιβάλλοντων οδηγεί σε σημαντικές αποκλίσεις ως προς την αντίληψη εκφράσεων προσώπου, συγκριτικά με περιπτώσεις όπου αυτές παρουσιάζονται σε ουδέτερα περιβάλλοντα [73].

Κατ' επέκταση, οι Weiser και Brosch [106] αναφέρουν ότι οι προϋπάρχουσες γνώσεις και εμπειρίες ενός δέκτη έχουν σημαντικό αντίκτυπο στον τρόπο αποκωδικοποίησης και αναγνώρισης συναισθηματικών καταστάσεων. Αυτή η γνώση περιλαμβάνει κυρίως φυλετικά στερεότυπα, συναισθηματικές σχέσεις, κοινωνικούς κανόνες, πολιτιστικές αξίες καθώς και την ψυχική τους κατάσταση, τις ανάγκες, τους στόχους και την εμπειρογνωμοσύνη τους. Με αυτόν τον τρόπο, η ενσωμάτωση της γνώσης και της εμπειρίας του δέκτη μπορεί να οδηγήσει στην ανακατασκευή και το φιλτράρισμα των συναισθηματικών πληροφοριών κατά τη διαδικασία αποκωδικοποίησης ενός συναισθήματος. Πειραματικά δεδομένα [8] υποδεικνύουν ότι καθώς οι συναισθηματικές λέξεις και έννοιες απομακρύνονται από την αντίληψη ενός ατόμου, παρουσιάζεται αυξανόμενη δυσκολία στην αποτελεσματική αναγνώριση του εκάστοτε συναισθήματος ακόμη και σε ελεγχόμενα περιβάλλοντα. Έτσι, όταν αντιλαμβανόμαστε το συναισθήμα που εκφράζει ένα πρόσωπο, η υποκείμενη διεργασία που εκτελείται από τον δέκτη μπορεί να παρομοιαστεί με την ανάγνωση μιας λέξης σε μια γραπτή σελίδα.

B Εισαγωγή στο Πρόβλημα της Οπτικής Αναγνώρισης Συναισθήματος

Η παρούσα ενότητα αποτελεί μια εισαγωγή στην Οπτική Αναγνώριση Συναισθήματος και αποσκοπεί στην διερεύνηση των βασικών υποκείμενων εννοιών που σχετίζονται με το εν λόγω πρόβλημα. Αρχικά θα διακριθούν τα διάφορα είδη βάσεων δεδομένων που συναντώνται συχνά σε πειραματικό επίπεδο, ανάλογα με τις διαθέσιμες πηγές συναισθηματικής πληροφορίας που τις χαρακτηρίζουν. Εν συνεχεία, θα γίνει μια σύντομη αναφορά σε δείγματα πρώιμης σχετικής βιβλιογραφίας, δίνοντας έμφαση σε μεθοδολογίες που χρησιμοποιούν «χειροποίητα», hand-crafted χαρακτηριστικά για τη κωδικοποίηση της οπτικής συναισθηματικής πληροφορίας. Κατ' επέκταση, θα γίνει μετάβαση σε πιο σύγχρονες αρχιτεκτονικές, έχοντας ως επίκεντρο πλέον τη Βαθιά Μάθηση και τα τεχνητά νευρωνικά δίκτυα. Σε αυτό το σημείο θα διευκρινιστούν θέματα σχετικά με τη προεπεξεργασία δεδομένων καθώς και θα αναλυθούν μέθοδοι για την εξαγωγή deep-learned χαρακτηριστικών.

B.1 Βάσεις Δεδομένων

Η ολοένα και μεγαλύτερη επιρροή των τεχνικών βαθιάς μάθησης στο εφαρμοσμένο πεδίο της Όρασης Υπολογιστών, και κατ' επέκταση στην συναισθηματική πληροφορική, έχει οδηγήσει αναπόφευκτα σε μια αυξανόμενη ζήτηση για δεδομένα εκπαίδευσης. Για τη πλήρη αξιοποίηση των δυνατοτήτων των νευρωνικών δικτύων, στα πλαίσια της οπτικής αναγνώρισης συναισθήματος, βασική προϋπόθεση αποτελεί η παροχή ενός επαρκούς αριθμού παραδειγμάτων εκπαίδευσης, με πληθώρα παραλλαγών σε εικονιζόμενα πρόσωπα και περιβάλλοντα. Κατά αυτό το τρόπο, καθίσταται αναγκαία η επιλεκτική ανασκόπηση ορισμένων βάσεων που έχουν χρησιμοποιηθεί ευρέως σε ερευνητικό επίπεδο και θα βοηθήσει στην ανάδειξη των κύριων διακριτικών τους χαρακτηριστικών.

Βάσεις Δεδομένων Εκφράσεων του Προσώπου

Το μεγαλύτερο κομμάτι της έρευνας στο τομέα της οπτικής αναγνώρισης συναισθημάτων επικεντρώνεται αποκλειστικά στην ανάλυση των εκφράσεων του προσώπου, πράγμα που συνεπάγεται ότι το ίδιο θα ισχυρεί και στη πλειοψηφία των διαθέσιμων βάσεων. Αξιοσημείωτα

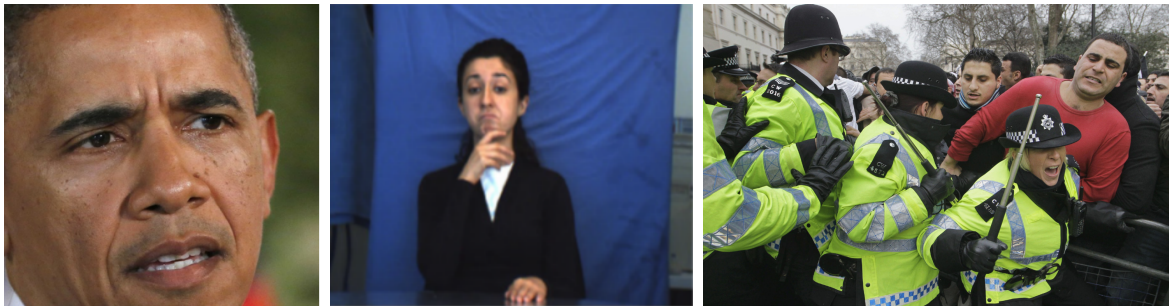
παραδείγματα αποτελούν οι Toronto Face Dataset (TFD) [97], FER2013 [43], Acted Facial Expressions in the Wild (AFEW) [28] και η AffectNet [75].

Διτροπικές Βάσεις Δεδομένων Βασισμένες σε Χειρονομίες

Πριν την κυριαρχία των σύγχρονων πρακτικών Βαθιάς Μάθησης, μεγάλη απήχηση γνώρισαν οι hand-crafted τεχνικές για την οπτική αναγνώριση συναισθημάτων, συνοδευόμενες από την ταυτόχρονη ανάπτυξη βάσεων δεδομένων που επικεντρώνονταν σε συναισθηματικές εκφράσεις μέσα από χειρονομίες και κινήσεις του σώματος. Οι εν λόγω βάσεις προάγουν τη χρήση πολλαπλών πηγών συναισθηματικής πληροφορίας, με το ανθρώπινο πρόσωπο και ευρύτερο σώμα να αποτελούν τις δύο κυριότερες. Παραδείγματα τέτοιων βάσεων δεδομένων αποτελούν οι Bi-modal Face and Body Gesture Database for Automatic Analysis of Human Nonverbal Affective Behavior (FABO) [45], GENEVA Multimodal Emotion Portrayal (GEMEP) [7] και η HUMAINE [17]. Κοινό χαρακτηριστικό τους αποτελεί το ελεγχόμενο, εργαστηριακό περιβάλλον που έχει χρησιμοποιηθεί κατά τη λήψη ή καταγραφή των εν λόγω εικόνων και βίντεο.

Βάσεις Δεδομένων Σημασιολογικού Περιεχομένου

Πιο πρόσφατα, ερευνητικές προσπάθειες έχουν στραφεί στην αναγνώριση συναισθημάτων με βάση το εικονιζόμενο σημασιολογικό περιεχόμενο. Στοιχεία σημασιολογικού περιεχομένου αποτελούν το περιφερειακό περιβάλλον, η εικονιζόμενη σκηνή, αντικείμενα και δευτερεύοντες ανθρώπινοι χαρακτήρες που μπορεί να είναι παρόντες. Οι EMOTIONS In Context (EMOTIC) [58], Context-Aware Emotion Recognition (CAER) benchmark [62] και Body Language Dataset (BoLD) [67] αποτελούν τις πιο αξιοσημείωτες δημοσίως διαθέσιμες βάσεις δεδομένων αυτού του είδους. Στιγμιότυπα από κάθε ένα εκ των προαναφερόμενων ειδών βάσεων δεδομένων, απεικονίζονται στο σχήμα B.1.



Σχήμα B.1: Στιγμιότυπα από τις βάσεις δεδομένων AffectNet (αριστερά), FABO (μέση) και EMOTIC (δεξιά).

B.2 Πρώιμη Σχετική Βιβλιογραφία

Πρώιμα δείγματα σχετικής βιβλιογραφίας αναδεικνύουν ένα σύνολο από hand-crafted τεχνικές για την κωδικοποίηση των διαθέσιμων συναισθηματικών πληροφοριών. Πειραματικά αποτελέσματα εξήχθησαν υπό ελεγχόμενες συνθήκες σε εργαστηριακά περιβάλλοντα με, κατά κύριο λόγο, χρήση διτροπικών βάσεων δεδομένων που επικεντρώνονταν σε συναισθηματικές εκφράσεις μέσα από χειρονομίες, κινήσεις του σώματος και μυϊκές δράσεις του προσώπου, ενώ η διαδικασία αναγνώρισης πραγματοποιούνταν εξ' ολοκλήρου στη βάση κατηγορικών συναισθηματικών μοντέλων και των έξι καθολικών συναισθημάτων.

Hand-Crafted Τεχνικές

Μια από τις πρώτες προσπάθειες σε αυτή τη κατεύθυνση αποτελεί το έργο των Gunes και Piccardi [47] οι οποίοι χρησιμοποίησαν μορφολογικά φίλτρα και κατάτμηση με βάση το χρώμα της επιδερμίδας, στον HSV χρωματικό χώρο για να εντοπίσουν και να απομωνώσουν τις περιοχές του προσώπου, κεφαλιού, ώμων και χεριών των εικονιζόμενων ατόμων. Αφού αφάιρεσαν το φόντο, υπολόγισαν οπτική ροή ανάμεσα σε ουδέτερα frame και frame συναισθηματικής κορύφωσης για να κωδικοποιήσουν τις μυϊκές δράσεις του προσώπου. Ομοiotρόπως, εντόπισαν τη σιλουέτα και οριοθέτησαν τα εικονιζόμενα ανθρώπινα σώματα έτσι ώστε να αποκτήσουν σημεία αναφοράς σχετικά με τη κίνηση τους. Η συναισθηματική πληροφορία που εμπεριέχεται σε κάθε βίντεο, κωδικοποιήθηκε σε διανύσματα 148 διαστάσεων για το πρόσωπο και 140 διαστάσεων για το ανθρώπινο σώμα.

Ο Castellano και άλλοι [18] πρότειναν την χρήση πέντε μετρικών για τη κωδικοποίηση των κινήσεων του σώματος. Οι εν λόγω μετρικές περιελάμβαναν τη ποσότητα κίνησης (Quantity of Motion, QoM), το δείκτη συστολής (Contraction Index, CI) καθώς και τη ταχύτητα, επιτάχυνση και ρευστότητα του βαρύκεντρου των χεριών. Ο δείκτης συστολής $CI \in [0, 1]$ μπορεί να υπολογιστεί ως ο λόγος ανάμεσα στο εμβαδό του μικρότερου οριοθετημένου κουτιού που περιέχει τα χέρια και το κεφάλι προς το εμβαδό που καλύπτει η σιλουέτα του σώματος. Αντίστοιχα, το QoM ποσοτικοποιεί την εντοπιζόμενη κίνηση με βάση τις σιλουέτες των εικονιζόμενων σωμάτων, κάνοντας χρήση των Silhouette Motion Images (SMIs).

Ο Chen και άλλοι, επιχείρησαν να πραγματοποιήσουν αυτόματη ταξινόμηση των χρονικών φάσεων των συναισθηματικών εκφράσεων (neutral, onset, apex, offset), κάνοντας χρήση δύο μετρικών της εικονιζόμενης κίνησης, του εμβαδού κίνησης (Motion Area, MA), της ουδέτερης απόκλισης (Neutral Divergence, ND) και των Motion History Images (MHI). Οι MHIs είναι δυαδικές εικόνες που λαμβάνουν τιμή ίση με μονάδα σε pixel στα οποία η διαφορά φωτεινότητας ανάμεσα στο τρέχον και προηγούμενο frame, ξεπερνάει κάποιο προκαθορισμένη τιμή κατωφλίου. Η MA αποτελεί το συνολικό αριθμό pixel με μη μηδενική τιμή φωτεινότητας στα MHI ενός frame. Επιπλέον, στο [21] χρησιμοποιούνται ιστογράμματα προσανατολισμένων κλίσεων (Histograms of Oriented Gradients, HOGs), τόσο σε εικόνες όσο και σε MHIs, σε συνδυασμό με Bag of Words (BoW) μοντέλα για να περιγραφεί αποτελεσματικά η κίνηση και η εμφάνιση των εικονιζόμενων συναισθηματικών εκφράσεων.

Επιπλέον, στο [93] προτάθηκε μια μεθοδολογία για τον αυτόματο εντοπισμό χωροχρονικών σημείων ενδιαφέροντος σε βίντεο μέσα από ενδείξεις του προσώπου και του σώματος. Πιο συγκεκριμένα, για κάθε frame ενός βίντεο, τα χωρο-χρονικά σημεία ενδιαφέροντος εντοπίζονται ως τα τοπικά μέγιστα μια συνάρτησης απόκρισης από μονοδιάστατα Gabor φίλτρα.

Χαρακτηριστικά Υφής

Σε αυτό το σημείο οφείλουμε να αναδείξουμε μια σειρά από τεχνητά χαρακτηριστικά υφής (texture features) τα οποία έχουν αξιοποιηθεί στα πλαίσια της οπτικής αναγνώρισης συναισθήματος. Κοινή βάση για την ανάπτυξη αυτών των χαρακτηριστικών υφής αποτέλεσαν τα τοπικά δυαδικά μοτίβα (Local Binary Patterns, LBP) [1]. Οι LBP ετικέτες κωδικοποιούν μοτίβα σε επίπεδο pixel, ενώ η συγκέντρωσή τους σε ιστογράμματα οδηγεί στο σχηματισμό καθολικών περιγραφητών του προσώπου για εικόνες, που είναι αναλλοίωτοι σε μεταβολές στο φωτισμό. Άμεση προέκταση των LBP περιγραφητών για εφαρμογές σε ακολουθίες βίντεο αποτελούν τα τοπικά δυαδικά μοτίβα όγκου (Volume Local Binary Patterns, VLBP) και τα τοπικά δυαδικά μοτίβα σε τρία ορθογώνια επίπεδα (Local Binary Patterns on Three Orthogonal Planes, LBP-TOP) [115]. Παραλλαγές των προαναφερόμενων τεχνικών, περιλαμβάνουν τη πρότερη συνέλιξη των εικόνων με Gabor φίλτρα πριν την εξαγωγή των χαρακτηριστικών υφής, αποτελώντας τα Local Gabor Binary Patterns (LGBP) [114] και LGBP-TOP [2], κατ' αντιστοιχία. Στο [79] προτάθηκε μια

προέκταση του LGBP-TOP περιγραφητή ο οποίος είναι αναλλοίωτος σε θωλώσεις της εικόνας, με την ονομασία Volume Phase Local Quantization (VLPQ).

Επιπρόσθετα, αξίζει να επισημάνουμε ορισμένα ακόμη χαρακτηριστικά υψής γενικού σκοπού τα οποία όμως έχουν βρει εφαρμογή στα πλαίσια του παρόντος προβλήματος. Το πρώτο από αυτά είναι οι πυραμίδες από ιστογράμματα προσανατολισμένων κλίσεων Pyramid HOGs [12]. Αυτά αποτελούν τη συνένωση πολλαπλών HOG διανυσμάτων, με το καθένα να έχει εξαχθεί σε διαφορετικά επίπεδα μιας χωρικής πυραμίδας. Εν συνεχεία, έχουμε το περιγραφητή κλιμακωτά αναλλοίωτου μετασχηματισμού χαρακτηριστικών (Scale-Invariant Feature Transform, SIFT) [65]. Ο SIFT περιγραφητής αποτελεί ένα διάνυσμα 128 διαστάσεων που προκύπτει από την συνένωση 16 ιστογραμμάτων σχετικά με το προσανατολισμό και τα πλάτη των κλίσεων της εκάστοτε εικόνας και είναι αναλλοίωτος σε περιστροφές, μετατοπίσεις, κλιμακώσεις και μεταβολές στο φωτισμό.

Αξίζει να επισημάνουμε πως μετά την επιτυχή εξαγωγή των hand-crafted χαρακτηριστικών, για την επίλυση των εκάστοτε προβλημάτων (συνήθως ταξινόμηση), χρησιμοποιούνταν κλασικές μέθοδοι μηχανικής μάθησης όπως: Naive-Bayes, Random Forest, k-Nearest Neighbors (k-NN) και Support Vector Machines (SVM).

B.3 Προσέγγιση του Προβλήματος με Τεχνικές Βαθιάς Μάθησης

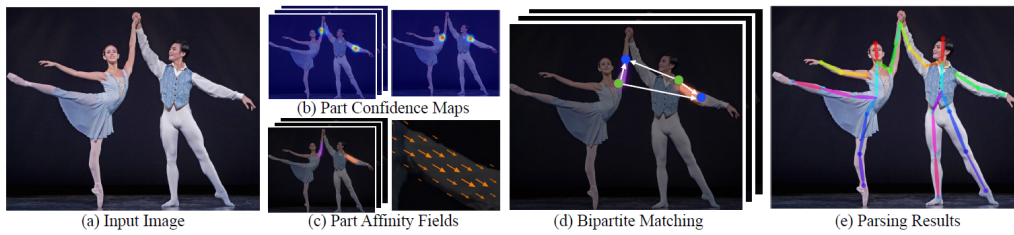
Στη παρούσα ενότητα, αναλύονται τα δύο βασικά στάδια που απαρτίζουν οποιαδήποτε μεθοδολογία η οποία βασίζεται σε τεχνικές Βαθιάς Μάθησης, κατά τη προσέγγιση του προβλήματος της οπτικής αναγνώρισης συναισθημάτων. Σε πρώτη φάση, υπάρχει το ευρύ στάδιο προεπεξεργασίας δεδομένων (data pre-processing) και εν συνεχεία, το στάδιο εξαγωγής βαθιών χαρακτηριστικών (deep feature extraction).

Προεπεξεργασία Δεδομένων

Η πολυτροπική οπτική αναγνώριση συναισθημάτων, με ταυτόχρονη αξιοποίηση πολλαπλών πηγών συναισθηματικής πληροφορίας, προϋποθέτει την εκτέλεση ορισμένων βημάτων προεπεξεργασίας σχετικά με τα εκάστοτε δεδομένα, όπως τον εντοπισμό του ανθρώπινου σώματος, την εκτίμηση πόζας, τον εντοπισμό και ευθυγράμμιση των χαρακτηριστικών του προσώπου.

Η πλειονότητα των διαθέσιμων βάσεων δεδομένων που προορίζονται για το πρόβλημα της οπτικής αναγνώρισης συναισθημάτων, παρέχουν εκ των προτέρων χωρικές οριοθετήσεις των σωμάτων που αντιστοιχούν στα εικονιζόμενα άτομα. Σε περίπτωση που κάτι τέτοιο δεν ισχύει, καθίσταται απαραίτητος ο εντοπισμός των επίμερους ανθρώπινων σωμάτων για κάθε διαθέσιμο δείγμα της εκάστοτε βάσης δεδομένων. Αυτό το στάδιο προεπεξεργασίας αναφέρεται ως human detection και περιλαμβάνει εν γένει τον εντοπισμό εικονιζόμενων περιοχών ενδιαφέροντος (regions of interest) και εν συνεχεία τη ταξινόμηση των εν λόγω περιοχών ανάλογα με τις εξαγόμενες προβλέψεις περί ύπαρξης ή μη ανθρώπινων χαρακτήρων εντός αυτών. Παλαιότερες προσεγγίσεις επί του θέματος περιελάμβαναν τη χρήση hand-crafted χαρακτηριστικών, όπως ιστογράμματα προσανατολισμένων κλίσεων (HOGs) [25], ενώ πιο πρόσφατα έχουν κυριαρχήσει μεθοδολογίες που βασίζονται στη χρήση βαθιών νευρωνικών δικτύων, ως ταξινομητές κυλιόμενου παραθύρου επί των εκάστοτε εικόνων εισόδου. Παράδειγμα μια τέτοιας προσέγγισης αποτελεί το Large-Field-Of-View (LFOV) δίκτυο [4].

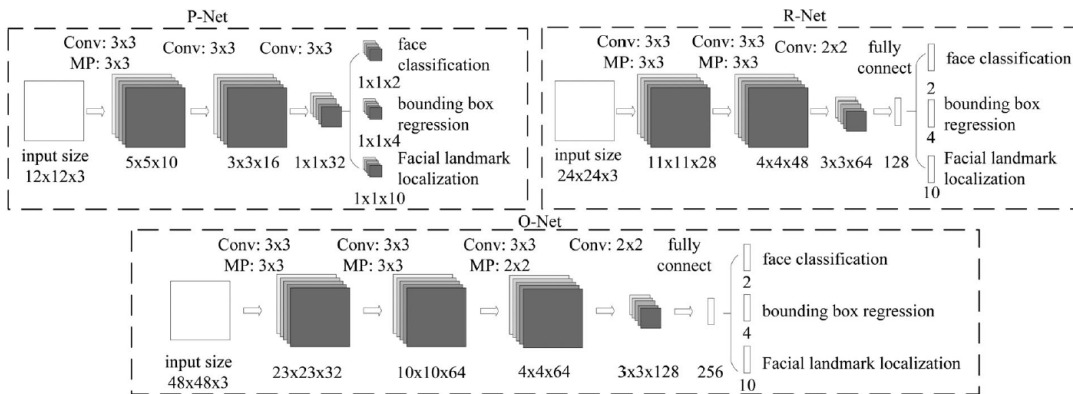
Αφού έχουν επιτυχώς οριοθετηθεί χωρικά οι ανθρώπινοι χαρακτήρες, συχνά απαιτείται η εξαγωγή επίμερους πληροφοριών σχετικά με τις πόζες των εικονιζόμενων σωμάτων. Αυτό το στάδιο προεπεξεργασίας αναφέρεται ως εκτίμηση πόζας (pose estimation) και στο οποίο πρωταγωνιστούν σχεδόν εξ' ολοκλήρου τεχνικές βαθιάς μάθησης. Η τελευταία σημαντική πρόοδος σχετικά με το πρόβλημα του pose estimation προέκυψε με την εισαγωγή του OpenPose [15] το οποίο αποτελεί ένα υπολογιστικό σύστημα για εκτίμησης πόζας πολλαπλών ατόμων σε πραγματικό



Σχήμα B.2: Περιγραφή υψηλού επιπέδου των επιμέρους σταδίων επεξεργασίας του υπολογιστικού συστήματος OpenPose. Πηγή: [15].

χρόνο με χρήση πεδίων συνάφειας μελών (Part Affinity Fields, PAFs). Τα στάδια επεξεργασίας που περιλαμβάνει το υπολογιστικό σύστημα OpenPose παρουσιάζονται στο σχήμα B.2.

Το επόμενο στάδιο κατά τη προεπεξεργασία δεδομένων σχετίζεται με τον εντοπισμό και απομόνωση των περιοχών που αντιστοιχούν σε εικονιζόμενα πρόσωπα καθώς και την ευθυγράμμιση των χαρακτηριστικών τους (face detection & alignment), όπως τα μάτια, μύτη, άκρες του στόματος, κτλ. Οι επικρατέστερες μέχρι σήμερα τεχνικές χρησιμοποιούν βαθιά συνελκτικά νευρωνικά δίκτυα με στόχο την από κοινού επίλυση των προβλημάτων του face detection και alignment. Παράδειγμα μιας τέτοιας προσέγγισης αποτελεί το Multi-Task Convolutional Neural Network (MTCNN) [112], η δομή του οποίου φαίνεται στο σχήμα B.3. Επιπλέον, η τελευταία σημαντική πρόοδος σχετικά με το εν λόγω πρόβλημα προέκυψε με την εισαγωγή του OpenFace 2.0 [6] το οποίο αποτελεί ένα δημοσίως διαθέσιμο, πλήρες σύστημα ανάλυσης ορατών συμπεριφορών σε ανθρώπινα πρόσωπα.



Σχήμα B.3: Υψηλού επιπέδου απεικόνιση της εσωτερικής δομής του δικτύου MTCNN. Πηγή: [112].

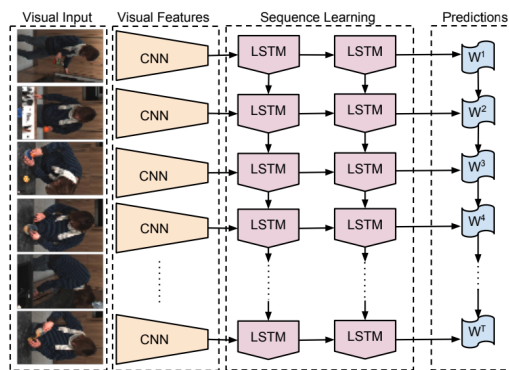
Εξαγωγή Deep-Learned Χαρακτηριστικών

Μετά την ολοκλήρωση των σταδίων προεπεξεργασίας, οι διάφορες ροές εισόδου τροφοδοτούν τα εκάστοτε βαθιά νευρωνικά δίκτυα με στόχο την εξαγωγή υψηλού επιπέδου αναπαράστασεων των δεδομένων που παρέχονται. Στη περίπτωση του προβλήματος της οπτικής αναγνώρισης συναισθημάτων, τα δεδομένα εισόδου αποτελούνται είτε από εικόνες είτε από ακολουθίες βίντεο. Κατά αυτό το τρόπο, τα δισδιάστατα ή και τρισδιάστατα συνελκτικά νευρωνικά δίκτυα (2D/3D Convolutional Neural Networks, CNN) καθώς και τα επαναληπτικά νευρωνικά δίκτυα (Recurrent Neural Networks, RNN) αποτελούν τις αρμόζουσες δομές βαθιάς μάθησης για το εν λόγω πρόβλημα.

Τα CNN απαρτίζονται από τεσσάρων ειδών στρώματα, τα συνελκτικά (convolutional), τα στρώματα κανονικοποίησης (regularization), τα στρώματα ομαδοποίησης (pooling) και τα

πλήρως διασυνδεδεμένα στρώματα (fully-connected). Τα πρώτα, χρησιμοποιούν σύνολα φίλτρων που συνελίσσονται με την αρχική εικόνα εισόδου με στόχο την εξαγωγή χαρτών από χαρακτηριστικά (feature maps) με διαφορετικά επίπεδα αφάιρεσης. Τα στρώματα κανονικοποίησης χρησιμεύουν στην αντιμετώπιση του πρόβληματος της υπερπροσαρμογής (overfitting) και στην επιτάχυνση της διαδικασίας εκπαίδευσης. Εν συνεχεία, τα στρώματα ομαδοποίησης υποδειγματοληπτούν τους εξαγόμενους χάρτες χαρακτηριστικών με στόχο τη μείωση της διαστατικότητας του. Τέλος, τα πλήρως διασυνδεδεμένα στρώματα μετατρέπουν τους πολυδιάστατους χάρτες χαρακτηριστικών σε μονοδιάστατα διανύσματα βάσει των οποίων παράγονται προβλέψεις για τα εκάστοτε πρόβλημα ταξινόμησης (classification) ή και παλινδρόμησης (regression).

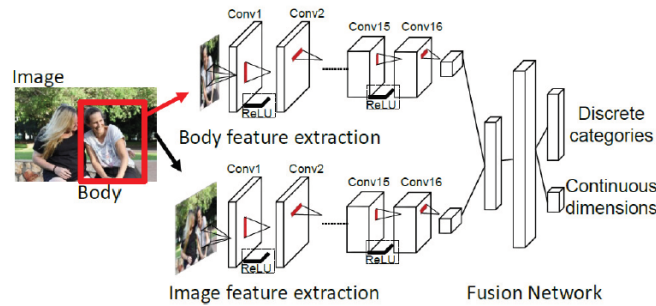
Σε περίπτωση που τα δεδομένα εισόδου απαρτίζονται από ακολουθίες βίντεο, αναδεικνύεται ως πιο αποτελεσματική η χρήση τρισδιάστατων συνελικτικών πυρήνων, με στόχο την εξαγωγή χωρο-χρονικών χαρακτηριστικών. Μία από τις πρώτες εφαρμογές 3D συνελικτικών δικτύων αφορούσε την αναγνώριση ανθρωπίνων δράσεων σε βίντεο [55] ενώ αξιοσημείωτο παράδειγμα αποτελεί και η C3D αρχιτεκτονική που εισήχθη στο [101]. Εκτός από τρισδιάστατα συνελικτικά δίκτυα, η επεξεργασία ακολουθιακών δεδομένων είναι εφικτή και μέσω επαναληπτικών νευρωνικών δικτύων RNN. Συνήθη και ευρέως διαδεδομένη πρακτική αποτελεί ο ακολουθιακός συνδυασμός CNN-RNN, όπως παροσιάζεται στο σχήμα B.4.



Σχήμα B.4: Το μακροπρόθεσμο επαναληπτικό συνελικτικό νευρωνικό δίκτυο (Long-term Recurrent Convolutional Networks, LRCN). Το LRCN επεξεργάζεται ακολουθίες μεταβλητού μήκους από frame (αριστερά) μέσω ενός CNN (μεσαία αριστερά), οι έξοδοι του οποίου τροφοδοτούν μία στοίβα από επαναληπτικά ακολουθιακά μοντέλα (μεσαία δεξιά) τα οποία εν τέλει παράγουν μεταβλητού μήκους διανύσματα εξόδου (δεξιά). Πηγή: [29].

Γ Οπτική-Σημασιολογική Αναγνώριση Συναισθημάτων σε Εικόνες

Μια σύντομη ανασκόπηση της σχετικής με την οπτική αναγνώριση συναισθημάτων βιβλιογραφίας αναδεικνύει την ανάγκη για περαιτέρω μελέτη και πειραματισμό σε μη ελεγχόμενα περιβάλλοντα, που χαρακτηρίζονται από μη προβλέψιμες κοινωνικές συνθήκες της καθημερινής πραγματικής ζωής, κατά τις οποίες μπορεί να είναι παρόντες πολλαπλοί ανθρώπινοι χαρακτήρες, λαμβάνοντας όλων των ειδών στάσης σώματος, καθιστώντας πολλές φορές αδύνατη την επεξεργασία των χαρακτηριστικών του προσώπου και προάγωντας, αντ' αυτού, τη χρήση του ολικού διαθέσιμου οπτικού περιεχομένου, με στόχο την πρόβλεψη των εκάστοτε συναισθηματικών καταστάσεων. Για αυτό το λόγο, αποφασίζουμε η πρώτη περίπτωση μελέτης μας επί του προβλήματος να σχετίζεται με την EMOTions In Context (EMOTIC) βάση δεδομένων.



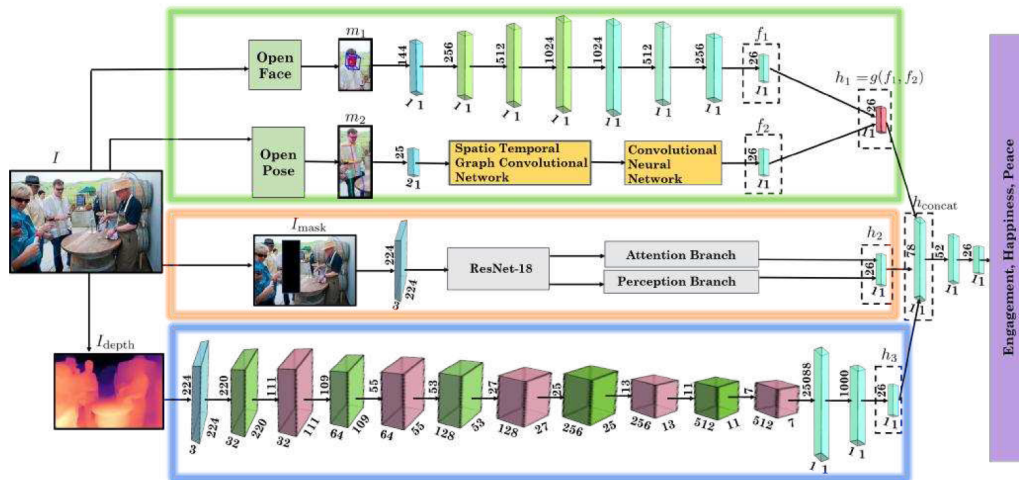
Σχήμα Γ.1: Δομή του baseline μοντέλου, για αναγνώριση συναισθημάτων σε κατηγορικό και συνεχές επίπεδο, στη βάση δεδομένων EMOTIC. Πηγή: [58].

Γ.1 Βασικές Αρχιτεκτονικές

Μια από τις πρώτες αξιοσημείωτες συνεισφορές προς τη κατεύθυνση της οπτικής αναγνώρισης συναισθημάτων με χρήση σημασιολογικού περιεχομένου, προέκυψε με την ίδια τη δημιουργία της βάσης δεδομένων EMOTIC [58]. Επιπλέον, προτάθηκε ένα πρώτο, βασικό μοντέλο για τη πρόβλεψη συναισθημάτων τόσο σε κατηγορικό (26 μη αμοιβαία αποκλειόμενα κατηγορικά συναισθήματα) όσο και σε συνεχές επίπεδο (VAD διαστάσεις) στην εν λόγω βάση. Το προτεινόμενο μοντέλο απαρτιζόταν από δύο κλάδους, ο καθένας εκ των οποίων αποτελούταν από ένα συνελκτικό δίκτυο (CNN). Ο ένας κλάδος τροφοδοτούνταν με ολόκληρη την εκάστοτε εικόνα (context stream), ενώ ο άλλος κλάδος δέχονταν ως είσοδο τη περιοχή της εν λόγω εικόνας που αντιστοιχούσε αυστηρά στο σώμα του πρωταρχικού επισημειωμένου εικονιζόμενου ανθρώπινου χαρακτήρα (body stream). Τα εξαγόμενα deep-learned χαρακτηριστικά των δύο κλάδων συνενώνονταν και εισάγονταν σε ένα δίκτυο σύντηξης (fusion network), προϊόντα του οποίου αποτελούσαν οι συναισθηματικές προβλέψεις σε κατηγορικό και συνεχές επίπεδο. Η δομή του εν λόγω δικτύου φαίνεται στο σχήμα Γ.1

Μια σημαντική βελτίωση στην απόδοση συναισθηματικής αναγνώρισης επί του βασικής προαναφερόμενης αρχιτεκτονικής, προέκυψε στα πλαίσια ανάπτυξης του μοντέλου EmotiCon [71]. Το εν λόγω μοντέλο, περιλαμβάνει πολλαπλές ροές πληροφορίας, όπως το πρόσωπο, τη πόζα, το σημασιολογικό και κοινωνικο-δυναμικό περιεχόμενο. Η εξαγωγή σημασιολογικού περιεχομένου γινόταν στη βάση εικόνων στις οποίες τα σώματα των πρωταρχικών εικονιζόμενων χαρακτήρων ήταν καλυμμένα (masked-out), ενώ ο συγκεκριμένος κλάδος περιελάμβανε και ένα πρόσθετο υποδίκτυο οπτικής προσοχής (Attention Branch Network, ABN) [38]. Για την εξαγωγή σημασιολογικού και κοινωνικού-δυναμικού περιεχομένου χρησιμοποιήθηκαν συνελκτικά δίκτυα. Πιο συγκεκριμένα, η εξαγωγή κοινωνικο-δυναμικού περιεχομένου (socio-dynamic context) έγινε με βάση εκτιμώμενους χάρτες βάθους, που εξήχθησαν με χρήση προ-εκπαιδευμένου δικτύου MegaDepth [63]. Επιπλέον, αξίζει να σημειωθεί πως για την εξαγωγή χαρακτηριστικών πόζας έγινε χρήση γραφο-συνελκτικών δικτύων (GCN) [57]. Τα εξαγόμενα deep-learned χαρακτηριστικά από τους κλάδους που αντιστοιχούν στο πρόσωπο και τη πόζα συνδυάζονται μέσω ενός γραμμικού σχηματισμού και την επιβολή ενός πρόσθετου όρου πολλαπλασιαστικού σφάλματος (multiplicative loss), όπως αρχικά προτάθηκε στο [72]. Τα εξαγόμενα χαρακτηριστικά όλων των κλάδων συνενώνονται σε ένα ενιαίο διάγραμμα χαρακτηριστικών βάσει του οποίου γίνεται συναισθηματική πρόβλεψη μόνο σε κατηγορικό επίπεδο. Η δομή του μοντέλου EmotiCon παρουσιάζεται αναλυτικά στο σχήμα Γ.2.

Μια εναλλακτική προσέγγιση επί του προβλήματος προτάθηκε στο [105], όπου τα εξαγόμενα deep-learned οπτικά χαρακτηριστικά συνδυάστηκαν με ενσωματώσεις λέξεων συναισθηματικού περιεχομένου από Word2Vec μοντέλο, με στόχο την παραγωγή κοινών συναισθηματικών αναπαραστάσεων. Πιο συγκεκριμένα, επιβλήθηκε ένα πρόσθετος όρος σφάλματος, βασισμένος στην



Σχήμα Γ.2: Επισκόπηση της αρχιτεκτονικής του μοντέλου EmotiCon. Πηγή: [71].

απόσταση ομοιότητας συνημιτόνου μεταξύ του διανύσματος οπτικών χαρακτηριστικών, όπως αυτό παράγεται μέσω ενός συνελικτικού δικτύου, και των ενσωματώσεων που αντιστοιχούν στις συναισθηματικές επισημειώσεις του εκάστοτε δείγματος εισόδου.

Γ.2 Προτεινόμενη Μέθοδος

Απώτερο σκοπό της προτεινόμενης μεθόδου αποτελεί η ενίσχυση της συναισθηματικής αντίληψης των κλασικών συνελικτικών αρχιτεκτονικών, όπως προορίζονται για εξαγωγή οπτικών χαρακτηριστικών γενικού σκοπού, μέσα από τη ταυτόχρονη και αποτελεσματική αξιοποίηση πολλαπλών ροών πληροφορίας. Ως τη συνελικτική δομική μονάδα του ευρύτερου μοντέλου μας, επιλέγουμε να χρησιμοποιήσουμε την αρχιτεκτονική υπολειπόμενων δικτύων ResNet [48], καθώς έχουν αναδειχθεί ως state-of-the-art εξαγωγείς οπτικών χαρακτηριστικών σε πληθώρα εφαρμογών, ενώ παράλληλα ο αποτελεσματικός τους σχεδιασμός, συμβάλλει σε χαμηλό υπολογιστικό κόστος κατά την διαδικασία εκπαίδευσης.

Βάση του υπολογιστικού μας μοντέλου αποτελεί ένας συνελικτικός κλάδος που τροφοδοτείται αποκλειστικά με το body crop του πρωταρχικού επισημειωμένου ανθρώπινου χαρακτήρα που απεικονίζεται στο εκάστοτε δείγμα εισόδου. Παράλληλα στον εν λόγω κλάδο, τοποθετούμε αρχικά ένα όμοιο συνελικτικό δίκτυο, προοριζόμενο για την εξαγωγή χαρακτηριστικών σημασιολογικού περιεχομένου (context). Ύστερα από πειραματισμούς με Class Activation Mappings (CAMs), καταλήγουμε στο ότι για να διαχωριστούν αποτελεσματικά οι περιοχές οπτικής προσοχής των δύο προαναφερόμενων κλάδων απαιτείται η κάλυψη των πρωταρχικών εικονιζόμενων χαρακτήρων, στις εικόνες που τροφοδοτούν τον context κλάδο.

Εν συνεχεία, προσθέτουμε ένα συνελικτικό κλάδο για τη κωδικοποίηση των χαρακτηριστικών του προσώπου. Ο εντοπισμός και απομόνωση των περιοχών κάθε εικόνας που αντιστοιχούν στα πρόσωπα των επισημειωμένων χαρακτήρων πραγματοποιήθηκε με χρήση του εργαλείου OpenFace 2.0 [6]. Επιπρόσθετα, επιχειρούμε να εξάγουμε χαρακτηριστικά σχετικά με τη πόζα των εικονιζόμενων ατόμων και εξετάζουμε την εφαρμογή δύο ξεχωριστών μεθοδολογιών, που βασίζονται σε μονοδιάστατα συνελικτικά (1D CNN) και γραφό-συνελικτικά δίκτυα (GCN), αντίστοιχα. Και στις δύο περιπτώσεις, τα δίκτυα τροφοδοτούνται με τις διαδιάστατες συντεταγμένες των αρθρώσεων που απαρτίζουν το σκελετό του εκάστοτε εικονιζόμενου χαρακτήρα, όπως αυτές εντοπίζονται με χρήση του OpenPose [15] μοντέλου. Το 1D CNN αποδείχτηκε ως το επικρατέστερο μεταξύ των δύο μεθόδων.

Κατ' επέκταση, παρατηρήσαμε πως η σχηγή και το περιφερειακό περιβάλλον σχετίζονται

άμεσα με τα συναισθήματα που μοιράζονται τα εκάστοτε εικονιζόμενα άτομα. Επομένως, συμπεριλάβαμε τα σκορ ταξινόμησης με βάση το είδος και τα χαρακτηριστικά των εικονιζόμενων σκηνών ως επιπρόσθετα χαρακτηριστικά στη διαδικασία αναγνώρισης συναισθημάτων και, απ' όσο γνωρίζουμε είμαστε οι πρώτοι που επιχειρούμε κάτι τέτοιο. Τα εξαγόμενα πειραματικά αποτελέσματα επιβεβαιώνουν την εγκυρότητα της αρχικής μας διαίσθησης.

Τέλος, επιδιώξαμε να εκμεταλλευτούμε τις αλληλοεξαρτήσεις μεταξύ των κατηγορικών συναισθηματικών επισημειώσεων, όπως εντοπίζονται εντός των εκάστοτε συνόλων δεδομένων, σε μια προσπάθεια να αξιοποιήσουμε περαιτέρω τις διαθέσιμη οπτικο-σημασιολογική πληροφορία. Εφαρμόσαμε δύο ξεχωριστές μεθοδολογίες, με τη μία να βασίζεται στο μηχανισμό ML-GCN [22] και η άλλη να βασίζεται στη μετρική μάθηση (metric learning) και την επιβολή σημασιολογικής συνάφειας μεταξύ των εξαγόμενων οπτικών χαρακτηριστικών και των ενσωματώσεων, μοντέλου GloVe [85], των συναισθηματικών επισημειώσεων, μέσω ενός επιπρόσθετου MSE όρου σφάλματος [37]. Και οι δύο προσεγγίσεις οδήγησαν σε βελτιώσεις στην αναγνώριση συναισθημάτων σε κατηγορικό επίπεδο.

Γ.3 Πειραματικά Αποτελέσματα και Σύγκριση με SOTA

Στη τρέχουσα ενότητα, συγκρίνουμε την απόδοση της προτεινόμενης μεθόδου με άλλα δημοσιευμένα μοντέλα, όπως αυτά περιγράφονται στη σχετική βιβλιογραφία [59], [71], [105], [113]. Οι πίνακες Γ.1 και Γ.2 συγκρίνουν τις αποδόσεις όλων των προαναφερόμενων μοντέλων, στο πλαίσιο αναγνώρισης συναισθημάτων τόσο σε κατηγορικό όσο και σε συνεχές επίπεδο.

Emotion Categories	Performance Comparison					
	Kosti et al. [59]	Zhang et al. [113]	Wei et al. [105]	Ours	Mittal et al. [71]	
					Without Depth	With Depth
1. Affection	27.85	46.89	-	34.02	41.83	45.23
2. Anger	09.49	10.87	-	22.22	11.41	15.46
3. Annoyance	14.06	11.27	-	22.91	17.37	21.92
4. Anticipation	58.64	62.64	-	57.94	67.59	72.12
5. Aversion	07.48	05.93	-	10.69	11.71	17.81
6. Confidence	78.35	72.49	-	76.31	65.27	68.85
7. Disapproval	14.97	11.28	-	20.35	17.35	19.82
8. Disconnection	21.32	26.91	-	31.61	41.46	43.12
9. Disquietment	16.89	16.94	-	22.08	12.69	18.73
10. Doubt/Confusion	29.63	18.68	-	23.95	31.28	35.12
11. Embarrassment	03.18	01.94	-	03.32	10.51	14.37
12. Engagement	87.53	88.56	-	85.85	84.62	91.12
13. Esteem	17.73	13.33	-	17.45	18.79	23.62
14. Excitement	77.16	71.89	-	71.70	80.54	83.26
15. Fatigue	09.70	13.26	-	15.98	11.95	16.23
16. Fear	14.14	04.21	-	10.41	21.36	23.65
17. Happiness	58.26	73.26	-	79.03	69.51	74.71
18. Pain	08.94	06.52	-	11.34	09.56	13.21
19. Peace	21.56	32.85	-	25.32	30.72	34.27
20. Pleasure	45.46	57.46	-	49.17	61.89	65.53
21. Sadness	19.66	25.42	-	33.66	19.74	23.41
22. Sensitivity	09.28	05.99	-	10.54	04.11	08.32
23. Suffering	18.84	23.39	-	38.78	20.92	26.39
24. Surprise	18.81	09.02	-	13.23	16.45	17.37
25. Sympathy	04.71	17.53	-	15.33	30.68	34.28
26. Yearning	08.34	10.55	-	10.41	10.53	14.29
Mean	27.38	28.42	30.96	31.29	31.53	35.48

Πίνακας Γ.1: Σύγκριση απόδοσης ως προς το *average precision* (AP, %), για κάθε κατηγορικό συνάισθημα, μεταξύ της προτεινόμενης μεθόδου μας και άλλων δημοσιευμένων μοντέλων, χρησιμοποιώντας το σύνολο δειγμάτων αξιολόγησης της βάσης δεδομένων EMOTIC.

Η προτεινόμενη μέθοδος μας πετυχαίνει συγκρίσιμα σκορ αναγνώρισης σε σχέση με τις τρέχουσες state-of-the-art υλοποιήσεις. Πιο συγκεκριμένα, πετυχαίνουμε 31.29% AP, υπολείποντας του υψηλότερου δημοσιευμένου ποσοστού αναγνώρισης κατά μόλις 0.2%, δεδομένου ότι

Continuous Dimensions	Performance Comparison				
	Kosti et al. [59]	Zhang et al. [113]	Wei et al. [105]	Ours	Mittal et al. [71]
Valence	0.0528	0.07	-	0.0788	-
Arousal	0.0611	0.1	-	0.0934	-
Dominance	0.0579	0.1	-	0.0898	-
Mean	0.0573	0.09	-	0.0873	-

Πίνακας Γ.2: Σύγκριση απόδοσης ως προς το *average absolute error* (AAE), για κάθε συναισθηματική διάσταση, μεταξύ της προτεινόμενης μεθόδου μας και άλλων δημοσιευμένων μοντέλων, χρησιμοποιώντας το σύνολο δειγμάτων αξιολόγησης της βάσης δεδομένων EMOTIC.

δεν έχουμε λάβει υπόψη τη χρήση δεδομένων βάθους. Επιπλέον, τα πειραματικά αποτελέσματα αναδεικνύουν την αδυναμία του μοντέλου μας να αποδόσει το ίδιο καλά και κατά την πρόβλεψη συναισθημάτων στις VAD διαστάσεις, σημειώνοντας αύξηση στο AAE περίπου ίση με 0.03 συγκριτικά με το αντίστοιχο baseline [59].

Δ Οπτική-Σημασιολογική Αναγνώριση Συναισθημάτων σε Βίντεο

Στη παρούσα ενότητα, επιχειρούμε να επεκτείνουμε τις ιδέες περί οπτικής αναγνώρισης συναισθημάτων, βασισμένη στο σημασιολογικό περιεχόμενο, υπό τις δυναμικές συνθήκες ακολουθιών βίντεο. Για τη μελέτη της αυτής της πτυχής του εν λόγω προβλήματος, πραγματοποιούμε εκτενή πειράματα στη νεοσύστατη και απαιτητική βάση δεδομένων Body Language Dataset (BoLD) [67].

Δ.1 Βασικές Αρχιτεκτονικές

Σε πρώτη φάση, καλούμαστε να θέσουμε το απαραίτητο θεωρητικό υπόβαθρο σχετικά με βασικές αρχιτεκτονικές βαθιών νευρωνικών δικτύων, οι οποίες έχουν χρησιμοποιηθεί ευρέως σε διάφορων ειδών εφαρμογές επεξεργασίας βίντεο και που αργότερα θα αποτελέσουν δομικές μονάδες των δικών μας υλοποιήσεων.

Συνελικτικά Δίκτυα Δύο Ροών

Η αρχιτεκτονική *συνελικτικών δικτύων δύο ροών* (two-stream convolutional networks) [94] είχε αρχικά προταθεί για αναγνώριση ανθρωπίνων δράσεων σε ακολουθίες βίντεο. Η εν λόγω δομή περιλαμβάνει δύο συνελικτικούς κλάδους, οι οποίοι είναι υπεύθυνοι για την ταυτόχρονη επεξεργασία του χωρικού και χρονικού περιεχομένου των βίντεο. Από κάθε ακολουθία εισόδου, δειγματοληπτείται με τυχαίο και ομοιόμορφο τρόπο, ένα μεμονωμένο frame το οποίο και τροφοδοτεί το χωρικό κλάδο. Έτσι, ο χωρικός κλάδος βοηθά στη σύλληψη της στατικής εμφάνισης και των χωρικών συσχετίσεων μεταξύ των εικονιζόμενων χαρακτήρων και αντικειμένων. Κατ'αντιστοιχία, ο χρονικός κλάδος τροφοδοτείται με $2L$ μονοκαναλικές εισόδους, οι οποίες αποτελούνται από τις οριζόντιες και κατακόρυφες συνιστώσες οπτικής ροής, όπως προκύπτουν από στοίβες L συνεχόμενων frames. Εν συνεχεία, οι επιμέρους προβλέψεις που εξάγονται από τους δύο κλάδους, συνδιάζονται έτσι ώστε να προκύψει μια κοινή πρόβλεψη για το εκάστοτε βίντεο.

Δίκτυα Χρονικών Τμημάτων

Άμεση προέκταση των *συνελικτικών δικτύων δύο ροών* συνιστούν τα *δίκτυα χρονικών τμημάτων* (Temporal Segment Networks) [104]. Βασικό στοιχείο την εν λόγω δομής, αποτελεί η εφαρμογή ενός σχήματος αραιής χρονικής δειγματοληψίας κατά τη λειτουργία, τόσο του χωρικού όσο και του χρονικού συνελικτικού κλάδου. Πιο συγκεκριμένα, κάθε ακολουθία βίντεο εισόδου,

χωρίζεται σε χρονικά ισομήκη τμήματα (segments), από κάθε ένα εκ των οποίων δειγματοληπείται ομοιόμορφα και τυχαία ένα μεμονωμένο frame (snippet). Τα δειγματοληπτημένα frames τροφοδοτούν ένα χωρικό συνελικτικό κλάδο και παράγουν ένα αντίστοιχο αριθμό προβλέψεων. Για να εξαχθεί απο κοινού πρόβλεψη σε επίπεδο βίντεο, οι μεμονωμένες προβλέψεις από τα snippets συνδιάζονται μέσω μια συνάρτησης διατμηματικής συμφωνίας (segmental consensus function), η οποία στη πλειοψηφία των περιπτώσεων είναι ένας απλός μέσος όρος. Το προαναφερόμενο σχήμα χρονικά αραιής δειγματοληψίας μπορεί να εφαρμοστεί, με τον ίδιο ακριβώς τρόπο, και σε χρονικό συνελικτικό κλάδο οποίος θα τροφοδοτείται με στρίβες συνιστωσών οπτικής ροής που θα αντιστοιχούν σε καθένα από τα δειγματοληπτημένα snippets του χωρικού κλάδου. Και σε αυτή τη περίπτωση, οι εξαγόμενες επιμέρους προβλέψεις των δύο κλάδων συνδιάζονται για την απόκτηση κοινής πρόβλεψης για κάθε βίντεο.

Χωρο-Χρονικά Γραφο-Συνελικτικά Δίκτυα

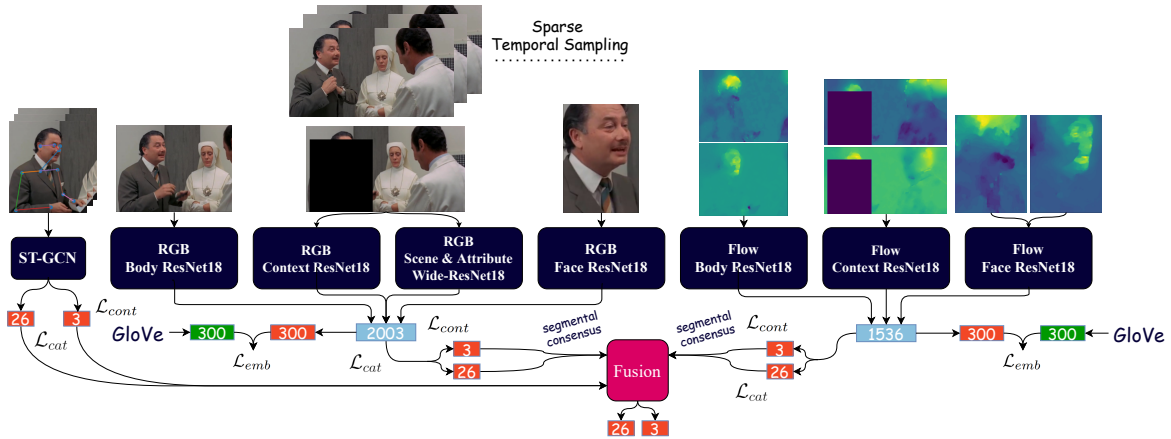
Τα χωρο-χρονικά γραφο-συνελικτικά δίκτυα (Spatial-Temporal Graph-Convolutional Networks, ST-GCN) [108], αποτελούν άμεση προέκταση των απλών γραφο-συνελικτικών δικτύων (GCN) [57] στη διάσταση του χρόνου, όπως είχαν προταθεί αρχικά για αναγνώριση ανθρωπίνων δράσεων μέσω ανάλυσης ακολουθιών ανθρωπίνων σκελετών. Τα ST-GCN δρουν πάνω σε ακολουθίες από γράφους, αυθαίρετης τοπολογίας, $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, όπου \mathcal{V} και \mathcal{E} αποτελούν τα σύνολα όλων των κόμβων και ακμών, κατά μήκος της εκάστοτε ακολουθίας. Το σύνολο ακμών \mathcal{E} απαρτίζεται από δύο υποσύνολα \mathcal{E}_S και \mathcal{E}_F , με το πρώτο να περιλαμβάνει τις εσωτερικές ακμές μεταξύ των κόμβων κάθε γράφου της ακολουθίας, και με το δεύτερο να αντιστοιχεί στις εξωτερικές ακμές οι οποίες συνδέουν τους ίδιους κόμβους μεταξύ διαδοχικών γράφων. Η χωρο-χρονική συνέλιξη γραφημάτων υλοποιείται σε δύο φάσεις, όπου κατά τη πρώτη πραγματοποιείται χωρική συνέλιξη βάση του κανόνα εμπρόσθιας διάδοσης των απλών GCN, το προϊόν της οποίας εν συνεχεία συνελίσσεται κατά μήκος του χρονικού άξονα. Δεδομένης αυτής της υψηλού επιπέδου περιγραφής της χωρο-χρονικής συνέλιξης γραφημάτων, σημαντικό ρόλο παίζει και η δομή του χρησιμοποιούμενου πίνακα γειτνίασης (adjacency matrix) ο οποίος εξαρτάται εν γένει από τη στρατηγική που εφαρμόζεται για τη διαμέριση των κόμβων κατά τη διαδικασία της συνέλιξης. Η διαμέριση των κόμβων μπορεί να γίνει ομοιόμορφα (uniform), με βάση την απόσταση μεταξύ των κόμβων (distance) καθώς και με βάση την απόσταση των κόμβων από το βαρυτικό κέντρο του γράφου (spatial).

Δ.2 Προτεινόμενη Μέθοδος

Η προτεινόμενη μέθοδος μας βασίζεται στον ύστερο συνδυασμό προβλέψεων (late score fusion) ανάμεσα σε μια τροποποιημένη εκδοχή των TSN και ενός κατάλληλα προ-εκπαιδευμένου ST-GCN. Ως τη συνελικτική δομική μονάδα του ευρύτερου μοντέλου μας, επιλέγουμε να χρησιμοποιήσουμε την αρχιτεκτονική υπολειπόμενων δικτύων ResNet-18 [48]. Η δομή του συνολικού προτεινόμενου μοντέλου φαίνεται στο σχήμα Δ.1.

TSN-RGB

Αρχικά εξετάζουμε την επέκταση της κλασικής δομής του TSN που δρα πάνω σε RGB frames, με την προσθήκη πολλαπλών ροών πληροφορίας που θα εστιάζουν σε διαφορετικά τμήματα των εκάστοτε δοσμένων εικόνων και ουσιαστικά θα κωδικοποιούν χαρακτηριστικά σχετικά με το πρόσωπο και το σώμα των εικονιζόμενων ατόμων καθώς και το σημασιολογικό περιεχόμενο που έγκειται στο εικονιζόμενο περιφερειακό περιβάλλον, αντικείμενα και δευτερεύοντες ανθρώπινους χαρακτήρες. Κάθε ροή πληροφορίας εντός του δικτύου υλοποιείται ως ένας κατάλληλα προεκπαιδευμένος συνελικτικός κλάδος. Ο βασικός συνελικτικός κλάδος του δικτύου τροφοδοτείται με τα body crops του πρωταρχικού επισημειωμένου ανθρώπινου χαρακτήρα που απεικονίζεται στο



Σχήμα Δ.1: Πλήρες σχηματικό διάγραμμα του προτεινόμενου μοντέλου για αναγνώριση συναισθημάτων στη βάση δεδομένων BoLD. Τα συνενωμένα διανύσματα χαρακτηριστικών απεικονίζονται με κυανό χρώμα, τα πλήρως συνδεδεμένα στρώματα απεικονίζονται με πορτοκαλί χρώμα και οι ενσωματώσεις GloVe απεικονίζονται με πράσινο χρώμα, μαζί με τη διάστασή τους ή τον αριθμό των κρυφών μονάδων. Το ST-GCN μοντέλο παράγει εγγενώς προβλέψεις σε επίπεδο βίντεο, ενώ στην περίπτωση των TSN-RGB και TSN-Flow, αυτό απαιτεί προηγουμένως, την εφαρμογή μια συνάρτησης διατηρηματικής συμφωνίας σχετικά με τις αντίστοιχες προβλέψεις σε επίπεδο snippet (26 πιθανότητες ταξινόμησης για διακριτά συναισθήματα, 3 συνεχείς τιμές για τις διαστάσεις VAD). Οι τελικές προβλέψεις λαμβάνονται μέσω εφαρμογής σταθμισμένου μέσου.

εκάστοτε δείγμα εισόδου (body stream). Παράλληλα σε αυτόν, τοποθετούμε ένα όμοιο κλάδο με σκοπό την εξαγωγή χαρακτηριστικών σημασιολογικού περιεχομένου (context stream) και ο οποίος τροφοδοτείται με RGB εικόνες στις οποίες έχουν εκ των προτέρων καλυφθεί (masked-out) οι πρωταρχικοί επισημειωμένοι εικονιζόμενοι χαρακτήρες. Κατ' επέκταση ενσωματώνουμε ένα κλάδο που επεξεργάζεται αποκλειστικά τις περιοχές των εικόνων εισόδου που αντιστοιχούν στα πρόσωπα των πρωταρχικών εικονιζόμενων χαρακτήρων (face stream). Τέλος, προσθέτουμε ένα συνελικτικό δίκτυο για την εξαγωγή των σκορ ταξινόμησης με βάση το είδος και τα χαρακτηριστικά των εικονιζόμενων σκηνών. Τα εξαγόμενα deep-learned χαρακτηριστικά όλων των κλάδων συνενώνονται σε ένα ενιαίο διάνυσμα βάσει του οποίου εξάγονται αρχικά προβλέψεις σε επίπεδο snippet και κατ' επέκταση σε επίπεδο βίντεο.

TSN-Flow

Ακολουθώντας τις ίδιες σχεδιαστικές αρχές, επιχειρούμε να εκπαιδύσουμε ένα TSN δίκτυο το οποίο θα επεξεργάζεται τα πεδία οπτικής ροής που αντιστοιχούν στα δειγματοληπτημένα frames του που τροφοδοτούν το TSN-RGB. Ο κύριος συνελικτικός κλάδος του εν λόγω δικτύου τροφοδοτείται με πεδία οπτικής ροής, εντοπισμένα στα σώματα των πρωταρχικών εικονιζόμενων χαρακτήρων (body stream), κωδικοποιώντας τις αντίστοιχες κινήσεις τους στη διάρκεια του βίντεο. Καλύπτοντας τα τμήματα οπτικής ροής που αντιστοιχούν στους πρωταρχικούς χαρακτήρες και τροφοδοτώντας τα σε ένα ξεχωριστό συνελικτικό κλάδο, επιδιώκουμε να κωδικοποιήσουμε τις κινήσεις των εκάστοτε δευτερευόντων εικονιζόμενων χαρακτήρων (context stream). Ακολούθως, τροφοδοτούμε ένα συνελικτικό δίκτυο αποκλειστικά με τμήματα οπτικής ροής που αντιστοιχούν αποκλειστικά στα πρόσωπα των πρωταρχικών εικονιζόμενων χαρακτήρων (face stream). Έτσι, ο συγκεκριμένος κλάδος εστιάζει στην πιο λεπτομερή κωδικοποίηση των ορατών μυϊκών δράσεων του προσώπου. Όπως συνέβη και στο TSN-RGB δίκτυο, τα εξαγόμενα deep-learned χαρακτηριστικά όλων των κλάδων συνενώνονται σε ένα ενιαίο διάνυσμα βάσει του οποίου εξάγονται αρχικά προβλέψεις σε επίπεδο snippet και κατ' επέκταση σε επίπεδο βίντεο. Για την εκπαίδευση τόσο του TSN-RGB όσο και του TSN-Flow δικτύου, χρησιμοποιούμε ένα πρόσθετο όρο σφάλματος βασισμένο στις ενσωματώσεις GloVe [85] των συναισθηματικών επισημειώσεων,

κατά παρόμοιο τρόπο, όπως παρουσιάστηκε στα προηγούμενα πειράματα μας, αναφορικά με τη βάση δεδομένων EMOTIC.

Μάθηση με Βάση τον Ανθρώπινο Σκελετό

Εξετάζουμε δύο διαφορετικές μεθοδολογίες για εξαγωγή συναισθηματικών αναπαραστάσεων με βάση τον ανθρώπινο σκελετό. Η πρώτη βασίζεται στη χρήση ενός ST-GCN μοντέλου το οποίο έχει προ-εκπαιδευτεί στη βάση δεδομένων Kinetics [16] και το οποίο τροφοδοτείται με ακολουθίες από τις διαστάσεις συντεταγμένες των αρθρώσεων που συνιστούν τον σκελετό του πρωταρχικού επισημειωμένου εικονιζόμενου χαρακτήρα, σε κάθε βίντεο. Επίπροσθετα, πειραματιζόμαστε με τη χρήση χαρακτηριστικών της κινησιακής ανάλυσης *Laban* (Laban Movement Analysis, LMA) [60] σε συνδυασμό με ταξινομητή (classifier) και παλινδρομητή (regressor) τύπου Random Forest. Βάσει των πειραματικών αποτελεσμάτων, η πρώτη μέθοδος αποδείχτηκε επικρατέστερη.

Δ.3 Πειραματικά Αποτελέσματα και Σύγκριση με SOTA

Για τον ύστερο συνδυασμό των επιμέρους προβλέψεων από τα TSN-RGB, TSN-Flow και ST-GCN μοντέλα, εξετάζουμε μεθόδους όπως τη λήψη μεγίστου, απλού μέσου όρου καθώς και σταθμισμένου μέσου όρου. Καλύτερη μέθοδος αναδείχτηκε αυτή του σταθμισμένου μέσου όρου με αναλογία βαρών 2:2:1, κατ' αντιστοιχία με τα τρία προαναφερόμενα μοντέλα. Επιλέγουμε το καλύτερο συνολικό μοντέλο μας, όπως αναδείχθηκε μέσα από πειράματα που διεξήχθησαν στο σύνολο δειγμάτων επικύρωσης (validation set) της βάσης δεδομένων BoLD και το συγκρίνουμε με τη baseline υλοποίηση του [67] καθώς και το state-of-the-art μοντέλο του [37], χρησιμοποιώντας το αντίστοιχο επίσημο σύνολο αξιολόγησης (test set). Στο πίνακα Δ.1 παρουσιάζονται αναλυτικά τα αποτελέσματα. Η αξιολόγηση της απόδοσης ως προς το πρόβλημα της ταξινόμησης γίνεται με βάση τις μετρικές της μέσης ακρίβειας (*average precision*, AP) και του εμβαδού κάτω από τη χαρακτηριστική καμπύλη λειτουργίας δέκτη (*area under the receiver operating characteristic*, RA). Η αξιολόγηση της απόδοσης ως προς το πρόβλημα της παλινδρόμησης γίνεται με βάση τη μετρική του συντελεστής προσδιορισμού (*coefficient of determination*, R^2). Στο πίνακα Δ.1 εμφανίζονται οι μέσες τιμές των παραπάνω μετρικών για το σύνολο των 26 πιθανών, μη αμοιβαία αποκλειόμενων κατηγορικών συναισθημάτων και των τριών συναισθηματικών διαστάσεων VAD, αντίστοιχα.

Τα παραπάνω πειραματικά δεδομένα αναδεικνύουν την υπεροχή της παρούσας υλοποίησης. Ο συνδυασμός ενός κατάλληλα προεκπαιδευμένου ST-GCN και του τροποποιημένου TSN συνέβαλε στο να πετύχουμε σημαντικές βελτιώσεις επί των state-of-the-art υλοποιήσεων αναφορικά με βάση δεδομένων BoLD, πετυχαίνοντας 0.3051 *emotion recognition score* (ERS) στο επίσημο σύνολο αξιολόγησης. Επιπλέον, καθώς χρησιμοποιούμε ρηχά συνελικτικά δίκτυα για εξαγωγή

Set	Model	Regression	Classification		ERS
		$mR^2 \uparrow$	mAP \uparrow	mRA \uparrow	
Valid.	TSN-RGB+TSN-Flow (ours)	0.1444	0.1883	0.6661	0.2858
	TSN-RGB+TSN-Flow+ST-GCN (ours)	0.1489	0.1929	0.6682	0.2897
Test	Luo et al. [67]	0.1030	0.1714	0.6352	0.2530
	Filntisis et al. [37]	0.1141	0.1796	0.6416	0.2624
	Ours	0.1597	0.2185	0.6826	0.3051

Πίνακας Δ.1: Ποσοτικά αποτελέσματα στα σύνολα επικύρωσης (validation) και αξιολόγησης (test) της βάσης δεδομένων BoLD αναφορικά με την προτεινόμενη υλοποίηση μας και άλλα δημοσιευμένα μοντέλα στην σχετική βιβλιογραφία. Οι μετρικές απόδοσης εκφράζονται στο διάστημα [0,1].

deep-learned χαρακτηριστικών, η παρατηρούμενη βελτίωση στην απόδοση δεν συνοδεύεται από ανάλογη αύξηση του υπολογιστικού κόστους εκπαίδευσης.

Ε Συνεισφορές και Μελλοντικές Προεκτάσεις

Στη παρούσα διπλωματική εργασία, αντιμετωπίσαμε το πρόβλημα της οπτικής αναγνώρισης συναισθημάτων με χρήση σημασιολογικού περιεχομένου. Πιο συγκεκριμένα, μελετήσαμε σε βάθος τόσο τη στατική όσο και τη δυναμική έκδοση του εν λόγω προβλήματος και αξιολογήσαμε τις προτεινόμενες μεθόδους μας, πραγματοποιώντας πειράματα σε δύο νεοσύστατες και απαιτητικές βάσεις δεδομένων, την EMOTions In Context (EMOTIC) και τη Body Language Dataset (BoLD).

Μέσα από εκτενή πειράματα στη βάση EMOTIC αποδείξαμε πως η απο κοινού χρήση πολλαπλών ροών πληροφορίας, όπως το ανθρώπινο σώμα, πρόσωπο, η πόζα και το σημασιολογικό περιεχόμενο συμβάλλουν ανθρωστικά στη βελτίωση της απόδοσης κατά την αναγνώριση συναισθημάτων. Αυτή η πολυτροπική προσέγγιση του εν λόγω προβλήματος αναδεικνύεται ως αναγκαία σε περιπτώσεις κατά τις οποίες η διαδικασία της αναγνώρισης συναισθημάτων πραγματοποιείται σε μη ελεγχόμενα περιβάλλοντα, όπου μη προβλέψιμες κοινωνικές συνθήκες και καταστάσεις μπορούν να καταστήσουν μία ή και πολλαπλές από τις προαναφερόμενες πηγές συναισθηματικής πληροφορίας, προσωρινά μη προσβάσιμες. Με τη προτεινόμενη μέθοδο μας καταφέραμε να πετύχουμε ένα ανταγωνιστικό σκορ αναγνώρισης 31.29% mAP, στο επίσημο σύνολο δεδομένων αξιολόγησης της βάσης EMOTIC, υπολείποντας του τρέχοντος υψηλότερου δημοσιευμένου σκορ κατά μόλις 0.2%, δεδομένου ότι έχουμε αποκλίσει τη χρήση δεδομένων βάθους.

Ακολούθως, επιχειρήσαμε να επεκτείνουμε την ιδέα της σημασιολογικής-οπτικής αναγνώρισης συναισθημάτων, υπό δυναμικές συνθήκες ακολουθιών βίντεο. Στόχο μας αποτέλεσε ο σχεδιασμός ενός ενοποιημένου μοντέλου που θα μοιραζόταν την ήδη πετυχημένη αρχιτεκτονική που υλοποιήσαμε κατά τις πειραματικές μελέτες που διεξήχθησαν στη βάση δεδομένων EMOTIC, ενώ ταυτόχρονα θα ήταν ικανό να επεξεργάζεται βίντεο. Επιλέξαμε τα Temporal Segment Networks (TSN) ως τη ραχοκοκαλιά του δικτύου μας, ενώ επίσης χρησιμοποιήσαμε στοιχεία της Laban Movement Analysis (LMA) καθώς και Spatial-Temporal Graph Convolutional Networks (ST-GCN) ως μέσα μάθησης βάσει του ανθρώπινου σκελετού.

Ακολουθώντας τις ίδιες σχεδιαστικές αρχές όπως στις προηγούμενες υλοποιήσεις μας, επεκτείνουμε την TSN αρχιτεκτονική ενσωματώνοντας πολλαπλές ροές πληροφορίας που ουσιαστικά κωδικοποιούν στοιχεία του ανθρώπινου σώματος και προσώπου, στοιχεία σημασιολογικού περιεχομένου καθώς και στοιχεία συσχετιζόμενα με το περιφερειακό εικονιζόμενο περιβάλλον, ενισχύοντας κατά αυτό το τρόπο την από κοινού αντίληψη του μοντέλου για τα ανθρώπινα συναισθήματα και το εικονιζόμενο οπτικό περιεχόμενο. Ο συνδυασμός ενός κατάλληλα προεκπαιδευμένου ST-GCN και του τροποποιημένου TSN συνέβαλε στο να πετύχουμε σημαντικές βελτιώσεις επί των state-of-the-art υλοποιήσεων αναφορικά με τη βάση δεδομένων BoLD, πετυχαίνοντας 0.3051 *emotion recognition score* (ERS) στο επίσημο σύνολο αξιολόγησης.

Τέλος, προκειμένου να επεκτείνουμε και να βελτιώσουμε την έρευνα που έγινε σε αυτή την διπλωματική εργασία προτείνουμε κάποιες μελλοντικές προεκτάσεις:

- Διερεύνηση των αιτιών λόγω των οποίων δεν παρατηρήθηκε η αναμενόμενη βελτίωση στην απόδοση αναγνώρισης συναισθημάτων σε συνεχές επίπεδο (VAD διαστάσεις), κατ' αναλογία με τις αυξήσεις των ποσοστών επιτυχούς αναγνώρισης σε κατηγορικό επίπεδο, κατά τη διάρκεια των πειραμάτων μας στη βάση δεδομένων EMOTIC.
- Χρήση βάσης δεδομένων τύπου RGB-D, που κατά προτίμηση θα προορίζεται για αναγνώριση ανθρωπίνων δράσεων, με στόχο τη προεκπαίδευση συνελκτικού δικτύου σε δεδομένα

βάθους, πριν την εφαρμογή του για εξαγωγή συναισθηματικής πληροφορίας, τόσο σε ει-
κόνες όσο και σε ακολουθίες βίντεο.

- Ενσωμάτωση του ML-GCN μηχανισμού στη TSN αρχιτεκτονική, με στόχο την περαι-
τέρω βελτίωση της απόδοσης σχετικά με την αναγνώριση συναισθημάτων σε κατηγορικό
επίπεδο.

Chapter 1

Introduction

The interpretation, perception and recognition of human affect has been a subject of rigorous studies and analysis across several scientific disciplines such as biology, psychology, sociology, neurology and last but not least, computer science. While the aforementioned cognitive sciences focus on the extraction of the available affective information, the fields of computer vision and machine learning aim at automating the recognition process through the development of novel techniques and algorithms which are capable of producing effective and robust encodings of such information. Automatic affect recognition bears an immense practical importance as it has extensive applications in environments that involve human-robot cooperation, sociable robotics, medical treatment, psychiatric patient surveillance, driver fatigue surveillance and many other human-computer interaction scenarios.

1.1 Models of Emotion

As a first step in understanding the concepts of emotion perception, interpretation and recognition, we ought to establish the theoretical foundations based on which emotional states are modeled. The best way of modelling affect has been a subject of debate for a long time and many perspectives upon the topic have been proposed. The most relevant models for affective computing can be classified in three main categories: categorical, dimensional and componential. In the following paragraphs we will try to introduce the main features, advantages and disadvantages of each model in relation to affective computing.

1.1.1 Categorical Models

Categorical models are used for classifying emotions in discrete categories that can be recognized and described easily in daily language. The major development in categorical emotion models is attributed to the work of Ekman and Friesen [33], [34] and their underlying assumptions about the universality of a set of six basic emotions, namely happiness, sadness, fear, anger, disgust and surprise. A depiction of the aforementioned emotions is illustrated in figure 1.1. To demonstrate the hypothesized universal element, Ekman and Friesen initially conducted experiments in which they showed still photographs of faces to people of different cultural backgrounds in order to test whether the participants would classify the depicted and narrated affective states as they same emotions, despite their cultural differences. The participants of the experiments belonged into several different culture groups, including college students from the United States, Argentina, Brazil, Chile and Japan, as well as two pre-literate cultures (the Sadong of Borneo and the Fore of New Guinea) which had extensive contact with Western cultures. The initial results suggested that emotions were perceived uniformly across all cultures, re-enforcing the concept of universal emotion perception.

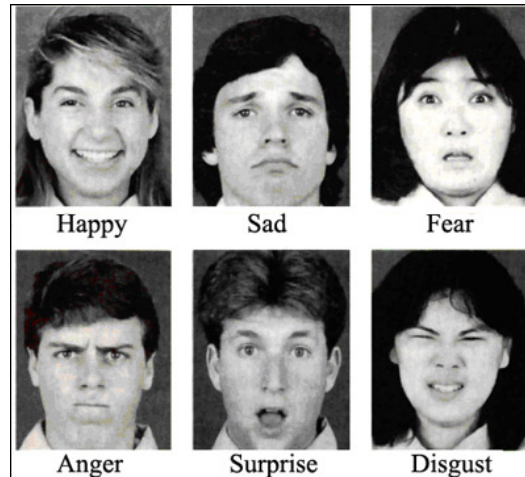


Figure 1.1: The categorical way of describing affect on the basis of the universal set of six emotions, i.e. happiness, sadness, fear, anger, surprise and disgust. Source: [77].

However, their interpretations were open for argument as all of the participants were found to have been exposed to the Western mass media portrayals of facial behavior. In that way, in order to overcome the difficulties in interpretation of their previous results, they repeated the experiments, using participants from the Fore linguistic-cultural group of New Guinea, who were selected on the condition that they did not understand nor speak English or Pidgin (grammatically simplified means of communication that develops between two or more groups that do not have a language in common), had not come in contact with a Caucasian, had not watched movies and had not visited any western settlement. A single test item during the evaluation of the participants consisted of an emotional story, a picture depicting a facial behavior that matched the one described in the story and two more irrelevant pictures. Every participant was asked to make at least three emotion discriminations, as a particular story was narrated more than once together with different combinations of matching and irrelevant photographs. The experimental results showed that the pre-literate participants, of all ages and both sexes, performed comparably with people of literate Western cultures in recognizing the depicted and narrated emotional states, supporting the authors' initial hypothesis for universality associated with the perception of the six aforementioned emotions.

On account of the simplicity of the categorical models of emotion, combined with the associated universality claim, the universal emotions hypothesis has been undoubtedly the primary tool in research relative to affective computing, as stated by Noroozi et al. [78]. However, it is the simplicity of the categorical models that limits their ability to capture and describe more complex emotional states.

1.1.2 Dimensional Models

Following the categorical models, dimensional models constitute the second most popular and widely used method for describing emotions. In a dimensional model, an affective state is represented as a point on a continuum spanned by a set of independent dimensions. Several dimensional models have emerged over the years through research conducted in the field of psychology. However, the Pleasure-Arousal-Dominance (PAD) model is the most commonly used in affective computing.

The aforementioned system was originally developed by Mehrabian and Russell [70], [91]. Extensive studies and experiments supported that the three factors of Pleasure-Arousal-Dominance are independent, as any value along one dimension can occur simultaneously with

any value on either of the other two dimensions. In addition, the three dimensions are defined as bipolar. Pleasure (or valence) describes how positive or negative a feeling is, ranging from extreme pain or unhappiness at one end to extreme happiness or ecstasy at the other end. Arousal corresponds to the level of activation, mental alertness and physical activity, ranging from sleep, through intermediate states of drowsiness and then frenzied excitement on the opposite extreme. Lastly, dominance represents the amount of control over others and the surrounding environment, ranging from total lack of control and a sense of vulnerability on one end to the opposite extreme of feeling influential and in control. An illustration of the 3D VAD space is shown in figure 1.2.

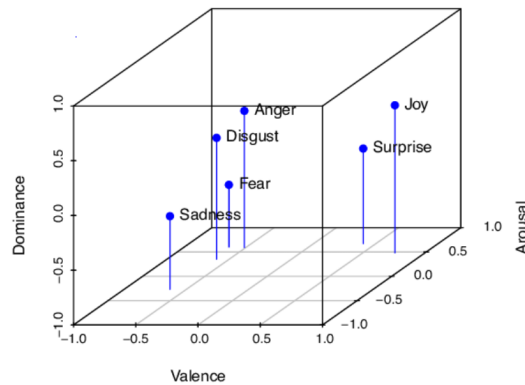


Figure 1.2: The continuous way of describing affect in the form of a point in 3D Valence-Arousal-Dominance (VAD) space. Source: [5].

The continuous nature of the dimensional systems accounts for more complete and rich representations of complex and sophisticated emotional states. The richness of the continuous space is more difficult to use for automatic recognition systems due to the fact that it can be challenging to associate a vectorized description of an emotion to a facial affective behavior or body movement. In that way, several automatic emotion recognition systems that utilize dimensional representations of emotions, simplify the problem by dividing the space to a limited set of categories, such as positive versus negative quadrants of the 2D space [111].

1.1.3 Componential Models

Componential models lie in between categorical and dimensional models, as far as descriptive capacity is concerned. Componential models arrange emotions in a hierarchical manner according to which, emotions that belong to higher or superior layers can be decomposed into a set of more primitive and basic emotions that belong to the exact precedent layers. The most notable example of a componential emotion model is the one introduced by Plutchik [87].

According to Plutchik, emotions constitute complex processes with functional value, both in communication and in increasing an individual's chances of survival, representing proximate methods to achieve evolutionary fitness. He considered there to be eight primary emotions, namely anger, fear, sadness, disgust, surprise, anticipation, trust, and joy, while his argument about primacy was supported on the fact that all the above mentioned motions triggered specific behaviors with a high value of survival, such as the way fear inspires the fight-or-flight response. In addition, he conceptualized primary emotions in a fashion analogous to a color wheel, placing similar emotions close together and opposites at 180 degrees apart, like complementary colors. More specifically, joy was placed against sadness; anger versus fear; trust versus disgust; and surprise versus anticipation. Other emotions can be

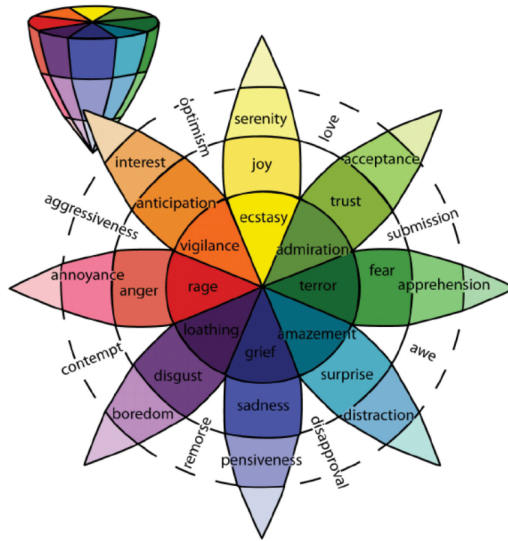


Figure 1.3: The componential way of describing affect on the basis Plutchik’s wheel of emotions. Source: [87].

conceived as mixtures of the primary emotions, just as some colors are primary while others are made by mixing the primary colors. Following this pattern of color theory, we can describe emotions which result from the combination of pairs of fundamental emotions, called dyads, in the same way that red and blue make purple. For example, mixing the primary emotions of joy and trust produces the mixed emotion of love, disgust plus anger results in contempt, anticipation plus joy results in optimism. All the above mentioned examples constitute primary dyads. Subsequently, more sophisticated combinations can be formed, namely secondary dyads, e.g. guilt as a product of joy and fear, as well as tertiary dyads, e.g. delight as a product of joy and surprise. Furthermore, Plutchik extended the initial circumplex model into a third dimension representing the intensity of emotions, resulting in a structured model that was shaped like a cone. The vertical dimension of the cone represents intensity while the horizontal plane represented degrees of similarity among the emotions. An illustration of Plutchik’s expanded circumplex model is illustrated in figure 1.3.

These types of models are rarely utilized in the context of affective computing and automatic emotion recognition related literature, in comparison to the previously mentioned, categorical and dimensional models. However, they should be taken into consideration as an effective compromise between ease of interpretation and expressive capacity.

1.2 Sources of Visual Affective Information

The subject of the current thesis revolves around automatic visual affect recognition in still images as well as videos. As far as visual emotion recognition is concerned, the primary sources of affective information are located at the face and body of the person in question. Most recent studies, e.g. [8], [106] also consider the surrounding environment and depicted scene to be a source of affective information of equal significance. The latter are commonly referred to as the context related features portrayed by an image or video sequence. The notion of context has also been addressed in the form of an implicit communication channel that transmits valuable affective information during interpersonal interactions and can be understood through face and speech signal analysis [23]. As a first step in understanding the nature of the problem, we ought to justify the usage and importance of each of the above

mentioned information cues.

1.2.1 Facial Features

The human face is commonly perceived as the window to the soul. In such way, the facial features are often considered as the primary source of affective information. Affect recognition systems that rely solely on the face aim at recognising the appearance of facial actions or the emotions conveyed by those actions on the basis of the Facial Action Coding System (FACS).

The Facial Action Coding System

The Facial Action Coding System (FACS) was based on a system originally developed by the Swedish anatomist Hjortsjö [50]. It was later adopted by Ekman and Friesen [32] who published it in 1978. Their work was focused around the creation of a universal measurement scheme that could distinguish among all visible facial behavior, so in that way, facial movement could be studied and analyzed both in research related and unrelated to emotion.

Since every facial movement is the result of muscular action, a system could be obtained on the basis of how each muscle contributes to the resulting change in visible facial appearance. In addition, FACS focuses on the movement and the muscular basis of appearance change and in that way it is capable of overcoming problems due to physiognomic differences, such as variations in size, shape, location of features and existence of wrinkles, bulges and other facial characteristics that gradually appear due to aging. After extensive experimentation and testing, the authors introduced a finite set of components of muscle movement, called Action Units (AUs), as well as a set of Action Descriptors (ADs). ADs differ from the AUs as they are unitary movements that may involve the actions of several muscle groups, the muscular basis of which hasn't been specified and specific behaviors haven't been distinguished as precisely as for the AUs. Examples of various detected AUs can be seen in figure 1.4. The criteria for observing and coding each Action Unit and Action Descriptor is described in the FAC manual, using which, human annotators can code almost every anatomically possible facial behavior by decomposing it into specific AUs and temporal segments that produced the expression.



Figure 1.4: Examples of detected Action Units (AUs) as they appear individually or in combinations during everyday social interactions. Source: [80].

The development of the FAC tool allows the objective and anatomical study of facial expressions in emotional contexts. Consequently, a collection of certain Action Units can be combined so as to provide information about which emotion is being displayed. For example, happiness is calculated from the combination of AU6 (cheek raiser) and AU12 (lip corner

puller), while sadness can be described as the combination of AU1 (inner brow raiser), AU4 (brow lowerer) and AU15 (lip corner depressor).

Facial Expression Dynamics

Facial actions are not produced instantaneously but rather evolve over a temporal interval. This very temporal evolution of the action units bears great importance in the interpretation of affective states as stated in [3]. Moreover, Ekman, Friesen, and Hager published a significant update to FACS [35], in which the temporal evolution of an expression was modelled with four temporal segments: neutral, onset, apex and offset. Neutral is the expressionless phase with no signs of muscular activity. Onset corresponds to the time period during which muscular contraction is present and the emotional state increases in intensity. Apex refers to the steady state during which emotional intensity remains stable and at its peak. Subsequently, offset corresponds to the period of time during which gradual muscular relaxation takes place and the emotional intensity degrades. The usual order of transitions among the aforementioned temporal phases is neutral-onset-apex-offset-neutral. The combined recognition of AUs and temporal segments accounts for the analysis of more complex emotional states and contributes to the distinction between natural and acted emotional behavior [102].

1.2.2 Body Features

Considering body movements as a modality for emotion perception and recognition is particularly relevant in situations during which affective information needs to be conveyed over long distances at which facial expressions and characteristics are no longer visible. Furthermore, a popular notion is that facial expressions are more easily controlled while body language often reveals our true emotions. For example, in stressful social environments, people tend to maintain a straight face or even smile so as to hide their nervousness and anxiety, as evidenced by quivering and sweaty hands, hand wringing and foot jiggling.

Evidence from Biology and Neuroscience

Investigations of the neuro-functional basis of observing bodily expressions have begun to show that they activate the same brain areas that were also associated with the perception of faces. Gelder [40] compared neutral and fearful expressions and found an increased activity for fearful bodily expressions in the amygdale (AMG) and in the form gyrus (FG). The area that showed body responsiveness in the FG was the same as the one identified in a separate study using a face localizer. In subsequent experiments, Gelder presented pictures of faces and bodies with blurred faces that depicted neutral, fearful and happy expressions and asked the participants to categorize the stimuli. The results state that the middle part of the FG which is typically associated with the perception of facial identity is more activated for bodies than for faces. Additionally, viewing whole body expressions evoked a larger set of brain areas in comparison with faces, including areas that were previously solely associated with the perception of faces such as the superior temporal sulcus (STS).

Movement Notation Systems

As a direct analogy to the aforementioned Facial Action Coding System (FACS), movement notation systems aim at providing a tool for systematic representation of movements which is capable of capturing both their structure as well as their expressiveness. Burgoon et al. [13], suggest that movement notation systems should be divided into functional and

structural approaches. Functional approaches describe the communicative function of a displayed movement using verbal labels. An example of such a system is the Ekman and Friesen formulation of kinesic behaviours into five categories [36], i.e. emblems, illustrators, affective displays, regulators and manipulators.

Structural approaches focus on the appearance of bodily movements and account for detailed notations of posture and movement dynamics, thus providing notation systems which are more suitable for computational movement analysis. An example of a structural notation system is the Laban Movement Analysis (LMA) [60]. More specifically, LMA constitutes a method and language for describing, visualizing, interpreting and documenting human movement. The Laban notation system has four major components: Body, Effort, Shape and Space. The first part of features in LMA, i.e. the Body component, captures the pose configuration. Such features include distances between joints, namely feet-hip, hands-shoulder, hands-head, centroid-pelvis, hands and feet. The second part of LMA features, i.e. the Effort component, captures body motion characteristics. These features include the joint velocity acceleration and jerk, as well as the angle, angular velocity and angular acceleration of pairwise limbs. The third part of features, namely the Shape component, captures body shape. These features correspond to the area that contains specific joints. Moreover, Shape consists of Shape Flow, Directional and Carving, features which according to Bartenieff [10], describe dynamic changes in movement form. Lastly, the fourth component of Space defines where in space a movement is happening and the directions of the body and body parts. Additionally, the space category notates choices which refer specifically to space, paying attention to the area that the body is moving within, the directions or points in space that the mover is identifying or using as well as geometrical observations of where the movement is being done, in terms of emphasis of directions, places in space and planar movement.

Body Language and Emotions

According to [78], the inner state of a person is expressed through elements such as gaze direction, position of hands, position of legs, style of sitting, walking, standing, body posture and movement. Therefore, it can be said that body language and human emotions are interdependent and the following examples aim to further establish this notion. Figure 1.5 illustrates various instances which verify the above concept.

Hands are probably the second richest source of affective information following the facial



Figure 1.5: Body language includes different types of nonverbal indicators such as facial expressions, body posture, gestures and eye movements. These are important markers of the emotional and cognitive inner state of a person and constitute significant sources of information relative to the task of human affect computing. Source: [78].

cue, as stated in [74], [83]. For example, sincerity and dishonesty can be potentially determined by the positions of a person’s hands. More specifically, if a person is being honest, they will most likely have their hands turned inside towards the interlocutor, while if they are being insincere, they will most likely hide their hands behind their back. In addition, practising open hand gestures during a conversation often gives the impression of a more reliable person.

The position of the head can convey valuable information of a person’s affective state. It is stated [84] that people tend to speak more when the listener shows approval and encourages them by nodding, while the pace of nodding is indicative of patience and lack thereof. Moreover, lifting of the chin, is a potential sign of superiority or arrogance and the exposure of the neck is often perceived as a signal of submission.

The torso by itself is not capable of conveying as much information as the aforementioned body parts but it can be indicative of one’s affective state in correlation with the rest of the body. For example, the angle of the torso with respect to the body can potentially reveal information about a person’s attitude in a conversation, as a frontal placement of the torso can be considered as a display of aggression, while a slight angle of the torso indicates self-confidence or lack of aggression. Furthermore, leaning forward combined with nodding or smiling is a typical sign of curiosity.

1.2.3 Contextual Features

In real life scenarios, when we observe an individual, we can probably estimate a lot of information about their emotional state despite the lack of additional specific knowledge about them. By analyzing a wider view, instead of focusing on the person in question, we can potentially collect affective information which can not be perceived provided the absence of context. This point is exemplified in figure 1.6, where both a face close-up as well as a wider view of a female athlete are shown. This concept holds true especially in situations during which the face is partially or completely occluded due to poor illumination conditions or obstacles, while the person may be found in unusual body configurations, hindering the extraction of information from the aforementioned sources. In summary, Dudzik et al. [31] highlight two primary sources of context which are used for interpreting emotional behavior, namely perceivable Perceivable Encoding Context and Perceiver Knowledge and Experience.



Figure 1.6: Example that illustrates the significance of contextual information for affective computing. For the image on the left, one would assume that the depicted woman is in a state of pain. However, with the introduction of context in the image on the right, it becomes evident that the woman is an athlete, she is celebrating and she is feeling ecstatic. Source: [8].

Perceivable Encoding Context

Perceivable Encoding Context includes factors that are perceived along the depicted emotional behavioral cues and that are experienced as having potentially influenced the encoding of an affective state. Wieser and Brosch [106] state that features which correspond to this context category revolve around demographic information related to the gender of the emotional behaviors as well as the situations and scenes in which they are embedded. As far as demographic features are concerned, some examples might include age, cultural background, gender and occupation. Additionally, examples of situational features are the location, depicted scene and illumination. In a hypothetical scenario [49], sadness often results from an irrevocable loss. However, for a toddler an ice cream that fell to the ground may be irrevocably lost, whereas an adult is aware of the fact that an ice cream is something that can easily be replenished. Such an example illustrates the effect of age as a situational context feature in emotion perception.

Additionally, the environment and scene depicted in an image or video can be closely related with the emotions of the people that are present. Barrett and Kensinger [9] report that the structural features of the face, when viewed in isolation, often prove to be insufficient for perceiving emotion. Furthermore, empirical findings suggest that the categorization of facial expressions is speeded up at the sight of congruent scenes [88], while both positive and negative contexts result in significantly different ratings of faces compared with those presented in neutral contexts [73]. For example, an image of a funeral that is located at a cemetery, suggests a strong correlation between the above oppressive setting and the generally negative and sad feelings shared among the depicted people.

Perceiver Knowledge and Experience

Weiser and Brosch [106] highlight empirical findings which suggest that the perceiver's pre-existing knowledge and experiences have a significant impact on the way they decode and recognize affective states. This knowledge mainly includes racial stereotypes, affective associations, social norms, cultural values as well as their mental state, needs, goals and expertise. In that way, the incorporation of Perceiver Knowledge and Experience may lead to the reconstruction and filtering of the affective information during the decoding process of a behavioral signal. Masuda et al. [68] describe an experiment based on a study about the effect of cultural background in emotion perception involved Asian and US Americans to whom cartoons and photos of a group of people were shown and they were asked to decode the central's person emotion. According to Barrett et al. [8], the study showed that the Asian participants made more strategic use of the information from the faces surrounding the target while Western participants seemed to rely mostly on the information deriving from the actual target. This finding was based on the fact that participants from a Western culture conceptualize emotions as located within the individual while participants of Asian culture conceptualize emotions as a reflection of the interpersonal relationships between people.

Moreover, the detection of facial actions might carry affective information; however, the discrete emotional concepts of affective states receive their meaning through context. In order to illustrate that, Barrett et al. [8] mention the case of a semantic dementia (SD) patient that participated in an emotion perception experiment. Semantic dementia can be described as a progressive neuro-degenerative disorder that is characterized by loss of semantic memory in both the verbal and non-verbal domains, causing patients to lose the ability to match words or images to their meanings and hindering spontaneous speech creation. The patient was asked to categorically sort 120 posed, stereotyped scowls "angry", pouting "sad", smiling "happy", nose-wrinkled "disgusted", startled "fearful" looking faces and neutral faces (20 each). The patient did not manage to distinguish the faces among the five emotional

categories (plus neutral) but managed to sort them according to the depicted valence levels, thus forming a positive (happy), neutral and negative stack (angry, sad, disgusted, fearful). Results suggested that as emotion words become more remote from the perception task, people face increasing difficulty in effectively recognizing emotion even in posed environments. Thus, it can be said that when seeing emotion in a face, the task performed by the perceiver can be compared with the reading of a word on a page.

1.3 Thesis Outline

In this final section of the introductory chapter, we briefly describe the structure as well as the topics which will be discussed throughout the remaining chapters of the current thesis.

To begin with, Chapter 2 lays the theoretical foundations associated with subjects such as Machine Learning, Representation Learning and Deep Learning. More specifically, we present the structure and features of all major, practical and modern deep neural network architectures. Furthermore, the necessary mathematical background is going to be established with the aim of obtaining a clearer insight of the functionalities of all the aforementioned modules.

Moreover, Chapter 3 will serve as an introduction to the task of Visual Emotion Recognition. Firstly, we present a summary of the most popular publicly available databases which have been used in research relative to visual affective computing. In addition, we analyze the most notable pieces of early related work, laying emphasis on techniques that utilize “hand-crafted” features for the encoding of visual affective information. Subsequently, a transition to more modern practices will be made with the emphasis being shifted towards Deep Learning where we will be analyzing topics associated with data pre-processing and deep feature extraction.

Moving on, Chapter 4 constitutes our first case study where we explicitly tackle the problem of image-based Visual Emotion Recognition in Context, using the EMOTIC dataset. We begin by presenting the most notable applied methodologies from all publicly available related work. Consequently, we propose possible extensions over the baseline models and lastly, we present our experimental results.

Chapter 5 extends the concept of Visual Emotion Recognition in Context as we now focus on video sequences instead of static image frames. To this end, we utilize the newly assembled Body Language Dataset (BoLD) as a counterpart of the EMOTIC dataset in dynamic temporal settings. For our network implementations we adopt the widely used Temporal Segment Networks (TSN) framework, providing an adaptation of our previous “static” model, capable of processing video sequences.

Finally, Chapter 6 completes our thesis with the inclusion of conclusive remarks and potential directions for future work.

Chapter 2

Deep Learning

The primary tools which will be utilized towards tackling the automatic affect recognition task belong in the fast-growing field of Deep Learning. In the current chapter, we will try to establish the theoretical foundations regarding Deep Learning, as well as accurately describe the basic architectures which will later constitute the building blocks of our own computational models. Therefore, with this specific goal in mind, we will begin by presenting the broader context of applied computer algorithms from which Deep Learning originated, and subsequently, we will briefly describe the evolution of this particular field from a historical perspective. Lastly, we will analyze the structure and features of all the major practical and modern deep networks. For guidance throughout this process, we consult the works of Goodfellow et al. [42] and Bishop [11], as our primary references.

2.1 Underlying Concepts

Deep Learning is closely related and shares several concepts with other fields such as Artificial Intelligence (AI), Machine Learning and Representation Learning. In fact, Deep Learning is considered to be a sub-field of Representation Learning, which in turn is a sub-field of Machine Learning, with the latter lying within the broad spectrum of Artificial Intelligence.

2.1.1 Artificial Intelligence

Artificial intelligence is a thriving scientific field that predates that of Deep Learning and revolves around techniques and algorithms which enable computers to mimic human behavior and solve problems which can be described by a set of formal mathematical rules. In earlier AI projects, programmers tried to tackle problems by hard-coding knowledge into machines in the form of logical inference rules using formal languages. However, those attempts failed due to the fact that it is extremely difficult to come up with a sufficiently complex and diverse set of formal rules to describe the ever growing complexity of the real world. Consequently, AI related techniques struggle to cope with problems for which no formal mathematical description can be given. The difficulties faced by AI systems, gave rise to a set of algorithms and methods which enabled computers to extract relative information directly from raw data and based on them, make predictions and choices in real-life scenarios. The subset of AI relative to this concept is called Machine Learning.

2.1.2 Machine Learning

Machine Learning (ML) algorithms are heavily dependent on the form or representation of the data with which they are provided. While AI focuses on mimicking human behavior,

ML focuses on mimicking how humans learn. In that way, depending on the nature of the task at hand, ML algorithms require relevant and robust data representations in order to function properly. In classic ML, those sought-after data representations come in the form of hand-crafted features which have been specifically designed with human supervision so as to contextually match the requirements of the problem. However, for many tasks, it is really difficult to either manually design accurate descriptions of potentially important pieces of data or even predict which pieces of the available information are worth encoding in the first place. Issues like these lead to severe degradation in performance or inhibit the use of ML algorithms in general.

2.1.3 Representation Learning

A potential solution to the aforementioned problem suggests extending the use of ML algorithms in learning not only the mapping from given data representations to the desired output but also the representations themselves. This technique is called Representation Learning (RL) and constitutes a sub-field of ML. RL enables AI systems to rapidly adapt to new tasks and provides them with effective data representations, that achieve performance which is comparable or even superior to that of hand-crafted features, while minimizing human intervention. The primary challenge involving RL in artificial intelligence applications is the disentanglement of the factors of variation found within the available data, namely the recognition and extraction of the factors that give meaning to the observed data.

Deep learning aims at solving the core problem of representation learning by not only mimicking the ways humans learn but also imitating the primary functionalities of the human brain. Drawing inspiration from the processes relative to biological learning, deep learning models share a property of self-organization analogous to the proposed learning dynamics of the human brain. In summary, deep learning comprises a set of ML methodologies that represent the world in a hierarchical fashion, according to which higher-level and more abstract representations of raw data are obtained on the basis of simpler and less abstract ones. Furthermore, in the context of modern ML, deep learning transcends its original neuro-scientific perspective and abides to a more general principal of learning multiple layers of composition. The basic modules that comprise deep learning models are called Artificial Neural Networks (ANNs). As a subsequent step in our introductory view in deep learning, we will briefly describe some of the most notable predecessors of modern ANNs.

2.2 Predecessors of Modern ANNs

In the current section we will dive into the primitive era (1943-1980) of neural networks and highlight the modules which have constituted the cornerstones of modern ANN architectures. More specifically, our brief analysis includes the McCulloch-Pitts artificial neuron (1943), Rosenblatt's perceptron (1958) and Fukushima's Neocognitron (1980), in that specific order.

2.2.1 The Artificial Neuron

The earliest realization of an artificial neural was introduced by McCulloch and Pitts in 1943 [69]. Their neural model is also known as a linear threshold gate. It is a computational model that involves a set of inputs $\{x_i\}_{i=1}^N$, a set of corresponding weights $\{w_i\}_{i=1}^N$, normalized in the range $[0,1]$ or $[-1,1]$ and an output y . The output of the linear threshold gate is binary,

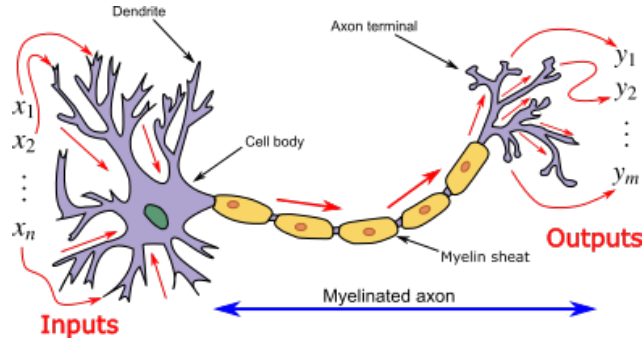


Figure 2.1: Illustration of a biological neuron and its myelinated axon are depicted, with signals flowing from inputs at dendrites to outputs at axon terminals. Source: https://en.wikipedia.org/wiki/Artificial_neural_network.

namely it can be described as follows:

$$z = \sum_{i=1}^N w_i x_i \quad (2.1)$$

$$y = f(z) = \begin{cases} 1 & \text{if } z \geq \tau \\ 0 & \text{if } z < \tau \end{cases} \quad (2.2)$$

where τ is a threshold constant. In that way, the function $f(\cdot)$ operates as a thresholded step function. An illustration of a biological neuron as well as its artificial counterpart can be seen in figures 2.1 and 2.2 respectively. Although being extremely simplistic, the McCulloch-Pitts artificial neuron model offers substantial computing potential. At this point, we ought to mention that the weights and threshold values are fixed. So in order for the model to respond to the input data in the desired way, the parameters of the model required manual fine-tuning which was carried out by a human operator. Thus, emerged the need for a neural model with more flexible computational features.

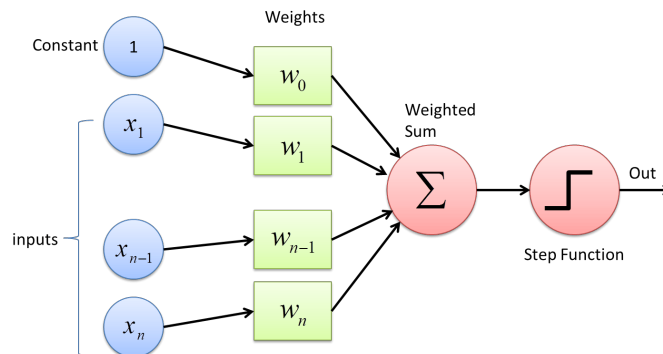


Figure 2.2: Illustration of the McCulloch-Pitts artificial neuron model where a weighted sum of the input signals is passed through a unit step function, resulting in the corresponding output. Source: [95].

2.2.2 The Perceptron

The perceptron was originally introduced by Rosenblatt in 1953 [89], as the first neural network model for binary classification which could learn the appropriate weights, given a set of data samples that belonged to each of the two categories. Consequently, it constituted the first algorithm of supervised learning for binary classifiers. The perceptron inherited its

structure from the McCulloch-Pitts artificial neuron and incorporated a training algorithm so as to achieve the weight adaptation.

Given m -dimensional input feature vectors $\mathbf{x} = [x_1, x_2, \dots, x_m]^\top$, corresponding weights $\mathbf{w} = [w_1, w_2, \dots, w_m]^\top$ and a bias b , the perceptron defines the m -dimensional separating hyperplane $\sum_{i=1}^m w_i x_i + b = 0$, according to which it classifies all potential input samples. In order to describe the perceptron training algorithm, we will expand the input and weight vectors in $(m + 1)$ dimensions, by incorporating the bias b into the weight vector \mathbf{w} , namely $\tilde{\mathbf{x}} = [1, x_1, x_2, \dots, x_m]^\top$ and $\tilde{\mathbf{w}} = [b, w_1, w_2, \dots, w_m]^\top$. Therefore, the separating hyperplane can be simply written as $\tilde{\mathbf{w}}^\top \tilde{\mathbf{x}} = 0$ and the perceptron output will be equal to $y = f(\tilde{\mathbf{w}}^\top \tilde{\mathbf{x}})$, where $f(\cdot)$ can be a Heavyside step function. As the separating surface is a hyperplane in m -dimensional space, the perceptron belongs to the broader family of linear models, and is capable of solving classification problems that involve data samples which are linearly separable in m dimensions.

At this point we are ready to present the perceptron training algorithm. We assume $D = \{(\mathbf{x}_n, d_n)\}_{n=1}^N$ to be the set of data samples, belonging in either of two possible classes, ω_1, ω_2 , together with their corresponding desired outputs. The algorithm is ran in iterations until convergence is achieved (if possible). During each iteration all data samples are examined. In order to highlight the time dependence of the parameters, we use the notation $\tilde{\mathbf{w}}(t)$ denoting the weight vector at time step t . The main steps of the algorithm are the following:

1. Initialization ($t = 0$): Set $\tilde{\mathbf{w}}(0) = \mathbf{0}$ and define the learning rate parameter $\eta \in (0, 1]$. Proceed with the computation of the following steps.
2. For every example $n = 1, 2, \dots, N$ in the dataset, do:
 - (a) Output Calculation: During time step t , compute the actual output $y_n = f(\tilde{\mathbf{w}}(t)^\top \tilde{\mathbf{x}}_n)$.
 - (b) Weight Adaptation: During time step t , update the weight vector according to the rule:

$$\tilde{\mathbf{w}}(t + 1) = \tilde{\mathbf{w}}(t) + \eta(d_n - y_n)\tilde{\mathbf{x}}_n$$

$$d_n = \begin{cases} 1 & \text{if } \mathbf{x}_n \in \omega_1 \\ 0 & \text{if } \mathbf{x}_n \in \omega_2 \end{cases}$$

- (c) Time Increment: $t = t + 1$.

3. Check if convergence has been reached and all data samples are correctly classified, else return at step (2).

Although the perceptron looked promising, it quickly became apparent that its potential to classify patterns was very limited. The most famous example of a task not capable of being solved by a perceptron is the XOR problem. That resulted in a significant drop in the popularity of artificial neural networks.

2.2.3 The Neocognitron

The next big breakthrough in ANNs came along the development of the Neocognitron by Fukushima in 1980 [39], which is a powerful hierarchical multilayered neural network capable of image processing and visual pattern recognition. The Neocognitron, as a neural network architecture, introduced two key features. Firstly, its structure and functionality draws inspiration from the mammalian visual nervous system, and secondly, it supports unsupervised learning, namely the process of its self-organization only requires the repeated

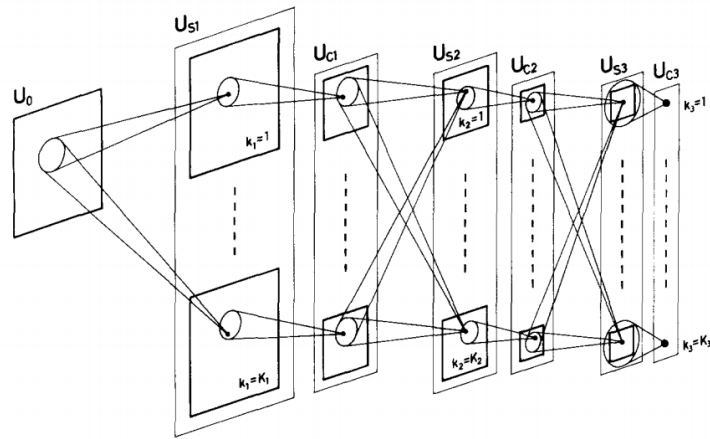


Figure 2.3: Schematic diagram illustrating the interconnections between layers in the neocognitron. Source: [39].

presence of input stimulus patterns at the input layer of the network, without the need of further human intervention.

The Neocognitron consists of a number of modular structures connected in a cascaded architecture, while the input layer is located at the lowest stage and is comprised of photo-receptive cells, forming a two-dimensional grid. Each modular structure consists of an S -layer, made up of S -cells, followed by a C -layer, made up of C -cells. S -cells correspond to simple cells of the primary visual cortex and their main purpose is centered around feature extraction. An illustration of the Neocognitron inter-layer connectivities is shown in figure 2.3. In addition, the weights of their input connections are variable and can be modified during the self-organization process. Each S -cell responds to specific features of the input stimulus patterns relative to its position in the S -layer and the size of its receptive field. C -cells on the other hand, correspond to the more complex cells of the visual cortex. Moreover, the input connections of C -cells which derive from a group of S -cells of the previous layer are constant and unmodifiable. Each C -cell serves as a receptor for the feature responses of a specific group of S -cells of the previous layer while achieving shift invariance. The cells in each layer are divided into subgroups in the form of smaller two-dimensional grids, called cell-planes according to their position in the cell-layer and the features to which they respond, while cells within one cell-plane share the same input connections. An illustration of cell interconnections within the same cell-plane can be seen in figure 2.4. Furthermore, a C -cell activates, provided that at least one of the S -cells in the corresponding S -plane emits a response. Even if a specific S -cell fails to respond due to a slight deformation or difference in the position of the presented input stimulus pattern, a nearby S -cell is highly likely to respond, thus resulting in the desired activation of the local C -cell. This functionality of the C -cells is analogous to a blurring operation. In general, lower layers extract local, low-level features such as edges, corners or contours while higher layers extract more abstract, global features related to the recognition of input patterns.

The exploration of the numerical expressions regarding the computation of the outputs of the two types of cells lies outside the scope of the current analysis. However, we ought to dive deeper into the self-organizing properties of the Neocognitron and evaluate its potential as a model for unsupervised learning. By considering the cells within all cell-planes of a cell-layer, a structure called cell-column or hypercolumn emerges. Cells within a hypercolumn, come from different planes of a single layer but their receptive fields are located at roughly the same position, namely each one is stacked on top of the others in the column. From

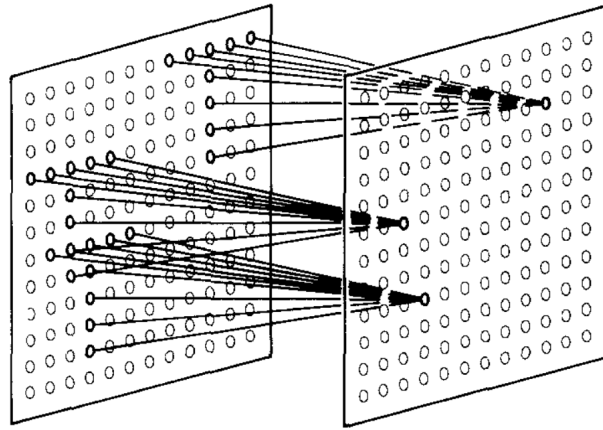


Figure 2.4: Illustration showing the input interconnections to the cells within a single cell-plane. Source: [39].

each S -column, a representative is selected as the cell with the largest response to the input stimulus patterns, so it is possible for more than one representatives per cell-plane to appear. If that is the case, only the one which yields the largest response among them is selected as the representative of that specific S -plane. The input connections of the representative cells as well the cells that belong in the same planes as the representatives are reinforced by an amount proportional to the intensity of their initial response. The S -planes which contain no representatives do not have their input connections modified.

The above self-organization process ensures that each S -plane becomes selectively sensitive to a specific type of features while preventing the formation of redundant connections. That is the case of two or more S -planes being used for the detection of the same feature. We note that the Neocognitron later became the basis for the modern convolutional network.

2.3 Modern Deep Neural Network Practices

In the current section, we will introduce the primary ANN modules which will later constitute the basis of our own models, towards the task of automatic visual affect recognition, as these architectures provide a powerful framework for supervised learning. To begin with, we will present the deep feed-forward networks known as Multilayer Perceptrons (MLPs) and we will analyze the popular Backpropagation (BP) algorithm as the primary tool in training such networks. Subsequently, we will focus on Convolutional Neural Networks (CNNs) for image processing and visual pattern recognition. Lastly, Recurrent Neural Networks (RNNs) and their variants will be introduced as being the main tool in temporal sequence processing.

2.3.1 The Multilayer Perceptron

A multilayer perceptron (MLP) is a type of feed-forward neural network as information travels in a forward direction through the network, without the existence of feedback loops that feed outputs back into itself. It consists mainly of three types of layers: the input layer, the hidden layers and the output layer. The input layer simply operates as a receptor for raw data and its size matches the dimensionality of the input features. In a similar fashion, the output layer emits feature responses and its size depends on the nature of the task. In a regression problem, the number of neurons in the output layer will equal the number of different quantities that are to be regressed, while in a multiclass classification problem, the

size of the output layer will be equal to the number of different possible classes in the given dataset. The hidden layers extract increasingly abstract features from the input features and are called “hidden” as their outputs are not available for observation. The only “visible” layers of the MLP are its input and output layers. The MLP is characterized primarily by the number of its hidden layers and the number of neurons or hidden units per layer, with each layer potentially having a different number of hidden units. In our implementations we only consider MLPs which are fully connected (FC), namely every neuron on a specific layer is connected with the neurons of the preceding and following layers. Fully connected layers appear in the vast majority of MLP applications.

Forward Pass

Let’s assume a fully connected MLP that takes an input vector $\mathbf{x} \in \mathbb{R}^{N_0}$, outputs a vector $\mathbf{y} \in \mathbb{R}^{N_{L+1}}$ and has L hidden layers where the l th hidden layer consists of N_l hidden units. By incorporating the biases in the corresponding weight vectors and expanding the feature vectors appropriately, as described in section 2.2.2, the output $h_j^{(l)}$ of the j th hidden unit of the l th hidden layer can be computed as follows:

$$h_j^{(l)} = \psi\left(\sum_{i=0}^{N_{l-1}} w_{ij}^{(l)} h_i^{(l-1)}\right) \quad (2.3)$$

where $w_{0j}^{(l)} = b_j^{(l)}$, namely the bias of the j th unit of the l th layer, $w_{ij}^{(l)}$ denotes the weight that connects the i th neuron of the $(l-1)$ th layer with the j th neuron of the l th layer, $\mathbf{h}^{(l-1)} = [1, h_1^{(l-1)}, \dots, h_{N_{l-1}}^{(l-1)}]^\top$ denotes the expanded hidden feature vector of the $(l-1)$ th layer and $\phi(\cdot)$ is a non-linear hidden unit activation function. In matrix form, the forward pass can be described by the following equations:

$$\begin{aligned} \mathbf{h}^{(0)} &= \mathbf{x} \\ \mathbf{h}^{(l)} &= \psi(\mathbf{W}_l^\top \mathbf{h}^{(l-1)}), \text{ if } 1 \leq l \leq L \\ \mathbf{y} &= \phi(\mathbf{W}_{L+1}^\top \mathbf{h}^{(L)}) \end{aligned} \quad (2.4)$$

where \mathbf{W}_l is the weight matrix of size $(N_{l-1} + 1) \times (N_l + 1)$, with elements:

$$\mathbf{W}_l = \begin{bmatrix} b_0^{(l)} & b_1^{(l)} & \cdots & b_{N_l}^{(l)} \\ w_{10}^{(l)} & w_{11}^{(l)} & \cdots & w_{1N_l}^{(l)} \\ \vdots & \vdots & \ddots & \vdots \\ w_{N_{l-1}0}^{(l)} & w_{N_{l-1}1}^{(l)} & \cdots & w_{N_{l-1}N_l}^{(l)} \end{bmatrix}$$

An illustration of a single hidden-layer MLP is provided in figure 2.5. We need to note that in equation 2.4, the non-linear hidden unit activation function $\psi(\cdot)$ and the output unit activation function $\phi(\cdot)$ are applied element-wise on the corresponding vectors. The vectors that are produced at each layer before the application of the aforementioned functions, are referred to as activations.

Activation Functions

In equations 2.3 and 2.4, we introduced the hidden unit non-linear activation function $\psi(\cdot)$ and the output unit activation function $\phi(\cdot)$ and we now ought to establish their purpose. The non-linear activation function is used in order to introduce non-linearity in an

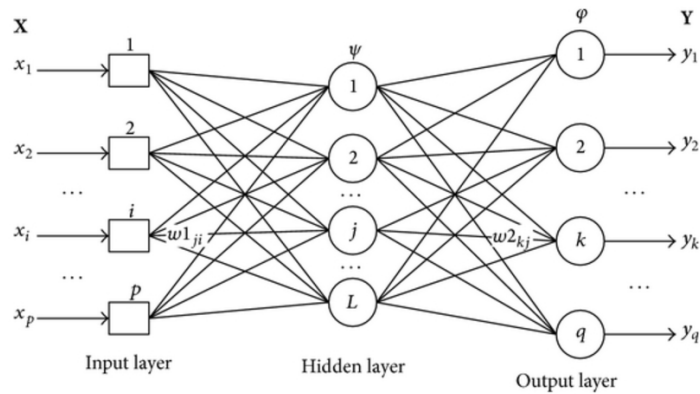


Figure 2.5: Schematic diagram of a multi-layer perceptron (MLP) with one hidden layer, p input units, L hidden units, q output units, with $\psi(\cdot)$ as the hidden unit activation functions and $\phi(\cdot)$ as the output unit activation functions.

otherwise linear model. In turn, this allows the network to model a response variable which depends non-linearly on its input explanatory features, namely it can not be reproduced by a simple linear combination of its inputs. In addition, if the activation function is non-linear, bounded, continuous and monotonically increasing, it has been proven that a MLP with one hidden layer constitutes a universal function approximator, namely it can approximate any target function with arbitrary precision. This property is also known as the Universal Approximation Theorem which was first proven for the sigmoidal function by Cybenko [24] and later generalized for other activation functions by Hornik [52]. In table 2.1 we present some of the most common activation functions which we will regularly come across in deep learning modules, along with their main features.

Name	Equation	Derivative	Range
Identity	$f(x) = x$	$f'(x) = 1$	$(-\infty, \infty)$
Sigmoid	$f(x) = \sigma(x) = \frac{1}{1+e^{-x}}$	$f'(x) = f(x)(1 - f(x))$	$(0, 1)$
Hyperbolic Tangent	$f(x) = \tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$	$f'(x) = 1 - f^2(x)$	$(-1, 1)$
Rectified Linear Unit (ReLU)	$f(x) = \begin{cases} x & \text{if } x > 0 \\ 0 & \text{if } x \leq 0 \end{cases} = \max\{0, x\}$	$f'(x) = \begin{cases} 1 & \text{if } x > 0 \\ 0 & \text{if } x \leq 0 \end{cases}$	$[0, \infty)$
Leaky ReLU	$f(x) = \begin{cases} x & \text{if } x > 0 \\ 0.01x & \text{if } x \leq 0 \end{cases}$	$f'(x) = \begin{cases} 1 & \text{if } x > 0 \\ 0.01 & \text{if } x \leq 0 \end{cases}$	$(-\infty, \infty)$
Softmax	$f_i(\mathbf{x}) = \frac{e^{x_i}}{\sum_{j=1}^J e^{x_j}}$, for $j = 1, 2, \dots, J$	$\frac{\partial f_i(\mathbf{x})}{\partial x_j} = f_j(\mathbf{x})(\delta_{ij} - f_i(\mathbf{x}))$	$(0, 1)$

Table 2.1: Common activation functions

It is worth noting that without non-linear activation functions, no matter how many layers an MLP had, it would be equivalent to a single-layer perceptron, as the aggregation of all the layers would result in another linear function. Furthermore, another important property of the above activation functions is that they are continuously differentiable, thus enabling gradient-based learning methods. The most widely used gradient-based algorithm

utilized in training ANNs for supervised learning, is called Backpropagation. However, in order to use gradient-based supervised learning, we first need to choose an appropriate loss function depending on the type of task that our model is designed to solve. Also the choice of output activation function $\psi(\cdot)$ depends on the form of the desired output data which in turn depends on the nature of the task that the network must perform. In that way, the output unit activation function is not necessarily non-linear. In general, the purpose of the output unit activation function is to provide an extra transformation to the hidden features so as to bring them into some suitable output format. As we will later see, in regression problems, the output unit activation function is simply the identity function.

2.3.2 Network Training and Loss Functions

In the current section, we will derive common loss functions used in supervised deep network training from a general probabilistic point of a view. For convenience, we will consider a feed-forward neural network model as a non-linear function that receives some input vector \mathbf{x} , outputs a vector \mathbf{y} and is characterized by a set of weight parameters \mathbf{w} , namely $\mathbf{y}(\mathbf{x}, \mathbf{w}) = f(g(\mathbf{x}, \mathbf{w}))$, where $f(\cdot)$ is the output activation function and $g(\mathbf{x}, \mathbf{w})$ is the activation of the output layer. The main two types of tasks which neural network models are designed to solve and which we will consider with regard to automatic affect recognition, are Classification and Regression.

Regression

In this type of task, the computational model is asked to predict a numerical value, given some input vector $\mathbf{x} \in \mathbb{R}^M$. In that way, the model effectively defines a mapping $f \circ g: \mathbb{R}^M \rightarrow \mathbb{R}$.

We now consider a set of training examples $D = \{(\mathbf{x}_n, t_n)\}_{n=1}^N$, together with their corresponding desired output values. We can assume that target variables t follow a Gaussian distribution such that their mean is dependent on the output of the network, namely $p(t|\mathbf{x}, \mathbf{w}) = \mathcal{N}(t|y(\mathbf{x}, \mathbf{w}), \beta^{-1})$, where β denotes the inverse variance of the Gaussian noise. If $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}$ and $\mathbf{t} = \{t_1, t_2, \dots, t_N\}$, under the assumption that the observations \mathbf{X} are independent and identically distributed, then the corresponding likelihood function is:

$$p(\mathbf{t}|\mathbf{X}, \mathbf{w}, \beta) = \prod_{n=1}^N \mathcal{N}(t_n|y(\mathbf{x}_n, \mathbf{w}), \beta^{-1}) \quad (2.5)$$

By taking the natural logarithm, we have:

$$\ln p(\mathbf{t}|\mathbf{X}, \mathbf{w}) = \frac{N}{2} \ln \beta - \frac{N}{2} \ln 2\pi - \frac{\beta}{2} \sum_{n=1}^N (t_n - y(\mathbf{x}_n, \mathbf{w}))^2 \quad (2.6)$$

Maximizing the log-likelihood with respect to \mathbf{w} is equivalent to minimizing the error term:

$$E(\mathbf{w}) = \sum_{n=1}^N (t_n - y(\mathbf{x}_n, \mathbf{w}))^2 \quad (2.7)$$

The minimization of the error term 2.7 constitutes a non-convex optimization problem due to the introduction of non-linearity by the non-linear hidden layer activation functions. Thus the optimal solution \mathbf{w}^* can be approximated using gradient based methods. Having found \mathbf{w}^* , we can substitute it in 2.6 and by optimizing with respect to β we obtain:

$$\frac{1}{\beta^*} = \frac{1}{N} \sum_{n=1}^N (t_n - y(\mathbf{x}_n, \mathbf{w}^*))^2 \quad (2.8)$$

In the case of regression, the output activation function $f(\cdot)$ is chosen to be the identity function. The error term 2.7 is a quadratic cost function which is also known as the mean squared error (MSE), if it is further divided by the number of samples N . If we expand the problem of regression on K target variables which are independent conditional on \mathbf{x} and \mathbf{w} , having a diagonal covariance matrix and a shared variance parameter β , then the corresponding conditional distribution of the target vectors would be:

$$p(\mathbf{t}|\mathbf{x}, \mathbf{w}) = \mathcal{N}(\mathbf{t}|\mathbf{y}(\mathbf{x}, \mathbf{w}), \beta^{-1}\mathbb{I}_K) \quad (2.9)$$

Following the same procedure, under the assumption of identically distributed observations $\mathbf{X} \in \mathbb{R}^{N \times M}$ and a target matrix $\mathbf{T} \in \mathbb{R}^{N \times K}$ containing the corresponding N target vectors of size K , the log-likelihood function would then be:

$$\ln p(\mathbf{T}|\mathbf{X}, \mathbf{w}) = \frac{NK}{2} \ln \beta - \frac{NK}{2} \ln 2\pi - \frac{\beta}{2} \sum_{n=1}^N \|\mathbf{t}_n - \mathbf{y}(\mathbf{x}_n, \mathbf{w})\|_2^2 \quad (2.10)$$

The corresponding error term deriving from the maximization of 2.10 with respect to \mathbf{w} , would then become:

$$E(\mathbf{w}) = \sum_{n=1}^N \|\mathbf{t}_n - \mathbf{y}(\mathbf{x}_n, \mathbf{w})\|_2^2 \quad (2.11)$$

Maximizing 2.10 with respect to β , after having found the optimal weights \mathbf{w}^* , yields:

$$\frac{1}{\beta^*} = \frac{1}{NK} \sum_{n=1}^N \|\mathbf{t}_n - \mathbf{y}(\mathbf{x}_n, \mathbf{w}^*)\|_2^2 \quad (2.12)$$

MSE is the most commonly used loss function used in regression problems. A widely used variant of the MSE loss is called Smooth- L^1 loss which was introduced by Girshick [41] and is less sensitive to outliers compared to MSE. Outliers are considered to be data samples that differ significantly from the rest of the observations, being a usual indication of measurement errors or a heavy-tailed distribution. Smooth- L^1 loss is defined as:

$$SL^1(\mathbf{x}, \mathbf{y}) = \frac{1}{N} \sum_{n=1}^N z(x_n, y_n) \quad (2.13)$$

$$z(x_n, y_n) = \begin{cases} 0.5(x_n - y_n)^2 & \text{if } |x_n - y_n| < 1 \\ |x_n - y_n| - 0.5 & \text{otherwise} \end{cases}$$

Classification

During this type of task, a computational model is asked to determine which of C possible categories some input vector $\mathbf{x} \in \mathbb{R}^M$ belongs to. In that way, the model effectively defines a mapping function $f \circ g: \mathbb{R}^M \rightarrow \{1, 2, \dots, C\}$.

To begin with, we consider the case of binary classification, with two possible classes C_1, C_2 , given a set of training examples $D = \{(\mathbf{x}_n, t_n)\}_{n=1}^N$, where $t_n = 0$ if $\mathbf{x}_n \in C_1$ and $t_n = 1$ if $\mathbf{x}_n \in C_2$. In this case, the model requires a single output that expresses the corresponding conditional probability of the input vector \mathbf{x} belonging to either of the two classes, namely $y(\mathbf{x}, \mathbf{w}) = P(C_1|\mathbf{x}) = 1 - P(C_2|\mathbf{x})$. Therefore, a proper output activation function would be the sigmoid ($f(\cdot) \equiv \sigma(\cdot)$):

$$y(\mathbf{x}, \mathbf{w}) = \frac{1}{1 + \exp(-g(\mathbf{x}, \mathbf{w}))} \quad (2.14)$$

In that way, a target variable t follows a Bernoulli distribution of the following form:

$$p(t|\mathbf{x}, \mathbf{w}) = y(\mathbf{x}, \mathbf{w})^t (1 - y(\mathbf{x}, \mathbf{w}))^{1-t} \quad (2.15)$$

If $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}$ and $\mathbf{t} = \{t_1, t_2, \dots, t_N\}$, under the assumption that the observations \mathbf{X} are independent and identically distributed, then the corresponding likelihood function is:

$$p(\mathbf{t}|\mathbf{X}, \mathbf{w}) = \prod_{n=1}^N y_n^{t_n} (1 - y_n)^{1-t_n} \quad (2.16)$$

By taking the natural logarithm of 2.16, we have:

$$\ln p(\mathbf{t}|\mathbf{X}, \mathbf{w}) = \sum_{n=1}^N [t_n \ln y_n + (1 - t_n) \ln (1 - y_n)] \quad (2.17)$$

Maximizing 2.17 is equivalent to minimizing the error term:

$$E(\mathbf{w}) = - \sum_{n=1}^N [t_n \ln y_n + (1 - t_n) \ln (1 - y_n)] \quad (2.18)$$

Similarly to the case of regression, the minimization of the error term 2.18 comprises a non-convex optimization problem and the optimal solution \mathbf{w}^* can be approximated using gradient based methods. The above error term is widely known as binary cross-entropy (BCE) loss.

Extending the problem to the classification of K binary variables, we can use a network with K separate outputs, each of which will have a sigmoid ($f(\cdot) \equiv \sigma(\cdot)$) as the output unit activation function, forming a probability vector $\mathbf{y}(\mathbf{x}, \mathbf{w}) \in \mathbb{R}^K$ with elements:

$$y_j(\mathbf{x}, \mathbf{w}) = \frac{1}{1 + \exp(-g_j(\mathbf{x}, \mathbf{w}))} \quad (2.19)$$

This task is known as single-class multi-label classification and in this case the target variables t become target vectors $\mathbf{t} \in \mathbb{R}^K$, where $t_k \in \{0, 1\}$. Assuming that the class labels are independent then the conditional distribution of a target vector will be:

$$p(\mathbf{t}|\mathbf{x}, \mathbf{w}) = \prod_{k=1}^K y_k^{t_k} (1 - y_k)^{1-t_k} \quad (2.20)$$

Again, we assume independent and identically distributed observations $\mathbf{X} \in \mathbb{R}^{N \times M}$ and a target matrix $\mathbf{T} \in \mathbb{R}^{N \times K}$ containing the corresponding N target vectors of size K . The likelihood function would then be:

$$p(\mathbf{T}|\mathbf{X}, \mathbf{w}) = \prod_{n=1}^N \prod_{k=1}^K y_{nk}^{t_{nk}} (1 - y_{nk})^{1-t_{nk}} \quad (2.21)$$

by taking the natural logarithm 2.21, we obtain the following expression for the corresponding cross entropy loss function:

$$E(\mathbf{w}) = - \sum_{n=1}^N \sum_{k=1}^K [t_{nk} \ln y_{nk} + (1 - t_{nk}) \ln (1 - y_{nk})] \quad (2.22)$$

where t_{nk} , y_{nk} are the k th elements of the label and output vectors respectively, corresponding to the n th training example. The error term 2.22 can be considered as the sum of BCE losses regarding each one of the separate K binary variables, across all of the N training examples.

Lastly we consider the case of multi-class classification, where each input vector \mathbf{x} is classified into one of K possible mutually exclusive classes. In this case, the target vectors $\mathbf{t} \in \mathbb{R}^K$ is one-hot encoded, namely $t_k \in \{0, 1\}$ and $\sum_{k=1}^K t_k = 1$. The corresponding model will have K separate outputs, where the k th output will be equal to the conditional probability of \mathbf{x} belonging to class k , namely $y_k = P(t_k = 1 | \mathbf{x})$. As each input vector is assigned into only one category and the outputs correspond to probabilities, the conditions $0 \leq y_k \leq 1$ and $\sum_{k=1}^K y_k = 1$ must also hold. Therefore, in this case, the softmax would be an appropriate output activation function satisfying both of the aforementioned constraints:

$$y_j(\mathbf{x}, \mathbf{w}) = \frac{\exp(g_j(\mathbf{x}, \mathbf{w}))}{\sum_{k=1}^K \exp(g_k(\mathbf{x}, \mathbf{w}))} \quad (2.23)$$

The conditional distribution of a target vector will be:

$$p(\mathbf{t} | \mathbf{x}, \mathbf{w}) = \prod_{k=1}^K y_k^{t_k} \quad (2.24)$$

Again, we assume independent and identically distributed observations $\mathbf{X} \in \mathbb{R}^{N \times M}$ and a target matrix $\mathbf{T} \in \mathbb{R}^{N \times K}$ containing the corresponding N one-hot encoded target vectors of size K . The likelihood function would then be:

$$p(\mathbf{T} | \mathbf{X}, \mathbf{w}) = \prod_{n=1}^N \prod_{k=1}^K y_{nk}^{t_{nk}} \quad (2.25)$$

By taking the natural logarithm of 2.25, we obtain the following expression for the multi-class cross entropy loss function:

$$E(\mathbf{w}) = - \sum_{n=1}^N \sum_{k=1}^K t_{nk} \ln y_{nk} \quad (2.26)$$

where t_{nk} , y_{nk} are the k th elements of the label and output vectors respectively, corresponding to the n th training example.

2.3.3 Gradient Descent Optimization

In the previous paragraph, we explored some of the loss functions which correspond in regression and classification tasks. We came to the realization that the minimization of those loss functions, with respect to the models parameters, constitute non-convex optimization problems. The solutions to these optimization problems utilize gradient-based methods.

A small variation in the weight space from \mathbf{w} to $\mathbf{w} + \delta \mathbf{w}$ will result in a small difference in the loss function equal to $\delta E(\mathbf{w}) \simeq \delta \mathbf{w}^\top \nabla E(\mathbf{w})$. The vector $\nabla E(\mathbf{w})$ denotes the direction at which the greatest rate of increase occurs for the loss function $E(\mathbf{w})$. As the error term $E(\mathbf{w})$ is a sufficiently smooth and continuous function of the model parameters \mathbf{w} , then we know that its minimum value will occur at a point \mathbf{w}^* at which $\nabla E(\mathbf{w}^*) = \mathbf{0}$. Therefore a potential scheme for minimizing the aforementioned loss functions would be to alter the weight parameters \mathbf{w} by iteratively taking steps in the direction of $-\nabla E(\mathbf{w})$. A general form of this process can be described in the following way:

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} - \eta \nabla E(\mathbf{w}^{(\tau)}) \quad (2.27)$$

where τ is used for indexing the iterations, while $\eta \in (0, 1]$, formally known as the learning rate, controls the rate of change in the weight parameters \mathbf{w} .

Stochastic Gradient Descent

The optimization process is performed on the basis of a training set $D = \{(\mathbf{x}_n, \mathbf{t}_n)\}_{n=1}^N$ and the total loss function can be expressed as a sum of N separate loss functions $E_n(\mathbf{w})$, with each one corresponding to its respective data sample:

$$E(\mathbf{w}) = \sum_{n=1}^N E_n(\mathbf{w}) \quad (2.28)$$

Therefore, at each iteration of the aforementioned descent mechanism, the changes in the weights \mathbf{w} are performed with respect to the current training example:

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} - \eta \nabla E_n(\mathbf{w}^{(\tau)}) \quad (2.29)$$

This approach constitutes an online gradient descent method, better known as Stochastic Gradient Descent (SGD), as introduced by Rumelhart et al. [90]. A variation of the simple SGD method includes a momentum term \mathbf{v} that effectively depends on the updates of all the previous iterations:

$$\begin{aligned} \mathbf{v}^{(\tau)} &= \alpha \mathbf{v}^{(\tau-1)} - \eta \nabla E(\mathbf{w}^{(\tau)}) \\ \mathbf{w}^{(\tau+1)} &= \mathbf{w}^{(\tau)} + \mathbf{v}^{(\tau)} \end{aligned} \quad (2.30)$$

where $\alpha \in [0, 1]$ is the momentum coefficient parameter that controls the accumulated contribution of the gradients from the previous iterations. This modification aims at accelerating the convergence process and avoid ending up at local minima.

A slightly different version of momentum known as Nesterov Accelerated Momentum (NAG), was introduced by Nesterov [76]. Sutskever et al. [98] modified the original momentum accelerated update of equation 2.30 in the following way:

$$\begin{aligned} \mathbf{v}^{(\tau)} &= \alpha \mathbf{v}^{(\tau-1)} - \eta \nabla E(\mathbf{w}^{(\tau)} + \alpha \mathbf{v}^{(\tau-1)}) \\ \mathbf{w}^{(\tau+1)} &= \mathbf{w}^{(\tau)} + \mathbf{v}^{(\tau)} \end{aligned} \quad (2.31)$$

Instead of evaluating gradient at the current position $\mathbf{w}^{(\tau)}$, we know that the classical momentum update is going to move the weight vector to the position $\mathbf{w}^{(\tau)} + \alpha \mathbf{v}^{(\tau-1)}$. Nesterov momentum evaluates the gradient at this "looked-ahead" position. Suppose that the addition of $\alpha \mathbf{v}^{(\tau-1)}$ results in an undesirable update in the loss function $E(\mathbf{w}^{(\tau)})$. The gradient correction to the term $\mathbf{v}^{(\tau-1)}$ is computed at position $\mathbf{w}^{(\tau)} + \alpha \mathbf{v}^{(\tau-1)}$ and if $\alpha \mathbf{v}^{(\tau-1)}$ is a poor update, then the correction term $\nabla E(\mathbf{w}^{(\tau)} + \alpha \mathbf{v}^{(\tau-1)})$ will point back towards $\mathbf{w}^{(\tau)}$ more strongly than $\nabla E(\mathbf{w}^{(\tau)})$. In that way, it will provide a larger and more appropriate correction to $\mathbf{v}^{(\tau)}$.

Adaptive Gradients

Adaptive Gradients (AdaGrad) is a variant of the SGD method which was introduced by Duchi et al. [30] and utilizes per-parameter learning rate that changes over time. This is important for adapting to the differences in datasets such as size and sparsity. Generally, Adagrad performs better compared to SGD when it is applied on sparse data where sparse parameters are more informative.

We assume that $\mathbf{w} \in \mathbb{R}^d$, namely that the model contains d separate trainable weight parameters. Firstly, a diagonal matrix $\mathbf{G}^{(\tau)} \in \mathbb{R}^{d \times d}$ is introduced:

$$\mathbf{G}^{(\tau)} = \sum_{t=1}^{\tau} \nabla E(\mathbf{w}^{(t)}) (\nabla E(\mathbf{w}^{(t)}))^{\top} \quad (2.32)$$

The acquisition of $\mathbf{G}^{(\tau)}$ for every iteration is computationally impractical, so as an alternative, only the diagonal elements of the aforementioned matrix were considered. The j th element $G_{jj}^{(\tau)}$ in the diagonal, is the sum of squared partial derivatives over all previous iterations $t = 1, 2, \dots, \tau$, with respect to the j th element of the weight vector:

$$G_{jj}^{(\tau)} = \sum_{t=1}^{\tau} \left(\frac{\partial E(\mathbf{w}^{(t)})}{\partial w_j^{(t)}} \right)^2 \quad (2.33)$$

The per parameter update rule takes the following form:

$$w_j^{(\tau+1)} = w_j^{(\tau)} - \frac{\eta}{\sqrt{\epsilon + G_{jj}^{(\tau)}}} \frac{\partial E(\mathbf{w}^{(\tau)})}{\partial w_j^{(\tau)}} \quad (2.34)$$

where $\eta \in (0, 1]$ is the initial learning rate and $\epsilon \sim 10^{-8}$ in order to avoid division by zero. For convenience, we can rewrite 2.34 in vectorized form:

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} - (\epsilon \mathbb{I}_d + \text{diag}(\mathbf{G}^{(\tau)})^{-\frac{1}{2}} \nabla E(\mathbf{w}^{(\tau)}) \quad (2.35)$$

The terms $G_{jj}^{(\tau)}$ constitute scaling factors that differentiate the learning rates for each one of the parameters. As τ increases, gradients for parameters that initially received large updates get more dampened compared to parameters that got few and small updates. Thus, one main benefit of this algorithm is that it does not require manual tuning of the learning rate. However, the increasing accumulation of squared gradients in the denominator causes the learning rates to eventually become infinitesimally small, thus effectively stopping the learning process.

Root Mean Square Propagation

An adaptation of the AdaGrad method that deals with the problem of diminishing gradients is the Root Mean Square Propagation (RMSprop) algorithm, as proposed by Tieleman and Hinton [99]. Instead of dividing the learning rate with the sum of squared gradients, it uses an exponentially decaying average of the squared gradients $u^{(\tau)}$ that is defined recursively. The parameter update rule for the j th element of the weight vector \mathbf{w} takes the following form:

$$\begin{aligned} u_j^{(\tau)} &= \gamma u_j^{(\tau-1)} + (1 - \gamma) \left(\frac{\partial E(\mathbf{w}^{(\tau)})}{\partial w_j^{(\tau)}} \right)^2 \\ w_j^{(\tau+1)} &= w_j^{(\tau)} - \frac{\eta}{\sqrt{\epsilon + u_j^{(\tau)}}} \frac{\partial E(\mathbf{w}^{(\tau)})}{\partial w_j^{(\tau)}} \end{aligned} \quad (2.36)$$

A typical value for the decay parameter is $\gamma = 0.9$ and $\epsilon \sim 10^{-8}$. In this case, RMSprop still modulates the learning rate of each weight parameter based on the magnitudes of its gradients, which has a beneficial equalizing effect. As in the case of SGD with momentum, the exponential average helps us to weigh the more recent gradient updates more than the less recent ones. Therefore, in this case, the updates do not get monotonically smaller. We also need to note that RMSprop implicitly performs simulated annealing. Suppose if we are heading towards the minima, and we want to slow down so as to not to overshoot the minima. RMSprop automatically will decrease the size of the gradient steps towards minima when the steps are too large.

Adaptive Moment Optimization

During our previous analyses we have seen that momentum aims at accelerating the search in the direction of minima, AdaGrad improves performance in problems with sparse gradients while RMSprop impedes the search in the direction of oscillations. The Adaptive Moment Optimization (Adam) algorithm, introduced by Kingma and Ba [56] combines the heuristics of both momentum, AdaGrad and RMSprop. In this optimization algorithm, running averages $m^{(\tau)}$, $u^{(\tau)}$ of both the gradients and the second moments of the gradients are used. The parameter update rule for the j th element of the weight vector \mathbf{w} takes the following form:

$$m_j^{(\tau)} = \beta_1 m_j^{(\tau-1)} + (1 - \beta_1) \frac{\partial E(\mathbf{w}^{(\tau)})}{\partial w_j^{(\tau)}} \quad (2.37)$$

$$u_j^{(\tau)} = \beta_2 u_j^{(\tau-1)} + (1 - \beta_2) \left(\frac{\partial E(\mathbf{w}^{(\tau)})}{\partial w_j^{(\tau)}} \right)^2 \quad (2.38)$$

$$\hat{m}_j^{(\tau)} = \frac{m_j^{(\tau)}}{1 - \beta_1^\tau} \quad (2.39)$$

$$\hat{u}_j^{(\tau)} = \frac{u_j^{(\tau)}}{1 - \beta_2^\tau} \quad (2.40)$$

$$w_j^{(\tau+1)} = w_j^{(\tau)} - \eta \frac{\hat{m}_j^{(\tau)}}{\sqrt{\epsilon + \hat{u}_j^{(\tau)}}} \quad (2.41)$$

Typical values for the parameters are $\eta = 10^{-3}$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon \sim 10^{-8}$. In this case, equation 2.37 updates the exponential average of gradients while equation 2.38 updates the exponential average of squared gradients. In addition, equations 2.39 and 2.40 perform bias correction on the first and second moment estimates respectively. Finally, equation 2.41 performs the per weight parameter update for the current iteration. Adam realizes the benefits of all the aforementioned optimization algorithms, namely momentum AdaGrad and RMSprop. In [56], Adam was compared with NAG, Adagrad and RMSprop, during the training of a MLP on the MNIST dataset [61]. Results showed that Adam converged faster than all of the other optimizers.

2.3.4 Backpropagation

The application of any of the aforementioned optimization algorithms requires the computation of the gradient vector $\nabla E(\mathbf{w})$ or equivalently the computation of the partial derivatives of the loss function $E(\mathbf{w})$ with respect to each one of the trainable weights of the model. This is achieved through the Backpropagation (BP) algorithm which was originally introduced by Rumelhart et al. [90]. The BP algorithm can be applied to any general network that has an arbitrary feed-forward topology, arbitrary differentiable nonlinear activation functions and a broad class of error functions. As we have seen, in a general feed-forward network, each neuron computes an activation form:

$$z_j = \sum_i w_{ij} h_i \quad (2.42)$$

supposing that the weights contain the bias terms and the input feature vectors are expanded accordingly, as discussed in section 2.2.2. The output activations z_j are weighted sums of the

outputs from the hidden units of the previous layers. We have omitted the layer indexing in order to simplify the numerical expressions for the upcoming analysis. The final output of each unit is obtained after the application of an activation function $f(\cdot)$, namely $h_j = f(z_j)$. Assuming we have a training dataset $D = \{(\mathbf{x}_n, \mathbf{t}_n)\}_{n=1}^N$, we consider the case of online training, expressing the total loss function $E(\mathbf{w})$ as the sum of N separate loss functions $E_n(\mathbf{w})$, each one corresponding to its respective data sample. Now we need to compute the partial derivative of $E_n(\mathbf{w})$ with respect to some weight w_{ij} . By using the chain rule of differential calculus we obtain:

$$\begin{aligned} \frac{\partial E_n(\mathbf{w})}{\partial w_{ij}} &= \frac{\partial E_n(\mathbf{w})}{\partial z_j} \frac{\partial z_j}{\partial w_{ij}} \\ &= \frac{\partial E_n(\mathbf{w})}{\partial z_j} \frac{\partial}{\partial w_{ij}} \left(\sum_m w_{mj} h_m \right) \\ &= \frac{\partial E_n(\mathbf{w})}{\partial z_j} h_i \end{aligned} \quad (2.43)$$

By setting $\delta_j \equiv \partial E_n(\mathbf{w}) / \partial z_j$, we can rewrite 2.43 in the following compact form:

$$\frac{\partial E_n(\mathbf{w})}{\partial w_{ij}} = \delta_j h_i \quad (2.44)$$

Now we need to consider two separate cases. In the first case, the unit j is considered to be one of the output units of the network. Therefore w_{ij} denotes the hidden to output weight connection from hidden unit i to output unit j . In this case, δ_j can be directly computed and equation 2.44 becomes:

$$\begin{aligned} \frac{\partial E_n(\mathbf{w})}{\partial w_{ij}} &= \frac{\partial E_n(\mathbf{w})}{\partial h_j} \frac{\partial h_j}{\partial z_j} h_i \\ &= \frac{\partial E_n(\mathbf{w})}{\partial h_j} f'(z_j) h_i \end{aligned} \quad (2.45)$$

where the terms $\partial E_n(\mathbf{w}) / \partial h_j$ and $f'(z_j)$ can be directly computed based on the form of the loss function $E_n(\mathbf{w})$ and the chosen activation functions $f(\cdot)$. For example in the simple case of regression, as mentioned in section(), the output unit activation functions would be the identity functions ($f'(\cdot) = 1$), while the loss function would be:

$$E_n(\mathbf{w}) = \sum_k (t_{nk} - h_{nk})^2$$

where $h_{nk} = h_k(\mathbf{x}_n, \mathbf{w})$, namely the index n refers to the current training sample and index k refers to the k th element of both the output and target vectors. By ignoring the subscripts n and by substituting in equation 2.45 we obtain:

$$\frac{\partial E_n(\mathbf{w})}{\partial w_{ij}} = 2(t_j - h_j) h_i \quad (2.46)$$

On the other hand, if the unit j is a hidden unit, then the computation of the δ_j term is more complicated. We start by assuming that the unit in question belongs to the last hidden layer of the network. In that way, provided the network is fully-connected, unit j influences

all the separate outputs of the network. Terms δ_j can now be written in the following form:

$$\begin{aligned}
\delta_j &\equiv \frac{\partial E_n(\mathbf{w})}{\partial z_j} = \sum_k \frac{\partial E_n(\mathbf{w})}{\partial z_k} \frac{\partial z_k}{\partial z_j} \\
&= \sum_k \delta_k \frac{\partial z_k}{\partial z_j} \\
&= \sum_k \delta_k \frac{\partial}{\partial z_j} \left(\sum_m w_{mk} h_m \right) \\
&= \sum_k \delta_k \frac{\partial}{\partial z_j} \left(\sum_m w_{mk} f(z_m) \right) \\
&= \sum_k \delta_k w_{jk} f'(z_j) \\
&= f'(z_j) \sum_k \delta_k w_{jk}
\end{aligned} \tag{2.47}$$

By substituting the above result in 2.44, assuming that w_{ij} is the weight connecting the i th unit of the $(l-1)$ th hidden layer with the j th unit of the l th hidden layer and with the application of the proper layer indexing, the following expression for the partial derivative of $E_n(\mathbf{w})$ with respect to $w_{ij}^{(l)}$ can be obtained:

$$\frac{\partial E_n(\mathbf{w})}{\partial w_{ij}^{(l)}} = f'(z_j^{(l)}) \left(\sum_k \delta_k^{(l+1)} w_{jk}^{(l+1)} \right) h_i^{(l-1)} \tag{2.48}$$

Equation 2.3.4 indicates that in order to compute the δ_j term for some hidden unit j , we need to first compute and then backpropagate the δ terms that correspond to all the units of the following layer. By recursively applying equation 2.3.4 it is possible to compute the required δ terms of all the hidden units of the network and thus be able to compute the gradient of the loss function that is required for the application gradient-based optimization.

2.3.5 Regularization

Given a training set, the training process involves the application of gradient-based minimization of one of the aforementioned loss functions with respect to the model's trainable parameters. The computed error between the predicted and ground-truth values is referred to as training error. However, the ultimate goal of a deep network model is to be able to make correct predictions about new samples that it has never encountered before. These samples constitute another labeled set known as the test set. The test set is used to measure the generalization capabilities of a particular model. Generally the test error is expected to be greater or equal to the observed training error. Consequently, the goal of supervised learning is to achieve a low training error, while maintaining a small gap between the training and test errors.

Based on this criterion, we can distinguish two different challenges regarding the performance of a neural network model: underfitting and overfitting. Underfitting refers to the inability of the model to achieve a small training error and results from either poor architectural design or extreme learning rate values. In the latter case, small learning rates may render the training process extremely slow, while big learning rates may result in oscillations and therefore lack of convergence. While it is simpler to cope with underfitting as a problem, the occurrence of overfitting poses a far more complicated challenge. Overfitting refers to the inability of the model to generalize well on test data, namely the difference between

the training and test errors is large. The predisposition of a model toward underfitting or overfitting is dependent on its capacity or equivalently its complexity, that is its ability to fit a large variety of target output distributions. If a model has low capacity then it is likely to underfit on the training data, while if it has high capacity, it is more likely to memorize certain aspects of the training examples that are not representative of the test samples, thus overfitting on the training data. As mentioned above, overfitting being a more prominent challenge in deep learning, gave rise to several algorithms and methodologies which aim at reducing the generalization error at a potential expense of increased training error. These techniques are generally referred to as regularization. The most common and widely used regularization strategies will be analyzed in the following paragraphs.

Parameter Regularization

The most common way of controlling the complexity of a model during the training process is by adding a complexity-representing term R to the existing loss function $E(\mathbf{w})$. In the case of neural networks, the complexity representing term is simply either the absolute-value norm or the squared Euclidean norm of the model's weight parameters, that is L^1 and L^2 regularization respectively. In this case, the regularized loss function $\tilde{E}(\mathbf{w})$ becomes:

$$\tilde{E}(\mathbf{w}) = E(\mathbf{w}) + \lambda R(\mathbf{w}) \quad (2.49)$$

The regularization coefficient $\lambda \in [0, \infty)$ controls the contribution of the regularization term in relation to the original objective function. Obviously, if $\lambda = 0$ then this results in the unregularized case. The regularization term $R(\mathbf{w})$ provides a measurement of the size of the model parameters. Therefore during the training process, the optimization algorithm will minimize both the original objective function $E(\mathbf{w})$ as well as reduce the element values of the weight vector \mathbf{w} . The choice between L^1 and L^2 based regularization will result in different update rules during each optimization step and will have different effects on the eventual form of the model weight parameters.

L^2 Regularization In the case of L^2 weight regularization, the regularization term is equal to the squared Euclidean norm of the weight vector \mathbf{w} , that is $R(\mathbf{w}) = \frac{1}{2}\|\mathbf{w}\|_2^2$. This methodology is commonly known as weight decay. The regularized loss function $\tilde{E}(\mathbf{w})$ takes the following form:

$$\tilde{E}(\mathbf{w}) = E(\mathbf{w}) + \frac{\lambda}{2}\|\mathbf{w}\|_2^2 \quad (2.50)$$

The above modification will result in the following changes in the update rule 2.27 of the generic gradient-descent optimizer:

$$\begin{aligned} \mathbf{w}^{(\tau+1)} &= \mathbf{w}^{(\tau)} - \eta \nabla \tilde{E}(\mathbf{w}^{(\tau)}) \\ &= \mathbf{w}^{(\tau)} - \eta \nabla (E(\mathbf{w}^{(\tau)}) + \frac{\lambda}{2}\|\mathbf{w}^{(\tau)}\|_2^2) \\ &= \mathbf{w}^{(\tau)} - \eta (\nabla E(\mathbf{w}^{(\tau)}) + \lambda \mathbf{w}^{(\tau)}) \\ &= (1 - \eta\lambda)\mathbf{w}^{(\tau)} - \eta \nabla E(\mathbf{w}^{(\tau)}) \end{aligned} \quad (2.51)$$

Assuming that $\lambda < 1/\eta$, the last equality of update rule 2.51 indicates that the weight vector at each iteration is being multiplied by a shrinking factor $(1 - \eta\lambda)$, thus giving the weights the tendency to decay towards zero. By making a quadratic approximation of the loss function $E(\mathbf{w})$, a clearer insight into the effect that weight decay has on the training process can

be obtained. The approximation is calculated around a potential minimum point \mathbf{w}^* in the weight space:

$$E(\mathbf{w}) = E(\mathbf{w}^*) + \nabla E(\mathbf{w}^*)^\top (\mathbf{w} - \mathbf{w}^*) + \frac{1}{2}(\mathbf{w} - \mathbf{w}^*)^\top \mathbf{H}(\mathbf{w} - \mathbf{w}^*) \quad (2.52)$$

The initial hypothesis states that \mathbf{w}^* constitutes a local minimum. Therefore, we can conclude that $\nabla E(\mathbf{w}^*) = \mathbf{0}$. Moreover, \mathbf{H} denotes the Hessian matrix of $E(\mathbf{w})$, calculated on the point $\mathbf{w} = \mathbf{w}^*$ and is comprised of the second derivatives of $E(\mathbf{w})$, namely $\mathbf{H} = \nabla^2 E(\mathbf{w})$. By substituting equation 2.52 in the regularized loss function 2.50, we get:

$$\tilde{E}(\mathbf{w}) = E(\mathbf{w}^*) + \frac{1}{2}(\mathbf{w} - \mathbf{w}^*)^\top \mathbf{H}(\mathbf{w} - \mathbf{w}^*) + \frac{\lambda}{2} \|\mathbf{w}\|_2^2 \quad (2.53)$$

On the basis of the assumption that \mathbf{w}^* constitutes a minimum for the loss function $E(\mathbf{w})$, it can be concluded that \mathbf{H} is semi-positive definite. Considering a small deviation around the location of the minimum, $\mathbf{w} = \mathbf{w}^* + \epsilon \mathbf{v}$, where ϵ is an arbitrarily small scaling factor and \mathbf{v} is a non-zero translation weight vector, the quadratic approximation of $E(\mathbf{w})$ around these positions yields:

$$\begin{aligned} E(\mathbf{w}^* + \epsilon \mathbf{v}) &= E(\mathbf{w}^*) + \frac{\epsilon^2}{2} \mathbf{v}^\top \mathbf{H} \mathbf{v} \\ \frac{\partial^2 E}{\partial \epsilon^2} &= \mathbf{v}^\top \mathbf{H} \mathbf{v} \end{aligned} \quad (2.54)$$

The existence of a minimum enforces the curvature to be non-negative near the the location $\mathbf{w} = \mathbf{w}^*$, that is $\mathbf{v}^\top \mathbf{H} \mathbf{v} \geq 0$, or equivalently that the Hessian matrix is semi-positive definite. The minimum of the regularized loss function $\tilde{E}(\mathbf{w})$ will occur at a point $\tilde{\mathbf{w}}$ at which $\nabla \tilde{E}(\tilde{\mathbf{w}}) = \mathbf{0}$. Calculating the gradient of equation 2.53 and setting it equal to zero, yields:

$$\begin{aligned} \lambda \tilde{\mathbf{w}} + \mathbf{H}(\tilde{\mathbf{w}} - \mathbf{w}^*) &= \mathbf{0} \\ (\lambda \mathbb{I} + \mathbf{H})\tilde{\mathbf{w}} - \mathbf{H}\mathbf{w}^* &= \mathbf{0} \\ \tilde{\mathbf{w}} &= (\lambda \mathbb{I} + \mathbf{H})^{-1} \mathbf{H} \mathbf{w}^* \end{aligned} \quad (2.55)$$

Assuming that the second partial derivatives are continuous and with the use of Schwarz's theorem regarding the symmetry of second derivatives, we can conclude that the Hessian matrix will be symmetric. Obviously, it is also real, therefore by using eigenvalue decomposition, it can be written as:

$$\mathbf{H} = \mathbf{Q} \mathbf{\Lambda} \mathbf{Q}^\top \quad (2.56)$$

where \mathbf{Q} is an orthogonal matrix ($\mathbf{Q} \mathbf{Q}^\top = \mathbf{Q}^\top \mathbf{Q} = \mathbb{I}$) composed of the eigenvectors of \mathbf{H} and $\mathbf{\Lambda}$ is a diagonal matrix composed of the corresponding eigenvalues. Substituting equation 2.56 in equation 2.55, we get:

$$\begin{aligned} \tilde{\mathbf{w}} &= (\lambda \mathbb{I} + \mathbf{Q} \mathbf{\Lambda} \mathbf{Q}^\top)^{-1} \mathbf{Q} \mathbf{\Lambda} \mathbf{Q}^\top \mathbf{w}^* \\ &= [\mathbf{Q}(\lambda \mathbb{I} + \mathbf{\Lambda})\mathbf{Q}^\top]^{-1} \mathbf{Q} \mathbf{\Lambda} \mathbf{Q}^\top \mathbf{w}^* \\ &= \mathbf{Q}^{-\top} (\lambda \mathbb{I} + \mathbf{\Lambda})^{-1} \mathbf{Q}^{-1} \mathbf{Q} \mathbf{\Lambda} \mathbf{Q}^\top \mathbf{w}^* \\ &= \mathbf{Q}(\lambda \mathbb{I} + \mathbf{\Lambda})^{-1} \mathbf{\Lambda} \mathbf{Q}^\top \mathbf{w}^* \end{aligned} \quad (2.57)$$

Equation 2.57 indicates that the application of weight decay results in the rescaling of the weights along the directions defined by the eigenvectors of the Hessian matrix by a factor equal to $\frac{\Lambda_{ii}}{\lambda + \Lambda_{ii}}$. Directions along which the corresponding eigenvalues Λ_{ii} are large indicate strong contribution in the reduction of the loss function. In these directions, where $\lambda \ll \Lambda_{ii}$, the regularization effect is negligible. However, components with $\lambda \gg \Lambda_{ii}$ correspond to

irrelevant directions along which the gradient will not significantly increase and which will be regularized to have nearly zero magnitude.

Additionally, in the scope of stochastic gradient descent (SGD), the gradient of the loss function is evaluated over a single training example at a time. In that way, the regularization term needs to be rescaled so as to make the loss function comparable to datasets of different sizes. More specifically, given a training set $D = \{(\mathbf{x}_n, \mathbf{t}_n)\}_{n=1}^N$, the total loss function can be expressed as a sum of N separate loss functions $E_n(\mathbf{w})$ and their corresponding regularized versions would be:

$$\tilde{E}_n(\mathbf{w}) = E_n(\mathbf{w}) + \frac{\lambda}{2N} \|\mathbf{w}\|_2^2 \quad (2.58)$$

Adding weight decay usually results in much smaller weights across the entire model, without zero value enforcement. In that way, L^2 regularization does not promote sparsity and its application in high-dimensional data may result in uninterpretable models.

L^1 Regularization In the case of L^1 weight regularization, the regularization term is equal to the absolute-value norm of the weight vector \mathbf{w} , that is $R(\mathbf{w}) = \|\mathbf{w}\|_1$. The regularized loss function $\tilde{E}(\mathbf{w})$ takes the following form:

$$\begin{aligned} \tilde{E}(\mathbf{w}) &= E(\mathbf{w}) + \lambda \|\mathbf{w}\|_1 \\ &= E(\mathbf{w}) + \lambda \sum_i |w_i| \end{aligned} \quad (2.59)$$

The above modification will result in the following changes in the update rule 2.27 of the generic gradient-descent optimizer:

$$\begin{aligned} \mathbf{w}^{(\tau+1)} &= \mathbf{w}^{(\tau)} - \eta \nabla \tilde{E}(\mathbf{w}^{(\tau)}) \\ &= \mathbf{w}^{(\tau)} - \eta \nabla (E(\mathbf{w}^{(\tau)}) + \lambda \|\mathbf{w}^{(\tau)}\|_1) \\ &= \mathbf{w}^{(\tau)} - \eta (\nabla E(\mathbf{w}^{(\tau)}) + \lambda \text{sgn}(\mathbf{w}^{(\tau)})) \\ &= \mathbf{w}^{(\tau)} - \eta \lambda \text{sgn}(\mathbf{w}^{(\tau)}) - \eta \nabla E(\mathbf{w}^{(\tau)}) \end{aligned} \quad (2.60)$$

Consequently, the contribution of the regularization term to the gradient does not scale linearly with respect to each element w_i of the weight vector, as it now is a constant term with a sign equal to $\text{sgn}(w_i)$ and a magnitude that depends on both the learning rate and the regularization coefficient. For convenience, it is also assumed that the Hessian matrix \mathbf{H} is diagonal with non-zero elements $H_{ii} > 0$. Substituting the quadratic approximation described by equation 2.52 in the L^1 regularized loss function 2.59, would yield:

$$\tilde{E}(\mathbf{w}) = E(\mathbf{w}^*) + \sum_i \lambda |\tilde{w}_i| + \frac{1}{2} H_{ii} (w_i - w_i^*)^2 \quad (2.61)$$

A graphical inspection of equation 2.61 with respect to some parameter w_i reveals that if $w_i^* > 0$ then $\tilde{w}_i \in [0, w_i^*]$, else if $w_i^* < 0$ then $\tilde{w}_i \in [-|w_i^*|, 0]$, that is \tilde{w}_i and w_i^* must share the same sign. Computing the partial derivatives of equation 2.61 with respect to each element w_i (assuming $w_i \neq 0$) and setting them equal to zero, results in the following expression for the regularized optimal weight vector \tilde{w}_i :

$$\begin{aligned} \lambda \text{sgn}(\tilde{w}_i) + H_{ii} (\tilde{w}_i - w_i^*) &= 0 \\ \tilde{w}_i &= w_i^* - \frac{\lambda \text{sgn}(\tilde{w}_i)}{H_{ii}} \\ \tilde{w}_i &= \begin{cases} \text{sgn}(w_i^*) (|w_i^*| - \frac{\lambda}{H_{ii}}) & \text{if } |w_i^*| > \frac{\lambda}{H_{ii}} \\ 0 & \text{otherwise} \end{cases} \end{aligned} \quad (2.62)$$

In that way two possibilities emerge. In the first case, if $|w_i^*| > \frac{\lambda}{H_{ii}}$, then the regularized optimal solution is shifted closer to the origin by a distance equal to $\frac{\lambda}{H_{ii}}$. On the other hand, if $|w_i^*| \leq \frac{\lambda}{H_{ii}}$, then a large enough value of λ is capable of forcing \tilde{w}_i to go to zero. The above formulation constitutes the main advantage of L^1 regularization, which is the fact that its application enforces model sparsity. That is, L^1 regularization results in sparser solutions, where more parameters will end up with a value of zero while only the ones exhibiting higher variance will survive. This is especially useful when the given data have many dimensions that are not correlated. On the other hand, the L^1 regularizer comes at a disadvantage when the number of training samples is lower than the number of dimensions or the data dimensions are highly correlated.

Early Stopping

Another popular method for controlling the variable complexity of a model during the training procedure is called Early Stopping. The application of this technique involves the usage of an auxiliary set of independent data, aside from the aforementioned training and test sets. This set of data is referred to as the validation set and constitutes a small subset of the original training set which is distinguished from the rest of the data. As a model is being trained on the remaining training data, it does not get exposed to any of the validation data as the validation set is only utilized as a measure of evaluating the generalization capability of the model during each epoch of training.

The iterative training process of a neural network regularly results in a steady and gradual reduction in the training error as being measured by some loss function with respect to the training data. In that way, the training error is a decreasing function of the training iteration index. On the other hand, the error corresponding to new, unseen data always exhibits a different kind of progression. In general, during the first training iterations, the validation error shows a steady but often slower decrease relative to the training error. Moreover, as the training proceeds and the network begins to overfit, the validation error starts to increase again, resulting in a characteristic U-shaped curve. This is because, after some point during the training process, the network focuses on patterns of the training examples that are irrelevant and lack discriminative power among other data of the same kind. Therefore, solving this problem involves a reduction in the capacity/complexity of the network in order to prevent it from memorizing training examples.

This can be achieved, if the training process is halted near the point in time at which the increase in validation error starts to occur. The application of this technique requires keeping a copy of the networks parameters as they were configured during the training iteration with the lowest monitored validation error. Additionally, every time the validation error reaches a new minimum, the parameter copies are overwritten. The process is terminated when no improvement in the validation error is observed after a particular pre-specified number of iterations, which also constitutes the only hyperparameter of the current method and is commonly known as patience. After some potentially needed fine-tuning of this hyperparameter, the training can effectively be stopped close to the point of the smallest observed error with respect to the validation data set, thus obtaining a network with good generalization performance.

The main costs of this method are related with the storage capacity needed for the required maintenance of copies for the model's best parameters as well as the training runs required for fine-tuning the patience hyperparameter. However, the application of early stopping not only provides a model together with a good estimate of its generalization performance, but also decreases the overall number of training iterations and therefore, the computational cost of the training procedure. Lastly, it is worth noting that the Early Stopping technique is

closely related with the weight decay regularizer. More specifically, the relationship between early stopping and weight decay can be quantified, as the term $\tau\eta$ plays the role of the reciprocal of the regularization coefficient λ . In that way, the effective number of parameters in the network grows during the course of training.

Dropout

Dropout as originally introduced in [96], constitutes another very popular method of regularization. The central concept behind Dropout, is that the combination of several neural network models and the averaging of their separate predictions is likely to improve the performance of machine learning tasks. This technique is commonly known as Bagging and involves training multiple models and collectively evaluating them on each test example. However, when it comes at training multiple large neural networks, the computational cost of this procedure becomes intractable. Dropout, offers a practical and inexpensive solution to bagging, by combining an exponential number of learned models that share parameters.

The term “dropout” refers to the deactivation of hidden unit nodes of the network during the training phase. During each iteration of the training process, each hidden unit has a probability p of being retained and $1 - p$ of being deactivated. The dropping out of some hidden unit includes the deactivation of all the incoming and outgoing connections of that particular node. A neural network model composed of N total hidden units, can be perceived as an ensemble of 2^N different possible sub-networks. Even though the number of the included networks is exponentially large, the weight sharing hypothesis ensures that the total number of trainable parameters remains quadratic with respect to the number of hidden units. More formally, considering a fully connected feed-forward neural network consisting of L hidden layers, with $l \in \{1, \dots, L\}$ being the layer index, $f(\cdot)$ denoting the hidden layer activation functions, $\mathbf{z}^{(l)}$, $\mathbf{h}^{(l)}$ denoting the activations and outputs of the l th hidden layer respectively and $\mathbf{h}^{(0)} = \mathbf{x}$ being some input vector, then, by modifying equations 2.4, the forward-pass operations with the application of dropout can be described by the following set of equations:

$$\begin{aligned} r_j &\sim \text{Bernoulli}(p) \\ \hat{\mathbf{h}}^{(l-1)} &= \mathbf{r}^{(l-1)} \odot \mathbf{h}^{(l-1)} \\ \mathbf{z}^{(l)} &= \mathbf{W}_l^\top \hat{\mathbf{h}}^{(l-1)} \\ \mathbf{h}^{(l)} &= f(\mathbf{z}^{(l)}) \end{aligned} \tag{2.63}$$

where \mathbf{W}_l denotes the parameter weight matrix of the l th layer and \mathbf{r} are vectors of independent Bernoulli random variables that have a probability p of being equal to 1. Additionally, during training, hidden units are forced to learn not to rely on other hidden units being present, preventing complex co-adaptations relative to the training data and thus reducing overfitting.

During inference, an average version of the network is used that includes all the hidden units, after getting their weighted connections scaled appropriately to compensate for the fact that none of the units is deactivated. This scaling is applied in order to ensure that the output of any hidden unit at test time matches the expected value of the same output during training. More specifically, during training, the expected output of some hidden unit j is:

$$\mathbb{E}[z_j] = \sum_i r_i w_{ij} h_i = \sum_i \mathbb{E}[r_i] w_{ij} h_i = p \sum_i w_{ij} h_i \tag{2.64}$$

as the expected value of $r_j \sim \text{Bernoulli}(p)$ is equal to p . It is worth mentioning that for the simple case of multi-class classification using a neural network with one hidden layer,

N hidden units and a softmax output layer, the use of the aforementioned average model is equivalent to using the geometric mean of the output probability distributions of the 2^N possible sub-networks for the actual final prediction of the target labels.

Batch Normalization

Batch normalization was introduced by Ioffe and Szegedy [54] and it constitutes another notable regularization technique that reduces overfitting and accelerates the training procedure. The batch normalization technique was introduced as a means of solving the internal covariate shift problem. This term refers to the change in the distribution of the activations of the network, caused either due to variations in the network parameters during training or the distribution of the input data. In neural networks, the input layer feeds the first hidden layer and the output of every other layer directly influences the distribution of the outputs in all the subsequent layers. This avalanche effect becomes more prominent as the depth of a neural network increases and the potential effect of even the slightest changes in model parameters receives gradual amplification during its propagation towards the output layer. Moreover, fluctuations in the output activations of each layer would result in oscillations in the computed loss function, therefore requiring longer training and smaller learning rates so as to achieve convergence. Batch normalization is a method intended to mitigate the aforementioned issues.

The suggested algorithm considers mini-batch training of a neural network model, according to which, the training dataset is split in small subgroups with no overlapping elements, called batches. At each training iteration, one of them is used to calculate the error and update the model parameters and after all batches have been used, the process repeats for a predefined number of training epochs. Moreover, the computed errors are summed or averaged over the mini-batches, resulting in a reduction in the variance of the back-propagated gradients.

Batch normalization aims at standardizing the distribution of output activations at each layer of the network before the application of the non-linear activation functions, resulting in them having zero mean and unit variance. This procedure is applied on each individual mini-batch. Considering a d -dimensional layer activation $\mathbf{z} = [z^{(1)}, \dots, z^{(d)}]^\top$, the normalization is applied to each dimension separately. Additionally, considering a mini-batch $B = \{\mathbf{z}_1, \dots, \mathbf{z}_m\}$ containing m such activations, the mean μ and variance σ^2 of the latter are calculated using the corresponding mini-batch statistics. As the normalization of the layer activations may constraint the outputs on the linear regime of the corresponding non-linearities, two extra learnable parameters γ and β are introduced so as to ensure that the overall batch normalization transform, denoted as $\text{BN}_{\gamma, \beta}(\cdot)$, is able to represent the identity function, if needed. The equations describing the batch normalization transform over a mini batch B , with respect to the i th dimension of the layer activations, are the following:

$$\begin{aligned} \mu_B^{(i)} &= \frac{1}{m} \sum_{j=1}^m z_j^{(i)} \\ (\sigma_B^{(i)})^2 &= \frac{1}{m} \sum_{j=1}^m (z_j^{(i)} - \mu_B^{(i)})^2 \\ \hat{z}_j^{(i)} &= \frac{z_j^{(i)} - \mu_B^{(i)}}{\sqrt{(\sigma_B^{(i)})^2 + \epsilon}} \\ \mathbf{y}_j^{(i)} &= \gamma^{(i)} \hat{z}_j^{(i)} + \beta^{(i)} = \text{BN}_{\gamma^{(i)}, \beta^{(i)}}(z_j^{(i)}) \end{aligned} \tag{2.65}$$

In fact, if $\gamma^{(i)} = \sqrt{(\sigma_B^{(i)})^2}$, neglecting the infinitesimal constant ϵ and $\beta^{(i)} = \mu_B^{(i)}$, then $\text{BN}_{\gamma^{(i)}, \beta^{(i)}}(\cdot)$ matches the identity transform. Fixing the layer activation distributions also has a beneficial effect on the gradient flow through the network, as it reduces the dependence of gradients on the scale of the parameters or of their initial values.

During inference the network outputs are expected to depend deterministically on the corresponding input. Therefore the parameters of the batch normalization layers are fixed. More, specifically, the unbiased variance estimates $\text{Var}[z^{(i)}] = \frac{m}{m-1} \mathbb{E}_B[(\sigma_B^{(i)})^2]$ and the mean estimates $\mathbb{E}[z^{(i)}] = \mathbb{E}_B[\mu_B^{(i)}]$ are used, along with the per dimension learnt parameters $\gamma^{(i)}, \beta^{(i)}$. The expected values are calculated over all training mini-batches B . As the means and variances are fixed during inference, then the per dimension batch normalization transform is replaced by the following simple linear transformation:

$$y_j^{(i)} = \frac{\gamma^{(i)}}{\sqrt{\text{Var}[z^{(i)}] + \epsilon}} z_j^{(i)} + \left(\beta^{(i)} - \frac{\gamma^{(i)} \mathbb{E}[z^{(i)}]}{\sqrt{\text{Var}[z^{(i)}] + \epsilon}} \right) \quad (2.66)$$

Batch normalization, as a regularization technique, is very similar to dropout. It multiplies each hidden unit by a random value at each step of training. More specifically, the random value is the standard deviation of all the hidden units present in the mini-batch and as different samples are chosen, supposedly at random, for inclusion in the mini-batch at each iteration, the statistical properties of the mini-batch randomly fluctuate. Batch normalization also subtracts a random value, the mean activation of the mini-batch from each hidden unit at each iteration. Both of these sources of noise force every layer to become more robust to a lot of variation in its input, similarly to the effect of dropout.

Data Augmentation

Data augmentation is another more general but equally useful and efficient regularization technique. It involves the synthesis of additional data samples through the transformation of the existing training data. The core concept behind this method is the fact that the best way to make a neural network model to generalize better is to expose it to a larger and more diverse training dataset. Moreover, training models with a high number of parameters requires a proportional amount of available examples that public datasets often lack. The process of creating synthetic examples based on current available data reduces the effect of overfitting while training.

More specifically, data augmentation techniques have been extensively applied on image-based datasets. Data augmentation techniques of this kind can be separated into two types: online (on-the-fly) and offline. On-the-fly methods often include the extraction of random crops centered on the four corners and center of the input training images. The crops can also be flipped horizontally, resulting in a synthetic dataset of even ten times the initial size. In addition, offline methods have also been developed in order to infuse greater diversity and variance in the training dataset. These utilize various affine transformation matrices such as scaling, rotation, skewing and shifting, as well as appearance filters such as noise injection, averaging, color and contrast jittering and motion blurring. In that way, the introduction of additional variance in the training data, encourages the model to become more robust to noise and variations in its input.

In general, the application of data augmentation is likely to have a more prominent effect on the actual performance of a network than the usage of different network architectures or machine learning algorithms. When comparing the performance of separate models which utilize different algorithms and also do not share the same data augmentation schemes, then any observed increase in performance of one of the models over the others is more likely to be caused by a particular choice of data augmentation transformations.

2.3.6 Convolutional Neural Networks

Convolutional Neural Networks, ConvNets or CNNs, comprise a class of neural network models which find extensive applications in data that share some type of grid-shaped arrangement. These compatible types of data can be found in multi-dimensional settings. For example, CNNs can be applied on 1D time-series, such as audio input data, for the extraction of useful patterns that can be found across the signals. The most common and popular application of CNNs is related with processing and pattern recognition in images. In that case, the performed operations are referred to as convolutions over volume as the multiple input image channels are processed simultaneously. A high-level schematic diagram of a modern CNN can be seen in figure 2.6. Lastly, 3D variants of CNNs can be applied in the analysis of volumetric data such as videos, which can be perceived as a stack or 2D image frames. In that case the 3D aspect is based on the presence of the additional time axis aside from the two space dimensional axes.

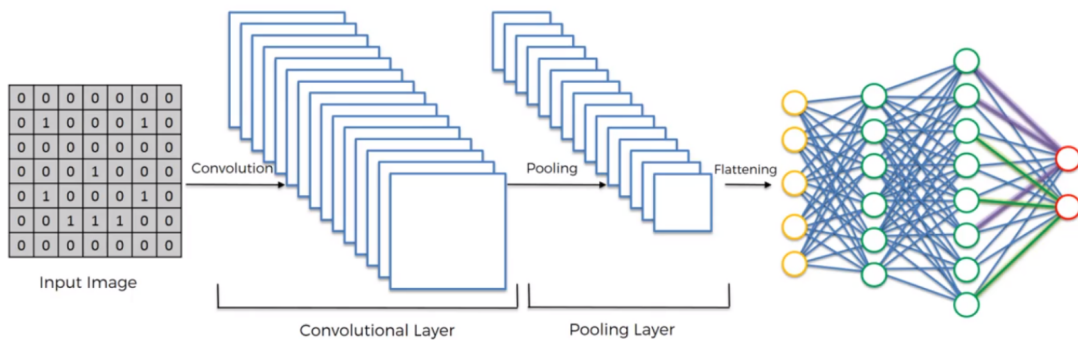


Figure 2.6: Illustration of a standard 2D Convolutional Neural Network (CNN), consisting of a series of convolutional and pooling layers, responsible for feature extraction, followed by fully-connected (FC) layers which perform either classification or regression, depending on the task. Adapted from <https://www.andreaperlato.com/aipost/cnn-and-softmax/>.

In the context of deep learning, the convolution operation refers to the multiplication of a multidimensional input tensor \mathbf{I} with a small multidimensional array of weights \mathbf{K} , which is called kernel. This operation is performed across the span of the entire input array, as the multiplications with the weight kernel are performed in a sliding window fashion. As the input data often have multiple channels of information, such as RGB images or stereo audio signals, the convolution operation applies separate weight kernels for each one of the input data channels. That is, in the multi-channel case, which is the most common in practice, the convolution operations utilize filters which are collections of kernels, with there being one kernel for every single input channel, and each kernel being unique. With that being said, the main structural block of the CNN is the convolutional layer. It is comprised of a predefined number of filters, containing sets of trainable weight parameters, each one intended at extracting different kinds of features from the input data. As the convolution operation is performed as a sliding window multiplication, the weights of a particular filter belonging to some convolutional layer, are used across all positions of the layer input data. This weight sharing feature of convolutional layers leads to sparse connectivities across the CNN model, in contrast with the fully connected feed forward models, where each unit of one layer is connected with all the units of both the preceding and subsequent layers.

Convolution Operation

For convenience, the following analysis considers specifically the case of 2D convolutions, as it is the most common one and is highly relevant to the task of affect recognition in images.

Given an input map \mathbf{I} and a convolutional kernel \mathbf{K} , the convolution operation is given by:

$$\begin{aligned} (\mathbf{I} * \mathbf{K})_{i,j} &= \sum_m \sum_n I_{i-m,i-n} K_{m,n} \\ &= \sum_m \sum_n I_{i+m,i+n} K_{-m,-n} \end{aligned} \quad (2.67)$$

Another similar operation, is called cross-correlation and is described by the following expression:

$$(\mathbf{I} \otimes \mathbf{K})_{i,j} = \sum_m \sum_n I_{i+m,i+n} K_{m,n} \quad (2.68)$$

Indices m, n traverse the spatial dimensions of the kernel \mathbf{K} , while indices i, j span across the valid locations of the input image. Based on the aforementioned definitions, convolution is the same as cross-correlation, except from the fact that the kernel \mathbf{K} is flipped both horizontally and vertically. Usually in the context of deep learning, convolution and cross-correlation are identical operations and the flipping of the weight kernel is irrelevant and does not affect the end result.

Convolutional Layers and Forward Propagation

As mentioned above, the structural block of a CNN is the convolutional layer analogously to the fully connected layers of feed-forward networks. A convolutional layer is characterized by an input map \mathbf{I} , a set of filters \mathbf{K} and biases \mathbf{b} . Input images are characterized by their height H_{in} and width W_{in} as well as their channels C (for grayscale images $C = 1$, for RGB images $C = 3$), such that $\mathbf{I} \in \mathbb{R}^{H_{in} \times W_{in} \times C}$. The convolutional filters consist of an arbitrary number of kernels. As kernels and feature maps are characterized by three or even more dimensions, in all following equations, subscript indices are separated with commas in an attempt of rendering mathematical expressions more legible. Considering a set of D filters, then $\mathbf{K} \in \mathbb{R}^{k_1 \times k_2 \times C \times D}$ and biases $\mathbf{b} \in \mathbb{R}^D$, one for each filter. The output of a convolutional layer is referred to as a feature map $\mathbf{F} \in \mathbb{R}^{H_{out} \times W_{out} \times D}$, and its d th channel can be computed as follows:

$$F_{i,j,d} = (\mathbf{I} * \mathbf{K})_{i,j,d} = \sum_m \sum_n \sum_{c=1}^C K_{m,n,c,d} I_{i-m,j-n,c} + b_d \quad (2.69)$$

As a CNN is composed of several such convolutional layers together with non-linear layer activation functions, it is necessary to employ a notation suitable for describing the forward-pass operations, similar to the ones described by equations 2.4 for the MLP. More specifically for the l th convolutional layer ($1 \leq l \leq L$), the activation map resulting from the convolution operation is denoted as $\mathbf{X}^{(l)}$ and the corresponding output feature map after the application of the non-linear activation function as $\mathbf{Z}^{(l)}$, with size $H^{(l)} \times W^{(l)} \times C^{(l)}$. Additionally, the set of filters is denoted as $\mathbf{W}^{(l)}$ with size $k_1^{(l)} \times k_2^{(l)} \times C^{(l-1)} \times C^{(l)}$ and the corresponding biases as $\mathbf{b}^{(l)}$. With \mathbf{I} denoting the input tensor of size $H^{(0)} \times W^{(0)} \times C^{(0)}$, the forward propagation operation can be described by the following set of equations:

$$\begin{aligned} Z_{i,j}^{(0)} &= I_{i,j} \\ X_{i,j,d}^{(l)} &= \sum_m \sum_n \sum_{c=1}^{C^{(l-1)}} W_{m,n,c,d}^{(l)} Z_{i-m,j-n,c}^{(l-1)} + b_d^{(l)} \\ Z_{i,j}^{(l)} &= f(X_{i,j}^{(l)}) \end{aligned} \quad (2.70)$$

The indices i, j span across the input and output feature map spatial dimensions, c is the input channel index, d is the output channel index and m, n indices span across the spatial

dimensions of the filters. In the above formulation, the channel index is omitted from the first and third equations for simplicity as it is assumed that the corresponding operations are performed for each channel independently. Moreover, $f(\cdot)$ is a non-linear activation function which is applied element-wise on each activation map $\mathbf{X}^{(l)}$.

Feature Map and Receptive Field Output Dimensions

Another important concept relative to the convolutional operation is the receptive field. The receptive field is defined as the region in the input space that a particular feature of a convolutional layer is looking at. The receptive field size is directly dependent on the size of the convolutional kernels as well as another set of parameters that characterize the convolutional operation, as well as define the size of the output feature map. These parameters include the size of the convolutional kernel, the stride, padding and dilation of the convolution. The stride refers to the number of locations skipped during the sliding window multiplication. A stride of 1 means to pick slides a pixel apart, so basically every single slide, acting as a standard convolution. A stride of 2 means picking slides 2 pixels apart, skipping every other slide in the process, downsizing by roughly a factor of 2, and so on. Padding refers to the addition of extra “fake” pixels around the edges of the input feature map. This way, the kernel when sliding can allow the original edge pixels to be at its center, while extending into the fake pixels beyond the edge, producing an output the same size as the input. Lastly, dilation refers to the spaces inserted between kernel elements during the convolution operation. This type of operation is called dilated convolution and are used to cheaply increase the receptive field of output units without increasing the kernel size. More specifically, with (s_1, s_2) , (p_1, p_2) , (d_1, d_2) denoting the stride, padding and dilation rates respectively in the two separate spatial dimensions, and considering an input feature map with of size $H_{in} \times W_{in}$, the output feature map dimensions after the convolution with kernel of size $k_1 \times k_2$ will be:

$$\begin{aligned} H_{out} &= \left\lfloor \frac{H_{in} + 2p_1 - d_1(k_1 - 1) - 1}{s_1} \right\rfloor + 1 \\ W_{out} &= \left\lfloor \frac{W_{in} + 2p_2 - d_2(k_2 - 1) - 1}{s_2} \right\rfloor + 1 \end{aligned} \quad (2.71)$$

A more simplified setting of the convolutional operation considers square kernels ($k_1 = k_2 = k$) as well as equal strides s , dilations d and paddings p across the two spatial dimensions. Moreover, with $k^{(l)}$, $s^{(l)}$, $d^{(l)}$ denoting the kernel size, stride and dilation of the l th convolutional layer, then the corresponding receptive field size $r^{(l)}$, can be computed using the following formula:

$$\begin{aligned} \hat{k}^{(l)} &= k^{(l)} + (k^{(l)} - 1)(d^{(l)} - 1) \\ r^{(l)} &= r^{(l-1)} + (\hat{k}^{(l)} - 1) \prod_{k=1}^{l-1} s^{(k)} \end{aligned} \quad (2.72)$$

where $\hat{k}^{(l)}$ is the effective kernel size after the effect of a dilation rate $d^{(l)}$. The receptive field at the l th layer covers $(k^{(l)} - 1)s^{(l-1)}$ more pixels than the $(l - 1)$ th receptive field. However receptive field size is computed with respect to the input image. Therefore, the product $\prod_{k=1}^{l-1} s^{(k)}$ is equal to the distance between two consecutive features, as observed on the $(l - 1)$ th output map, with respect to the original input feature map.

Pooling Layers

In most modern CNN architectures, convolutional layers are often followed by another type of layer that performs a sample-based discretization process. These layers are referred to as pooling layers and their purpose is to provide invariance to small translations of their input as well as reduce the dimensionality of a given input representation. The aggregation strategy used in the pooling operation defines the type of the layer. Pooling operations include max pooling which is performed by applying a max-out filter over non-overlapping regions of the input feature map, average pooling which is performed by computing the mean over elements contained in the pooling blocks, or even L^2 -norm pooling, which involves the computation of the L^2 norm of the elements inside a pooling block. The most common pooling operations are max and average pooling. All of the aforementioned pooling operations may involve zero-padding. Considering the case of a square pooling kernel of size k , with strides s and dilations d along the two spatial dimensions, then the relationship between an input feature map \mathbf{X} and a corresponding output feature map \mathbf{Z} , for the case of max and average pooling, would be the following:

$$\begin{aligned} Z_{i,j}^{\max} &= \max_{m,n \in \{0,1,\dots,k-1\}} X_{si+dm,sj+dn} \\ Z_{i,j}^{\text{av}} &= \frac{1}{k^2} \sum_{m=0}^{k-1} \sum_{n=0}^{k-1} X_{si+dm,sj+dn} \end{aligned} \quad (2.73)$$

The channel indices are omitted as the pooling operations are performed for each on of the channels independently. The size of the output feature maps is again given by equations 2.71.

Backpropagation for Convolutional Layers

For the sake of simplifying numerical computations in the derivation of the backpropagation algorithm for convolutional layers, the indices c, d of equations 2.70 are omitted, as they only appear in the form of summations over all input or output channels. In that way, the aforementioned equations describing the forward propagation operations in convolutional layers are modified as follows:

$$\begin{aligned} X_{i,j}^{(l)} &= (\mathbf{W}^{(l)} * \mathbf{Z}^{(l-1)})_{i,j} + b^{(l)} = \sum_m \sum_n W_{m,n}^{(l)} Z_{i-m,j-n}^{(l-1)} + b^{(l)} \\ Z_{i,j}^{(l)} &= f(X_{i,j}^{(l)}) \end{aligned} \quad (2.74)$$

The derivation of the backpropagation equations will be carried out under the assumption of regular, non-dilated, non-strided convolutions, that is $s = d = 1$. In the setting of a CNN containing L convolutional layers, the network outputs coincide with the outputs of the last convolutional block, $\mathbf{Z}^{(L)}$. Let \mathcal{L} be a loss function applied on each one of the elements of $Z_{i,j}^{(L)}$. Backpropagation aims at computing the partial derivatives $\frac{\partial \mathcal{L}}{\partial W_{m,n}^{(l)}}$ for all indices m, n and layers l . The application of the chain rule for derivatives yields:

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial W_{m,n}^{(l)}} &= \sum_{m'} \sum_{n'} \frac{\partial \mathcal{L}}{\partial X_{m',n'}^{(l)}} \frac{\partial X_{m',n'}^{(l)}}{\partial W_{m,n}^{(l)}} \\ &= \sum_{m'} \sum_{n'} \delta_{m',n'}^{(l)} \frac{\partial X_{m',n'}^{(l)}}{\partial W_{m,n}^{(l)}} \end{aligned} \quad (2.75)$$

where $\delta_{m',n'}^{(l)} \equiv \partial\mathcal{L}/\partial X_{m',n'}^{(l)}$. Application of the convolution operation 2.74 in the above expression yields:

$$\begin{aligned} \frac{\partial X_{m',n'}^{(l)}}{\partial W_{m,n}^{(l)}} &= \frac{\partial}{\partial W_{m,n}^{(l)}} \left(\sum_{m''} \sum_{n''} W_{m'',n''}^{(l)} Z_{m'-m'',n'-n''}^{(l-1)} + b^{(l)} \right) \\ &= \frac{\partial}{\partial W_{m,n}^{(l)}} \left(\sum_{m''} \sum_{n''} W_{m'',n''}^{(l)} f(X_{m'-m'',n'-n''}^{(l-1)}) + b^{(l)} \right) \\ &= \frac{\partial}{\partial W_{m,n}^{(l)}} \left(W_{m,n}^{(l)} f(X_{m'-m,n'-n}^{(l-1)}) \right) = f(X_{m'-m,n'-n}^{(l-1)}) \end{aligned} \quad (2.76)$$

Substituting the result of equation 2.76 in equation 2.75, yields:

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial W_{m,n}^{(l)}} &= \sum_{m'} \sum_{n'} \delta_{m',n'}^{(l)} f(X_{m'-m,n'-n}^{(l-1)}) \\ &= \left(\boldsymbol{\delta}^{(l)} \otimes f(\mathbf{X}^{(l-1)}) \right)_{-m,-n} = \left\{ \boldsymbol{\delta}^{(l)} * \text{rot}_{180^\circ} [f(\mathbf{X}^{(l-1)})] \right\}_{m,n} \end{aligned} \quad (2.77)$$

where the operator $\text{rot}_{180^\circ}(\cdot)$ denotes the horizontal and vertical flipping of a kernel. In similar fashion, the chain rule can be applied for the computation of the $\boldsymbol{\delta}$ terms:

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial X_{m,n}^{(l)}} &= \sum_{m'} \sum_{n'} \frac{\partial \mathcal{L}}{\partial X_{m',n'}^{(l+1)}} \frac{\partial X_{m',n'}^{(l+1)}}{\partial X_{m,n}^{(l)}} \\ &= \sum_{m'} \sum_{n'} \delta_{m',n'}^{(l+1)} \frac{\partial X_{m',n'}^{(l+1)}}{\partial X_{m,n}^{(l)}} \end{aligned} \quad (2.78)$$

By again using the convolution operation and taking the partial derivatives w.r.t all the components, the term $\partial X_{m',n'}^{(l+1)}/\partial X_{m,n}^{(l)}$ can be written as follows:

$$\begin{aligned} \frac{\partial X_{m',n'}^{(l+1)}}{\partial X_{m,n}^{(l)}} &= \frac{\partial}{\partial X_{m,n}^{(l)}} \left(\sum_{m''} \sum_{n''} W_{m'',n''}^{(l+1)} Z_{m'-m'',n'-n''}^{(l)} + b^{(l+1)} \right) \\ &= \frac{\partial}{\partial X_{m,n}^{(l)}} \left(\sum_{m''} \sum_{n''} W_{m'',n''}^{(l+1)} f(X_{m'-m'',n'-n''}^{(l)}) + b^{(l+1)} \right) \\ &= \frac{\partial}{\partial X_{m,n}^{(l)}} \left(W_{m'-m,n'-n}^{(l+1)} f(X_{m,n}^{(l)}) \right) = W_{m'-m,n'-n}^{(l+1)} f'(X_{m,n}^{(l)}) \end{aligned} \quad (2.79)$$

Again, by substituting the above result into equation 2.78, the following recursive formula for the $\boldsymbol{\delta}$ terms is obtained:

$$\begin{aligned} \delta_{m,n}^{(l)} &= \frac{\partial \mathcal{L}}{\partial X_{m,n}^{(l)}} = f'(X_{m,n}^{(l)}) \sum_{m'} \sum_{n'} \delta_{m',n'}^{(l+1)} W_{m'-m,n'-n}^{(l+1)} \\ &= f'(X_{m,n}^{(l)}) \left(\boldsymbol{\delta}^{(l+1)} \otimes \mathbf{W}^{(l+1)} \right)_{-m,-n} \\ &= f'(X_{m,n}^{(l)}) \left[\boldsymbol{\delta}^{(l+1)} * \text{rot}_{180^\circ} (\mathbf{W}^{(l+1)}) \right]_{m,n} \end{aligned} \quad (2.80)$$

Combining equations 2.80 and 2.77, a recursive formula can be obtained, according to which, the partial derivatives of the loss function w.r.t the weights of some convolutional layer, are dependent on the $\boldsymbol{\delta}$ terms of both the current layer and all following layers as well as the activations of the preceding layer.

In addition there is the possibility of pooling layers being placed in between the convolutional layers. The most common instances of pooling layers are max and average-pooling layers. The application of pooling layers with square kernels of size k , results in a block of $k \times k$ values to be reduced to a single value that is either the maximum value located inside the block or the average of those values. Backpropagation through the pooling layers considers only the error w.r.t to that single maximum or average value of the surviving unit. To keep track of the surviving unit, its index is saved during the forward pass and used for gradient routing during backpropagation. Gradient routing differs relative to each type of pooling layer used. For the case of max-pooling, the gradient is assigned to the surviving unit and the rest of the units in the layer's pooling blocks are assigned with zero values as they do not contribute in error propagation. On the other hand, in the case of average pooling, the gradients are multiplied by a factor of $1/k^2$ and assigned to all the values within the pooling blocks.

It is assumed that the strides s and dilations d of the square pooling kernels are equal along the two spatial dimensions. Moreover, if index l corresponds to a pooling layer with an input feature map $\mathbf{Z}^{(l-1)} = \mathbf{X}^{(l)}$ and a corresponding output feature map $\mathbf{Z}^{(l)}$, then for the case of max pooling, gradients are computed in the following way:

$$\begin{aligned} Z_{i,j}^{(l)} &= X_{i^*,j^*}^{(l)} \text{ where } (i^*, j^*) = \underset{p,q \in \{0,1,\dots,k-1\}}{\operatorname{argmax}} X_{si+dp,sj+dq}^{(l)} \\ \frac{\partial \mathcal{L}}{\partial X_{m,n}^{(l)}} &= \begin{cases} \frac{\partial \mathcal{L}}{\partial Z_{i,j}^{(l)}} & \text{if } (m, n) = (i^*, j^*) \\ 0 & \text{otherwise} \end{cases} \end{aligned} \quad (2.81)$$

while the case of average pooling is as follows:

$$\frac{\partial \mathcal{L}}{\partial X_{m,n}^{(l)}} = \begin{cases} \frac{1}{k^2} \frac{\partial \mathcal{L}}{\partial Z_{i,j}^{(l)}} & \text{if } (m, n) = (si + dp, sj + dq) \text{ for any } p, q \in \{0, 1, \dots, k-1\} \\ 0 & \text{otherwise} \end{cases} \quad (2.82)$$

Fully Connected Layers

Modern CNN architectures are comprised of two basic parts of feature extraction and classification or regression. The feature extraction part is comprised of all the convolutional and pooling layers, as well as regularization layers such as dropout and batch-normalization layers. These types of regularization layers perform the same kind of operations as the ones described in section 2.3.5, with the sole difference of considering each feature map channel independently. For the first part, stacks of convolutional blocks constitute high-level feature extractors. After the last convolutional block, it is common practice to flatten the corresponding feature map into a single feature vector.

Subsequently, the classification and regression part of a CNN consists of fully connected (FC) layers. The objective of a fully connected layer is to take the results of the convolution/pooling process and use them to perform regression or classification, depending on the type of task. With the use of FC layers, the network aims at learning combinations of the extracted features, as neurons in a FC layer have full connections to all activations in the preceding and following layers. Additionally, the difference between FC and convolutional layers is that the neurons in the latter are connected only to a local region of the input and that many of the neurons in a convolutional layer share parameters. In addition, for any convolutional layer there is an FC layer that implements the same forward function. The weight matrix would be a large matrix that is mostly zero except for at certain blocks, due to local connectivity where the weights in many of the blocks are equal, due to parameter sharing.

Furthermore, FC layers can be converted to convolutional layers. More specifically, a feature volume of size $k \times k \times F$ where k corresponds to the spatial dimensions and F corresponds to the channel depth, can be converted to a feature vector of size D by applying a convolutional layer with kernel size equal to k , unitary stride s and dilation d and D filters, resulting in a output feature volume of size $1 \times 1 \times D$. It is worth noting that FC layers can only deal with input of a fixed size, due to the fact that it requires a certain amount of parameters to fully connect the input and output. On the other hand, convolutional layers just slide the same filters across the input, so they can basically deal with input of an arbitrary spatial size.

Skip Connections and Residual Architectures

Several problems may arise during the training of deep neural networks, like the problem of vanishing/exploding gradients as well as that of degradation. In the first case, gradient-related problems are encountered when gradient-based optimization and backpropagation are used. Backpropagation relies on the recursive application of the chain rule for the computation of the partial derivatives of some loss function w.r.t all the hidden layer activations. If the activation functions that are used have gradients in the range $(0, 1)$, then the application of the backpropagation algorithm will result in the multiplication of several small values during the computation of the error signals for the higher layers of the network. In that way, the gradients decrease exponentially w.r.t the number of layers resulting in a very slow training process or even inhibiting the training process completely, for the higher layers. On the other hand, when the derivatives of the activation functions are allowed to take larger values, there exists the risk of exploding gradients, respectively.

Even if deep neural networks start converging, there exists the problem of degradation. Degradation refers to the fact that variants of networks which utilize deeper architectures, namely a larger number of layers, exhibit higher error rates compared to their shallower counterparts. A deeper model is always expected to produce error rates that are no higher than that of their shallower counterparts as the former should be able to learn identity mappings for all the added layers, while maintaining the weights of all the layers from the learnt shallower model. The problem suggests that the solvers find difficulties in learning those identity mappings through multiple non-linear layers.

A solution to the aforementioned issues was provided by He et al. [48] in the form of residual blocks and skip connections. The output $\mathcal{H}(\mathbf{x})$ of a residual block effectively aggregate the output $\mathcal{F}(\mathbf{x})$ of one layer with the input \mathbf{x} of an earlier layer, that is $\mathcal{H}(\mathbf{x}) = \mathcal{F}(\mathbf{x}) + \mathbf{x}$. In that way, the stacked non-linear layers are expected to fit a different mapping $\mathcal{F}(\mathbf{x}) := \mathcal{H}(\mathbf{x}) - \mathbf{x}$ which is referred to as the residual function. Moreover, it is easier to optimize the residual function $\mathcal{F}(\mathbf{x})$ compared to the original mapping $\mathcal{H}(\mathbf{x})$, as in the case of an identity mapping being optimal, it would be easier to push the residual to zero rather than to learn an identity mapping with a stack of non-linear layers. An illustration of a simple residual block can be seen in figure 2.7. In order for the skip connections to work, the dimensions of $\mathcal{F}(\mathbf{x})$ and \mathbf{x} must have equal dimensions. Dimension equality can be enforced, if needed, through the multiplication of \mathbf{x} with a projection matrix \mathbf{W}_s , such that $\mathcal{H}(\mathbf{x}) = \mathcal{F}(\mathbf{x}) + \mathbf{W}_s \mathbf{x}$.

Additionally, residual blocks contribute in the preservation of gradients. Assuming L in total, stacked residual blocks, with $\mathbf{x}^{(l)}$ denoting the input of the l th layer, $\mathcal{F}(\mathbf{x}^{(l)})$ and $\mathcal{H}(\mathbf{x}^{(l)})$ denoting the common residual and output functions (all layer inputs must have equal

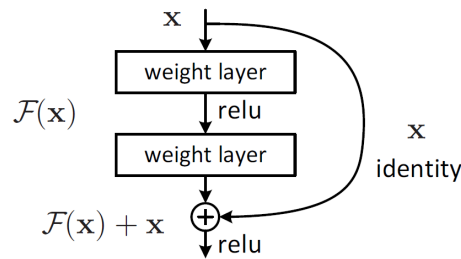


Figure 2.7: Illustration of a residual connection, the building block of residual learning. Source: [48].

dimensions), then the forward propagation is as follows:

$$\begin{aligned}
 x_j^{(L)} &= \mathcal{F}(x_j^{(L-1)}) + x_j^{(L-1)} \\
 &= \mathcal{F}(x_j^{(L-1)}) + \mathcal{F}(x_j^{(L-2)}) + x_j^{(L-2)} \\
 &\vdots \\
 &= x_j^{(l)} + \sum_{i=l}^{L-1} \mathcal{F}(x_j^{(i)})
 \end{aligned}$$

and given a loss function E , the partial derivative of the loss function w.r.t to some element $x_j^{(l)}$ can be computed using the following expression:

$$\begin{aligned}
 \frac{\partial E}{\partial x_j^{(l)}} &= \frac{\partial E}{\partial x_j^{(L)}} \frac{\partial x_j^{(L)}}{\partial x_j^{(l)}} \\
 &= \frac{\partial E}{\partial x_j^{(L)}} \frac{\partial}{\partial x_j^{(l)}} \left(x_j^{(l)} + \sum_{i=l}^{L-1} \mathcal{F}(x_j^{(i)}) \right) \\
 &= \frac{\partial E}{\partial x_j^{(L)}} \left(1 + \frac{\partial}{\partial x_j^{(l)}} \sum_{i=l}^{L-1} \mathcal{F}(x_j^{(i)}) \right)
 \end{aligned} \tag{2.83}$$

The last equation indicates that every $\partial E / \partial x_j^{(L)}$ comes in additive form rather than multiplicative, thus reducing the occurrence probability of the vanishing gradient problem. Furthermore, skip connections allow information. Furthermore, skip connections allow lower level semantic information, that is extracted in earlier layers to propagate intact through the network without becoming too abstract.

This type of identity shortcut connection constitutes the main idea behind one of the most popular modern CNN architectures, called Residual Neural Network (ResNet). The ResNet includes variants with 18, 34, 50, 101 and 152 layers. For the following analysis, convolutional layers are denoted as tuples of the form (k, k, f, s) , while pooling layers are denoted as (k, k, s) , where k is the square kernel size, f is the number of filters used and s is the stride along the two spatial dimensions. There are two types of residual blocks used, one that is two layers deep and is used in ResNet-18 & 34 and one that is three layers deep and is used in all the deeper variants. Stacks of these residual blocks constitute larger convolutional blocks. All variants are comprised of four convolutional blocks containing a different number of stacked residual blocks. In the first case, the residual block consists of two stacked $(3, 3, f, s)$ convolutional layers. In the deeper variants the residual blocks include a series of $(1, 1, f, s)$, $(3, 3, f, s)$ and $(1, 1, 4f, s)$ stacked convolutional layers. Whenever the output feature map size is halved, the

number of filters f of the convolutional layers doubles, ranging from 64 up to 512. Feature map downsampling is performed with strided convolutions. With the exception of the first convolutional block, the first layer of the first residual block of every other convolutional block, applies a stride $s = 2$. All other convolutional layers apply a stride $s = 1$. In all the variants, the first residual block is preceded by a $(7, 7, 64, 2)$ convolutional layer and a $(3, 3, 2)$ max pooling layer. Moreover, the networks utilize batch normalization layers right after every convolutional layer and before ReLU non-linearities. Lastly, the last convolutional block in all variants is followed by an average pooling layer, outputting a flattened feature vector that is to be fed to any potential subsequent FC layer.

The Dense Convolutional Network (DenseNet) constitutes another notable residual architecture, introduced by Huang et al. [53]. Without going into the same level of detail, instead of additive residual blocks, the DenseNet utilizes skip connections via concatenation so as to ensure maximum information flow between layers in the network. This is achieved by concatenating feature maps of all preceding layers directly with each other. More specifically, supposing that $\mathbf{x}^{(l)}$ is the output of the l th layer of the DenseNet and $\mathcal{H}^{(l)}(\cdot)$ is the corresponding non-linear mapping implemented at that layer, then the l th layer receives the feature maps of all preceding layers $\mathbf{x}^{(0)}, \mathbf{x}^{(1)}, \dots, \mathbf{x}^{(l-1)}$:

$$\mathbf{x}^{(l)} = \mathcal{H}^{(l)}([\mathbf{x}^{(0)}, \mathbf{x}^{(1)}, \dots, \mathbf{x}^{(l-1)}]) \quad (2.84)$$

where $[\mathbf{x}^{(0)}, \mathbf{x}^{(1)}, \dots, \mathbf{x}^{(l-1)}]$ is the channel-wise concatenation of the corresponding feature maps. In that way, utilizing feature map concatenation, instead of addition, exploits the networks full potential through feature reusability and leads to more compact models that are easier to train.

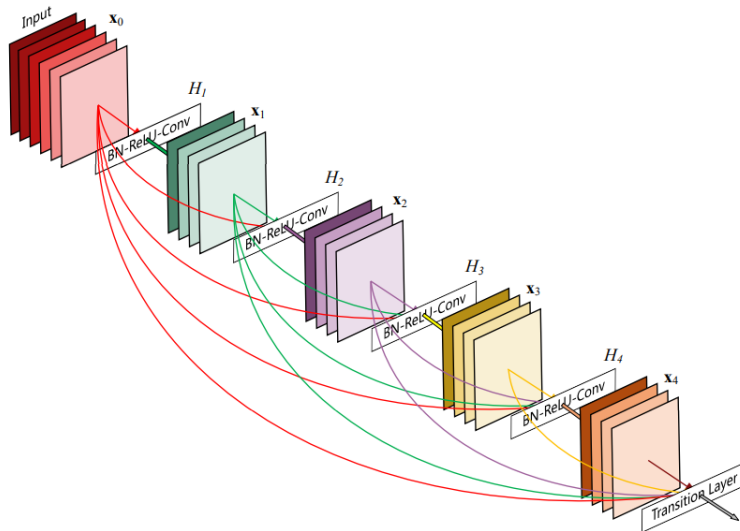


Figure 2.8: A 5-layer dense block with a growth rate of $k = 4$. Each layer takes all preceding feature-maps as input. Source: [53].

Similarly to the case of additive feature map aggregation, skip connections via concatenation require equal feature map sizes. As ResNets are comprised of residual blocks, DenseNets are divided into dense blocks, where the dimensions of the feature maps remain constant within a block, but the number of filters change between them. An illustration of a 5-layer dense block is provided in figure 2.8. Dense blocks are also divided into dense layers, with each dense layer producing a constant amount of output feature maps. The layers between

consecutive dense blocks are referred to as transition layers and perform downsampling by applying a batch normalization layer, a ReLU activation, a 1×1 convolutional layer with stride $s = 1$ and a 2×2 average pooling layer with stride $s = 2$, in that order. Lastly, it is worth noting that as feature maps are concatenated, the channel dimension increases at every layer. Assuming a dense block containing L dense layers and given that each dense layer function $\mathcal{H}^{(l)}(\cdot)$ produces k feature maps, then the output feature map corresponding to the l th dense layer will have $k^{(l)} = k^{(0)} + k(l - 1)$ channels. The parameter k is called growth rate and regulates the amount of information that is added to the network at each layer.

2.3.7 Graph Convolutional Networks

In the current section, we consider the problem of classifying nodes in a graph, including the case where labels are only available for a small subset of nodes. This problem can be formatted as graph-based learning, where label information is smoothed over the graph via some form of explicit graph-based regularization. Kipf et al. [57] proposed a scalable approach for supervised/semi-supervised learning on graph-structured data that is based on an efficient variant of convolutional neural networks which operate directly on graphs. In their work, they encoded the graph structure directly using a neural network model $f(\mathbf{X}, \mathbf{A})$, called Graph Convolutional Network (GCN), with \mathbf{X} denoting the input node feature vectors and \mathbf{A} denoting the graph adjacency matrix. A high-level representation of a GCN can be seen in figure 2.9. The GCN is trained on a supervised target L_0 for all nodes with labels, avoiding explicit graph-based regularization in the loss function. Conditioning $f(\cdot)$ on the adjacency matrix of the graph allows the model to distribute gradient information from the supervised loss L_0 and enables it to learn representations of nodes both with and without labels.

Forward Rule

The current section presents the forward propagation rule that is employed in GCNs. Considering a multi-layer GCN, the following layer-wise propagation rule is applied:

$$\mathbf{H}^{(l+1)} = h(\tilde{\mathbf{D}}^{-\frac{1}{2}} \tilde{\mathbf{A}} \tilde{\mathbf{D}}^{-\frac{1}{2}} \mathbf{H}^{(l)} \mathbf{W}^{(l)}) \quad (2.85)$$

where $\tilde{\mathbf{A}} = \mathbf{A} + \mathbb{I}_N$ is the adjacency matrix of the undirected graph with added self-connections and $\mathbf{H}^{(l+1)} \in \mathbb{R}^{N \times F}$ denotes the output of the l th hidden layer. Moreover, $\tilde{\mathbf{D}}$ is a diagonal matrix with elements $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$, $\mathbf{W}^{(l)} \in \mathbb{R}^{D \times F}$ is the weight matrix of the l th layer, $h(\cdot)$ denotes a non-linear activation function, $\mathbf{H}^{(l)} \in \mathbb{R}^{N \times D}$ is the matrix of activations in the l th layer, $\mathbf{H}^{(0)} = \mathbf{X}$. Without going into further details, the authors have proved that the aforementioned propagation rule is derived as a first-order approximation of spectral filters on graphs.

Node Classification

We consider a GCN with L hidden layers for node classification on a graph with symmetric adjacency matrix \mathbf{A} that can be either binary or weighted. After calculating the normalized adjacency matrix $\hat{\mathbf{A}} = \tilde{\mathbf{D}}^{-\frac{1}{2}} \tilde{\mathbf{A}} \tilde{\mathbf{D}}^{-\frac{1}{2}}$, the forward propagation rule is applied in order to obtain the hidden layer activation of the last GCN layer, namely $\mathbf{H}^{(L+1)} = f(\mathbf{X}, \mathbf{A})$. Given that the classification task features K mutually exclusive classes, the softmax activation function is applied on each row of the hidden layer activations $\mathbf{H}^{(L+1)} \in \mathbb{R}^{N \times K}$, producing a probability

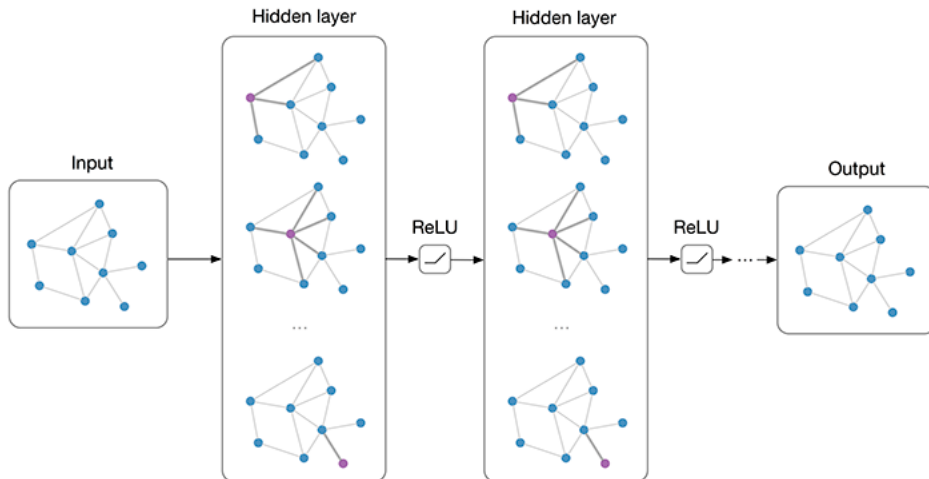


Figure 2.9: Illustration of a graph convolutional network (GCN) with two hidden layers and ReLU activation functions. Source: <https://github.com/tkipf/pygcn>.

matrix \mathbf{Z} containing the probability scores for all N graph nodes. For semi-supervised multi-class classification, a cross-entropy loss function is evaluated over all labeled instances:

$$\mathcal{L}_{\text{CE}} = - \sum_{l \in \mathcal{Y}_L} \sum_{k=1}^K Y_{lk} \ln Z_{lk} \quad (2.86)$$

where \mathcal{Y}_L is the set of node indices that have labels and Y_{lk} denotes the k th groundtruth label of the l th graph node. Classification can also be performed on a graph-level. However that would require some kind of attention mechanism that would be able to extract a single feature vector from all N node feature vectors of $\mathbf{H}^{(L+1)}$.

2.3.8 Recurrent Neural Networks

Recurrent neural networks or RNNs, as introduced by Rumelhart et al. [90], constitute a class of artificial neural networks related with sequential data modelling. Some of the most common applications involving sequential data include natural language processing, speech recognition and text classification. An RNN receives input vectors $\mathbf{x}^{(t)}$ where t denotes the time step index, ranging from 1 up to the sequence length. More specifically, given a sequence of observations $\mathbf{X} = \{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \dots, \mathbf{x}^{(\tau)}\}$ and a corresponding label set $\mathbf{Y} = \{\mathbf{y}^{(1)}, \mathbf{y}^{(2)}, \dots, \mathbf{y}^{(\tau)}\}$ of length τ , then a recurrent neural network aims at learning a mapping $f: \mathbf{X} \rightarrow \mathbf{Y}$. An important feature of most RNNs, is that they are capable of processing sequences of arbitrary length. Additionally, in practice RNNs operate on mini-batches of sequences but in order to simplify the expressions the batch indices are omitted.

Recurrent neural networks are used to model dynamical systems. An example of a classical dynamical system would have the following form:

$$\mathbf{s}^{(t)} = f(\mathbf{s}^{(t-1)}, \mathbf{x}^{(t)}, \boldsymbol{\theta}) \quad (2.87)$$

where $\boldsymbol{\theta}$ is a set of parameters, $\mathbf{s}^{(t)}$ denotes the state of the model and $\mathbf{x}^{(t)}$ is an external input signal at time step t . The above equation is recurrent as the state of the model at some time step t is defined w.r.t to the exact previous state at time step $t - 1$. Given a finite number of time steps τ , then the recurrent model can be unfolded by applying the above equation $\tau - 1$ times, meaning that state $\mathbf{s}^{(\tau)}$ can be expressed w.r.t the parameters $\boldsymbol{\theta}$, the input vector sequence $\mathbf{x}^{(\tau)}, \mathbf{x}^{(\tau-1)}, \dots, \mathbf{x}^{(1)}$ and potentially the initial model state $\mathbf{s}^{(1)}$. This

procedure removes any kind of recurrence leading to an expression that can be described by a directed, acyclic and unfolded computational graph across time.

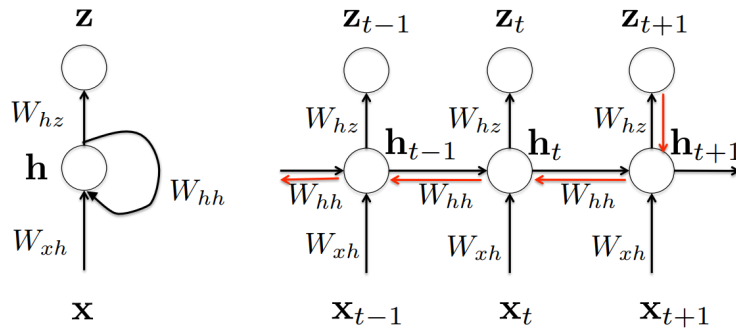


Figure 2.10: It is a RNN example: the left recursive description for RNNs, and the right is the corresponding extended RNN model in a time sequential manner. Source: [19].

Unidirectional Recurrent Neural Networks

The most widely used design pattern for RNNs, considers a dynamical system where the state at each time step is dependent on the previous state and the current observation $\mathbf{x}^{(t)} \in \mathbb{R}^d$. Moreover, the states of the network are characterized as “hidden” and as a convention, they are denoted as $\mathbf{h}^{(t)} \in \mathbb{R}^m$. This type of RNN features recurrent connections between hidden states and also produces an output $\mathbf{y}^{(t)} \in \mathbb{R}^q$ at each time step. Therefore, the model maps an input sequence to an output sequence of the same length. The equations describing the operations of this type of RNN, are the following:

$$\begin{aligned}
 \mathbf{a}^{(t)} &= \mathbf{W}_{hh}\mathbf{h}^{(t-1)} + \mathbf{W}_{xh}\mathbf{x}^{(t)} + \mathbf{b}_h \\
 \mathbf{h}^{(t)} &= \tanh(\mathbf{a}^{(t)}) \\
 \mathbf{z}^{(t)} &= \mathbf{W}_{hz}\mathbf{h}^{(t)} + \mathbf{b}_z \\
 \hat{\mathbf{y}}^{(t)} &= g(\mathbf{z}^{(t)})
 \end{aligned} \tag{2.88}$$

where $\mathbf{W}_{xh} \in \mathbb{R}^{m \times d}$, $\mathbf{W}_{hh} \in \mathbb{R}^{m \times m}$, $\mathbf{W}_{hz} \in \mathbb{R}^{q \times m}$ denote the input to hidden, hidden to hidden and hidden to output weights respectively, $\mathbf{b}_h \in \mathbb{R}^m$, $\mathbf{b}_z \in \mathbb{R}^q$ denote the hidden and input layer biases, while $g(\cdot)$ denotes the output unit activation function. Both the recursive as well as the unfolded representations of an RNN are illustrated in figure 2.10. Typical options for the final non-linearity are the softmax function for multi-class categorical prediction tasks, the sigmoid function for the prediction of independent Bernoulli variables and the identity function in the case of regression. The input size is kept constant, while the same parameters and non-linear functions are used at each time step, resulting in a model that can operate on sequences of variable length.

Backpropagation Through Time

Given an input sequence $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \dots, \mathbf{x}^{(t)}$ and a corresponding sequence of target labels $\mathbf{y}^{(1)}, \mathbf{y}^{(2)}, \dots, \mathbf{y}^{(t)}$, then the total loss can be computed as the sum of losses over all time steps. Considering the task of multi-class classification, the loss $\mathcal{L}^{(t)}$ at each time step will be the negative log-likelihood of $\mathbf{y}^{(t)}$ given all input vectors up to time step t . In that way the total

loss will be equal to:

$$\mathcal{L} = \sum_t \mathcal{L}^{(t)} = - \sum_t \mathbf{y}^{(t)} \ln \hat{\mathbf{y}}^{(t)} = - \sum_t \sum_i \llbracket y_i^{(t)} = 1 \rrbracket \ln \hat{y}_i^{(t)} \quad (2.89)$$

where i corresponds to each element of the output and one-hot encoded target vectors. Based on the task, it follows that $g(\cdot)$ would be the softmax function.

Training RNNs involves the application of a variant of the classic backpropagation algorithm which was analyzed in sections 2.3.4 and 2.3.6 and is referred to as backpropagation through time (BPTT). Conceptually, BPTT works by unrolling all timesteps with each timestep having one input, one copy of the network, and one output. Errors are then calculated and accumulated for each timestep. The network is rolled back up and the weights are updated. The trainable parameters are comprised of the weight matrices \mathbf{W}_{xh} , \mathbf{W}_{hh} , \mathbf{W}_{hz} and the biases \mathbf{b}_h , \mathbf{b}_z . Therefore, the end goal is to compute the gradients of the loss function \mathcal{L} w.r.t to each one of the elements of the aforementioned trainable parameters. The aforementioned partial derivatives will be expressed relative to the sequence of nodes of the unfolded computational graph indexed by t , including $\mathbf{x}^{(t)}$, $\mathbf{h}^{(t)}$, $\mathbf{z}^{(t)}$ as well as the gradients of the loss function \mathcal{L} w.r.t them.

Assuming the same loss formulation as described in the previous section, the gradient of \mathcal{L} relative to each element of the output vectors $\mathbf{z}^{(t)}$, is as follows:

$$\begin{aligned} \left(\frac{\partial \mathcal{L}}{\partial \mathbf{z}^{(t)}} \right)_j &= \frac{\partial \mathcal{L}}{\partial z_j^{(t)}} = \frac{\partial \mathcal{L}}{\partial \mathcal{L}^{(t)}} \frac{\partial \mathcal{L}^{(t)}}{\partial z_j^{(t)}} \\ &= \frac{\partial}{\partial z_j^{(t)}} \left(\sum_i \llbracket y_i^{(t)} = 1 \rrbracket \ln \hat{y}_i^{(t)} \right) = - \sum_i \llbracket y_i^{(t)} = 1 \rrbracket \frac{1}{\hat{y}_i^{(t)}} \frac{\partial \hat{y}_i^{(t)}}{\partial z_j^{(t)}} \\ &= - \sum_i \llbracket y_i^{(t)} = 1 \rrbracket \frac{1}{\hat{y}_i^{(t)}} \frac{\partial}{\partial z_j^{(t)}} (\text{softmax}(z_i^{(t)})) \\ &= - \sum_i \llbracket y_i^{(t)} = 1 \rrbracket \frac{1}{\hat{y}_i^{(t)}} \hat{y}_i^{(t)} (\delta_{ij} - \hat{y}_j^{(t)}) = - \sum_i \llbracket y_i^{(t)} = 1 \rrbracket (\delta_{ij} - \hat{y}_j^{(t)}) \\ &= - \sum_i \llbracket y_i^{(t)} = 1 \rrbracket \llbracket i = j \rrbracket + \sum_i \llbracket y_i^{(t)} = 1 \rrbracket \hat{y}_j^{(t)} = \hat{y}_j^{(t)} - \llbracket y_j^{(t)} = 1 \rrbracket \end{aligned} \quad (2.90)$$

Assuming input and output sequences of length τ , then at the last time step, $\mathbf{h}^{(\tau)}$ influences only $\mathbf{z}^{(\tau)}$, therefore its gradient is as follows:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{h}^{(\tau)}} = \frac{\partial \mathcal{L}}{\partial \mathbf{z}^{(\tau)}} \frac{\partial \mathbf{z}^{(\tau)}}{\partial \mathbf{h}^{(\tau)}} = \frac{\partial \mathcal{L}}{\partial \mathbf{z}^{(\tau)}} \frac{\partial (\mathbf{W}_{hz} \mathbf{h}^{(\tau)} + \mathbf{b}_z)}{\partial \mathbf{h}^{(\tau)}} = \frac{\partial \mathcal{L}}{\partial \mathbf{z}^{(\tau)}} \mathbf{W}_{hz} \quad (2.91)$$

For all other time steps $t < \tau$, the hidden node $\mathbf{h}^{(t)}$ affects the current output node $\mathbf{z}^{(t)}$ as well as the following hidden node $\mathbf{h}^{(t+1)}$ in the computational graph, therefore its gradient is as follows:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{h}^{(t)}} = \sum_{t \leq i \leq \tau} \frac{\partial \mathcal{L}}{\partial \mathbf{z}^{(i)}} \frac{\partial \mathbf{z}^{(i)}}{\partial \mathbf{h}^{(i)}} \frac{\partial \mathbf{h}^{(i)}}{\partial \mathbf{h}^{(t)}} \quad (2.92)$$

As mentioned above, the weight parameters and biases are shared among all time steps. In that way, in order to accumulate the gradients over all time steps, a copy of each one of the parameters $\mathbf{W}_{xh}^{(t)}$, $\mathbf{W}_{hh}^{(t)}$, $\mathbf{W}_{hz}^{(t)}$, $\mathbf{b}_h^{(t)}$, $\mathbf{b}_z^{(t)}$ can be considered for each time step t . For the biases,

the corresponding gradients are as follows:

$$\begin{aligned}
 \frac{\partial \mathcal{L}}{\partial \mathbf{b}_z} &= \sum_t \frac{\partial \mathcal{L}}{\partial \mathbf{z}^{(t)}} \frac{\partial \mathbf{z}^{(t)}}{\partial \mathbf{b}_z^{(t)}} = \sum_t \frac{\partial \mathcal{L}}{\partial \mathbf{z}^{(t)}} \frac{\partial (\mathbf{W}_{hz}^{(t)} \mathbf{h}^{(t)} + \mathbf{b}_z^{(t)})}{\partial \mathbf{b}_z^{(t)}} = \sum_t \frac{\partial \mathcal{L}}{\partial \mathbf{z}^{(t)}} \\
 \frac{\partial \mathcal{L}}{\partial \mathbf{b}_h} &= \sum_t \frac{\partial \mathcal{L}}{\partial \mathbf{h}^{(t)}} \frac{\partial \mathbf{h}^{(t)}}{\partial \mathbf{a}^{(t)}} \frac{\partial \mathbf{a}^{(t)}}{\partial \mathbf{b}_h^{(t)}} = \sum_t \frac{\partial \mathcal{L}}{\partial \mathbf{h}^{(t)}} \frac{\partial (\tanh(\mathbf{a}^{(t)}))}{\partial \mathbf{a}^{(t)}} \frac{\partial (\mathbf{W}_{hh}^{(t)} \mathbf{h}^{(t-1)} + \mathbf{W}_{xh}^{(t)} \mathbf{x}^{(t)} + \mathbf{b}_h)}{\partial \mathbf{b}_h^{(t)}} \\
 &= \sum_t \frac{\partial \mathcal{L}}{\partial \mathbf{h}^{(t)}} \text{diag}[1 - (\mathbf{h}^{(t)})^2]
 \end{aligned} \tag{2.93}$$

Subsequently, the gradients of the loss function with respect to the weight matrices are more complicated to compute. Firstly, the derivative of the scalar loss function w.r.t to a weight matrix needs to be decomposed into simpler derivatives with respect to vectors or scalars. The case of \mathbf{W}_{hz} is as follows:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{W}_{hz}} = \left[\left(\frac{\partial \mathcal{L}}{\partial \mathbf{w}_{hz,1}} \right)^\top, \dots, \left(\frac{\partial \mathcal{L}}{\partial \mathbf{w}_{hz,m}} \right)^\top \right]^\top \tag{2.94}$$

where $\mathbf{W}_{hz} = [\mathbf{w}_{hz,1}^\top, \dots, \mathbf{w}_{hz,m}^\top]^\top$ and $\mathbf{w}_{hz,i}^\top$ is the i th row of the weight matrix \mathbf{W}_{hz} . Each row of the gradient can be computed as follows:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{w}_{hz,j}} = \sum_t \frac{\partial \mathcal{L}}{\partial z_j^{(t)}} \frac{\partial z_j^{(t)}}{\partial \mathbf{w}_{hz,j}^{(t)}} = \sum_t \frac{\partial \mathcal{L}}{\partial z_j^{(t)}} \frac{\partial [(\mathbf{w}_{hz,j}^{(t)})^\top \mathbf{h}^{(t)} + b_{z,j}]}{\partial \mathbf{w}_{hz,j}^{(t)}} = \sum_t \frac{\partial \mathcal{L}}{\partial z_j^{(t)}} \mathbf{h}^{(t)} \tag{2.95}$$

Substituting the above expression in every row of the weight matrix \mathbf{W}_{hz} in equation 2.94, yields the following result:

$$\begin{aligned}
 \frac{\partial \mathcal{L}}{\partial \mathbf{W}_{hz}} &= \left[\left(\sum_t (\mathbf{h}^{(t)})^\top \frac{\partial \mathcal{L}}{\partial z_1^{(t)}} \right), \dots, \left(\sum_t (\mathbf{h}^{(t)})^\top \frac{\partial \mathcal{L}}{\partial z_m^{(t)}} \right) \right]^\top \\
 &= \sum_t \left(\frac{\partial \mathcal{L}}{\partial \mathbf{z}^{(t)}} \right)^\top (\mathbf{h}^{(t)})^\top
 \end{aligned} \tag{2.96}$$

Following the exact same procedure, the gradient of the loss function w.r.t the weight matrix \mathbf{W}_{hh} can be computed as follows:

$$\begin{aligned}
 \frac{\partial \mathcal{L}}{\partial \mathbf{w}_{hh,j}} &= \sum_t \frac{\partial \mathcal{L}}{\partial h_j^{(t)}} \frac{\partial h_j^{(t)}}{\partial a_j^{(t)}} \frac{\partial a_j^{(t)}}{\partial \mathbf{w}_{hh,j}} \\
 &= \sum_t \frac{\partial \mathcal{L}}{\partial h_j^{(t)}} [1 - (h_j^{(t)})^2] \frac{\partial [(\mathbf{w}_{hh,j}^{(t)})^\top \mathbf{h}^{(t-1)} + (\mathbf{w}_{xh,j}^{(t)})^\top \mathbf{x}^{(t)} + b_{h,j}]}{\partial \mathbf{w}_{hh,j}} \\
 &= \sum_t \frac{\partial \mathcal{L}}{\partial h_j^{(t)}} [1 - (h_j^{(t)})^2] \mathbf{h}^{(t-1)} \\
 \frac{\partial \mathcal{L}}{\partial \mathbf{W}_{hh}} &= \sum_t \text{diag}[1 - (\mathbf{h}^{(t)})^2] \left(\frac{\partial \mathcal{L}}{\partial \mathbf{h}^{(t)}} \right)^\top (\mathbf{h}^{(t-1)})^\top
 \end{aligned} \tag{2.97}$$

In similar fashion, the gradient of the loss function w.r.t the weight matrix \mathbf{W}_{xh} can be

computed as follows:

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \mathbf{w}_{xh,j}} &= \sum_t \frac{\partial \mathcal{L}}{\partial h_j^{(t)}} \frac{\partial h_j^{(t)}}{\partial a_j^{(t)}} \frac{\partial a_j^{(t)}}{\partial \mathbf{w}_{xh,j}} \\
&= \sum_t \frac{\partial \mathcal{L}}{\partial h_j^{(t)}} [1 - (h_j^{(t)})^2] \frac{\partial [(\mathbf{w}_{hh,j}^{(t)})^\top \mathbf{h}^{(t-1)} + (\mathbf{w}_{xh,j}^{(t)})^\top \mathbf{x}^{(t)} + b_{h,j}]}{\partial \mathbf{w}_{xh,j}} \\
&= \sum_t \frac{\partial \mathcal{L}}{\partial h_j^{(t)}} [1 - (h_j^{(t)})^2] \mathbf{x}^{(t)} \\
\frac{\partial \mathcal{L}}{\partial \mathbf{W}_{xh}} &= \sum_t \text{diag}[1 - (\mathbf{h}^{(t)})^2] \left(\frac{\partial \mathcal{L}}{\partial \mathbf{h}^{(t)}} \right)^\top (\mathbf{x}^{(t)})^\top
\end{aligned} \tag{2.98}$$

Bidirectional Recurrent Neural Networks

A unidirectional RNN whose forward operations are described by equations 2.88, preserves information of the past because the only inputs it has seen are from the past. The hidden state $\mathbf{h}^{(t)}$ at time step t directly depends on the state $\mathbf{h}^{(t-1)}$ of the previous time step and the current observation $\mathbf{x}^{(t)}$. By unfolding the computational graph and removing the recurrence, it becomes evident that the hidden state at time step t depends indirectly on the entire sequence of past observations up to that point. The hidden to hidden connections traverse the computational graph forwards in time, namely from past to present. Subsequently, unidirectional RNNs are limited in the sense that at any particular node, can only have access to the past information and hence the output can only be generated based on what the network has seen.

Bidirectional RNNs (BRNNs) [92], constitute a direct extension of their unidirectional counterparts. They introduce a second hidden layer in which the hidden to hidden connections flow in reverse temporal order, that is from future to present. In that way, a BRNN includes two sub-networks, one that accesses information in forward direction and another which accesses it in the reverse direction. These networks have access to the past as well as the future information and hence the output is generated from both the past and future context. This type of networks has been successfully applied on tasks such as image captioning, language translation, part-of-speech tagging and protein structure prediction.

Considering a BRNN, the forward and backward hidden states are assumed to be $\mathbf{h}_f^{(t)}, \mathbf{h}_b^{(t)} \in \mathbb{R}^m$ respectively, where m indicates the number of hidden units. Given observations $\mathbf{x}^{(t)} \in \mathbb{R}^d$ and predicted outputs $\hat{\mathbf{y}}^{(t)} \in \mathbb{R}^q$ at each time step t and assuming corresponding input to hidden, $\mathbf{W}_{xh}^{(f)}, \mathbf{W}_{xh}^{(b)} \in \mathbb{R}^{m \times d}$, hidden to hidden $\mathbf{W}_{hh}^{(f)}, \mathbf{W}_{hh}^{(b)} \in \mathbb{R}^{m \times m}$ and hidden to output $\mathbf{W}_{hz} \in \mathbb{R}^{q \times 2m}$ weight matrices as well as biases $\mathbf{b}_h^{(f)}, \mathbf{b}_h^{(b)} \in \mathbb{R}^m$ and $\mathbf{b}_z \in \mathbb{R}^q$, where the superscripts f and b denote the forward and backward operations respectively, then the model's dynamics can be described by the following operations:

$$\begin{aligned}
\mathbf{h}_f^{(t)} &= \phi(\mathbf{W}_{hh}^{(f)} \mathbf{h}_f^{(t-1)} + \mathbf{W}_{xh}^{(f)} \mathbf{x}^{(t)} + \mathbf{b}_h^{(f)}) \\
\mathbf{h}_b^{(t)} &= \phi(\mathbf{W}_{hh}^{(b)} \mathbf{h}_b^{(t-1)} + \mathbf{W}_{xh}^{(b)} \mathbf{x}^{(t)} + \mathbf{b}_h^{(b)})
\end{aligned} \tag{2.99}$$

The non-linear activation $\phi(\cdot)$ can be chosen to be the hyperbolic tangent similarly to the unidirectional case. The forward and backward hidden states are concatenated into a single hidden state vector $\mathbf{h}^{(t)} \in \mathbb{R}^{2m}$. The final predicted output at $\hat{\mathbf{y}}^{(t)}$ at time step t , is computed as follows:

$$\hat{\mathbf{y}}^{(t)} = g(\mathbf{W}_{hz} \mathbf{h}^{(t)} + \mathbf{b}_z) \tag{2.100}$$

where $g(\cdot)$ is the standard output activation function. The corresponding gradient derivation with through the application of the BPTT algorithm will be omitted. However, it is worth noting that for example, the backward pass procedure is slightly more complicated because the update of state and output neurons can no longer be done one at a time. In this case, supposing input and output sequences of length T , the BPTT algorithm will first perform the backward pass for the forward states (from $t = T$ to $t = 1$) and then for the backward states (from $t = 1$ to $t = T$).

Deep Recurrent Neural Networks

Up to this point, the recurrent neural network architectures which have been discussed include unidirectional and bidirectional RNNs with one hidden layer (forward or backward). As MLPs utilize multiple stacked fully connected layers and CNNs employ blocks of consecutive convolutional layers, RNNs can have multiple hidden layer as well where each hidden state is continuously passed to both the next timestep of the current layer and the current timestep of the next layer. This family of recurrent models is referred to as Deep RNNs. Assuming input vectors $\mathbf{x}^{(t)} \in \mathbb{R}^d$ and hidden states of the l th hidden layer $\mathbf{h}_l^{(t)} \in \mathbb{R}^m$, where m is the number of hidden units as well as output prediction vector $\hat{\mathbf{y}}^{(t)} \in \mathbb{R}^q$ at time step t , with $f_l(\cdot)$ being the hidden unit activation function for the l th layer, then the forward propagation equations for the first and all subsequent layers are as follows:

$$\begin{aligned}\mathbf{h}_1^{(t)} &= f_1(\mathbf{x}^{(t)}, \mathbf{h}_1^{(t-1)}) \\ \mathbf{h}_l^{(t)} &= f_l(\mathbf{h}_{l-1}^{(t)}, \mathbf{h}_l^{(t-1)})\end{aligned}\tag{2.101}$$

For all layers $1 < l \leq L$, the hidden state of the previous layer is used in its place. The output layer depends only on the hidden states of the last hidden layer. If $g(\cdot)$ is the output function, then the output vector $\hat{\mathbf{y}}^{(t)}$ at time step t is as follows:

$$\hat{\mathbf{y}}^{(t)} = g(\mathbf{h}_L^{(t)})\tag{2.102}$$

In deep RNNs, the number of hidden layers L as well as the number of hidden units m constitute hyperparameters for the model. Additionally, bidirectional RNN hidden layers can also be utilized, in which each hidden layer will be comprised of a forward and backward hidden states, that will be concatenated and fed as input into the subsequent hidden layers.

Long Term Dependencies

The long-term dependency problem refers to the fact that the computation of the gradients of some loss function relative to the network's parameters depends on the computation of gradients w.r.t to hidden state vectors over all past time steps within a given sequence. As indicated by equation 2.92, given an input sequence of length τ , the computation of $\frac{\partial \mathcal{L}}{\partial \mathbf{h}^{(t)}}$ for time steps $t < \tau$, is expressed in terms of Jacobian matrices $\frac{\partial \mathbf{h}^{(t+1)}}{\partial \mathbf{h}^{(k)}}$. Further expansion of the above partial derivative yields the following:

$$\begin{aligned}\frac{\partial \mathbf{h}^{(t+1)}}{\partial \mathbf{h}^{(k)}} &= \prod_{k \leq i \leq t} \frac{\partial \mathbf{h}^{(i+1)}}{\partial \mathbf{h}^{(i)}} = \prod_{k \leq i \leq t} \frac{\partial \mathbf{h}^{(i+1)}}{\partial \mathbf{a}^{(i+1)}} \frac{\partial \mathbf{a}^{(i+1)}}{\partial \mathbf{h}^{(i)}} \\ &= \prod_{k \leq i \leq t} \frac{\partial(\tanh(\mathbf{a}^{(i+1)}))}{\partial \mathbf{a}^{(i+1)}} \frac{\partial(\mathbf{W}_{hh}\mathbf{h}^{(i)} + \mathbf{W}_{xh}\mathbf{x}^{(i+1)} + \mathbf{b}_h)}{\partial \mathbf{h}^{(i)}} \\ &= \prod_{k \leq i \leq t} \text{diag}[1 - (\mathbf{h}^{(i+1)})^2] \mathbf{W}_{hh}\end{aligned}\tag{2.103}$$

Therefore, the Jacobian matrix $\frac{\partial \mathbf{h}^{(t+1)}}{\partial \mathbf{h}^{(k)}}$ indicates matrix multiplication over part of the entire sequence. During BPTT, the network backpropagates gradients over a long sequence and if the aforementioned matrix multiplication involves small values, then the gradient values will shrink layer by layer and will eventually vanish after a few time steps. On the other hand, there is the possibility of exploding gradients, which is respectively attributed to large values in the matrix multiplication.

More specifically, if eigendecomposition is applied on the Jacobian matrix $\frac{\partial \mathbf{h}^{(t+1)}}{\partial \mathbf{h}^{(k)}}$, then eigenvalues $\lambda_1, \lambda_2, \dots, \lambda_m$ with $|\lambda_1| > |\lambda_2| > \dots > |\lambda_m|$ and corresponding eigenvectors $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_m$ will be obtained where m is the dimensionality of the hidden states. Any change on the hidden state in the direction of a vector \mathbf{v}_j has the effect of multiplying the change with the eigenvalue associated with this eigenvector. The product of these Jacobians implies that subsequent time steps, will result in scaling the change with a factor equivalent to λ_j^t , where λ_j^t represents the i th eigenvalue raised to the power of the current time step t . The terms λ_j^t will grow or decay exponentially fast as $t \rightarrow \infty$ and λ_1^t will dominate the final result. According to [81], if the $\lambda_1 < 1$ then the gradients will exponentially shrink and eventually vanish, while if $\lambda_1 > 1$, the gradients will exponentially grow and eventually explode.

Long Short-Term Memory

As described in the previous sections, all RNNs have feedback loops in the recurrent layer. This allows them to maintain information in memory over time. However, it can be difficult to train vanilla RNNs to solve problems that require learning long-term temporal dependencies. This is because the gradient of the loss function may decay or grow exponentially over time, resulting in the vanishing and exploding gradient problems, respectively.

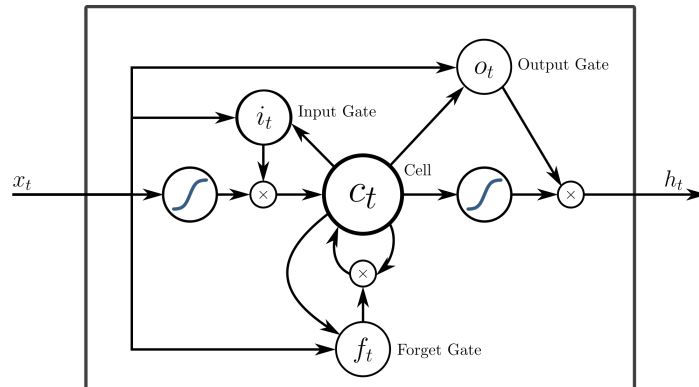


Figure 2.11: Structure example of a Long Short-Term Memory (LSTM) block. Adapted from [44].

Long Short-Term Memory networks (LSTM) [51] constitute a type of RNN that utilize special units in addition to standard hidden state and output units, resulting in better handling of long sequential dependencies. LSTM units include a memory cell that can maintain information in memory for long periods of time. A set of gates is used to control when information enters the memory, when its output, and when its forgotten. This architecture lets them learn longer-term dependencies. LSTMs belong in a family of models which are referred to as gated RNNs. There exist many gated RNN variants, with the most prominent being the Gated Recurrent Unit (GRU) along side the LSTM. GRUs are similar to LSTMs, but use a simplified structure. They also use a set of gates to control the flow of information, while they don't use separate memory cells, and they use fewer gates. The structure of a typical LSTM block can be seen in figure 2.11. There is no conclusive evidence suggesting that

one consistently performs better than the other. By a rule of thumb, LSTMs constitute the primary choice when it comes to learning long dependencies, with GRUs being a secondary but less computationally expensive choice. The current analysis will focus only on LSTMs.

Given an LSTM, at every time step t , the input is $\mathbf{x}^{(t)} \in \mathbb{R}^d$ and the hidden state is $\mathbf{h}^{(t)} \in \mathbb{R}^m$. The memory control gates include the forget gate $\mathbf{f}^{(t)} \in \mathbb{R}^m$, the input gate $\mathbf{i}^{(t)} \in \mathbb{R}^m$ and the output gate $\mathbf{o}^{(t)} \in \mathbb{R}^m$. The gates are calculated as follows:

$$\begin{aligned}\mathbf{i}^{(t)} &= \sigma(\mathbf{W}_{hi}\mathbf{h}^{(t-1)} + \mathbf{W}_{xi}\mathbf{x}^{(t)} + \mathbf{b}_i) \\ \mathbf{f}^{(t)} &= \sigma(\mathbf{W}_{hf}\mathbf{h}^{(t-1)} + \mathbf{W}_{xf}\mathbf{x}^{(t)} + \mathbf{b}_f) \\ \mathbf{o}^{(t)} &= \sigma(\mathbf{W}_{ho}\mathbf{h}^{(t-1)} + \mathbf{W}_{xo}\mathbf{x}^{(t)} + \mathbf{b}_o)\end{aligned}\tag{2.104}$$

where $\mathbf{W}_{hi}, \mathbf{W}_{hf}, \mathbf{W}_{ho} \in \mathbb{R}^{m \times m}$ and $\mathbf{W}_{xi}, \mathbf{W}_{xf}, \mathbf{W}_{xo} \in \mathbb{R}^{m \times d}$ are the weight matrices and $\mathbf{b}_i, \mathbf{b}_f, \mathbf{b}_o \in \mathbb{R}^m$ are the corresponding biases, while $\sigma(\cdot)$ denotes the sigmoid function, thus the output values of each gate is in the range $[0, 1]$. The next part of the LSTM is related with the memory cell. The candidate memory cell $\tilde{\mathbf{c}}^{(t)} \in \mathbb{R}^m$ is calculated similarly to the control gates but with a hyperbolic tangent as the non-linear activation function so as to obtain a value in the range $[-1, 1]$. Its computation is as follows:

$$\tilde{\mathbf{c}}^{(t)} = \tanh(\mathbf{W}_{hc}\mathbf{h}^{(t-1)} + \mathbf{W}_{xc}\mathbf{x}^{(t)} + \mathbf{b}_c)\tag{2.105}$$

where $\mathbf{W}_{hc} \in \mathbb{R}^{m \times m}$, $\mathbf{W}_{xc} \in \mathbb{R}^{m \times d}$ are the weight matrices and $\mathbf{b}_c \in \mathbb{R}^m$ is the corresponding bias. The parameter $\mathbf{i}^{(t)}$ governs how much of new data is taken into account via $\tilde{\mathbf{c}}_t$ and the forget gate $\mathbf{f}^{(t)}$ addresses how much of the old memory cell content $\mathbf{c}^{(t-1)}$ is retained. Using pointwise multiplication the corresponding update equation is as follows:

$$\mathbf{c}^{(t)} = \mathbf{f}^{(t)} \odot \mathbf{c}^{(t-1)} + \mathbf{i}^{(t)} \odot \tilde{\mathbf{c}}^{(t)}\tag{2.106}$$

If the forget gate is always approximately 1 and the input gate is always approximately 0, the past memory cells $\mathbf{c}^{(t-1)}$ will be saved over time and passed to the current timestep. This design was introduced to alleviate the vanishing gradient problem and to better capture dependencies for time series with long range dependencies. The hidden states $\mathbf{h}^{(t)}$ of the LSTM unit and are derived as gated versions of the hyperbolic tangent of the memory cell, namely:

$$\mathbf{h}^{(t)} = \mathbf{o}^{(t)} \odot \tanh(\mathbf{c}^{(t)})\tag{2.107}$$

Whenever the output gate is 1 we effectively pass all memory information through to the predictor, whereas for output 0 all the information are retained within the memory cell and receive no further processing. Lastly, a prediction layer can be added, which will first apply a linear model on the hidden states $\mathbf{h}^{(t)}$ and then a non-linear output activation function $g(\cdot)$ so as to produce predictions $\hat{\mathbf{y}}^{(t)}$ relative to the task for which the LSTM is employed. The above operations are described by the following expressions:

$$\begin{aligned}\mathbf{z}^{(t)} &= \mathbf{W}_{hz}\mathbf{h}^{(t)} + \mathbf{b}_z \\ \hat{\mathbf{y}}^{(t)} &= g(\mathbf{z}^{(t)})\end{aligned}\tag{2.108}$$

LSTMs can also utilize bidirectional configurations just like vanilla RNNs, having forward and backward hidden states which are eventually concatenated to form a single hidden state vector. Additionally LSTMs can be utilized in deep architectures in which the output of one hidden layer will constitute the input vector for the following hidden layer, while the actual output of the network will only depend on the hidden states of the final hidden layer.

The corresponding equations relative to the application of the BPPT algorithm in the case of LSTM units, will not be derived within the scope of the current analysis, as they are similar to the ones described in section 2.3.8, during gradient propagation through regular RNNs.

Chapter 3

Introduction to Visual Emotion Recognition

The current chapter serves as an introduction to Visual Emotion Recognition and aims at exploring the basic concepts underlying the aforementioned task. The first section of the current chapter is dedicated to presenting selected databases that have been widely used during research in the field of visual emotion recognition. In the second section, a brief review of earlier published work will be made, laying emphasis on techniques which have utilized “hand-crafted” features for the encoding of visual affective information. Subsequently, a transition to more modern practices will be made, with the emphasis now being shifted towards Deep Learning and ANN architectures. Furthermore, the section dedicated to deep learning methodologies will be divided into two subsections, with the first one focusing on certain common data preprocessing steps relative to deep visual emotion recognition, while the second one will be centered around the most common feature extraction methodologies, considering both cases of static images as well as dynamic video sequences.

3.1 Databases

The ever so growing influence of deep learning in the applied field of computer vision, and subsequently in affective computing, has inadvertently led into an increasing demand for training data. As far as visual emotion recognition is concerned, ANNs can reach their full potential only when they are provided with a sufficient amount of data examples that feature an abundance of variations in depicted populations and environments. Therefore, the current section serves as a review of selected publicly available databases which have found extensive application during research in the field of visual affective computing. For each of the following databases, emphasis will be laid on some key features, such as the type of data used, the total number of instances, the available modalities of information provided by the database, the models of emotion used for annotation, as well as the setting/source of the recorded instances.

3.1.1 Facial Expression Databases

The following databases focus primarily on the face as a source of affective information. More specifically, this collection of databases includes:

TFD

The Toronto Face Dataset (TFD), as introduced by Susskind et al. [97] contains 112,234 images, 4,178 of which are annotated with one of seven expression labels: anger, disgust, fear, happiness, sadness, surprise and neutral. In addition, five data folds are provided in the TFD, with each fold allocating 70%, 10% and 20% of the total images for training, validation and testing, respectively.

FER2013

The FER2013 database was introduced by Goodfellow et al. [43] during the ICML 2013 Challenges in Representation Learning. The images feature unconstrained settings and the database is assembled on the basis of the Google search API. The database contains 28,709 training images, 3,589 validation images and 3,589 test images with seven expression labels of anger, disgust, fear, happiness, sadness, surprise and neutral.

AFEW

The Acted Facial Expressions in the Wild (AFEW) database was introduced by Dhall et al. [28] and has served as an evaluation platform for the annual Emotion Recognition in the Wild (EmotiW) challenge since 2013. AFEW is an audiovisual dynamic database containing movie clips, featuring significant variations in illumination, head poses and background settings. Data instances are labeled with one of seven possible expressions: anger, disgust, fear, happiness, sadness, surprise and neutral. Different versions of the database have been published through the years, with a variable number of instances. The latest version, AFEW 7.0 in the EmotiW 2017 challenge, featured a data fold of 773 training, 383 validation and 653 test samples, with the three sets belonging to mutually exclusive movies and actors.

SFEW

The Static Facial Expressions in the Wild (SFEW) database [27] was created by selecting frames of the AFEW database after having computed key frames based on facial point clustering. The most widely used version, SFEW 2.0 contains a fold of 958 training, 436 validation and 372 test samples, while using the same categorical emotion annotation model as AFEW.

AffectNet

AffectNet, as introduced by Mollahosseini et al. [75], constitutes the largest and most diverse facial expression database, containing more than one million images gathered from the web using emotional related tags. The annotations provided utilize both a categorical model of 8 basic expressions, namely happy, sad, surprise, fear, disgust, anger, contempt, uncertainty, plus neutral, as well as a continuous model of the valence-arousal dimensions. Out of the total amount of images in the database, 450,000 of them have been manually annotated.

3.1.2 Gesture-Based Bimodal Databases

The current section is dedicated in presenting selected publicly available databases for recognizing gesture based expression of affect. These databases make use of multiple data streams, with the main two focusing on the face and body of the depicted people. These databases include:

FABO

The Bi-modal Face and Body Gesture Database for Automatic Analysis of Human Non-verbal Affective Behavior (FABO), was created by H. Gunes and M. Piccardi [45]. The database features 23 subjects, 12 male and 11 female, with various racial characteristics, ranging from the age of 18 to 50. Each subject was asked to enact 10 emotional states together with a particular gesture associated with each one of them. During the recordings, two digital cameras were used, one focused on the subjects' heads and one on their whole body.

GEMEP

The GENEva Multimodal Emotion Portrayal (GEMEP) corpus was introduced by Bänziger et al. [7] and it consists of over 7,000 audiovisual emotion portrayals, representing 18 emotions portrayed by 10 actors who were trained by a professional director. As the basis of their expressions, the actors were instructed to utter 2 pseudo-linguistic phoneme sequences or a sustained vowel 'aaa'. Of the total number of recordings, 1,260 portrayal were selected and included in a rating study to evaluate inter-judge reliability and recognition accuracy.

HUMAINE

The HUMAINE corpus was collected by Castellano et al. [17] during the Third Summer School of the HUMAINE EU-IST project, held in Genova in September 2006. The overall recording procedure was based on the GEMEP corpus. Ten participants of the summer school, distributed as evenly as possible concerning their gender, participated to the recordings. Subjects represented five different nationalities: French, German, Greek, Hebrew, Italian. Two DV cameras recorded the actors in frontal view, with one camera recording the actor's body and the other one focused on the actor's face. The subjects enacted 8 emotions together with 8 emotion specific gestures, resulting in 240 samples for each modality.

3.1.3 Context-Oriented Databases

More recently, research has been shifted towards emotion recognition in context, with the aim of integrating contextual information such as the scene, objects and surrounding environment in the affective computing process. Notable databases of this kind are the following:

EMOTIC

The EMOTIC dataset was introduced by Kosti et al. [58] and is comprised of images from MSCOCO [64], ADE20K [118] and images that were manually downloaded using the Google search engine. Instances within the dataset may depict multiple people, resulting in a total number of 18,316 images and 23,788 annotated instances. The images were manually annotated using the Amazon Mechanical Turk (AMT). For annotations, the dataset combines a categorical model of 26 non-mutually exclusive emotions as well as the continuous model of VAD dimensions (in the range [1-10]).

BoLD

The Body Language Dataset (BoLD), as it was assembled by Luo et al. [67], constitutes a dataset that focuses on bodily expressions of emotion. BoLD is comprised of 9,876 movie

video clips of body movements, depicting a total of 13,239 human characters. The annotation of the dataset was performed using a crowdsourcing pipeline based on the Amazon Mechanical Turk (AMT). Instances are annotated in both categorical and dimensional level. For categorical emotions, the 26 categories of the EMOTIC dataset were utilized, allowing multi-label classification, while for the continuous annotations, the VAD dimensional model was used.

CAER

The Context-Aware Emotion Recognition (CAER) benchmark was introduced by Lee et al. [62] and is comprised of 13,201 TV video clips, resulting in about 1.1M frames. The videos range from short (~ 30 frames) to longer clips (~ 120 frames), while the average sequence length is 90 frames. Each video clip was independently annotated with one of the six basic emotions, plus “neutral”, by three different annotators and if at least two annotators assigned the same emotion categories to a particular clip, then that instance remained in the dataset. If instances had low confidence scores (annotation reliability), below 0.5, then those clips were removed. Additionally, the CAER-S benchmark was formed as a static variant of the original dataset, as it contains 70K static frames from the initial dataset.

3.2 Hand-Crafted Features

The earlier published works in the field of multi-modal visual emotion recognition utilized “hand-crafted” features for the encoding of visual affective information. The majority of the experiments were conducted on datasets that included images and videos from constrained lab environments and the emotion recognition task was performed relative to a variable number of basic discrete emotions, with the most common case being that of the six universal emotions, plus the neutral emotion category. Additionally, for the actual classification task, common machine learning classifiers had been employed, such as Random Forest, k-Nearest Neighbors (k-NN), Naive-Bayes classifiers, as well as Support Vector Machines (SVM). Lastly, a series of engineered texture features will be presented that have been primarily used as a means of describing shape and appearance but have also found extensively application in affective analysis. In the following paragraphs we will briefly analyze the most significant earlier contributions that initialized the field of multi-modal visual emotion recognition and paved the way for further research.

3.2.1 Bimodal Visual Emotion Recognition using Body/Face

The earlier steps in bi-modal visual affective analysis, utilizing both facial and body cues, involved the use of hand-crafted features for the classification of discrete emotions. One of the most notable approaches of this kind is the work of Gunes and Piccardi [47]. They initially created the FABO database which included video sequences of the body and face of several subjects, enacting 6 emotional states, namely anxiety, anger, disgust, fear, happiness and uncertainty. The sequences were captured using two cameras, one focused on the face and one on the whole body. Instances of the FABO database can be seen in figure 3.1. The extracted feature vectors consist of displacement measures between a neutral and apex frames.

By using morphological filters and skin color segmentation on HSV color space they obtained the face region. Next, they used grayscale information and edge maps (extracted with the Canny edge detector [14]) in order to define bounding boxes centered on key facial features, such as eyes, eyebrows, mouth, nostrils and chin. After key facial features have

been located, they calculated optical flow using the Lukas-Kanade algorithm [66] between the neutral and apex frames, resulting in a 148-dim total feature vector.

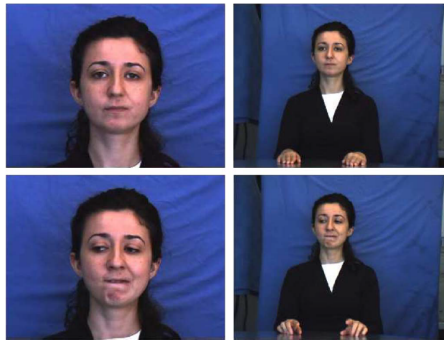


Figure 3.1: Sample images from the FABO database separately recorded by the (left) face and (right) body cameras. Source: [46].

The corresponding body model involved the detection and tracking of the head and hands. A similar approach, including background subtraction, silhouette extraction color segmentation and morphological filtering was used for the detection process. For each foreground object, they calculated a set of features including its centroid, area, bounding box and contraction index, resulting in a 140-dim feature vector.

Due to lack of enough sequence examples they applied a dimensionality reduction process, namely best-first-search, keeping only 29 facial features, 11 body features and 14 fused features in the bi-modal method. In all cases, a standard Naïve Bayes classifier was used for the classification of the input feature vectors into the six discrete emotion states. They examined both mono-modal and multi-modal approaches, considering either feature level or decision level fusion (sum/product/weighted-sum score aggregation). They concluded that the multi-modal approach with feature level fusion provides superior performance.

Castellano et al. [18] proposed another multi-modal approach for the recognition of eight emotional states (anger, despair, interest, pleasure, sadness, irritation, joy and pride) based on the ten-subject multi-modal corpus of the HUMAINE EU-IST project. This dataset included video footage of the body and face, recorded from two separate cameras, with the subjects enacting specific emotions while pronouncing a pseudo-linguistic sentence.

Face feature extraction, included face detection and segmentation in areas, namely eyes, eyebrows, nose and mouth. Each area contains feature specific boundaries, masks that are separately extracted and subsequently fused. The fused masks are used to extract 19 feature points (FPs) in the neutral and apex frames of each sequence. FPs between the neutral and apex frames were compared to produce facial animation parameters (FAPs) which represent 66 displacements and rotations of the feature points.

Body feature extraction involved the computation of five expressive motion cues, namely quantity of motion (QoM), contraction index (CI), velocity, acceleration and fluidity of the hands' barycenter. To begin with, QoM measures detected motion and is based on the extraction of body silhouette in each frame of a sequence. Silhouette motion images (SMI) are calculated as follows:

$$\text{SMI}[t] = \sum_i \text{Silhouette}[t - i] - \text{Silhouette}[t] \quad (3.1)$$

QoM is computed as area and is normalized to receive a value in the interval [0,1]:

$$\text{QoM}[t] = \frac{\text{area}(\text{SMI}[t])}{\text{area}(\text{Silhouette}[t])} \quad (3.2)$$

The contraction index $CI \in [0, 1]$ can be calculated as the ratio between the area of the minimum rectangle bounding the hands and head and the area covered by the silhouette. An example of QoM measurement using SMIs is illustrated in figure 3.2. Velocity and acceleration are related by the trajectory of the tracked hands, while fluidity (measure of uniformity of motion) is inversely proportional to the acceleration.



Figure 3.2: A measure of QoM using SMIs (the shadow along the arms and the body) and the tracking of the hand. Source: [18].

For a given sequence the above features can either be described as a time series and use them to perform temporal classification using Dynamic Time Warping (DTW) or convert them to a fixed set of meta features, such as maximum value, mean value, number of peaks, initial and final slopes, and symmetry index. That would result in a 80-dim motion feature vector for every video sequence that would be fed in a 1-NN or Bayesian Classifier. In the first approach, all possible warping paths \mathcal{W} would be computed between sequence pairs $\mathbf{q} = [q_1, \dots, q_n]^\top$ and $\mathbf{c} = [c_1, \dots, c_m]^\top$, resulting in a minimum warping cost:

$$\text{DTW}(\mathbf{q}, \mathbf{c}) = \min_{\mathcal{W}=\{w_1, w_2, \dots, w_K\}} \sqrt{\sum_{\substack{k=1 \\ w_k=(i,j)}}^K d(q_i, c_j)} \quad (3.3)$$

where $d(\cdot)$ is a selected distance metric. In that way, time series classification can be performed using 1-NN approach, namely a test series is classified at the same class as its nearest time series in the training dataset according to the DTW distance. After examining both approaches in mono-modal settings, the authors concluded that the use of meta features provides superior performance over 1-NN DTW.

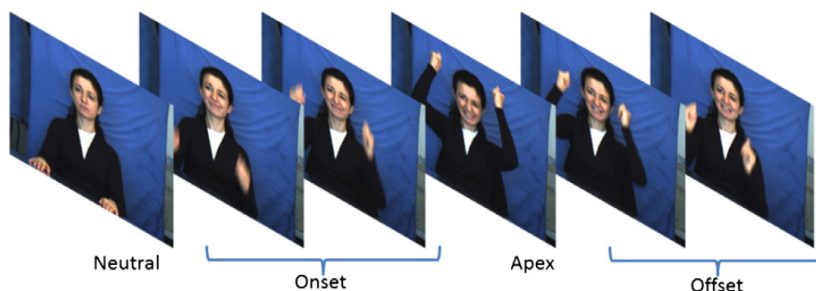


Figure 3.3: The temporal evolution of an expression of “Happiness”. Source: [21].

The above methods perform feature extraction on the basis of neutral and apex frames of body and face channels, therefore an automatic temporal segmentation method of the video sequences is required. To this end, an honorable contribution was made by Chen et al. [20]. They proposed the use of a gesture channel only for the extraction of two kinds of features, namely motion area (MA) and neutral divergence (ND). Motion area is based on motion

history images (MHIs) which can be calculated for each pixel (x, y) of a given frame t :

$$\begin{aligned} \text{MHI}_\tau(x, y, t) = & D(x, y, t) \times \tau + [1 + D(x, y, t)] \times U[\text{MHI}_\tau(x, y, t - 1)] \\ & \times \text{MHI}_\tau(x, y, t - 1) \end{aligned} \quad (3.4)$$

where $U[\cdot]$ is the unit step function, t represents the current frame index, τ is the duration of motion and \mathbf{D} is a binary image with a value 1 at pixel locations where the difference in intensity between the current frame and the previous one, is higher than a predefined threshold. The MHI is scaled to an 8-bit image. An example of MHI extraction around the hand and head regions is shown in figure 3.4. The motion area of a frame is the total number of pixels with non-zero intensity in the MHI:

$$\text{MA}_\tau(t) = \sum_{x=1}^H \sum_{y=1}^W U[\text{MHI}_\tau(x, y, t) - \epsilon] \quad (3.5)$$

where $0 < \epsilon < 1$ is a threshold parameter. Neutral divergence measures the degree of difference between a current frame t and the neutral frame t_0 , by summing absolute intensity differences over three color channels:

$$\text{ND}_\tau(t) = \sum_{c=1}^3 \sum_{x=1}^H \sum_{y=1}^W \text{abs}[I(x, y, t) - I(x, y, t_0)] \quad (3.6)$$

Both MA and ND are normalized in the interval $[0,1]$. In order to acquire corresponding feature vector forms, temporal windows of specified length are centered on each frame. Therefore, motion area and neutral divergence features have a dimensionality equal to the size of the used temporal windows. The key idea is to use these features both separately as well as in a fused manner, as inputs to a classifier so as to automatically perform temporal segmentation. The expressive phases (neutral/onset/offset/apex) detected during temporal segmentation for the emotion of ‘‘Happiness’’ are illustrated in figure 3.3.

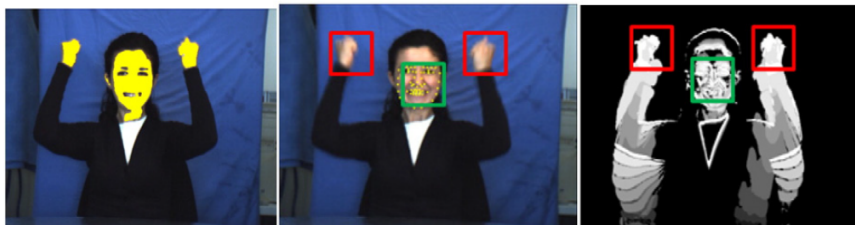


Figure 3.4: Hand tracking by skin color-based tracker (left); position of hands using skin color tracking and position of head using an Active Shape Model (ASM) (middle); extraction of motion areas of hand and head regions (right). Source: [21].

In later work, Chen et al. [21] used image HOGs and MHI-HOGs to describe the appearance and motion information of expressions respectively. In order to represent the dynamics over a complete motion cycle (onset-apex-offset), he introduced two methods, namely Bag of Words (BoW) model and Temporal Normalization (TN), as opposed to previous approaches that utilized max-voting schemes over apex frames only.

During the feature extraction process, both facial and body gesture features were used. After locating the face along with 53 key facial landmarks at each frame of an emotion cycle, $n \times n$ patches were centered on each interest point. Each patch was further divided into a $m \times m$ grid. For every patch, a HOG and MHI-HOG descriptor was formed as the concatenation of m^2 histograms with b_i and b_m bins, resulting in feature vectors of size $m^2 \cdot b_i$ and $m^2 \cdot b_m$, respectively, for each interest point in a frame of the expression cycle.

In the BoW approach, visual words for image HOGs and MHI-HOGs are found by applying k-means clustering separately for the facial and body gesture features and then using minimum Euclidean distance. Features are quantized to visual words to form HOG/MHI-HOG histograms that are concatenated so as to represent a complete expression cycle as a probability distribution. The codebooks of HOGs and MHI-HOGs for facial expressions had a size of 200 words each while their body counterparts had a size of 80 words each (less variations compared to the face). The alternative approach of expressing temporal dynamics involves temporal normalization (TN). TN essentially uses linear interpolation for each individual feature dimension along the temporal dimension in order to normalize each expression cycle to a fixed number of frames N_{fr} . By applying TN, the above features are extracted for every frame of a fixed, N_{fr} frame long expression cycle.

Lastly, Shan et al. [93] experimented on the FABO database and proposed a way of directly extracting spatio-temporal features from both facial and body cues. For every frame of a video, spatio-temporal interest points are detected by calculating the response function, after the application of separable linear filters:

$$R = (I * g * h_{ev})^2 + (I * g * h_{od})^2 \quad (3.7)$$

$I(x, y, t)$ denotes a single video frame, $g(x, y; \sigma)$ denotes a 2D Gaussian smoothing kernel and h_{ev}, h_{od} are a pair of 1D Gabor filters applied temporarily, defined as $h_{ev} = -\cos(2\pi t\omega)e^{-t^2/2}$ and $h_{od} = -\sin(2\pi t\omega)e^{-t^2/2}$. In all cases $\omega = 4/\tau$, while σ, τ correspond to the spatial and temporal scale of the detector. Interest points can be located as local maxima of the above response function. Around each interest point, a cuboid neighborhood can be extracted. These local regions represent the spatio-temporal information of each single frame, as the initial video can be discarded. Each cuboid can be described in various ways such as brightness gradients, windowed optical flow combined with global or local histogramming. By assembling a Bag of Features (BoF) model, every cuboid can be assigned to a closest prototype vector and a video can be described with the use of a histogram of cuboids.

3.2.2 Texture Features

At this point we would like to summarize a series of engineered features that have been used in affective computing as a means of describing shape and appearance.

Local Binary Patterns

To begin with, Ahonen et al. [1] introduced Local Binary Patterns (LBP) which later constituted the basis for several other subsequent texture feature variants. The operator assigns a binary label to every pixel of an image by thresholding a 3×3 neighborhood of the pixel. The intensity of the center pixel is compared to the intensities of the surrounding neighbor pixels. If the intensity value of a neighbor pixel is equal or higher than that of the center pixel, then a 1 is assigned to that pixel, else a 0 is assigned to the pixel. The 0s/1s that formed around the given pixel are read counterclockwise, forming an 8-bit binary number, with the most significant bit being the one that is left of the center pixel. The 8-bit number is then converted to a decimal number in the range 0-255. Neighborhoods can be generalized in the form (P, R) , where P is the number of sampling points in a circle of radius R . The functionality of the basic LBP operator is shown in figure 3.5.

This basic operator can be used in the extraction of texture descriptors. An image is divided in M rectangular regions. In each region a histogram of LBP codes is calculated independently. The resulting M histograms are combined resulting in a spatially enhanced

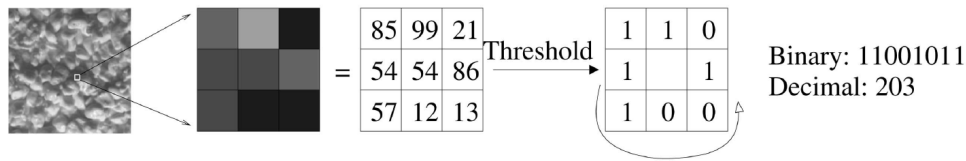


Figure 3.5: Illustration of the basic Local Binary Pattern (LBP) operator. Source: [1].

histogram of size $M \times N$, where N is the length of one histogram and depends on the neighborhood size (usually $N=256$). The LBP labels for the histogram contain information about patterns on a pixel level. The summation of labels on small regions produces information in a regional level. Lastly the local histograms are concatenated into a global descriptor of the face, preserving information in all 3 levels of locality. The LBP descriptor is robust against illumination variations.

Volume LBP and LBP on Three Orthogonal Planes

Two extensions of LBP for dynamic texture analysis, called VLBP (volume) and LBP-TOP (three orthogonal planes) were proposed by Zhao and Pietikainen [115]. Starting with VLBP, we assume a dynamic texture of size $X \times Y \times T$, where X, Y are the spatial dimensions and T are the number of frames. The extraction of VLBP considers three frames, equally spaced in the temporal direction with an interval of L frames in between. LBP codes are calculated for every pixel of the texture in the same manner as described above but for cuboid neighborhoods that span three frames and therefore contain $3P + 2$ neighbor pixels. In that way the size of the resulting feature descriptors will be equal to 2^{3P+2} , prohibiting the use of large neighborhoods due to increased computational cost.

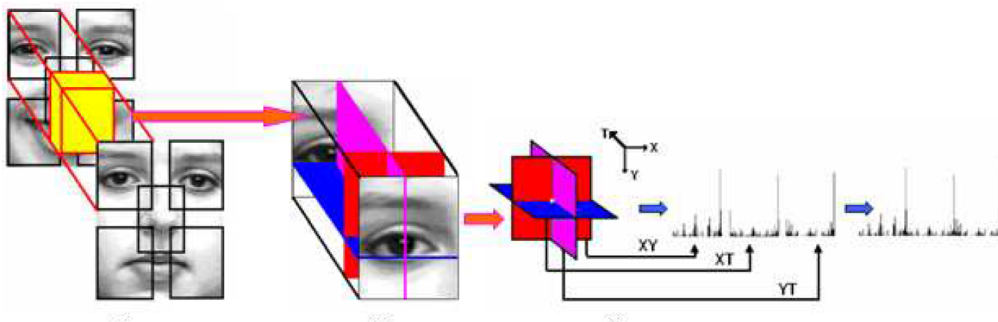


Figure 3.6: Illustration of the facial expression representation procedure using LBP-TOP descriptors. Localization of facial features in each block volume, extraction of LBP features in three orthogonal planes. Features are concatenated for each block volume in order to represent shape and appearance. Source: [115].

As an alternative the authors proposed the LBP-TOP variation which involves the extraction of LBP codes from three orthogonal planes, XY, YT, XT. LBP operators can have different number of sample pixels and different radii in each plane, extending the traditional circular sampling to elliptical sampling. The individual histograms from every plane are concatenated into a global histogram integrating temporal and spatial features. LBP-TOP can be used in facial expression recognition. In that case image frames can be divided in blocks (overlapping or not), where inside each block, LBP-TOP histograms will be calculated and then concatenated into a single histogram, effectively describing facial expressions at a pixel level, local, as well as global level. The process of encoding facial expressions using the LBP-TOP operators is illustrated in figure 3.6. Compared to VLBP, LBP-TOP descriptors

have a size of order 3^{2P} for P neighbor samples, thus being computationally more efficient while not sacrificing performance. A slight variation of the LPB-TOP operator involved the convolution of the image frames with banks of Gabor filters before the extraction of the actual texture features, resulting in the formulation of Local Gabor Binary Patterns (LGBP) and their LGBP-TOP extension. LGBP-TOP features have shown increased accuracy compared to LBP-TOP and have proven to be moderately robust to face-alignment errors.

Volume Local Phase Quantization

Subsequently, Päiväranta et al. [79] introduced Volume Local Phase Quantization (VLPQ) as an extension of the original LBP-TOP texture descriptor that is invariant to image blurring. A blurred version $g(\mathbf{x})$ of an image $s(\mathbf{x})$ can be modeled as the convolution of the image with a point spread function (PSF) $h(\mathbf{x})$ of the blur, namely $g(\mathbf{x}) = (s * h)(\mathbf{x})$. In the Fourier domain, considering the phase of the spectrum, $\angle G(\mathbf{u}) = \angle S(\mathbf{u}) + \angle H(\mathbf{u})$ and for a centrally symmetric PSF, its Fourier transform is real valued and resembles a low pass filter, therefore $\angle G(\mathbf{u}) = \angle S(\mathbf{u})$ and $\angle H(\mathbf{u}) = 0$. For every pixel \mathbf{x} in a sequence of frames $f(\mathbf{x})$, we consider a neighborhood \mathcal{N}_x of size $M \times M \times N$ pixels, where M is the spatial size and N the temporal length. The Fourier transform can be approximated locally in \mathcal{N}_x using STFT:

$$F(\mathbf{u}, \mathbf{x}) = \sum_{\mathbf{y} \in \mathcal{N}_x} f(\mathbf{x} - \mathbf{y}) e^{-j2\pi \mathbf{u}^\top \mathbf{y}} \quad (3.8)$$

where u is the 3D frequency. STFT can be rewritten as $F(\mathbf{u}, \mathbf{x}) = \mathbf{w}_{\mathbf{u}}^\top f_{\mathbf{x}}$, where $f_{\mathbf{x}}$ contains all neighbor pixels and $\mathbf{w}_{\mathbf{u}}^\top$ contains the basis vector of the 3D DFT at frequency \mathbf{u} . In order for the blur invariant property to hold, DFT is only calculated at 13 lowest non-zero frequencies. Moreover, the real and imaginary parts are separated, namely:

$$\begin{aligned} F_{\mathbf{x}} &= [\text{Re}\{F(\mathbf{u}_1, \mathbf{x})\}, \text{Im}\{F(\mathbf{u}_1, \mathbf{x})\}, \dots, \text{Re}\{F(\mathbf{u}_{13}, \mathbf{x})\}, \text{Im}\{F(\mathbf{u}_{13}, \mathbf{x})\}] \\ \mathbf{W} &= [\text{Re}\{\mathbf{w}_{\mathbf{u}_1}^\top\}, \text{Im}\{\mathbf{w}_{\mathbf{u}_1}^\top\}, \dots, \text{Re}\{\mathbf{w}_{\mathbf{u}_{13}}^\top\}, \text{Im}\{\mathbf{w}_{\mathbf{u}_{13}}^\top\}] \\ F_{\mathbf{x}} &= \mathbf{W} f_{\mathbf{x}} \end{aligned} \quad (3.9)$$

where $\text{Re}\{\cdot\}$ and $\text{Im}\{\cdot\}$ denote the real and imaginary parts respectively, while \mathbf{W} is a $26 \times M^2 \times N$ matrix. Due to excessive number of variables, dimensionality reduction must be applied. Assuming a simple covariance model based on the distance of adjacent pixels in the \mathcal{N}_x neighborhood, PCA can be applied, resulting in a reduced DFT vector G_x with $L < 26$ components. After calculating G_x for every volume position, a simple quantization method can be used:

$$q_{\mathbf{x}}(j) \begin{cases} 1 & \text{if } g_{\mathbf{x}}(j) \geq 0 \\ 0 & \text{otherwise} \end{cases} \quad (3.10)$$

where $g_{\mathbf{x}}(j)$ is the j th component of G_x . In that way, binary codes can be formed and then transformed into decimal to form histograms of 2^L bins. The derivation of VLPQ-TOP follows the same process but is calculated in three orthogonal planes. VLPQ has shown superior classification performance compared to LPQ-TOP and both methods score higher than LBP-TOP. In addition their method tolerates more centrally symmetric spatial blurring.

Pyramid HOGs and SIFT descriptors

Lastly, two more types of texture features need to be mentioned, namely Pyramid HOGs (PHOG), which were introduced by Bosch et al. [12] as well as the Scale Invariant Feature Transform (SIFT) descriptor, which was introduced by Lowe [65]. Both of the aforementioned descriptors have found extensive applications in facial expression recognition (FER).

In the first case, a HOG vector is computed for each grid cell at each spatial pyramid level. The PHOG descriptor for an image is a concatenation of all the HOG vectors. At level l of the pyramid, the image is divided into 2^l cells along each dimension, resulting in 4^l cells in total. Assuming each HOG vector that is extracted from every cell has K bins, then a pyramid of levels $l \in \{0, 1, \dots, L\}$ results in a HOG descriptor of size $K \sum_l 4^l$. The similarity $K(S_i, S_j)$ between two PHOG descriptors S_i, S_j is measured in terms of their chi-squared distance (χ^2) $d(S_i, S_j)$:

$$K(S_i, S_j) = \sum_l a_l d(S_i, S_j) \quad (3.11)$$

where $a_l = 1/2^{l-1}$. In general, PHOGs have exhibited higher performance compared to their single-level counterparts.

Last but not least, SIFT descriptors have been used in encoding shape and appearance information during FER. The SIFT descriptor is 128-dim vector that results from the concatenation of all 16 histograms and is invariant to image rotations, shifts, scalings and changes in illumination. Additionally, the large number of features in a typical image allow for robust recognition under partial occlusion in cluttered images.

3.3 Deep Learning Approach to Visual Emotion Recognition

Deep learning has recently become a hot research topic and has achieved state-of-the-art performance for a variety of applications. Deep learning attempts to extract high-level data representations through the use of hierarchical architectures of multiple non-linear transformations. Therefore, the current section will briefly present the two main steps related with deep visual emotion recognition, namely data preprocessing and feature extraction.

3.3.1 Data Preprocessing

Visual emotion recognition on the basis of both facial and body features, requires several data preprocessing steps, such as human body detection, pose estimation, as well as face detection and alignment. The basic concepts behind each preprocessing step as well as popular methodologies for each individual step are discussed in the following paragraphs.

Human Detection

Most of the existing bimodal databases used for emotion recognition provide body crops in the case of images or feature recordings focused solely on the bodies of the depicted people. In case this type of information is not directly supplied, human detection techniques need to be applied. The human detection task is based on determining rectangular bounding boxes that enclose a human body. This problem is basically decomposed as an object detection task, involving the extraction of potential regions of interest, representation and classification of those regions as human or not human and merging of the positively classified ones as final decisions.

Earlier methodologies for human detection utilized “hand-crafted” features. A notable example of such an approach is the use of gradient based methods and the so called histogram of oriented gradients (HOG), as proposed by Dallal and Triggs [25]. More recent methodologies rely on deep neural networks and especially CNNs, for the extraction of bounding boxes for the human bodies. One of the first steps in that direction was the Large Field of View (LFOV) network. It was introduced by Angelova et al. [4] and featured a cascaded architecture with three ConvNets. In general, DNN models, are known to be very slow when used as sliding-window classifiers which makes it challenging to use for human body detection. In order to

alleviate this problem cascaded networks are utilized. For example, a shallow network can be trained to greatly reduce the initially large number of candidate regions produced by the sliding window detector. Then in a second step, only high confidence regions are passed through a deep network obtaining in this way a trade-off between speed and accuracy.

Pose Estimation

After bounding boxes have been determined and the background has been subtracted, the second stage involves human pose estimation. In recent years, the pose estimation task has been dominated by deep network architectures.

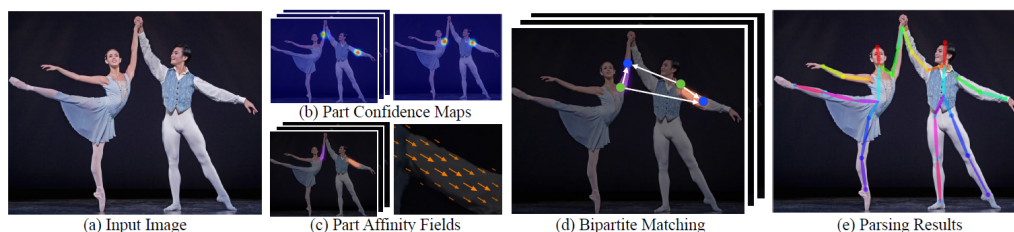


Figure 3.7: OpenPose processing pipeline. The entire image constitutes the input for a two-branch CNN to jointly predict confidence maps for body part detection, shown in (b), and part affinity fields for parts association, shown in (c). The parsing step performs a set of bipartite matchings to associate body parts candidates (d). Finally full body poses are assembled for all people in the image (e). Source: [15].

One honorable mention is the DeepPose network, introduced by A. Toshev and C. Szegedy [100], which formulates the pose estimation task as a joint regression problem. After an initial joint regression result has been obtained, a cascade of identical CNNs is employed to refine upon the latter, with each stage receiving as input, image crops around the predicted joints produced by the previous stage. Furthermore, the latest significant advancement in pose estimation came along the form of OpenPose, as it was introduced by Cao et al. [15]. OpenPose constitutes a realtime multi-person 2D pose estimation system which utilizes Part Affinity Fields (PAFs). The basis of this method relies on a two-branch multistage CNN, with the first stage extracting part confidence scores, while the second branch extracts vector fields, namely PAFs that encode the degree of association between candidate body parts. The maps are refined over successive stages and at each stage the predictions from the two branches as well as the original image features are concatenated and fed to the next stage. A high-level description of the OpenPose pipeline can be seen in figure 3.7.

Face Detection & Alignment

The next step in the preprocessing pipeline is related with the task of face detection and alignment. Face detection involves the extraction of bounding boxes around a human face while face-alignment considers the localization of facial landmarks (eyes, nose, mouth corners, etc.).

As of more recently, joint face detection and alignment has been achieved through the use of cascaded CNNs. An example of such an architecture is the Multi-Task Cascaded Convolutional Network (MTCNN), introduced by Zhang et al. [112], where a shallow fully convolutional network (with no fully connected layers) called proposal network (P-Net) produced candidate facial windows with their bounding box regression vectors as well as a coarse estimate of the facial landmark locations, while the second and third net, called R-Net and O-Net respectively, utilizing a more complex architecture, each refined the estimates produced by the previous stage. An overview of the MTCNN architecture is illustrated in figure 3.8.

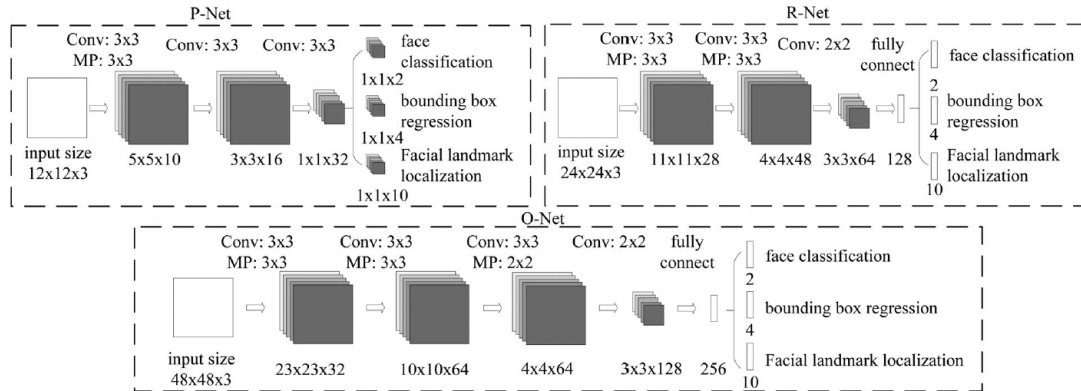


Figure 3.8: Illustration of the MTCNN inner structure. Architectures of P-Net, R-Net, and O-Net, where “MP” means max pooling and “Conv” means convolution. The step size in convolution and pooling is 1 and 2, respectively. Source: [112].

Moreover, the latest breakthrough in face detection and alignment came along the development of the OpenFace 2.0 toolkit, by Baltrusaitis et al. [6]. OpenFace features a complete facial behavior analysis pipeline capable of accurate facial landmark detection, head pose estimation, facial action unit recognition and eye-gaze estimation.

3.3.2 Feature Extraction

After all available data have been preprocessed, the input modalities are fed into deep neural networks which operate as feature extractors, with the aim of acquiring high-level deep feature representations. The feature extraction methodology usually depends on the nature of the available data, namely if they are static (images) or sequential (video clips). In both cases however, the CNN constitutes the common deep learning module for feature extraction.

In general, a CNN used for feature extraction, is comprised of four types of layers, namely convolutional layers, normalization layers, pooling layers and fully-connected (FC) layers. To begin with, the convolutional layer uses a set of learnable filters to convolve through the entire input image and produce multiple feature maps over different levels of abstraction, moving from low to high level representations as the layers are positioned deeper within the network. Moreover, the convolution operation attains three main benefits, namely local connectivity, weight sharing and shift-invariance. Local connectivity is associated with learning correlations among neighboring pixels. Subsequently, weight sharing refers to the common usage of weight parameters in the same feature maps, resulting in a considerable decrease in the total number of trainable parameters, while shift-invariance is related with the network’s ability in effectively addressing problems of translation, rotation and scale variations of the detected entities. While pooling layers regularly succeed convolutional layers and are used in reducing the spatial size of the produced feature maps, normalization layers, such as dropout as well as batch-normalization (BN) layers aid the network by reducing the effect of overfitting and accelerating the training process, respectively. Lastly, fully-connected layers convert the multi-dimensional feature maps produced by the previous layers into 1D feature vectors for further feature representation and classification.

The usage of CNNs for feature extraction can be extended in the domain of dynamic, sequential data such as video clips. This type of network is referred to as a 3D CNN and it was initially proposed for human action recognition, by Ji et al. [55]. Videos can be considered as stacks of consecutive frames, resulting in a data format that extends in two spatial and one temporal dimension. The 3D variants of CNNs utilize 3D convolutional kernels that have the ability to move in three directions during the convolution operation resulting in

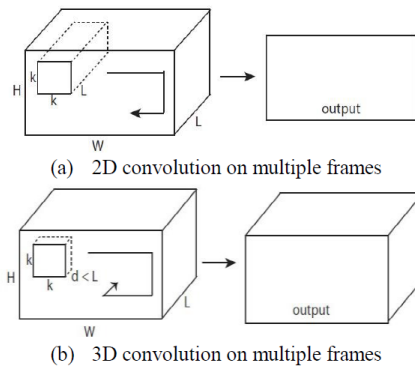


Figure 3.9: 2D and 3D convolution operations. a) Applying 2D convolution on a video volume (multiple frames as multiple channels) also results in an image. b) Applying 3D convolution on a video volume results in another volume, preserving temporal information of the input signal. Adapted from [101].

voluminous output shapes such as cuboids. Figure 3.9 illustrates the core differences between 2D and 3D convolutions when applied on multiple frames. Another notable example of such an architecture is the C3D, which was proposed by Tran et al. [101] for spatio-temporal feature extraction in large-scale training datasets.

Except for 3D CNNs, RNNs can also be effectively utilized for emotion recognition in dynamic image sequences, as they can robustly derive information from sequences by exploiting the interdependent relation between successive frames. LSTMs, as an improved variant of the RNN, are capable of handling sequential data of variable length, while reducing the effect of the vanishing gradient problem. Compared with RNN, CNN and its variants are more suitable for computer vision applications and thus for visual emotion recognition. However, the perceptual vision of CNNs and the strength of LSTMs for sequential data, can be effectively combined in cascaded architectures. More specifically, CNNs are firstly used to extract discriminative representations for each one of the frames contained within a video sequence and then these features are fed to sequential networks in order to reinforce the temporal information encoding. An example of a CNN-LSTM cascade that is both spatially and temporally deep is depicted in figure 3.10.

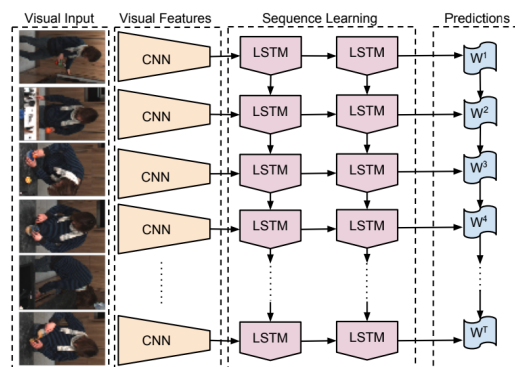


Figure 3.10: The Long-term Recurrent Convolutional Networks (LRCN). The LRCN processes the (possibly) variable-length visual input (left) with a CNN (middle-left), whose outputs are fed into a stack of recurrent sequence models (LSTMs, middle-right), which finally produce a variable-length prediction (right). Both the CNN and LSTM weights are shared across time, resulting in a representation that scales to arbitrarily long sequences. Source: [29].

Chapter 4

Image-Based Visual Emotion Recognition in Context

The majority of past efforts in visual emotion recognition have been mostly limited to the analysis of facial expressions, while some studies have either incorporated information relative to body pose or have attempted to perform emotion recognition, solely on the basis of body movements and gestures. While some of these approaches perform well in certain specified settings, in most cases they fail to interpret real-world scenarios. An example to illustrate this point is the fact that emotion recognition systems often deal with instances of people whose facial features are fully visible and their body joints are unoccluded, something which does not generally conform to reality. However, evidence from psychology related studies suggest that the scene context, in addition to facial expression and body pose, provides important information to the perception of people’s emotions.

To this end, we choose our first case study to be associated with context-based visual emotion recognition in static images. In that way we are presented with the opportunity to explore and potentially extend a relatively new domain of automated emotion recognition which promotes the use of all possible sources of visual affective information. Moreover, the newly assembled EMOTions In Context (EMOTIC) dataset is going to be used as a frame of reference for both of our theoretical analysis and experimental results. In the first section of the current chapter we will present the applied methodologies from all publicly available related work as well as propose possible extensions over the current models. Subsequently, a second section will be dedicated to our experimental results on the EMOTIC dataset.

4.1 Related Work

4.1.1 Baseline Model

Kosti et al. [58] made one of the first notable contributions towards context-based emotion recognition by introducing the EMOTIC dataset as well as providing a baseline model that was trained and evaluated on it. Their baseline model consisted of three modules: two feature extraction modules and one fusion network. The first feature extraction module received the whole image as input with the aim of extracting context-related features, while the second feature extraction module received the body image crops as input, generating body-related features.

Both branches used CNNs for feature extraction, but with each CNN having been pre-trained on different datasets. The context branch CNN was pre-trained on the Places365 [117] dataset, while the body branch CNN was pre-trained on the ImageNet [26] dataset.

The Places365 dataset is a scene-centric repository of 10M scene photographs, labeled with one of 365 different possible scene semantic categories, comprising a quasi-exhaustive list of the types of environments encountered in the world. Moreover, ImageNet is a large visual database designed for use in visual object recognition software research, containing a total of more than 14M images and over 20,000 annotated object categories. The data used for network pre-training are part of the ImageNet Large Scale Visual Recognition Challenge (ILSVRC), containing only a subset of 1,000 categories.

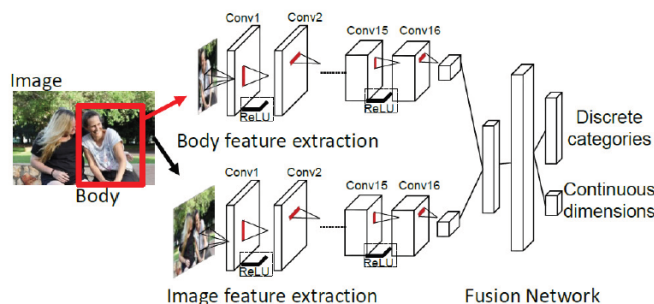


Figure 4.1: Baseline end-to-end model for emotion recognition in context. The model consists of two feature extraction modules and a fusion network for jointly estimating the discrete categories and the continuous dimensions. Source: [59].

The images contained in the EMOTIC dataset depict a variable number of annotated people for whom, body bounding boxes are provided. In that way, instances of the dataset come in the form of context-body pairs, with multiple instances sharing the same context image in case the latter depicts multiple annotated people. Furthermore, the features extracted from the two feature extraction modules are concatenated and fed to the fusion network. The fusion network consists of a fully-connected (FC) layer, followed by a batch-normalization layer, ReLU activation and a dropout layer. The output of the fusion network are fed to the classification and regression layers, performing emotion recognition in both categorical and continuous level. An overview of the aforementioned baseline model is depicted in figure 4.1.

As discussed in section 3.1, the EMOTIC dataset annotation system provides 26 discrete emotion categories for multi-label classification as well as VAD scores in the [1-10] range. The network is trained by backpropagating the gradients a weighted sum of the two separate losses produced while multitasking, namely $\mathcal{L} = \lambda_{\text{disc}}\mathcal{L}_{\text{disc}} + \lambda_{\text{cont}}\mathcal{L}_{\text{cont}}$, where $\mathcal{L}_{\text{disc}}$ represents the loss corresponding to the discrete categories and $\mathcal{L}_{\text{cont}}$ represents the loss corresponding to the continuous dimensions. Given a ground truth target vector $\mathbf{y} = (\mathbf{y}^{\text{disc}}, \mathbf{y}^{\text{cont}})$ and the corresponding predicted scores $\hat{\mathbf{y}} = (\hat{\mathbf{y}}^{\text{disc}}, \hat{\mathbf{y}}^{\text{cont}})$, where $\hat{\mathbf{y}}^{\text{disc}} = (\hat{y}_1^{\text{disc}}, \dots, \hat{y}_{26}^{\text{disc}})$, $\hat{\mathbf{y}}^{\text{cont}} = (\hat{y}_1^{\text{cont}}, \hat{y}_2^{\text{cont}}, \hat{y}_3^{\text{cont}})$, $\hat{y}_i^{\text{disc}} \in \{0, 1\}$ and $\hat{y}_i^{\text{cont}} \in [1, 10]$, then the discrete loss $\mathcal{L}_{\text{disc}}$ is given by:

$$\mathcal{L}_{\text{disc}} = \sum_{i=1}^{26} w_i (\hat{y}_i^{\text{disc}} - y_i^{\text{disc}})^2 \quad (4.1)$$

where w_i is a weighting factor that is inversely proportional to the number of occurrences of the i th category within each training batch. It is worth noting that the authors decided to use an L^2 loss function for a classification task instead of using the standard binary cross entropy, as discussed in section 2.3.2. Subsequently, two different loss functions for the regression task are examined, with the first one being a marginal Euclidean loss $\mathcal{L}_{2\text{cont}}$ of the following form:

$$\mathcal{L}_{\text{cont}}^2 = \sum_{k=1}^3 v_k (\hat{y}_k^{\text{cont}} - y_k^{\text{cont}})^2 \quad (4.2)$$

where $v_k \in \{0, 1\}$ represents the error margin. If $|\hat{y}_k^{\text{cont}} - y_k^{\text{cont}}| < \theta$ then $v_k = 0$, otherwise $v_k = 1$ where θ is a threshold parameter. Secondly, the smooth- L^1 loss function [41] is proposed as it is found to be less sensitive to outliers:

$$\mathcal{L}_{\text{cont}}^1 = \sum_{k=1}^3 \begin{cases} 0.5(\hat{y}_k^{\text{cont}} - y_k^{\text{cont}})^2 & \text{if } |\hat{y}_k^{\text{cont}} - y_k^{\text{cont}}| < 1 \\ |\hat{y}_k^{\text{cont}} - y_k^{\text{cont}}| - 0.5 & \text{otherwise} \end{cases} \quad (4.3)$$

The models were trained for 21 epochs, using the SGD optimizer. The baseline model is evaluated on the EMOTIC test set. The performance on each one of the discrete categories in terms of *average precision* (AP, %), is presented in table 4.1, while the *average absolute errors* (AAE) on each one of the continuous VAD dimensions are presented in table 4.2. Current and all subsequent results for the continuous task, correspond to the ground truth values after having been rescaled in the range [0,1]. Letters ‘‘B’’ and ‘‘C’’ denote the *body* and *context* input streams respectively.

Emotion Categories	CNN inputs and $\mathcal{L}_{\text{cont}}$ type			
	B (L^2)	B (SL^1)	BC (L^2)	BC (SL^1)
1. Affection	21.80	16.55	21.16	27.85
2. Anger	06.45	04.67	06.45	09.49
3. Annoyance	07.82	05.54	11.18	14.06
4. Anticipation	58.61	56.61	58.61	58.64
5. Aversion	05.08	03.64	06.45	07.48
6. Confidence	73.79	72.57	7.97	78.35
7. Disapproval	07.63	05.50	11.00	14.97
8. Disconnection	20.78	16.12	20.37	21.32
9. Disquietment	14.32	13.99	15.54	16.89
10. Doubt/Confusion	29.19	28.35	28.15	29.63
11. Embarrassment	02.38	02.15	02.44	03.18
12. Engagement	84.00	84.59	86.24	87.53
13. Esteem	18.36	19.48	17.35	17.73
14. Excitement	73.73	71.80	76.96	77.16
15. Fatigue	07.85	06.55	08.87	09.70
16. Fear	12.85	12.94	12.34	14.14
17. Happiness	58.71	51.56	60.69	58.26
18. Pain	03.65	02.71	04.42	08.94
19. Peace	17.85	17.09	19.43	21.56
20. Pleasure	42.58	40.98	42.12	45.46
21. Sadness	08.13	06.19	10.36	19.66
22. Sensitivity	04.23	03.60	04.82	09.28
23. Suffering	04.90	04.38	07.65	18.84
24. Surprise	17.20	17.03	16.42	18.81
25. Sympathy	10.66	09.35	11.44	4.71
26. Yearning	07.82	07.40	08.34	08.34
Mean	23.86	22.36	24.88	27.38

Table 4.1: *Average precision* (AP, %), per emotional category, of the baseline model [59], as obtained on the EMOTIC test set.

Continuous Dimensions	CNN inputs and $\mathcal{L}_{\text{cont}}$ type			
	B (L^2)	B (SL^1)	BC (L^2)	BC (SL^1)
Valence	0.0537	0.0545	0.0546	0.0528
Arousal	0.0600	0.0630	0.0648	0.0611
Dominance	0.0570	0.0567	0.0573	0.0579
Mean	0.0569	0.0581	0.0589	0.0573

Table 4.2: *Average absolute error* (AAE), per emotional dimension, of the baseline model [59], as obtained on the EMOTIC test set.

4.1.2 EmotiCon

A notable increase in performance over the baseline model in the EMOTIC dataset, came along the EmotiCon model, introduced by Mittal et al. [71]. The main contributions of

the current model are related with the incorporation of multiple modalities in the task of context-based emotion recognition, including the face, pose as well as inter-agent interactions and socio-dynamic context. Additionally the information from the various modalities were combined in a score-fusion manner by employing a modified cross-entropy-based loss function on each one of the separate modalities that underwent early fusion.

Semantic Context and the Attention Branch Network

Visual semantic context includes information about the scene the surrounding objects and the performed activities, excluding the primary agent depicted in an image. In order to capture this type of information, authors propose the use of the following schemes: the masking of the primary agent and the incorporation of an Attention Branch Network (ABN).

Firstly, masked versions I_{mask} of each original image I within the dataset are formed by either utilizing the already provided body bounding boxes or calculating them using a human pedestrian tracking method as discussed in 3.3.1. The masked image I_{mask} for a given input image I is calculated in the following way:

$$I_{\text{mask}} = \begin{cases} I(i, j) & \text{if } (i, j) \notin \text{bbox}_{\text{agent}} \\ 0 & \text{otherwise} \end{cases} \quad (4.4)$$

where the tuple (i, j) corresponds to the pixel locations and $\text{bbox}_{\text{agent}}$ denotes the bounding box of a particular agent in a scene.

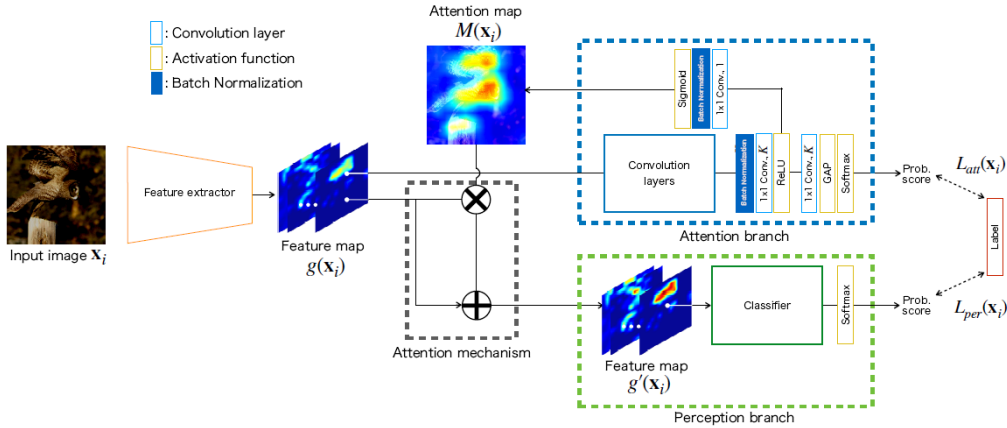


Figure 4.2: Overview of the Attention Branch Network (ABN). Source: [38].

Subsequently, the masked images I_{mask} are fed to the Attention Branch Network (ABN), as proposed by Fukui et al. [38]. ABN consists of three modules: the feature extractor, the attention branch, and the perception branch, while its functionality is associated with Class Activation Mappings (CAMs), as introduced by B. Zhou et al. [116]. The overall schematic of the ABN mechanism is illustrated in figure 4.2. The feature extractor contains multiple convolutional layers and extracts feature maps from an input image. The attention branch outputs the attention locations based on CAMs in the form of a weight map by using an attention mechanism. The perception branch outputs the probability of each class by receiving the feature map from the feature extractor and a corresponding attention map. Given an input image sample $\mathbf{X} \in \mathbb{R}^{D \times H \times W}$ (D : # of channels, H : height, W : width), the ABN mechanism produces CAMs during the training phase and applies them on the intermediate convolutional features $g(\mathbf{X}) \in \mathbb{R}^{C \times h \times w}$ of the feature extractor in order to highlight discriminative parts of the image. The FC layer used in the original CAM mechanism is replaced

with $K \times 1 \times 1$ convolutions, where K is the number of discrete emotion categories. The attention mechanism independently outputs both the attention maps $M(\mathbf{X}) \in \mathbb{R}^{1 \times h \times w}$ as well as independent category probability scores $p_{\text{att}} \in \mathbb{R}^{K \times 1}$ and calculates an attention loss in order to guide the attention maps to encode contextual information. The produced attention maps are multiplied bit-wise with each one of the channels of the intermediate convolutional feature maps:

$$\begin{aligned} g'_c(\mathbf{X}) &= M(\mathbf{X}) \cdot g_c(\mathbf{X}) \\ g''_c(\mathbf{X}) &= (1 + M(\mathbf{X})) \cdot g_c(\mathbf{X}) \end{aligned} \quad (4.5)$$

where $c \in \{1, \dots, K\}$ is the class index. The second formulation, $g''(\mathbf{X})$, can highlight the feature map at the peak of the attention map while preventing the lower value region of the attention map from degrading to zero. The output of the attention mechanism is fed to the remaining part of the network, namely the Perception Branch that is responsible for producing the final predictions.

Face and Gait Modalities

For every annotated agent within a particular image, additional features are extracted relative to the face and gait modalities. With the term gait, the authors refer to the body pose of a depicted agent.

As far as the face modality is concerned, the 2D locations of 67 facial landmarks are extracted, using the OpenFace [6] facial behavior toolkit, resulting in an input vector $\mathbf{m}_1 \in \mathbb{R}^{144}$. These vectors are used as input to a network consisting of three consecutive 1D convolutional layers, followed by three consecutive FC layers. Both convolutional and FC layers were followed by a batch-normalization layer and a ReLU non-linearity.

Subsequently, for the gait modality, the OpenPose [15] body pose estimation toolkit was used to extract the 2D locations of 25 body joints, resulting in a gait vector $\mathbf{m}_2 \in \mathbb{R}^{25 \times 2}$. The gait vectors were used as input to a Graph Convolutional Network (GCN) [57]. The output of the GCN was then fed to a CNN feature extractor, through which the final predictions based on the gait modality were produced.

Inter-Agent Interactions & Socio-Dynamic Context

The authors suggest that when an agent is located in an environment, surrounded by other agents, then their perceived emotional states are likely to change. If the agents share strong social relationships or are in close proximity, then they often coordinate their behaviors, while this effect degrades with interrelational and social distancing. Therefore, social interactions and proximity features can potentially aid the emotion recognition process.

The first proposed method for the encoding of socio-dynamic information is associated with the use of depth maps. More specifically, for each input image I , an estimated depth map I_{depth} is produced with the use of the MegaDepth [63] pre-trained model. Subsequently, the estimated depth maps I_{depth} are used as input to a CNN feature extractor, through which the final predictions for the depth modality are produced.

In addition to depth map-based representation, Graph Convolutional Networks (GCNs) were also used to model the proximity-based socio-dynamic interactions between agents. Given an input image containing n agents, the 2D coordinates of the body centroids of these depicted agents are calculated and used as input to a GCN which is comprised of 2 GCN layers, followed by two FC layers. The unweighted adjacency matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ used in the GCN model is calculated as follows:

$$A_{ij} = \begin{cases} e^{-d(v_i, v_j)} & \text{if } d(v_i, v_j) < \mu \\ 0 & \text{otherwise} \end{cases} \quad (4.6)$$

where v_i denotes the centroid of the i th agent and $d(v_i, v_j)$ denotes the pixel distance between the two agents.

Multiplicative Fusion

In real-life scenarios, people simultaneously process information from various modalities such as speech, text, face and body in order to infer the emotional state of a particular person. Taking that into consideration, the authors propose a way to combine the available input modalities in a complimentary manner to each other. More specifically, the predicted scores from each on of the available modalities are combined using a Multiplicative Fusion scheme [72].

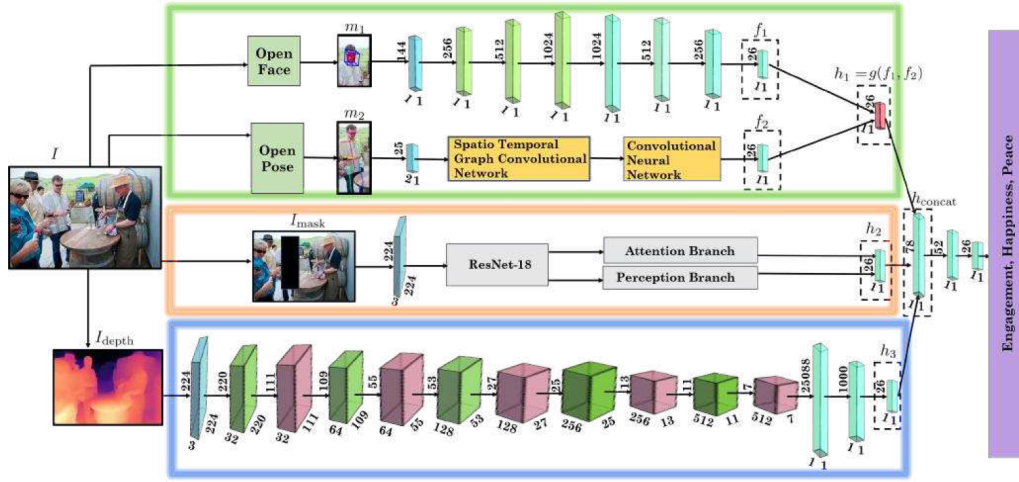


Figure 4.3: Overview of the EmotiCon model architecture. Source: [71].

Given n different input modalities with their corresponding inputs $\mathbf{m}_1, \mathbf{m}_2, \dots, \mathbf{m}_n$, then the applied multiplicative loss function is calculated as follows:

$$\mathcal{L}_{\text{multiplicative}} = - \sum_{i=1}^n (p_i^e)^{\frac{\beta}{n-1}} \ln p_i^e \quad (4.7)$$

where p_i^e denotes the predicted probability score of the emotion label e and i is the modality index. This scheme directly boosts the stronger and more relative modalities during the score fusion. In addition, the multiplicative loss function ignores the wrong predictions by simply not addressing them and rather focuses on modalities giving the right prediction. For well classified instances, $p_i^e \rightarrow 1$ and the predictions of the current modalities are left relatively intact, while if an example is miss-classified, then $p_i^e \rightarrow 0$ and the corresponding modality is suppressed. The score fusion of the multiple modalities with the application of the multiplicative loss function, constitutes the Multiplicative Fusion scheme.

Firstly, the individual prediction scores of the face and gait modalities are multiplicatively fused, producing the combined prediction probabilities \mathbf{h}_1 . Moreover, \mathbf{h}_1 together with the predicted probabilities $\mathbf{h}_2, \mathbf{h}_3$, from the context and socio-dynamic modalities respectively are concatenated, resulting in a score vector $\mathbf{h} = [\mathbf{h}_1, \mathbf{h}_2, \mathbf{h}_3]$. The score concatenation layer is followed by an FC layer of 52 hidden units and a classification layer of 26 hidden units. The network is trained using a combined loss function of the form:

$$\mathcal{L} = \lambda_1 \mathcal{L}_{\text{multiplicative}} + \lambda_2 \mathcal{L}_{\text{classification}} \quad (4.8)$$

where $\mathcal{L}_{\text{classification}}$ is the standard binary cross-entropy loss computed based on the final prediction scores.

An overview of the EmotiCon model architecture is shown in figure 4.3. The model was trained and evaluated on the EMOTIC dataset. The training configuration included the Adam optimizer with a learning rate of 10^{-4} , a batch size of 32 and a total of 75 epochs. The performance of the EmotiCon on each one of the discrete categories, with each one of the two suggested approaches for the socio-dynamic modality, namely GCN-based and Depth-based, is presented in table 4.3. The current model is also compared with the performance of the baseline model, presented in [59]. EmotiCon does not perform emotion regression on the continuous VAD dimensions.

Emotion Categories	Kosti et al. [59]	EmotiCon	
		GCN-Based	Depth-Based
1. Affection	27.85	36.78	45.23
2. Anger	09.49	14.92	15.46
3. Annoyance	14.06	18.45	21.92
4. Anticipation	58.64	68.12	72.12
5. Aversion	07.48	16.48	17.81
6. Confidence	78.35	59.23	68.65
7. Disapproval	14.97	21.21	19.82
8. Disconnection	21.32	25.17	43.12
9. Disquietment	16.89	16.41	18.73
10. Doubt/Confusion	29.63	33.15	35.12
11. Embarrassment	03.18	11.25	14.37
12. Engagement	87.53	90.45	91.12
13. Esteem	17.73	22.23	23.62
14. Excitement	77.16	82.21	83.26
15. Fatigue	09.70	19.15	16.23
16. Fear	14.14	11.32	23.65
17. Happiness	58.26	68.21	74.71
18. Pain	08.94	12.54	13.21
19. Peace	21.56	35.14	34.27
20. Pleasure	45.46	61.34	65.53
21. Sadness	19.66	26.15	23.41
22. Sensitivity	09.28	9.21	8.32
23. Suffering	18.84	22.81	26.39
24. Surprise	18.81	14.21	17.37
25. Sympath	4.71	24.63	34.28
26. Yearning	08.34	12.23	14.29
Mean	27.38	32.03	35.48

Table 4.3: Average precision (AP, %), per emotional category, of EmotiCon [71], as obtained on the EMOTIC test set and performance comparison with baseline model [59].

4.1.3 Leveraging Categorical Label Dependencies

The classification challenge presented by the EMOTIC dataset is two-fold as it includes regression on the continuous VAD dimensions and prediction of 26 discrete emotion categories. The discrete classification task is also a multi-label problem, where one instance can have more than one categorical emotion labels. However, unlike single-label emotion classification, in the multi-label setting the labels that appear simultaneously in an annotated instance of the dataset are always correlated and usually appear in groups. Thus, we focus on different ways of incorporating our prior knowledge of the publicly available EMOTIC dataset and its labels distribution in the learning process with the aim of boosting the overall recognition performance.

ML-GCN

To this end, one notable methodology was introduced by Chen et al. [22]. The proposed approach is based on a dual branch network which utilizes a feature extractor as a means of

image representation learning as well as a GCN that leverages label dependencies and learns inter-dependent classifiers. The overall design of the ML-GCN mechanism can be seen in figure 4.4.

The features of a given input image are learned through a CNN feature extractor with a set of trainable parameters θ_{CNN} . The last convolutional block of the CNN outputs feature maps $\mathbf{X} \in \mathbb{R}^{d \times h \times w}$ (d : # of output channels, h : height, w : width) and through the application of global average or max-pooling, a feature vector $\mathbf{x} \in \mathbb{R}^d$ is produced.

The second branch learns inter-dependent classifiers $\mathbf{W} = \{\mathbf{w}_i\}_{i=1}^C$, $\mathbf{W} \in \mathbb{R}^{d \times C}$, where d is the dimensionality of the extracted feature vector and C is the number of different classes present in the dataset. The branch is modeled by stacked GCN layers. The l th layer uses as input the hidden representations \mathbf{H}_l produced by the previous layer and outputs hidden representations using the following forward rule:

$$\mathbf{H}_{l+1} = h(\tilde{\mathbf{A}}\mathbf{H}_l\mathbf{W}_l) \quad (4.9)$$

where $h(\cdot)$ is a non-linear activation function, $\tilde{\mathbf{A}}$ is a normalized version of the original adjacency matrix and \mathbf{W}_l are the weights of the l th layer. As initial input for the GCN, the authors used the word embeddings of the C different labels in the form of a matrix $\mathbf{Z} \in \mathbb{R}^{C \times M}$, where M is the dimensionality of the label-level word embedding. For the acquisition of the word embeddings the authors suggested the use of a publicly available GloVe model, pre-trained on Wikipedia and Gigaword 5 data.

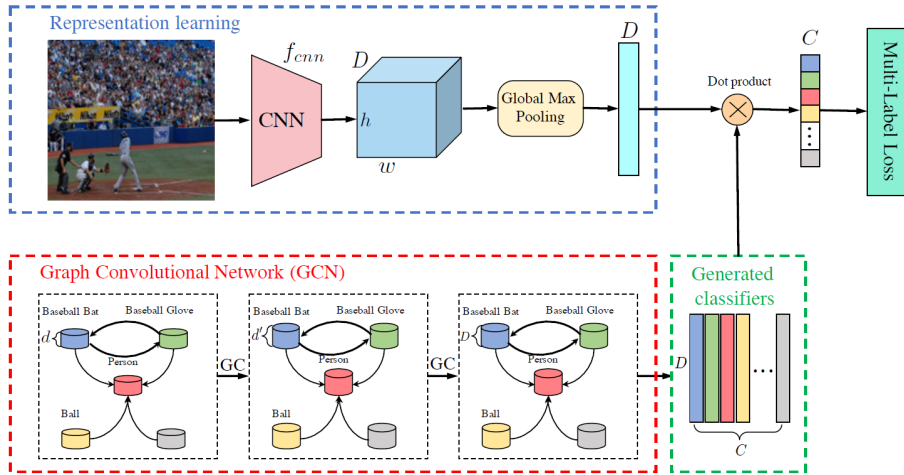


Figure 4.4: Overview of the ML-GCN model architecture. Source: [22].

The construction of the adjacency matrix is based on the prior knowledge available about label correlations within the given dataset. A co-occurrence matrix $\mathbf{M} \in \mathbb{R}^{C \times C}$ is formed by counting the occurrence of label pairs in the training set. Then the conditional probability matrix $\mathbf{P} \in \mathbb{R}^{C \times C}$ is calculated, with elements $P_{ij} = P(L_j|L_i) = \frac{M_{ij}}{N_i}$, where M_{ij} is equal to the number of times the labels L_i, L_j appear together, N_i denotes the occurrence times of label L_i and $P(L_j|L_i)$ denotes the conditional occurrence probability of label L_j given the occurrence of label L_i . The conditional probability matrix \mathbf{P} is binarized using a threshold τ so as to filter out noisy edges:

$$A_{ij} = \begin{cases} 0 & \text{if } P_{ij} < \tau \\ 1 & \text{if } P_{ij} \geq \tau \end{cases} \quad (4.10)$$

By using this binarized adjacency matrix, after the application of the GCN layers, the nodes' features will be a weighted sum of their own features and the adjacent nodes' features and

that can lead to potential over-smoothing. To alleviate this problem the authors propose the following re-weighting scheme:

$$A'_{ij} = \begin{cases} \left(p / \sum_{\substack{i=1 \\ i \neq j}}^C A_{ij} \right) \times A_{ij} & \text{if } i \neq j \\ 1 - p & \text{if } i = j \end{cases} \quad (4.11)$$

In that way, the weights assigned to the nodes themselves and their neighbors depend on the parameter p . When updating the nodes' features after each GCN layer, the nodes themselves will have fixed weights and the weights of the correlated nodes will be determined by the neighborhood distribution. More specifically, the weight A'_{nm} assigned to the edge that connects nodes n and m , of the label correlation graph, is proportional to p , as it is divided by the total number of nodes that are connected to node m , in an attempt to equally distribute the information transferred to node m from all neighboring nodes. In addition, when $p \rightarrow 1$, the weight of the node itself will not be considered while when $p \rightarrow 0$, neighboring information tends to be ignored. The output of the last GCN layer is the weight matrix $\mathbf{W} \in \mathbb{R}^{d \times C}$, with d denoting the dimensionality of the image deep feature representation, as extracted by the CNN. By applying the learned classifiers, the class confidence scores are obtained:

$$\hat{\mathbf{y}} = \mathbf{W}^\top \mathbf{x} \quad (4.12)$$

Given the corresponding ground-truth target vectors $\mathbf{y} \in \mathbb{R}^C$ with $y^i = \{0, 1\}$ and $i = 1, 2, \dots, C$, the network can be trained in an end to end fashion using a binary cross-entropy loss.

Categorical Label Embedding Loss

A different kind of methodology for leveraging categorical label dependencies during multi-label classification, is based on the application of metric learning on semantic embeddings built from words. One notable example of such an approach was introduced by Wei et al. [105]. In their work, they extracted both visual and text embeddings and used them in order to form joint emotion representations.

Given a standard CNN model for image feature extraction, visual embeddings $f_v(\mathbf{X}) \in \mathbb{R}^d$ can be obtained after the application of average or max-pooling on the CNN output feature maps, with \mathbf{X} denoting the corresponding input map and d being the dimensionality of the extracted feature vector. In that way, text-based embeddings can be used so as to aid the training of the aforementioned visual embeddings. The main idea is to ensure that the visual emotion features are compatible with the text-based features. For a given input image instance and the corresponding ground truth multi-hot target vector \mathbf{y} , a standard GloVe (Global Vectors) [85] pre-trained model can be employed so as to extract text-based label embeddings for each one of the different categorical labels, as in [37]. In the case of the EMOTIC dataset, the latter are comprised of discrete emotional states, namely $f_t(y^i) \in \mathbb{R}^k$, $i = 1, 2, \dots, C$ with k denoting the dimensionality of the GloVe model, i being a categorical label index and C denoting the total number of categories. Using an FC layer with weights $\mathbf{W}_{\text{emb}} \in \mathbb{R}^{k \times d}$, the visual embedding vector $f_v(\mathbf{X})$ can be mapped in the same space as the text embeddings. Subsequently, an additional loss term can be produced in the form of mean-squared error between the transformed visual embeddings and the average text embeddings for only the positive labels of the corresponding ground truth target vector:

$$\mathcal{L}_{\text{emb}} = \|\mathbf{W}_{\text{emb}} f_v(\mathbf{X}) - \frac{1}{|\mathcal{P}|} \sum_{y^i \in \mathcal{P}} f_t(y^i)\|_2^2 \quad (4.13)$$

where \mathcal{P} denotes the set of positive labels for a given target vector \mathbf{y} and $|\mathcal{P}|$ denotes the cardinality of that set. The whole network can be trained in an end-to-end manner by minimizing the combined loss between the embedding regularization term \mathcal{L}_{emb} and a standard binary cross-entropy based classification loss \mathcal{L}_{cls} , namely $\mathcal{L} = \mathcal{L}_{\text{cls}} + \lambda\mathcal{L}_{\text{emb}}$, where λ is weighting parameter.

4.2 Experimental Results

In the current section, we present our experimental results on the EMOTIC dataset. Across all of our experiments, we use the standard train, validation and test splits provided by the official distributors of the dataset. The first part of the presentation constitutes an ablation study relative to the effect of incorporating multiple modalities in the emotion recognition process.

4.2.1 Ablation Studies

The following experiments are dedicated in acquiring a deeper insight into the effect that each additional modality of affective information has on the emotion recognition process. The first step in this direction, is to verify that the addition of contextual information acts in a complementary way towards emotion recognition that is solely based on the body modality.

Context

For this type of experiment we adopt the configuration presented in [58], namely in the first case, we will train one network with body crops of the depicted people from each image, while in the second case we will train a network that consists of a body as well as a context branch, with the latter being fed the whole images as inputs. We choose to utilize 18-layer ResNet [48] feature extractors for both branches of our network. ResNets constitute state-of-the-art ConvNet backbones, offering a valuable trade-off between performance and computational complexity. In addition, the context feature encoding branch is pre-trained on the Places365-Standard [117] dataset while the body feature extractor is pre-trained on ImageNet [26]. We train both networks for 20 epochs, using the SGD optimizer, a constant learning rate of 10^{-2} , momentum equal to 0.9, weight decay equal to 10^{-5} and a batch size of 32. The modality aggregation method used is direct feature fusion. As far as discrete categories are concerned, the concatenated features are directly fed to the categorical prediction layer, where a binary cross-entropy loss is applied. In order to acquire continuous emotion predictions, the concatenated features are passed through a series of two FC layers, with 512 and 256 hidden units and a ReLU non-linearity, with the corresponding loss function being a smooth- L^1 loss as it has been found to perform better than the standard mean squared error (MSE). Additionally, the fusion network presented in [58] was neglected as it produced inferior results compared to the aforementioned one. A performance comparison between the two networks is presented in table 4.4, namely the *average precision* (AP, %) per category as well as the *mean average precision* (mAP, %). A performance comparison relative to the continuous dimension, measured in terms of *absolute average error* (AAE) is presented in table 4.5. In every case, the best performing epoch is selected based on the performance of the models on the validation set, namely the ratio between mAP and AAE. Letters “B” and “C” denote the *body* and *context* input streams, respectively.

It is obvious that the inclusion of contextual features boosts the model’s performance in categorical emotion recognition. For the majority of the emotion categories, the second model surpasses the AP score of the single branch counterpart with the exception of “Anticipation”,

Emotion Categories	CNN inputs	
	B	BC
1. Affection	28.72	32.67
2. Anger	11.04	11.21
3. Annoyance	15.73	16.39
4. Anticipation	56.38	56.30
5. Aversion	07.29	07.10
6. Confidence	75.33	73.59
7. Disapproval	13.21	14.93
8. Disconnection	25.12	28.55
9. Disquietment	17.72	18.80
10. Doubt/Confusion	19.59	21.44
11. Embarrassment	02.43	02.34
12. Engagement	84.69	85.37
13. Esteem	16.59	16.78
14. Excitement	68.76	68.40
15. Fatigue	11.52	12.66
16. Fear	06.03	06.40
17. Happiness	70.77	71.82
18. Pain	05.75	08.40
19. Peace	23.55	24.96
20. Pleasure	42.32	44.23
21. Sadness	18.42	25.41
22. Sensitivity	07.92	07.73
23. Suffering	18.68	24.81
24. Surprise	08.35	09.71
25. Sympath	12.82	13.40
26. Yearning	08.60	08.70
Mean	26.05	27.39

Table 4.4: Effect of including the context modality on *average precision* (AP, %), per emotional category, as obtained on the EMOTIC test set.

Continuous Dimensions	CNN inputs	
	B	BC
Valence	0.0904	0.0863
Arousal	0.0995	0.0982
Dominance	0.0929	0.0924
Mean	0.0943	0.0923

Table 4.5: Effect of including the context modality on *average absolute error* (AAE), per continuous emotional dimension, as obtained on the EMOTIC test set.

“Aversion”, “Embarrassment”, “Excitement” and “Sensitivity” which can be considered to constitute body-dominant emotions. The inclusion of the context modality also results in a reduction in absolute error in all three continuous VAD dimensions, confirming the assistive role of visual context in emotion perception.

Context Masking

Based on [71], the given input images should undergo a masking operation, during which the primary agents of each instance receive a mask based on the provided body bounding boxes. In order to evaluate the significance of this scheme, we use Class Activation Maps (CAMs), as proposed by Zhou et al. [116] so as to evaluate the ability of the context network to focus on context-related features. To this end, after having trained a network using the context-body modalities, without the application of any kind of masking operation, we perform inference and plot CAMs for multiple test images. Each displayed CAM comes with its corresponding categorical emotion label. Each row of figure 4.5 displays the whole unmasked image, the CAM from the context branch and the body branch overlaid on the original image and the corresponding body crop respectively. The CAMs of the first row correspond to the “Esteem” emotion category, the second row corresponds to the emotion of “Yearning” while the last row corresponds to the emotion of “Confidence”.

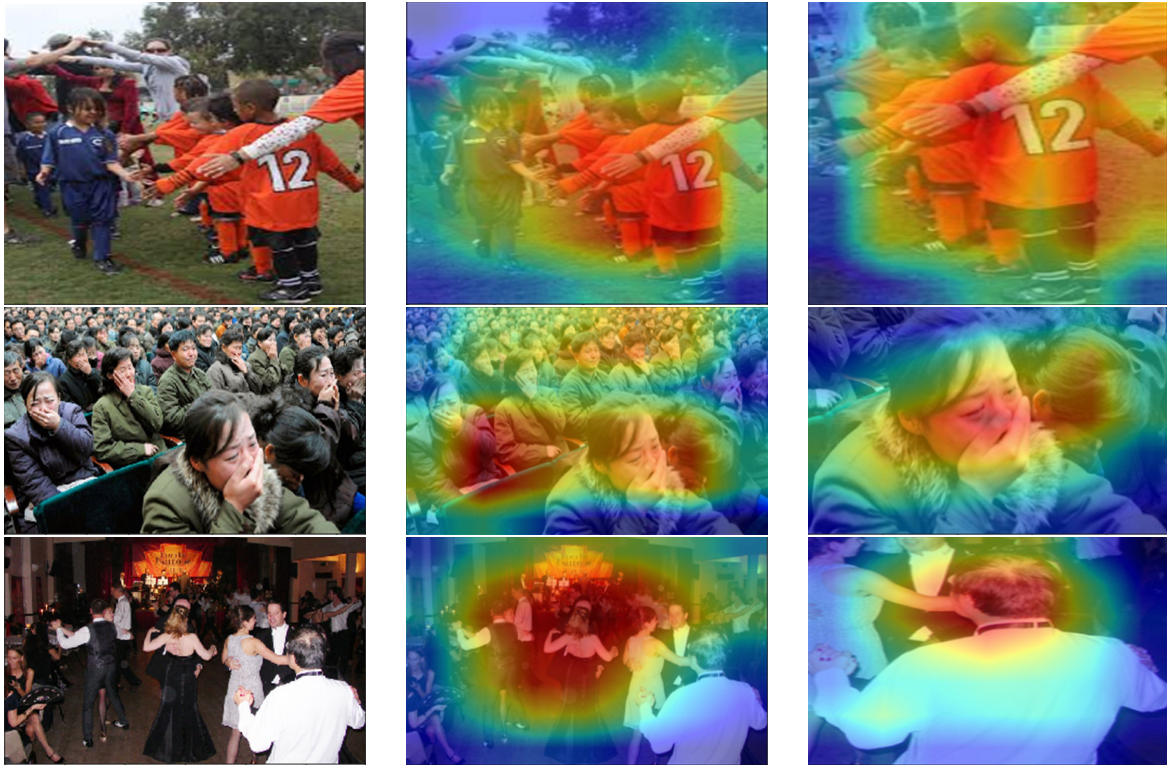


Figure 4.5: CNN localization using CAMs. The first column depicts the input images without the application of context masking, the second and third column depict the CAMs from the context and body branches, overlaid on the original input image and the corresponding agent body crop.

Although the context branch focuses on relevant features, we notice that it focuses mostly on bodies and faces, rather than scenes and objects. For example, in the first row it is obvious that both the context and body branches focus on the same parts of the image, something which is clearly not desirable. In order to push the context branch to focus elsewhere we use the aforementioned masking scheme, that is for every tuple of context and body images, we apply a mask over the corresponding primary body of the context image. In that way the context branch is obliged to focus on objects and scenes rather than bodies and faces which are more likely to be detected by the body accordingly. Again, after having trained a network using the context-body modalities, this time with the application of masking, we perform inference and plot CAMs for the same test images. The results of CAM visualization after the application of context masking are depicted in figure 4.6.

In all three of the above cases, context masking has effectively distinguished the attention regions of the two CNN branches. Additionally, evaluating the effect of masking on the recognition, we noticed a slight increase in mAP equal to 0.2% over the baseline model, while no change in AAE was detected. Agent masking is adopted for all subsequent experiments.

Facial Features

The face is considered to be the window to the soul, meaning that the face constitutes the most significant source of affective information along with the human body. Therefore, the inclusion of the face modality is expected to effectively assist the overall recognition process. For the detection, alignment and extraction of face crops, we use the publicly available OpenFace 2.0 toolkit, as introduced by Baltrušaitis et al. [6]. Moreover, we extend our current model by introducing an additional feature extraction branch, whose inputs are

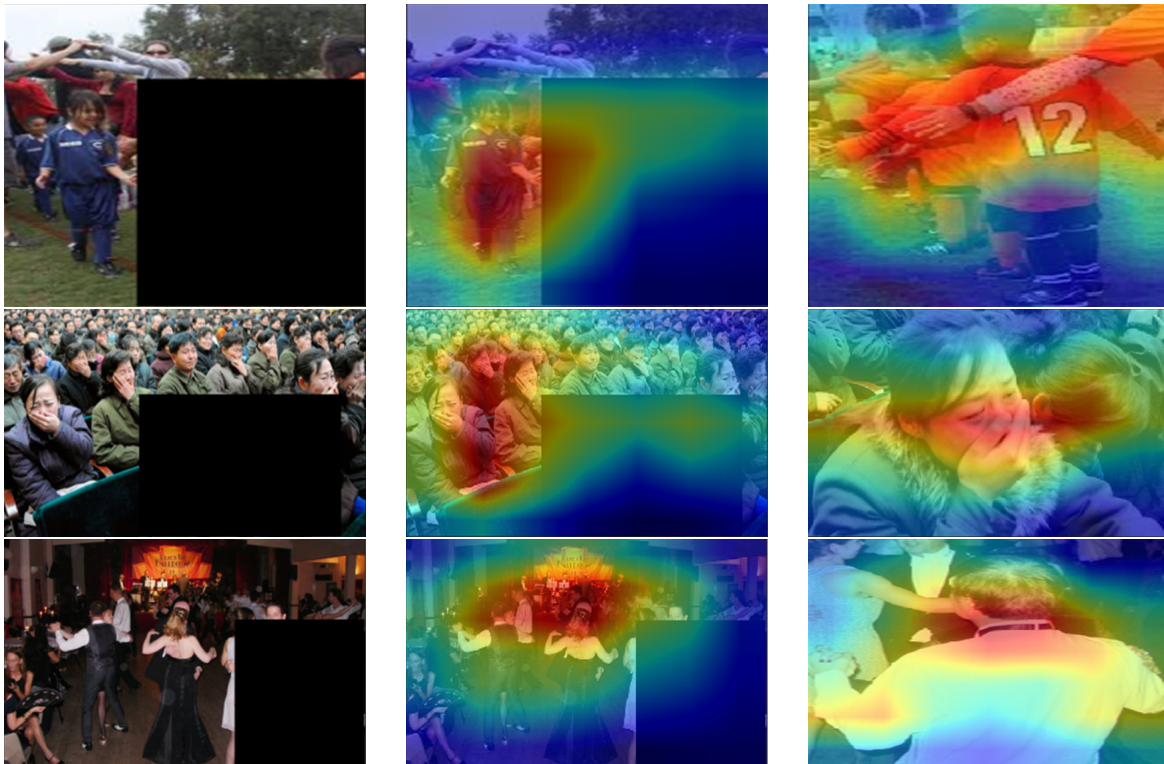


Figure 4.6: Application of context masking. The first column depicts the input images after having masked the primary agents, the second and third column depicts the CAMs from the context and body branches, overlaid on the masked input image and the corresponding agent body crop.

comprised of the facial crops of the corresponding agents. However, in the EMOTIC dataset, more than 25% of all the depicted people have their faces partially occluded or with very low resolution, resulting in the model’s failure in detecting any face for these particular instances. In this case, we feed the face feature extractor with a black proxy image, namely a 3-channel tensor filled with zeros. As a backbone for the feature extractor we employ a ResNet-18, pre-trained on the AffectNet [75] database. More specifically, we utilized 8 emotion categories, including “Neutral”, “Happy”, “Sad”, “Surprise”, “Fear”, “Disgust”, “Anger” and “Contempt” as well as the continuous annotations of valence and arousal, after rescaling them from the $[-1, 1]$ range to $[0, 1]$. We trained the network for 20 epochs using the Adam optimizer, with a learning rate of 10^{-5} , weight decay equal to 10^{-5} and a batch size of 64, reaching a maximum accuracy of 58.16% and minimum AAE of 0.164 on the 9th training epoch.

The modality aggregation method used is direct feature fusion and the training configuration is the same as the one employed in our previous experiments. The continuous prediction branch is now comprised of three FC layers, with 1,024-512-256 hidden units and ReLU nonlinearities. A performance comparison between the current model and our previous baseline model is presented in tables 4.6 and 4.7. Letter “F” denotes the *face* input stream.

According to table 4.6, the inclusion of the face modality results in an increase in AP for all discrete categories, with the exception of “Engagement” and “Peace” where a slight decrease of less than 1% is noticed and that could be considered by all means as transcendental. For all other emotions, there is an increase of up to 10% in AP, with “Anger”, “Happiness”, “Sadness”, “Annoyance” and “Fear” showcasing the largest boosts in performance. We speculate that this is related with the fact that the aforementioned emotions belong or are closely related to the basic eight emotional categories included in the AffectNet database, used in the pre-training phase. Subsequently, a decrease in AAE is also evident for the continuous emotion

Emotion Categories	CNN inputs	
	BC	BCF
1. Affection	32.67	34.34
2. Anger	11.21	21.27
3. Annoyance	16.39	20.03
4. Anticipation	56.30	56.98
5. Aversion	07.10	09.38
6. Confidence	73.59	74.20
7. Disapproval	14.93	18.26
8. Disconnection	28.55	29.49
9. Disquietment	18.80	20.22
10. Doubt/Confusion	21.44	23.03
11. Embarrassment	02.34	03.05
12. Engagement	85.37	84.77
13. Esteem	16.78	17.21
14. Excitement	68.40	69.14
15. Fatigue	11.52	13.53
16. Fear	06.40	09.26
17. Happiness	71.82	78.04
18. Pain	08.40	09.79
19. Peace	24.96	24.85
20. Pleasure	44.23	47.59
21. Sadness	25.41	29.35
22. Sensitivity	07.73	09.66
23. Suffering	24.81	26.64
24. Surprise	09.71	11.63
25. Sympath	13.40	13.56
26. Yearning	08.70	08.94
Mean	27.39	29.39

Table 4.6: Effect of including the face modality on *average precision* (AP, %), per emotional category, as obtained on the EMOTIC test set.

Continuous Dimensions	CNN inputs	
	BC	BCF
Valence	0.0863	0.0826
Arousal	0.0982	0.0991
Dominance	0.0924	0.0914
Mean	0.0923	0.0910

Table 4.7: Effect of including the face modality on *average absolute error* (AAE), per emotional dimension, as obtained on the EMOTIC test set.

prediction branch, confirming our assumptions relative to the face modality being a rich source of affective information.

Pose

Bodily expressions have been recognized as important for non-verbal communication, as changes in a person’s affective state are also reflected by changes in body posture, while some affective expressions may indeed be better conveyed by the body than the face. For example, for the emotion of fear, by focusing on the body posture, one can perceive the cause of the threat as well as any action tendency relative to the human subject. Therefore, we ought to study the role of the body in communicating emotions when affective displays containing a combination of facial expressions, posture or movement are presented, like in the case of the EMOTIC dataset.

To this end, we include pose information in the emotion recognition process in the form of 2D body joint coordinates, as extracted with the OpenPose [15] publicly available toolkit. More specifically, for each body crop instance provided by the EMOTIC dataset, we extract 2D coordinates for 25 body joints, as well as 21 joints for each one of the two hands. The coordinates that have been extracted undergo filtering based on the localization confidence score provided by the pose estimation model. All joints coordinates with a confidence score

Emotion Categories	CNN inputs		
	BCF	BCFP	
		GCN	Conv1D
1. Affection	34.34	34.85	34.51
2. Anger	21.27	23.11	22.99
3. Annoyance	20.03	21.12	20.45
4. Anticipation	56.98	57.63	56.70
5. Aversion	09.38	09.74	10.22
6. Confidence	74.20	72.07	73.46
7. Disapproval	18.26	18.75	17.24
8. Disconnection	29.49	29.39	30.01
9. Disquietment	20.22	20.49	21.34
10. Doubt/Confusion	23.03	22.87	23.05
11. Embarrassment	03.05	03.01	02.66
12. Engagement	84.77	85.03	84.30
13. Esteem	17.21	17.32	17.31
14. Excitement	69.14	70.24	69.64
15. Fatigue	13.53	13.26	13.96
16. Fear	09.26	09.55	10.32
17. Happiness	78.04	77.96	77.12
18. Pain	09.79	08.98	10.37
19. Peace	24.85	25.09	25.05
20. Pleasure	47.59	47.34	46.40
21. Sadness	29.35	28.86	30.89
22. Sensitivity	09.66	08.79	09.97
23. Suffering	26.64	26.92	30.96
24. Surprise	11.63	10.87	13.46
25. Sympath	13.56	13.18	13.83
26. Yearning	08.94	09.47	08.95
Mean	29.39	29.45	29.81

Table 4.8: Effect of including the pose modality on *average precision* (AP, %), per emotional category, as obtained on the EMOTIC test set. “GCN” and “Conv1D” denote the pose feature extraction method that has been used.

Continuous Dimensions	CNN inputs		
	BCF	BCFP	
		GCN	Conv1D
Valence	0.0826	0.0819	0.0787
Arousal	0.0991	0.0946	0.0969
Dominance	0.0914	0.0917	0.0900
Mean	0.0910	0.0894	0.0885

Table 4.9: Effect of including the pose modality on *average absolute error* (AAE), per emotional dimension, as obtained on the EMOTIC test set. “GCN” and “Conv1D” denote the pose feature extraction method that has been used.

of less than 10% are discarded and replaced with zeros. Moreover, the joint coordinates are normalized in the range $[0, 1]$, with $(0, 0)$ denoting the top left corner and $(1, 1)$ denoting the bottom right corner of an image. For the body joints, we use the nose as a point of reference as it constitutes the least occluded body joint, while for the hand joints, we use the wrists as points of reference. In addition we extract 7 LMA [60] features, namely the feet-hip, hands-shoulder, hands-head, centroid-pelvis, foot, hand distances and torso height. The aforementioned features constitute the *body components* and according to Luo et al. [67], they exhibit a strong correlation with the arousal emotion dimension. After concatenating all the above features, we obtain a 141-dim pose feature vector.

As far as feature extraction is concerned, we evaluate two different methods. The first one is based on 1D convolutions, while the second one utilizes a GCN for feature extraction. For the first method, we use three consecutive 1D convolutional layers, with 3×3 kernels, unitary padding and 256-512-1,024 filters. The last convolutional layer is followed by 3 FC layers, with 1,024-512-256 hidden units. Both convolutional layers and FC layers are followed by a batch-normalization layer and ReLU non-linearities. According to the second approach, each one of the 1D convolutional layers is replaced with a GCN layer, with 32-64-64 hidden

units. The output of the last GCN layer is flattened to form a feature vector that is fed into a series of 3 FC layers similarly to the first approach. All hidden layers are succeeded by batch-normalization and ReLU non-linearities. Both approaches are evaluated after being incorporated in the previous fusion network, introducing the pose modality in the recognition process, along with the existing context, body and face modalities. A performance comparison between the current two proposed models and our previously best model is presented in tables 4.8 and 4.9. Letter “P” now denotes the *pose* input stream.

With the introduction of pose information, the performance of the network increases in both the discrete categories as well as the continuous dimensions, with the latter showcasing a more significant boost over our previously best model. The pose branch based on 1D convolutions shows a 0.4% increase in mAP over its GCN-based counterpart and 0.5% increase over the previous model. As far continuous emotion recognition is concerned, again the pose branch based on 1D convolutions exhibits the best performance, with an AAE reduction of $2.5 \cdot 10^{-3}$ compared to the previously best model.

Scene Classification Scores and Attributes

As we have seen through our experiments, the context branch often fails to focus on actual scene features of the depicted image, despite the application of explicit agent masking. This is especially evident in image instances in which multiple people are depicted and consequently a large part of the image is covered by human bodies. In order to counter this problem, we try to directly incorporate scene-related scores and attributes, obtained with the usage of an 18-layer Wide-ResNet [110] that has been pre-trained on the Places365 [117] and Scene-Understanding (SUN) [82] databases.

The Places [117] database is a quasi-exhaustive repository of 10M scene photographs, labeled with 476 scene semantic categories. We use only a subset of the latter, namely the Places365-Standard which features 1.8M images and 365 scene categories. Moreover, the SUN attribute database [82] constitutes a subset of the SUN categorical database [107], comprised of 14,000 images that are annotated using a taxonomy of 102 scene attributes. Some of the categories that are included in the Places365 dataset are: amusement park, basketball court, cemetery, jail, cell, lecture room, museum, office, sauna, soccer field, etc. Some of the scene attributes included in the SUN dataset are: competing, socializing, working, exercise, praying, open-area, enclosed-area, stressful, etc. It is quite evident that the environment and scene depicted in an image can be closely related with the emotions of the people that are present. For example, an image of a funeral that is located at a cemetery, suggests a strong correlation between the above oppressive setting and the generally negative and sad feelings shared among the depicted people. Provided that our model is capable of leveraging the hinted correlations, incorporating scene specific information can potentially boost its overall emotion recognition performance.

Given an input image, the feature extractor, through each last convolutional block produces feature maps $\mathbf{Z} \in \mathbb{R}^{512 \times 14 \times 14}$. After the application of an average pooling layer, a deep feature vector representation $\mathbf{z} \in \mathbb{R}^{1 \times 512}$ is formed and fed to the FC layer with weights $\mathbf{W}_{\text{scenes}} \in \mathbb{R}^{512 \times 365}$, producing class confidence scores $\hat{\mathbf{y}}_{\text{scenes}} \in \mathbb{R}^{365}$. The corresponding class probabilities $\mathbf{p}_{\text{scenes}} \in [0, 1]^{365}$ are calculated after the application of the softmax function. The provided model also includes a set of pre-trained weights relative to the SUN dataset, namely $\mathbf{W}_{\text{attr}} \in \mathbb{R}^{512 \times 102}$, that can be used for the prediction of the confidence scores $\hat{\mathbf{y}}_{\text{attr}} \in \mathbb{R}^{102}$ for 102 scene attributes that are included in the SUN dataset. The corresponding attribute classification probabilities $\mathbf{p}_{\text{attr}} \in [0, 1]^{102}$ are again calculated after the application of the softmax function. Subsequently, the produced scene and attribute probability scores are concatenated with the extracted deep features from all the aforemen-

tioned input streams. After the initialization of the feature extractor with the aforementioned pre-trained models, weight parameters are kept frozen during the training phase.

Emotion Categories	CNN inputs		
	BCFP	+Scenes	+Attributes
1. Affection	34.51	35.46	34.34
2. Anger	22.99	22.81	23.51
3. Annoyance	20.45	20.08	21.94
4. Anticipation	56.70	56.42	56.45
5. Aversion	10.22	10.44	09.80
6. Confidence	73.46	75.22	73.33
7. Disapproval	17.24	20.03	20.42
8. Disconnection	30.01	28.99	30.12
9. Disquietment	21.34	20.83	20.24
10. Doubt/Confusion	23.05	22.84	22.91
11. Embarrassment	02.66	02.58	02.61
12. Engagement	84.30	84.77	85.69
13. Esteem	17.31	16.81	17.06
14. Excitement	69.64	69.96	69.97
15. Fatigue	13.96	14.05	14.66
16. Fear	10.32	08.93	10.47
17. Happiness	77.12	78.65	78.42
18. Pain	10.37	11.61	11.18
19. Peace	25.05	25.25	25.94
20. Pleasure	46.40	49.08	47.67
21. Sadness	30.89	29.96	31.23
22. Sensitivity	09.97	09.15	09.83
23. Suffering	30.96	28.19	31.59
24. Surprise	13.46	11.16	11.97
25. Sympath	13.83	13.33	14.01
26. Yearning	08.95	09.80	09.73
Mean	29.81	29.86	30.19

Table 4.10: Effect of including scene scores and attributes on *average precision* (AP, %), per emotional category, as obtained on the EMOTIC test set. “+Scenes” column corresponds to the model after incorporating only the scene scores, “+Attributes” column corresponds to the model after incorporating both scene scores and attributes.

The difference in performance given by the inclusion of the scene probabilities and scene attributes is going to be examined upon our current best performing network. The extracted *body, context, face, pose* features are going to be concatenated, firstly with the predicted scene classification probabilities and later on with both scene scores and attributes. The model structure remains the same, as well as the applied training configuration. A performance comparison of the current configurations with our previously best model is presented in tables 4.10 and 4.11.

Continuous Dimensions	CNN inputs		
	BCFP	+Scenes	+Attributes
Valence	0.0787	0.0758	0.0777
Arousal	0.0969	0.0980	0.0951
Dominance	0.0900	0.0889	0.0896
Mean	0.0885	0.0876	0.0874

Table 4.11: Effect of including the the scene scores and attributes on *average absolute error* (AAE), per emotional dimension, as obtained on the EMOTIC test set. “+Scenes” column corresponds to the model after incorporating only the scene scores, “+Attributes” column corresponds to the model after incorporating both scene scores and attributes.

The introduction of scene classification scores and attributes boosts the performance of the model in both classification and regression tasks. The inclusion of scene scores results in a trivial increase in mAP and a 10^{-3} decrease in AAE. Furthermore, incorporating the scene attributes along with the scene scores results in a 0.4% increase in mAP over our previously best model and no further decrease in AAE.

Socio-Dynamic Context

Mittal et al. [71] proposed the incorporation of inter-agent interactions and socio-dynamic context as an additional modality for categorical emotion recognition in the EMOTIC dataset. As it was described in section 4.1.2 of the theoretical analysis, they examined a GCN-based as well as a depth-based approach, with the latter exhibiting superior performance and an additional boost 3% boost in mAP over the GCN-based method. Due to the increased complexity of the GCN-based method as well as the lack of sufficient implementation details, we focus solely on the depth-based approach for socio-dynamic contextual feature extraction.

For each image of the EMOTIC dataset, we use the publicly available MegaDepth pre-trained model [63] with the aim of acquiring the corresponding depth estimation maps. As seen in figure 4.7, the given depth estimation model produces grayscale depth maps, where the same set of intensity values is repeated over three channels so as to match the input size requirements of subsequent CNN feature extractors.

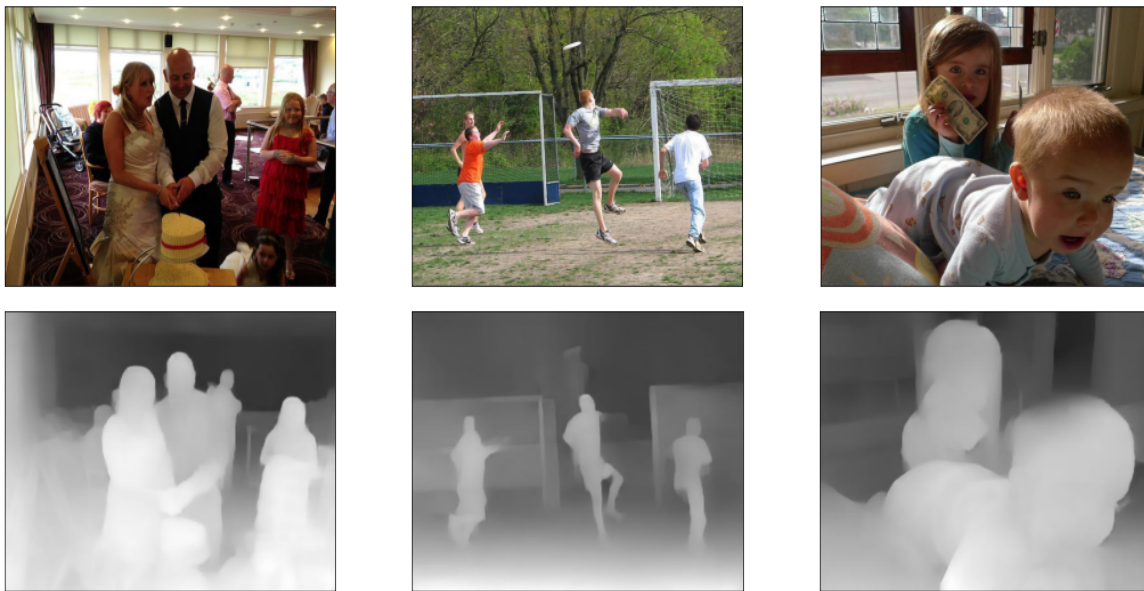


Figure 4.7: Examples of estimated depth maps for 3 random images of the EMOTIC train set. The top row illustrates the original RGB images while the bottom row illustrates the corresponding grayscale depth maps as produced by the MegaDepth [63] pre-trained model.

After manually calculating the mean and standard deviation of intensity values of the single repeated channel, we performed intensity normalization. For the extraction of socio-dynamic contextual features from the produced depth maps, we choose between two different CNN backbones, namely the custom 5-layer CNN as proposed in [71], as well as a standard ResNet-18 pre-trained on ImageNet. The provided description does not mention the kernel sizes and paddings for the convolutional layers of the first CNN backbone, but only the feature map output sizes after each convolutional block. We therefore improvise and use 5×5 kernels for the first convolutional layer, and 3×3 kernels for the remaining four layers with no zero padding. For the max-pooling layers, we use 2×2 kernels with unitary zero padding. All convolutional layers and the FC hidden layer are succeeded by a batch normalization layer and ReLU non-linearity. The features extracted from the estimated depth images are concatenated with the existing *context*, *body*, *face*, *pose* features, scene scores and attributes. We compare the AP and AAE performance of the current model, including socio-dynamic contextual features extracted from the two different CNN backbones, with our previously best model. The training configuration is identical to the one used in our previous experiments.

The results are presented in tables 4.12 and 4.13. Letters ‘‘S’’ and ‘‘A’’ denote the Places365 scene scores and SUN attributes respectively.

Emotion Categories	CNN inputs		
	Previous Best	+Depth	
		Baseline [71]	ResNet-18
1. Affection	34.34	33.89	35.50
2. Anger	23.51	22.07	22.68
3. Annoyance	21.94	21.46	21.39
4. Anticipation	56.45	56.96	56.39
5. Aversion	09.80	10.03	09.89
6. Confidence	73.33	72.64	74.06
7. Disapproval	20.42	18.81	18.61
8. Disconnection	30.12	29.76	31.26
9. Disquietment	20.24	21.30	20.03
10. Doubt/Confusion	22.91	23.90	23.28
11. Embarrassment	02.61	02.81	02.98
12. Engagement	85.69	84.11	84.60
13. Esteem	17.06	17.41	16.77
14. Excitement	69.97	69.68	70.21
15. Fatigue	14.66	13.76	13.72
16. Fear	10.47	08.69	08.53
17. Happiness	78.42	78.63	77.75
18. Pain	11.18	09.00	09.56
19. Peace	25.94	25.18	25.34
20. Pleasure	47.67	48.59	48.32
21. Sadness	31.23	27.57	32.43
22. Sensitivity	09.83	08.56	08.97
23. Suffering	31.59	27.94	28.08
24. Surprise	11.97	11.79	12.21
25. Sympath	14.01	12.65	13.12
26. Yearning	09.73	09.65	09.39
Mean	30.19	29.49	29.81

Table 4.12: Effect of including socio-dynamic contextual features on *average precision* (AP, %), per emotional category, as obtained on the EMOTIC test set. Two CNN backbones are evaluated for depth feature extraction, namely the baseline model [71] as well as a ResNet-18 pre-trained on ImageNet.

Continuous Dimensions	CNN inputs		
	Previous Best	+Depth	
		Baseline [71]	ResNet-18
Valence	0.0777	0.0803	0.0786
Arousal	0.0951	0.0947	0.0948
Dominance	0.0896	0.0901	0.0891
Mean	0.0874	0.0884	0.0876

Table 4.13: Effect of including the socio-dynamic contextual features on *average absolute error* (AAE), per emotional dimension, as obtained on the EMOTIC test set. Two CNN backbones are evaluated for depth feature extraction, namely the baseline model as well as a ResNet-18 pre-trained on ImageNet.

Although socio-dynamic context in the form of estimated depth maps is supposed to complement the other affective modalities, through our experiments, we came to the conclusion that this is not the case. Firstly, the model which utilizes a ResNet-18 backbone pre-trained on ImageNet, for the extraction of socio-dynamic contextual features from the estimated depth maps, performs better than its baseline counterpart in both categorical and continuous tasks. However, the introduction of the depth modality actually results in a reduction in the overall performance compared to our previously best model. Even though the current model performs slightly better in AP remotely for specific emotions like ‘‘Affection’’, ‘‘Confidence’’, ‘‘Disconnection’’, ‘‘Embarrassment’’, ‘‘Excitement’’, ‘‘Sadness’’ and ‘‘Surprise’’, all other emotion categories feature lower recognition rate and the overall mAP score drops by 0.4%. Additionally, the AAE performance of the current model along the continuous emotion dimensions does not improve.

The above results mark the end of our input modality-related experiments. For all of our

subsequent experiments we make explicit use of the *body*, *context*, *face*, *pose* input streams as well as the Places365 scene scores and SUN attributes. The use of socio-dynamic context is excluded, as it results in a decrease in network performance.

Attention Branch Network Extension

Mittal et al. [71] suggested the extension of the context feature encoding branch with the use of the Attention Branch Network (ABN), as it was originally proposed by Fukui et al. [38]. However, they did not provide any comparative study relative to the performance of their network, with and without the inclusion of the aforementioned module. We therefore conduct ablation experiments in order to examine whether the inclusion of the ABN mechanism will result in an improvement over our current best model. Furthermore, we consider the incorporation of the ABN mechanism not only in the context branch but in the body branch as well.

The following experiments are performed on the basis of our current best performing model, utilizing the same training configuration as before. Network training is driven by a combined loss of the form $\mathcal{L} = \mathcal{L}_{\text{cont}} + \mathcal{L}_{\text{cat}} + \lambda_{\text{att}}\mathcal{L}_{\text{att}}$, where \mathcal{L}_{att} denotes the binary cross-entropy loss produced from the attention branch, while $\mathcal{L}_{\text{cont}}$ and \mathcal{L}_{cat} denote the continuous and categorical losses produced at the final prediction layers after feature fusion is applied. The graphs of figure 4.8 depict the mAP and AAE scores obtained on the EMOTIC test set for various values of the weight parameter λ_{att} , after incorporating the ABN mechanism on the context feature encoding branch.

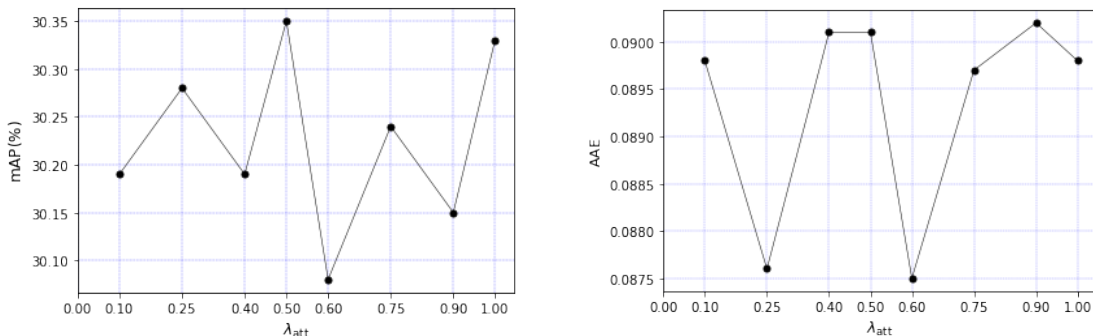


Figure 4.8: Mean average precision (mAP, %) and average absolute error (AAE) scores for various values of λ_{att} , as obtained on the EMOTIC test set after incorporating the ABN mechanism in the context feature encoding branch.

The maximum gain in mAP performance is achieved for $\lambda_{\text{att}} = 0.5$, at the cost of an increase in AAE equal to 0.005 over our previous best result. For that value of the aforementioned weight parameter, tables 4.14 and 4.15 present the experimental results associated with the extended networks in comparison with our current best model. For the “BC-ABN” configuration, weight parameter λ_{att} is divided in half for both branches so as to keep the total attention loss at the same level of magnitude as in the “C-ABN” configuration, for a more fair comparison.

The introduction of the attention loss \mathcal{L}_{att} results in the network diverging away from the continuous emotion prediction task due to an overwhelming increase of categorical losses during the training procedure. Additionally, the maximum performance increase in mAP is limited to 0.2% over our previous best model. As the performance boost associated with the categorical task is relatively small and the AAE scores tend to increase, we decide to exclude the ABN mechanism from all of our following experiments. The failure of the aforementioned module in providing a considerable difference in the overall performance of the network may

Emotion Categories	Network Configuration		
	Without ABN	+ C-ABN	+ BC-ABN
1. Affection	34.34	36.08	35.90
2. Anger	23.51	23.15	24.51
3. Annoyance	21.94	21.75	22.47
4. Anticipation	56.45	57.14	57.30
5. Aversion	09.80	10.17	10.22
6. Confidence	73.33	73.28	72.88
7. Disapproval	20.42	18.29	19.22
8. Disconnection	30.12	30.39	30.08
9. Disquietment	20.24	20.95	21.26
10. Doubt/Confusion	22.91	23.39	23.01
11. Embarrassment	02.61	02.86	02.54
12. Engagement	85.69	85.03	84.75
13. Esteem	17.06	17.35	16.86
14. Excitement	69.97	70.79	70.59
15. Fatigue	14.66	13.84	14.26
16. Fear	10.47	09.93	09.32
17. Happiness	78.42	78.87	79.24
18. Pain	11.18	11.31	09.69
19. Peace	25.94	25.78	25.74
20. Pleasure	47.67	48.71	49.59
21. Sadness	31.23	31.95	31.26
22. Sensitivity	09.83	09.98	09.40
23. Suffering	31.59	31.93	29.10
24. Surprise	11.97	12.26	11.73
25. Sympathy	14.01	14.17	14.67
26. Yearning	09.73	09.79	09.68
Mean	30.19	30.35	30.20

Table 4.14: Performance comparison in terms of *average precision* (AP, %), per emotional category, of different network configurations after the incorporation of the Attention Branch Network (ABN) in the context feature encoding branch (“C-ABN”) and body branch (“BC-ABN”). Results are obtained on the EMOTIC test set, using $\lambda_{\text{att}} = 0.5$.

Continuous Dimensions	Network Configuration		
	Without ABN	+ C-ABN	+ BC-ABN
Valence	0.0777	0.0852	0.0819
Arousal	0.0951	0.0943	0.0947
Dominance	0.0896	0.0907	0.0899
Mean	0.0874	0.0901	0.0888

Table 4.15: Performance comparison in terms of *average absolute error* (AAE), per emotional dimension, of different network configurations after the incorporation of the Attention Branch Network (ABN) in the context feature encoding branch (“C-ABN”) as well as in the body branch (“BC-ABN”). Results are obtained on the EMOTIC test set, using $\lambda_{\text{att}} = 0.5$.

be associated with the fact that the original ABN mechanism was implemented for multi-class single-label rather than multi-label prediction tasks.

Deeper Convolutional Backbones

At this section we investigate whether the utilization of deeper CNN backbones for feature extraction will result in an increase in the performance of the network in both categorical and continuous tasks. More specifically, for the context feature encoding branch we utilize a ResNet-50 pre-trained on the Places365 dataset, for the body branch we adopt a standard ResNet-50 pre-trained on ImageNet, while for the facial feature encoding branch we use a ResNet-50 which we have manually pre-trained on the AffectNet database. The latter was trained for 20 epochs with the Adam optimizer, learning rate equal to $5 \cdot 10^{-5}$ and weight decay equal to 10^{-5} , reaching a maximum accuracy of 60.16% and a minimum AAE of 0.140 on the validation set.

The network configuration includes the *context*, *body*, *face*, *pose* input streams as well as the Places365 scenes classification scores and SUN attributes. All extracted features are

combined using the standard early fusion scheme. An additional change in the network architecture is made relative to the FC layers in the continuous emotion prediction branch, where two more layers are added on top of the existing ones with 4,096 and 2,048 hidden units, followed by ReLU activations. The corresponding model is trained with the SGD optimizer for 20 epochs with weight decay equal to 10^{-5} and an initial learning rate of 10^{-2} which is reduced by a factor of 0.2 after 5 epochs. In table 4.16 we present only the average performance statistics of mAP and AAE of the current model in comparison with its previously best and shallower counterpart.

Evaluation Metrics	CNN Backbones	
	ResNet-18	ResNet-50
mAP \uparrow	30.19	30.74
mAAE \downarrow	0.0874	0.0873

Table 4.16: Effect of utilizing deeper CNN backbones for feature extraction in performance, measured in terms of *mean average precision* (mAP, %) and *mean average absolute error* (mAAE), as obtained on the EMOTIC test set.

Both categorical and continuous prediction branches benefit from the deeper architectures as expected, with the latter indicating a more significant improvement. Limitations in available GPU memory inhibit the usage of even deeper CNN backbones such as the ResNet-101 and ResNet-152, which would probably provide additional boosts in performance.

ML-GCN Network Extension

As a subsequent step in the process of improving the performance of the current model, we try to infuse additional prior knowledge into our model and leverage the categorical label dependencies existent within the EMOTIC dataset. To this end, we conduct a series of experiments which aim at adapting the widely used and promising ML-GCN module, as introduced it by Chen et al. [22], to our current best model.

In order to find optimal parameters for the adjacency matrix binarization threshold τ and probability p , we conducted experiments with a ResNet-50, feeding it body crops while having the ML-GCN attached to the categorical classification layer. The categorical label embeddings were acquired through a 300-dim GloVe model pre-trained on Wikipedia and Gigaword 5 data. Furthermore, two GCN layers are used, with 1,024 and 2,048 hidden units, and a Leaky ReLU non-linear activation function. Figure 4.9 provides graphs that depict the performance of the aforementioned network, in terms of mAP, for various values of the parameters τ and p . Firstly, we fixed the probability parameter $p = 0.25$ as it was the default suggested value and varied the threshold parameter τ . After finding the optimal threshold, we fixed τ to its optimal value and varied p instead.

The best parameter combination proved to be $\tau = 0.4$ and $p = 0.25$, reaching a maximum mAP of 28.3% in the EMOTIC test set and providing nearly 1% boost in mAP performance over the barebones body ResNet-50. With this hyperparameter configuration, we incorporate the ML-GCN mechanism on our previous best model. The number of GCN layers remains the same but the number of hidden units is increased to 2,048 and 6,867, with the latter being the dimensionality of the concatenated feature vector, resulting from the early fusion of the extracted feature vectors corresponding to all of the input modalities. The training configuration featured 20 epochs using the SGD optimizer, with momentum equal to 0.9, weight decay equal to 10^{-5} and with an initial learning rate of $5 \cdot 10^{-3}$ that was reduced by a factor of 0.2 after 5 epochs. Tables 4.17 and 4.18 present a performance comparison of our current model, before and after the addition of the ML-GCN module.

The incorporation of the ML-GCN module with the aim of leveraging categorical label

4.2. EXPERIMENTAL RESULTS

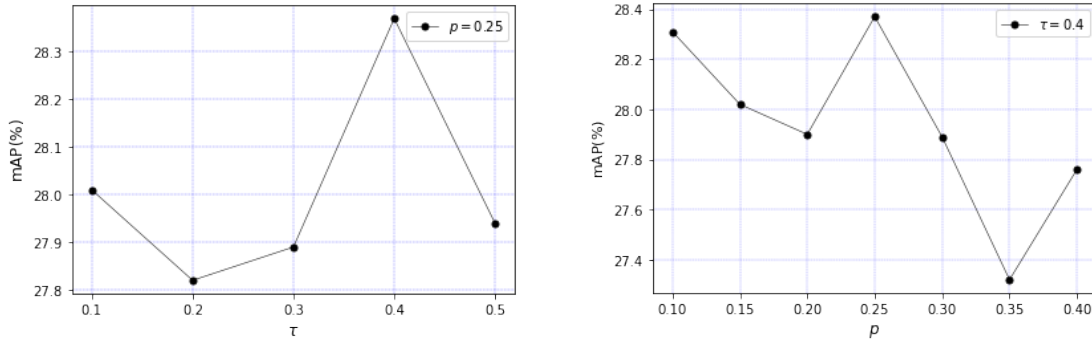


Figure 4.9: Mean average precision (mAP, %) scores for various combinations of the parameters τ and p as obtained on the EMOTIC test set after incorporating the ML-GCN module in the the body feature encoding network.

Emotion Categories	Network Configuration	
	Previous Best	ML-GCN
1. Affection	34.79	33.60
2. Anger	22.05	22.24
3. Annoyance	21.70	23.02
4. Anticipation	57.64	57.76
5. Aversion	09.90	10.81
6. Confidence	74.87	76.19
7. Disapproval	19.94	20.32
8. Disconnection	31.04	31.36
9. Disquietment	22.11	21.78
10. Doubt/Confusion	22.97	23.58
11. Embarrassment	02.70	03.08
12. Engagement	84.96	85.77
13. Esteem	16.99	17.07
14. Excitement	71.03	71.83
15. Fatigue	14.54	15.54
16. Fear	09.39	10.06
17. Happiness	78.92	79.22
18. Pain	11.12	12.03
19. Peace	25.16	25.24
20. Pleasure	48.79	49.53
21. Sadness	35.54	33.82
22. Sensitivity	09.70	09.30
23. Suffering	36.83	38.36
24. Surprise	13.13	12.72
25. Sympathy	13.46	13.99
26. Yearning	10.03	10.02
Mean	30.74	31.09

Table 4.17: Performance comparison in terms of *average precision* (AP, %), per emotional category, of different network configurations after the incorporation of the ML-GCN module in our previously best model, as obtained on the EMOTIC test set. The selected parameter settings are $\tau = 0.4$ and $p = 0.25$.

Continuous Dimensions	Network Configuration	
	Previous Best	ML-GCN
Valence	0.0788	0.0812
Arousal	0.0934	0.0947
Dominance	0.0898	0.0902
Mean	0.0873	0.0885

Table 4.18: Performance comparison in terms of *average absolute error* (AAE), per emotional dimension, of different network configurations after the incorporation of the ML-GCN module in our previously best model, as obtained on the EMOTIC test set. The selected parameter settings are $\tau = 0.4$ and $p = 0.25$.

dependencies resulted in an increase in AP performance for almost all emotion categories except of “Affection”, “Disquietment”, “Sadness”, “Sensitivity” and “Surprise”, where small decreases in score are noticed. The mean performance of the network after the aforementioned

modification is increased by 0.3% over our previous best model. However an increase in the continuous emotion prediction metrics for all dimensions is evident. After closer inspection of the network’s behavior, we notice that this drop in AAE performance is most likely caused by the fact that the categorical prediction branch seems to learn a lot faster compared to the continuous branch, causing the two sub-networks to reach their peak performance at different epochs.

Categorical Label Embedding Loss

On top of our previous model, we investigate whether the addition of a categorical label embedding loss [37], [105], will result in an increase in recognition performance. The categorical label embeddings are still provided by a 300-dim GloVe model, pre-trained on Wikipedia and Gigaword 5 data. After experimenting with various configurations we came down to the conclusion that the direct application of an embedding loss on the final fused feature vector does not constitute the best practice as it may contain irrelevant elements due to the inability of the OpenFace and OpenPose toolkits to successfully detect face regions and body joints in every input instance respectively. In that way, the categorical label embedding loss is applied solely with respect to the concatenated visual embeddings that originate from the context and body feature encoding branches. Therefore, an additional FC layer is incorporated with the purpose of linearly transforming the visual embeddings to the same dimensionality as the word embeddings. Network training is driven by a combined loss $\mathcal{L} = \mathcal{L}_{\text{cont}} + \mathcal{L}_{\text{cat}} + \lambda_{\text{emb}}\mathcal{L}_{\text{emb}}$, where \mathcal{L}_{emb} is the categorical label embedding loss and λ_{emb} is the corresponding weight parameter.

Emotion Categories	Network Configuration	
	ML-GCN	+ Embedding Loss
1. Affection	33.60	34.02
2. Anger	22.24	22.22
3. Annoyance	23.02	22.91
4. Anticipation	57.76	57.94
5. Aversion	10.81	10.69
6. Confidence	76.19	76.31
7. Disapproval	20.32	20.35
8. Disconnection	31.36	31.61
9. Disquietment	21.78	22.08
10. Doubt/Confusion	23.58	23.95
11. Embarrassment	03.08	03.32
12. Engagement	85.77	85.85
13. Esteem	17.07	17.45
14. Excitement	71.83	71.70
15. Fatigue	15.54	15.98
16. Fear	10.06	10.41
17. Happiness	79.22	79.03
18. Pain	12.03	11.34
19. Peace	25.24	25.32
20. Pleasure	49.53	49.17
21. Sadness	33.82	33.66
22. Sensitivity	09.30	10.54
23. Suffering	38.36	38.78
24. Surprise	12.72	13.23
25. Sympathy	13.99	15.33
26. Yearning	10.02	10.41
Mean	31.09	31.29

Table 4.19: Performance comparison in terms of *average precision* (AP, %), per emotional category, of different network configurations after the addition of the embedding loss \mathcal{L}_{emb} on top of our ML-GCN model, as obtained on the EMOTIC test set. Results correspond to $\lambda_{\text{emb}} = 0.4$.

The graphs of figure 4.10 depict the mAP and AAE scores obtained on the EMOTIC test set for various values of the weight parameter λ_{emb} , after adding the categorical label embedding loss \mathcal{L}_{emb} . The best performance is achieved for $\lambda_{\text{emb}} = 0.4$. For this parameter

Continuous Dimensions	Network Configuration	
	ML-GCN	+ Embedding Loss
Valence	0.0812	0.0816
Arousal	0.0934	0.0949
Dominance	0.0898	0.0901
Mean	0.0885	0.0889

Table 4.20: Performance comparison in terms of *average absolute error* (AAE), per emotional dimension, of different network configurations after the addition of the embedding loss \mathcal{L}_{emb} on top of our ML-GCN model, as obtained on the EMOTIC test set. Results correspond to $\lambda_{\text{emb}} = 0.4$.

setting, tables 4.19 and 4.20 present performance comparisons between our standard ML-GCN model and the current one.

The addition of the categorical label embedding loss led to a maximum increase in mAP equal to 0.2% over our previous best model. Moreover, the current model exhibits a rather

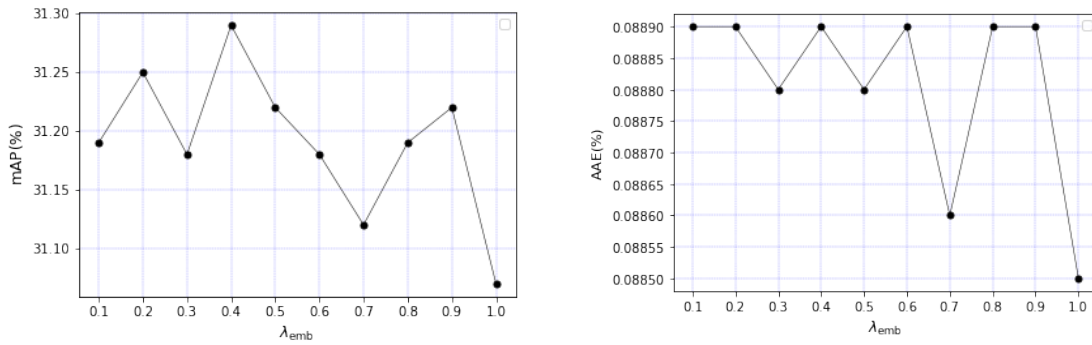


Figure 4.10: Mean average precision (mAP, %) and average absolute error (AAE) scores for various values of λ_{emb} , as obtained on the EMOTIC test set after adding the categorical label embedding loss \mathcal{L}_{emb} to the standard ML-GCN model.

negligible increase in AAE which is probably caused by the increased loss, namely categorical and embedding, that is used during the training phase of the categorical emotion prediction branch.

4.2.2 Cross-Study Performance Comparison

In the current section, we compare our best performing models on the categorical and continuous emotion prediction tasks with respect to the corresponding results, as presented by other published works of relative literature, namely Kosti et al. [59], Zhang et al. [113], Wei et al. [105] and Mittal et al. [71]. As some models focus solely on categorical emotion prediction, in order to provide more fair comparisons, we present our best results on the two tasks independently. Table 4.21 compares our model with the aforementioned pieces of relative literature on the basis of the categorical emotion prediction task, while table 4.22 makes the same comparisons relative to continuous emotion prediction.

Emotion Categories	Performance Comparison					
	Kosti et al. [59]	Zhang et al. [113]	Wei et al. [105]	Ours	Mittal et al. [71]	
					Without Depth	With Depth
1. Affection	27.85	46.89	-	34.02	41.83	45.23
2. Anger	09.49	10.87	-	22.22	11.41	15.46
3. Annoyance	14.06	11.27	-	22.91	17.37	21.92
4. Anticipation	58.64	62.64	-	57.94	67.59	72.12
5. Aversion	07.48	05.93	-	10.69	11.71	17.81
6. Confidence	78.35	72.49	-	76.31	65.27	68.85
7. Disapproval	14.97	11.28	-	20.35	17.35	19.82
8. Disconnection	21.32	26.91	-	31.61	41.46	43.12
9. Disquietment	16.89	16.94	-	22.08	12.69	18.73
10. Doubt/Confusion	29.63	18.68	-	23.95	31.28	35.12
11. Embarrassment	03.18	01.94	-	03.32	10.51	14.37
12. Engagement	87.53	88.56	-	85.85	84.62	91.12
13. Esteem	17.73	13.33	-	17.45	18.79	23.62
14. Excitement	77.16	71.89	-	71.70	80.54	83.26
15. Fatigue	09.70	13.26	-	15.98	11.95	16.23
16. Fear	14.14	04.21	-	10.41	21.36	23.65
17. Happiness	58.26	73.26	-	79.03	69.51	74.71
18. Pain	08.94	06.52	-	11.34	09.56	13.21
19. Peace	21.56	32.85	-	25.32	30.72	34.27
20. Pleasure	45.46	57.46	-	49.17	61.89	65.53
21. Sadness	19.66	25.42	-	33.66	19.74	23.41
22. Sensitivity	09.28	05.99	-	10.54	04.11	08.32
23. Suffering	18.84	23.39	-	38.78	20.92	26.39
24. Surprise	18.81	09.02	-	13.23	16.45	17.37
25. Sympathy	04.71	17.53	-	15.33	30.68	34.28
26. Yearning	08.34	10.55	-	10.41	10.53	14.29
Mean	27.38	28.42	30.96	31.29	31.53	35.48

Table 4.21: Performance comparison in terms of *average precision* (AP, %), per emotional category, of our best model with other publicized works relative to categorical emotion recognition, on the basis of the EMOTIC test set.

The work of Mittal et al. [71] constitutes the current SOTA in the categorical emotion recognition task, utilizing the *context*, *body*, *face*, *pose* and *depth* modalities. Considering that our model does not utilize depth information, it is more appropriate to compare the latter with the second-to-last column which describes the SOTA performance excluding the use of estimated depth maps. We notice that our model achieves comparable results, falling shy of the current best of 31.53% mAP by a mere 0.2%. Additionally, it is worth noting that our model achieves SOTA recognition performance in the “Anger”, “Annoyance”, “Disapproval”, “Disquietment”, “Happiness”, “Sadness”, “Sensitivity” and “Suffering”, despite the lack of the depth modality.

Continuous Dimensions	Performance Comparison				
	Kosti et al. [59]	Zhang et al. [113]	Wei et al. [105]	Ours	Mittal et al. [71]
Valence	0.0528	0.07	-	0.0788	-
Arousal	0.0611	0.1	-	0.0934	-
Dominance	0.0579	0.1	-	0.0898	-
Mean	0.0573	0.09	-	0.0873	-

Table 4.22: Performance comparison in terms of *average absolute error* (AAE), per emotional dimension, of our best model with other publicized works relative to continuous emotion recognition, on the basis of the EMOTIC test set.

Subsequently, we also manage to achieve comparable results in the continuous emotion prediction task even though the gap to the best publicized result of Kosti et al. [59] is quite large. We presume that this large difference is caused by the increased complexity of our model as well as the accumulated categorical losses which we utilize and are necessary for the proper functionality of the categorical emotion prediction branch.

Chapter 5

Video-Based Visual Emotion Recognition in Context

In the last chapter, we tackled the problem of image-based visual emotion recognition in context. Through our experiments, we showed that the utilization of an early fusion scheme among various input modalities such as the body, context, face as well as the pose, cumulatively boost the recognition performance. Additionally, we exploited the statistical label dependencies within the EMOTIC dataset in two separate ways, namely by adapting the ML-GCN mechanism in the categorical emotion prediction branch, and by enforcing semantic congruity between the extracted deep feature representations and the categorical label word embeddings from a pre-trained GloVe model. The latter was achieved by imposing a metric learning-inspired mean-squared error loss between the extracted feature vectors and the aforementioned word embeddings. The combination of the above proposed methodologies produced satisfactory results that are comparable with the current state-of-the-art on the EMOTIC dataset.

The current chapter aims at extending the concept of context-based visual emotion recognition in the dynamic setting of video sequences. To this end, we choose to use the newly assembled Body Language Dataset (BoLD) as a frame of reference for both the required theoretical analysis as well as our experimental results. Our model implementation will be based on the widely used Temporal Segment Network (TSN) architecture as it constitutes a flexible framework capable of enriching our previous, already successful multi-modal feature fusion design with the ability of processing video sequences.

The structure of this chapter is as follows: in the first section we establish the necessary theoretical background associated with the deep learning practices and network architectures which will be later used in our own implementation while the second section will be dedicated solely to presenting our experimental results and comparing them with other published works of relative literature.

5.1 Related Work

5.1.1 Two-Stream Convolutional Networks

The two-stream convolutional network architecture, as introduced by Simonyan and Zissermann [94], constitutes a powerful and versatile learning framework which was originally proposed for video-based action recognition but can also be adapted for the task of dynamic affective computing. The core functionality of the two-stream architecture is based on the decomposition of video-sequences into their spatial and temporal components, namely the shape and appearance depicted within static frames as well as the motion across consecutive frames. Each video component is encoded separately by a deep ConvNet, while the final video-level predictions are formed after the application of a late fusion scheme upon the separate scores that are produced through each information stream. An overview of the two-stream architecture for video processing is depicted in figure 5.1.

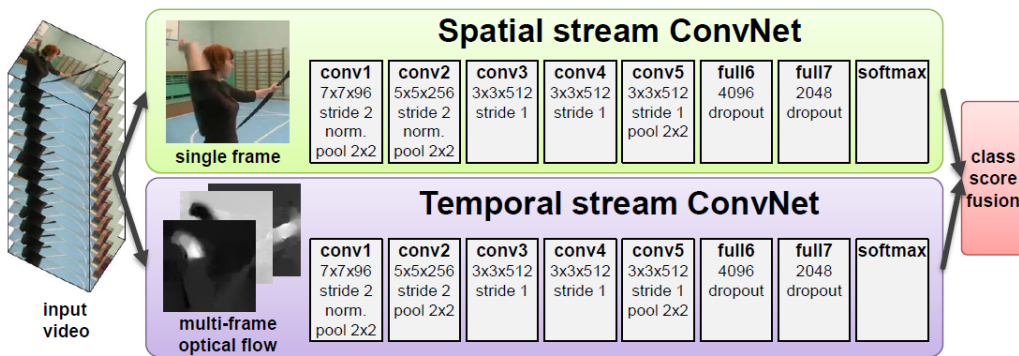


Figure 5.1: Overview of the two-stream convolutional architecture for action recognition in videos. Source: [94].

The spatial stream ConvNet operates on still image frames, effectively capturing information associated with the shape and appearance of people, objects as well as the scene and surrounding depicted environment. In that way, the spatial ConvNet constitutes a standard image classification architecture and therefore can be pre-trained on large-scale image recognition databases, depending on the type of features that need to be encoded relative to the task at hand.

On the other hand, the temporal stream ConvNet operates on stacks of optical flow displacement fields, or dense optical flow, as extracted from several consecutive video frames. Such type of input aims at explicitly capturing the motion between frames, thus making the overall recognition process easier as the network does not have to estimate motion implicitly. Dense optical flow can be formulated as a set of displacement vectors $\mathbf{d}_t = [d_t^x, d_t^y]^\top$, between pairs of consecutive frames t and $t + 1$, with d_t^x and d_t^y denoting the horizontal and vertical optical flow components, respectively. Additionally, $\mathbf{d}_t(u, v)$ denotes the displacement vector at point (u, v) and at frame t , which moves from the current location to the corresponding point at the following frame $t + 1$. Moreover, each optical flow component can be seen as two separate image channels, with their values discretized in the range $[0, 255]$. To represent the motion across a sequence of frames, the flow channels $d_t^{x,y}$ of L consecutive frames are stacked, forming a total of $2L$ input channels. More specifically, for an arbitrary frame τ of width w and height h , the temporal ConvNet input volume $\mathbf{I}_\tau \in \mathbb{R}^{w \times h \times 2L}$ is constructed in the following way:

$$\begin{aligned} I_\tau(u, v, 2k - 1) &= d_{\tau+k-1}^x(u, v) \\ I_\tau(u, v, 2k) &= d_{\tau+k-1}^y(u, v), u = [1; w], v = [1; h], k = [1; L] \end{aligned} \quad (5.1)$$

The formulation of equations 5.1 deals explicitly with forward optical flow, namely the displacement vector \mathbf{d}_t at frame t defines the location of its pixels in the following frame $t+1$. Bidirectional optical flow constitutes a natural extension of forward optical flow and allows the computation of displacement fields in the opposite temporal direction. In this case, the input volume I_τ is formed by stacking $L/2$ forward displacement fields between frames τ and $\tau + L/2$ as well as $L/2$ backward fields between frames τ and $\tau - L/2$, resulting in the same number of $2L$ input channels.

5.1.2 Temporal Segment Networks

Temporal Segment Networks (TSN), as proposed by Wang et al. [104], constitute an effective and efficient video-level framework for learning video representation, capable of capturing long-range temporal structures. Mainstream ConvNet frameworks usually focus on appearances and short-term motions, thus lacking the capacity to incorporate long-range temporal structure. This problem is often tackled through dense temporal sampling with a predefined sampling interval, which however leads to highly similar, unnecessary sampled frames and excessive computational cost. The TSN framework solves these issues by applying sparse temporal sampling on top of the already successful two-stream ConvNet architecture. An overview of the TSN framework is illustrated in figure 5.2.

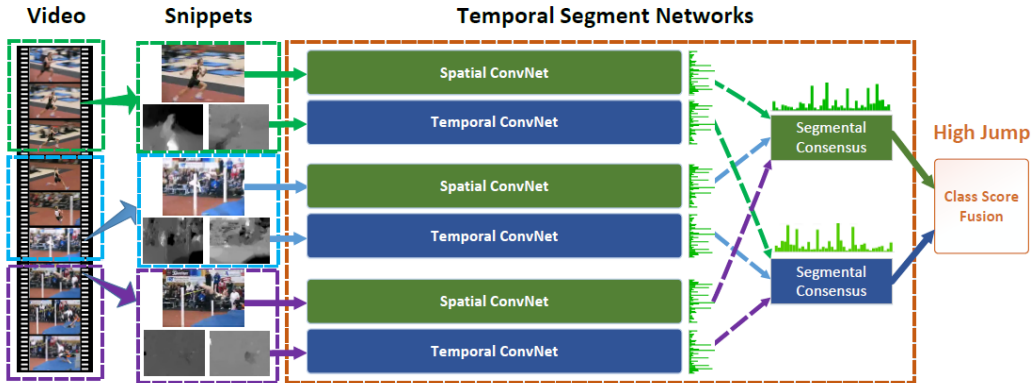


Figure 5.2: Overview of the TSN architecture. One input video is divided into K segments and a short snippet is randomly selected from each segment. The class scores of different snippets are fused by an the segmental consensus function to yield segmental consensus, which is a video-level prediction. Predictions from all modalities are then fused to produce the final predictions. Source: [104].

Framework Overview

Any given input video sequence is firstly divided into K segments $\{S_1, S_2, \dots, S_K\}$ of equal durations. The TSN operates on a set of K snippets, with each snippet constituting an instance which has been randomly sampled from a corresponding segment. More formally, the output of a TSN network is modeled as follows:

$$\text{TSN}(T_1, T_2, \dots, T_K) = \mathcal{H}\left(\mathcal{G}\left(\mathcal{F}(T_1; \mathbf{W}), \mathcal{F}(T_2; \mathbf{W}), \dots, \mathcal{F}(T_K; \mathbf{W})\right)\right) \quad (5.2)$$

where $\{T_1, T_2, \dots, T_K\}$ denote the snippets, \mathbf{W} denotes the network trainable parameters, \mathcal{F} denotes the snippet-level network predictions, \mathcal{G} denotes a segmental consensus function and

last but not least, \mathcal{H} denotes a prediction function, such as the softmax or the sigmoid function for the multi-class and multi-label classification tasks, respectively. For example, in the case of multi-label classification, given groundtruth target vectors $\mathbf{y} \in \{0, 1\}^C$, where C is the number of classes and a segmental consensus $\mathbf{G} = \mathcal{G}(\mathcal{F}(T_1; \mathbf{W}), \mathcal{F}(T_2; \mathbf{W}), \dots, \mathcal{F}(T_K; \mathbf{W}))$, the binary cross-entropy loss function takes the following form:

$$\mathcal{L}(\mathbf{y}, \mathbf{G}) = \sum_{c=1}^C [(1 - y_c)G_c + \ln(1 + e^{-G_c})] \quad (5.3)$$

where G_c denotes the class-specific score inferred from the scores of the same class on all the snippets, using the aggregation function \mathcal{G} .

Additionally, the segmental consensus function \mathcal{G} aims at aggregating the produced snippet level predictions through the application of one of several possible operators, such as simple average, weighted average, maximum, etc. This temporal segment network is differentiable or at least has subgradients, depending on the choice of \mathcal{G} . This allows the utilization of multiple snippets for the joint optimization of the model parameters \mathbf{W} with standard gradient descent algorithms. In the backpropagation process, the gradients of model parameters \mathbf{W} with respect to a loss function \mathcal{L} can be derived as follows:

$$\frac{\partial \mathcal{L}(\mathbf{y}, \mathbf{G})}{\partial \mathbf{W}} = \frac{\partial \mathcal{L}}{\partial \mathbf{G}} \sum_{k=1}^K \frac{\partial \mathcal{G}}{\partial \mathcal{F}(T_k; \mathbf{W})} \frac{\partial \mathcal{F}(T_k; \mathbf{W})}{\partial \mathbf{W}} \quad (5.4)$$

where K is the number of segments that the TSN is configured to use. The above equation guarantees that the parameter updates during backpropagation utilize the segmental consensus \mathbf{G} derived from the snippet-level predictions, thus the TSN learns from the entire video sequence and not just a few short snippets. Additionally, fixing the number of segments K enforces a sparse temporal sampling strategy that leads to a drastic reduction in the overall computational cost.

Aggregation Functions

Aggregation or consensus functions constitute a primary component of the TSN framework. In the current section we provide a brief analysis of the aggregation functions that will later be utilized in our model implementations. Amongst the aggregation functions which will be considered are: averaging pooling, linear weighting and attention weighting, as initially proposed by Wang et al. [103].

To begin with, given the segmental class scores $\mathbf{F}^k = \mathcal{F}(T_k; \mathbf{W})$ and a segmental consensus \mathbf{G} , with f_i^k being the i th class score corresponding to the k th segment and g_i being the i th dimension of \mathbf{G} , then for the case of average pooling, $g_i = \frac{1}{K} \sum_k f_i^k$. The gradient of the average pooling aggregation function g_i , with respect to f_i^k , is computed as follows:

$$\frac{\partial g_i}{\partial f_i^k} = \frac{1}{K} \quad (5.5)$$

Averaging pooling computes the mean activations from all snippets in order to provide video-level predictions. However, no distinction among the segments is made based on the quality of their content and as a result they all contribute equally for the generation of the final predictions.

Another approach is to form a weighted sum of the class scores from each segment, and let the network define the weights during backpropagation. In this case, the aggregation function is defined as $g_i = \sum_k \omega_k f_i^k$, where ω_k is the weight corresponding to the k th snippet. In this

case, the set of learnable parameters ω are introduced while the gradients of g_i , with respect to f_i^k and ω_k , are computed as follows:

$$\frac{\partial g_i}{\partial f_i^k} = \omega_k, \quad \frac{\partial g_i}{\partial \omega_k} = f_i^k \quad (5.6)$$

Compared to averaging pooling, in the current implementation, each snippet-level prediction is assigned with a different importance weight and therefore plays a unique role in the formulation of the video-level predictions.

Attention weighting constitutes an extension of linear weighting according to which the segment specific weights depend on the content of the corresponding snippets. In this case, the aggregation function is defined as $g_i = \sum_k A(T_k) f_i^k$, where $A(T_k)$ is the attention weight of snippet T_k . The gradients of g_i with respect to f_i^k and $A(T_k)$ are computed as follows:

$$\frac{\partial g_i}{\partial f_i^k} = A(T_k), \quad \frac{\partial g_i}{\partial \omega_k} = f_i^k \quad (5.7)$$

By denoting as $\mathbf{R} = \mathcal{R}(T_k)$, the visual embeddings which have been extracted from snippet T_k , then the attention weights are computed as follows:

$$e_k = \omega_{\text{att}} \mathcal{R}(T_k), \quad A(T_k) = \frac{\exp(e_k)}{\sum_{j=1}^K \exp(e_j)} \quad (5.8)$$

where ω_{att} is a set of new trainable parameters that are jointly learned with the rest of the network parameters. The gradient of $A(T_k)$ with respect to ω_{att} is calculated as follows:

$$\frac{\partial A(T_k)}{\partial \omega_{\text{att}}} = \sum_{j=1}^K \frac{A(T_k)}{\partial e_j} \mathcal{R}(T_j), \quad \frac{\partial A(T_k)}{\partial e_j} = \begin{cases} A(T_k)(1 - A(T_j)) & \text{if } k = j \\ -A(T_k)A(T_j) & \text{otherwise} \end{cases} \quad (5.9)$$

The attention model enhances the modeling capacity of the TSN framework by automatically estimating the importance weight of each snippet based on the video content. Moreover, as the attention mechanism is based on the visual embeddings \mathbf{R} , it leverages extra backpropagation information that guide the learning process and may accelerate the convergence of training.

5.1.3 Spatial-Temporal Graph Convolutional Networks

As mentioned in section 2.3.7, Graph Convolutional Networks (GCN) [57] generalize the convolutional operation on graphs of arbitrary structures, providing a flexible learning framework that can be applied on various image classification tasks. However, GCN operate on fixed rather than dynamic graph inputs, such as human skeleton sequences. Yan et al. [108] proposed an extension of graph neural networks to a spatial-temporal graph model, called Spatial-Temporal Graph Convolutional Network (ST-GCN), with the initial purpose of representing human skeleton sequences for action recognition.

A skeleton sequence is usually represented by 2D or 3D coordinates of each human joint in each frame. A hierarchical representation of skeleton sequences can be constructed in the form of an undirected spatial-temporal graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, with N joints and T frames, featuring both intra-body and inter-frame connections. The graph node set $\mathcal{V} = \{v_{ti} | t = 1, \dots, T, i = 1, \dots, N\}$ includes all the joints within a skeleton sequence. In addition, every node or joint in the graph comes with a set of features $F(v_{ti})$ that usually consist of the current joints coordinates, plus the corresponding detection confidence. The joints within one frame are connected with edges according to the connectivity of the human body structure which in turn depends on the dataset as well as the joint detection and tracking methodology that has been employed.

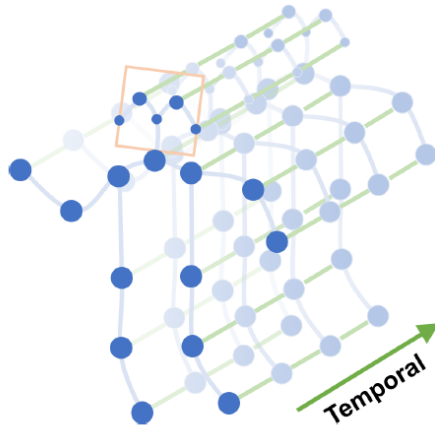


Figure 5.3: The spatial temporal graph of a skeleton sequence. Blue dots denote the body joints. The intra-body edges between body joints are defined based on the natural connections in human bodies. The inter-frame edges connect the same joints between consecutive frames. Joint coordinates are used as inputs to the ST-GCN. Source: [108].

For example, the OpenPose [15] toolkit offers multiple output formats, namely “BODY 25”, “COCO”, “MPI” that differ both in the number of output joints as well as the modeled inter-joint connections. Furthermore, each joint will be connected to the same joint in the consecutive frame. More specifically, the graph edge set \mathcal{E} is comprised of two subsets. The first subset features the intra-skeleton connections at each frame $\mathcal{E}_S = \{v_{ti}v_{tj} | (i, j) \in \mathcal{H}\}$ where \mathcal{H} is the set of naturally connected human joints. The second subset contains the inter-frame edges that connect the same joints in consecutive frames, $\mathcal{E}_F = \{v_{ti}v_{(t+1)i} | i = 1, \dots, N\}$. A spatial-temporal representation of a skeleton sequence is illustrated in figure 5.3.

Assuming that the output feature map of a spatial convolution operation has the same size as an input feature map and given a convolution operator with a kernel size of $K_v \times K_v$, then the output value of a single channel at a given spatial location \mathbf{x} can be written as:

$$f_{\text{out}} = \sum_{w=1}^{K_v} \sum_{h=1}^{K_v} f_{\text{in}}(\mathbf{p}(\mathbf{x}, w, h)) \cdot \mathbf{w}(h, w) \quad (5.10)$$

where the sampling function \mathbf{p} enumerates the neighbor joints at location \mathbf{x} and the weight function $\mathbf{w} : \mathbb{Z}^2 \rightarrow \mathbb{R}^c$ provides a weight vector in c -dimensional real space for computing the inner product with the sampled input vector of dimension c . On graphs, the sampling function can be defined on the neighbor set $B(v_{ti}) = \{v_{tj} | d(v_{ti}, v_{tj}) \leq D\}$ of node v_{ti} , where $d(v_{ti}, v_{tj})$ denotes the minimum length of any path from v_{ti} to v_{tj} . Therefore, the sampling function $\mathbf{p} : B(v_{ti}) \rightarrow \mathcal{V}$ can be written simply as $\mathbf{p}(v_{ti}, v_{tj}) = v_{tj}$.

As far the weight function is concerned, it can not be simply implemented by indexing a tensor of shape (c, K_v, K_v) due to the fact that graphs do not have a fixed spatial configuration. This problem is solved by using a graph labeling process. Instead of assigning a unique weight to each neighbor node, the neighborhood $B(v_{ti})$ of node v_{ti} is partitioned into a fixed number of K_v subsets, thus resulting in a mapping $l_{ti} : B(v_{ti}) \rightarrow \{0, \dots, K_v - 1\}$. In that way the weight function $\mathbf{w} : B(v_{ti}) \rightarrow \mathbb{R}^c$ can be implemented by indexing a tensor of shape (c, K_v) , with $\mathbf{w}(v_{ti}, v_{tj}) = \mathbf{w}'(l_{ti}(v_{tj}))$.

After having formulated the spatial graph convolutions, the concept of neighborhood needs to be extended so as to include temporally connected joints as well, namely $B(v_{ti}) = \{v_{qj} | d(v_{ti}, v_{tj}) \leq D, q - t \leq \lfloor \Gamma/2 \rfloor\}$, where Γ is a hyperparameter that controls the temporal range of the neighborhood. In addition, the labeling map needs to be modified so as to

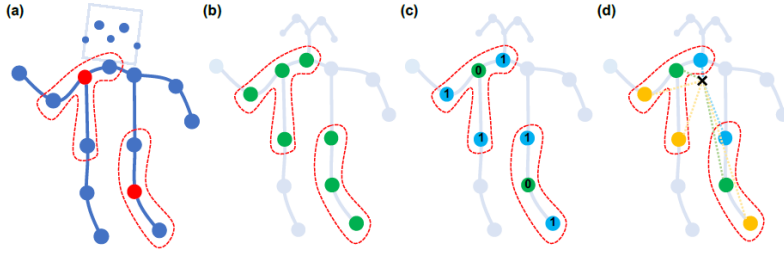


Figure 5.4: Partitioning strategies for constructing convolution operations. From left to right: (a) An example frame of input skeleton. Body joints are drawn with blue dots. The receptive fields of a filter with $D = 1$ are drawn with red dashed circles. (b) Uniform labeling partitioning strategy, where all nodes in a neighborhood has the same label (green). (c) Distance partitioning. The two subsets are the root node itself with distance 0 (green) and other neighboring points with distance 1 (blue). (d) Spatial partitioning. The nodes are labeled according to their distances to the skeleton gravity center (black cross) compared with that of the root node (green). Centripetal nodes have shorter distances (blue), while centrifugal nodes have longer distances (yellow) than the root node. Source: [108].

incorporate a temporal ordering on top of the existing spatial graph labeling process. The spatio-temporal labeling map is defined as $l_{ST}(v_{ti}) = l_{ti}(v_{tj}) + (q - t + \lfloor \Gamma/2 \rfloor) \times K_v$ where $l_{ti}(v_{tj})$ is the label map for the single frame case at for node v_{ti} .

In discussion of possible spatial partitioning strategies, three cases need to be highlighted, namely *uniform*, *distance* and *spatial* labeling. With *uniform* being the simplest labeling strategy, all joints that are connected through a limb belong in the same subset, resulting in $K_v = 1$ total subsets. Formally, $K_v = 1$ and $l_{ti}(v_{tj}) = 0, \forall i, j \in \mathcal{V}$. The *distance* labeling strategy extends the concept of neighboring joints, as pairs of joints that are connected through a sequence of limbs are also taken into consideration. Formally, $l_{ti}(v_{tj}) = d(v_{ti}, v_{tj})$, where D is the maximum allowed distance between two neighboring joints (we choose $D = 1$ for simplicity), resulting in a total of $K_v = D + 1$ subsets. Last but not least, the *spatial* configuration partitioning strategy divides the neighbor joint set in into three subsets, namely the root joint itself, the centripetal group which consists of the neighboring joints that are closer to the gravitational center of the skeleton than the root nodes and the opposite of the latter, called the centrifugal group. The skeleton center of gravity is calculated as the average coordinate of all joints locations at a single frame. All node distances from the gravity centers are averaged over all frames in the sequence. Formally:

$$l_{ti}(v_{tj}) = \begin{cases} 0 & \text{if } r_j = r_i \\ 1 & \text{if } r_j < r_i \\ 2 & \text{if } r_j > r_i \end{cases} \quad (5.11)$$

In practice, the gravitational center is replaced by a fixed root (the neck in our case). Given that the maximum allowed distance between two neighboring joints $D = 1$, then this results in $K_v = 3$ subsets. The aforementioned partitioning strategies are visualized in figure 5.4.

Lastly, according to the *spatial* labeling strategy, neighboring joints are distinguished based on their individual distances from a fixed root (the neck), resulting in $K_v = 3$ subsets.

In the spatial-temporal case, the input feature map \mathbf{H}_{in} of a ST-GCN unit is represented as a tensor of shape (C_{in}, T_{in}, V) , where C_{in} denotes the number of input channels, T_{in} denotes the number of frames in the skeleton sequence and V denotes the number of nodes. Firstly, the input tensor undergoes a $(K_v \cdot C_{out}) \times 1 \times 1$ spatial graph convolution operation, with C_{out} being the desired number of output channels and K_v being the number of joint subsets that are formed based on the chosen labeling strategy. The resulting tensor is reshaped into

$(K_v, C_{\text{out}}, T_{\text{in}}, V)$ and multiplied with the normalized adjacency matrix $\mathbf{D}^{-\frac{1}{2}} \hat{\mathbf{A}} \mathbf{D}^{-\frac{1}{2}}$, where $\hat{\mathbf{A}} = \mathbb{I} + \mathbf{A}$ and \mathbf{D} is a diagonal matrix with elements $D^{ii} = \sum_j \hat{A}^{ij}$. In case of the distance and spatial partitioning strategies ($K_v > 1$), the adjacency matrix \mathbf{A} is formed by stacking K_v matrices \mathbf{A}_k , with each one corresponding to one of the K_v joint subsets. If we ignore interlayer nonlinearities, then the aforementioned spatial convolution operation is equivalent to the original GCN [57] formula:

$$\mathbf{H}_{\text{out}} = \sum_k \mathbf{W}_k \mathbf{H}_{\text{in}} \mathbf{D}_k^{-\frac{1}{2}} \hat{\mathbf{A}}_k \mathbf{D}_k^{-\frac{1}{2}} \quad (5.12)$$

where \mathbf{W}_k are $C_{\text{out}} \times C_{\text{in}} \times 1 \times 1$ weight matrices (the multiplication is replicated T_{in} times in the temporal dimension and V times in the spatial dimension). $D_k^{ii} = \sum_j \hat{A}_k^{ij} + \alpha$ is the normalized diagonal matrix and α is set to 0.001 to avoid empty rows. Additionally, learnable edge importance weighting can be implemented simply by multiplying element-wise the adjacency matrices $\hat{\mathbf{A}}_k$ of Eq. 5.12 with a weight mask \mathbf{M} , namely $\hat{\mathbf{A}}_k \odot \mathbf{M}$. The output feature map resulting from the spatial graph convolution undergoes a $C_{\text{out}} \times \Gamma \times 1$ temporal convolution, with Γ denoting the temporal kernel size, completing the processing pipeline of a single ST-GCN unit.

5.1.4 Laban Movement Analysis

Laban Movement Analysis (LMA), sometimes called Laban/Bartenieff Movement Analysis, constitutes a framework for describing, visualizing, interpreting and documenting human movement, as it is based on the initial work of Laban and Ullman [60].

Laban movement analysis divides body movements into four separate components, namely “Body”, “Effort”, “Shape” and “Space”. More specifically, the “Body” category describes structural and physical characteristics of the human body while moving. This category is responsible for describing which body parts are moving, which parts are connected, which parts are influenced by others, and general statements about body organization. “Effort” comprises a system for understanding the more subtle characteristics about movement with respect to inner intention. For example, the difference between punching someone in anger and reaching for a glass is slight in terms of body organization – both rely on extension of the arm, whereas the attention to the strength of the movement, the control of the movement and the timing of the movement are very different. Furthermore, the way the body changes shape during movement is further experienced and analyzed through the “Shape” category. Lastly, the “Space” category features a greater theoretical depth in comparison with the rest of the LMA system, as it combines body movement analysis with ideas from the realm of Space Harmony. The latter constitutes a separate movement theory and practice which was created by Laban himself and was based on universal patterns of nature and of man as part of a universal design.

Luo et al. [67] utilized the LMA framework with the aim of analyzing human skeleton sequences for the task of visual emotion recognition in videos. Firstly, they extracted the 2D coordinates for 18 body joints, with $\mathbf{p}_i^t \in \mathbb{R}^2$ denoting the coordinate vector of joint i at frame t of a video sequence. As far as scaling is concerned, they normalized each pose by the average length of all visible limbs in each sequence. Let \mathcal{V} be the set of all limbs with visible joints and $|\mathcal{V}|$ its cardinality. For each sequence, of T frames, the scaling factor s and the normalized coordinate vectors are calculated in the following way:

$$s = \frac{1}{T|\mathcal{V}|} \sum_{(i,j) \in \mathcal{V}} \sum_t \|\mathbf{p}_i^t - \mathbf{p}_j^t\|, \quad \hat{\mathbf{p}}_i^t = \frac{\mathbf{p}_i^t}{s} \quad (5.13)$$

Subsequently, they extracted 54 LMA features for each frame of every sequence. An analytical listing of the extracted LMA features is presented in table 5.1. Features $f_1, f_2, f_3, f_4, f_8, f_9$ correspond to the body component and capture the pose configuration. For symmetric joints, the mean distances were used. The mean values were also used for the calculation of all other features that involved symmetric joints. For f_4 the centroid was calculated as the average of all visible joints while the pelvis is located in the middle of the two hips.

f_i	Description	f_i	Description
f_1	Feet-hip dist.	f_2	Hands-shoulder dist.
f_3	Hands dist.	f_4	Hands-head dist.
f_8	Centroid-pelvis dist.	f_9	Gait size (foot dist.)
f_{10}	Angles		
f_{29}	Shoulders velocity	f_{32}	Elbow velocity
f_{13}	Hands velocity	f_{12}	Hip velocity
f_{35}	Knee velocity	f_{14}	Feet velocity
f_{38}	Angular velocities		
f_{30}	Shoulder accel.	f_{33}	Elbow accel.
f_{16}	Hands accel.	f_{15}	Hip accel.
f_{36}	Knee accel.	f_{17}	Feet accel.
f_{39}	Angular accel.		
f_{31}	Shoulders jerk	f_{34}	Elbow jerk
f_{40}	Hands jerk	f_{18}	Hip jerk
f_{37}	Knee jerk	f_{41}	Feet jerk
f_{19}	Volume		
f_{21}	Volume (lower body)	f_{20}	Volume (upper body)
f_{23}	Volume (right side)	f_{22}	Volume (left side)
		f_{24}	Torso height

Table 5.1: Laban Movement Analysis (LMA) features (f_i : categories; dist.: distance; accel.: acceleration).

The second part of LMA features, corresponds to the effort component which captures body motion dynamics. Based on the normalized joint coordinates, normalized joint velocities \hat{v}_i^t , accelerations \hat{a}_i^t and jerks \hat{j}_i^t were calculated:

$$\mathbf{v}_i^t = \frac{\hat{\mathbf{p}}_i^{t+\tau} - \hat{\mathbf{p}}_i^t}{\tau}, \quad \mathbf{a}_i^t = \frac{\mathbf{v}_i^{t+\tau} - \mathbf{v}_i^t}{\tau}, \quad \mathbf{j}_i^t = \frac{\mathbf{a}_i^{t+\tau} - \mathbf{a}_i^t}{\tau} \quad (5.14)$$

$$\hat{v}_i^t = \|\mathbf{v}_i^t\|, \quad \hat{a}_i^t = \|\mathbf{a}_i^t\|, \quad \hat{j}_i^t = \|\mathbf{j}_i^t\|$$

Additionally, angles θ , angular velocities ω and accelerations α_θ were calculated for specific pairs of limbs, namely feet-knee and knee-hip, hip-neck and neck-shoulder, shoulder-elbow and elbow-wrist, shoulder-neck and neck-nose, neck-nose and nose-ear, nose-ear and ear-eye, hip-neck and neck-nose, knee-hip and hip-neck. Calculations were carried out as follows:

$$\theta^t(i, j, m, n) = \arccos \left(\frac{(\hat{\mathbf{p}}_i^t - \hat{\mathbf{p}}_j^t) \cdot (\hat{\mathbf{p}}_m^t - \hat{\mathbf{p}}_n^t)}{\|\hat{\mathbf{p}}_i^t - \hat{\mathbf{p}}_j^t\| \|\hat{\mathbf{p}}_m^t - \hat{\mathbf{p}}_n^t\|} \right)$$

$$\omega^t(i, j, m, n) = \frac{\theta^{t+\tau}(i, j, m, n) - \theta^t(i, j, m, n)}{\tau} \quad (5.15)$$

$$\alpha_\theta^t(i, j, m, n) = \frac{\omega^{t+\tau}(i, j, m, n) - \omega^t(i, j, m, n)}{\tau}$$

The third part of LMA features, the shape component, captures the body shape. More specifically, $f_{19} - f_{23}$ comprise a series of volume features and are approximated with the area of the bounding boxes that contain the corresponding set of joints.

Subsequently, descriptive statistics, namely the maximum, minimum, mean and standard deviation of each feature value were extracted resulting in 216-dim feature vectors that effectively describe an entire video sequence.

5.1.5 Baseline Model

After successfully assembling the BoLD dataset, Luo et al. [67] furthered their contributions by comparing various network configurations and finally providing a baseline model for the task of categorical and continuous emotion prediction. Among the examined methodologies are: usage of motion-based descriptors such as histograms of optical flow (HOF) and motion boundary histograms (MBH), skeleton-based learning through LMA features and the ST-GCN model and last but not least, two-stream convolutional architectures and the TSN extension.

As far evaluation metrics are concerned, for the task of categorical emotion prediction they used *average precision* (AP), namely the area under the precision-recall curve as well as the *area under the receiver operating characteristic* (ROC-AUC). For continuous emotion regression along the VAD dimensions, the *coefficient of determination* (R^2) is used. Performance comparison among different models is carried out on the basis of an aggregatory *emotion recognition score* (ERS) which is calculated as follows:

$$\text{ERS} = \frac{1}{2}(\text{m}R^2 + \frac{1}{2}(\text{mAP} + \text{mRA})) \quad (5.16)$$

where $\text{m}R^2$ denotes the *mean coefficient of determination* along the VAD dimensions while mAP and mRA denote the *mean average precision* and mean ROC-AUC over the 26 emotion categories respectively.

Model	Regression	Classification		ERS
	$\text{m}R^2\uparrow$	mAP \uparrow	mRA \uparrow	
TSN-Body	0.095	0.1702	0.6270	0.247
TSN-Body + LMA	0.101	0.1670	0.6275	0.249
TSN-Body+TSN-Face	0.101	0.1731	0.6346	0.252
TSN-Body+TSN-Face +LMA	0.103	0.1714	0.6352	0.253

Table 5.2: Baseline performance comparison among various network ensemble configurations, on the BoLD test set, as achieved by Luo et al. [67]. Performance metrics are presented in the range [0,1].

The proposed baseline model and the corresponding performance results in the BoLD test set are presented in table 5.2. The proposed emotion recognition system, named ARBEE which is short for Automated Recognition of Bodily Expression of Emotion, constitutes an ensemble of multiple sub-modules that operate on different input modalities. More specifically, they averaged the predictions from a TSN trained on RGB body crops, a TSN trained on RGB face crops as well as a Random Forest classifier operating on the 216-dim LMA feature vectors. According to table 5.2, the combination of all modalities, i.e., body, face and skeleton, achieves the best performance.

5.2 Experimental Results

In the current section, we present our experimental results on BoLD. Across all of our experiments, we use the standard train and validation splits provided by the official distributors of the dataset. The first part of the presentation constitutes an ablation study relative to the effect of incorporating multiple input modalities on the emotion recognition process.

5.2.1 Ablation Studies

The backbone of our network implementations resides in a combination of the two-stream convolutional and TSN architectures. As a first step, we experiment with separately training

TSN using the *RGB*, *Optical Flow* and *RGB Difference* modalities. Subsequently, we shift our attention to skeleton-based learning through the usage of LMA features and the ST-GCN mechanism. Both the TSN and ST-GCN constitute flexible and lightweight frameworks that have been extensively utilized in action recognition related literature.

TSN-RGB

A single RGB image usually encodes static appearance at a specific point in time but lacks the contextual information about previous and next frames. During our experiments on the EMOTIC dataset, we have showed that the emotion recognition process benefits greatly from the feature level combination of multiple feature extractors that focus on different parts of the human instance. In our previous implementations, we made extensive use of the human body, face as well as the surrounding depicted environment in the form of visual context. In order to replicate our previous methodology and adapt it in the dynamic setting of video sequences, we modify the standard TSN framework by including multiple input streams, which later undergo feature level fusion, before the application of segmental consensus for the prediction of both categorical and continuous targets.

We begin by training a standard TSN using the RGB modality and the body crops of each frame instance. For the calculation of the necessary body bounding boxes, we make use of the coordinates of 18 body joints that have been successfully tracked along the entirety of each video sequence and are being provided by the distributors of the dataset. As a feature extractor backbone, we use a standard 18-layer ResNet [48] pre-trained on ImageNet [26]. Unless specified otherwise, all of the other CNN backbones that will later be used for other input streams, utilize the same architecture and only differ in the pre-training aspect.

Subsequently, we incorporate a context stream in the form of RGB frames whose primary depicted agents have been masked out. For the acquisition of the masks we use the body bounding boxes that we have previously calculated and multiply them element-wise with a constant value of zero. The context branch feature extractor is trained on the Places365-Standard [117] dataset, following the exact same approach as in our first case study.

Furthermore, we introduce an input stream which explicitly operates on extracted face crops. For the localization and extraction of faces we use the first and last four body joints that correspond to the eyes, ears and nose of each depicted instance. We use these joints to calculate the largest bounding box that contains the head of the agent. However, as the pose of an agent might result in partial or complete occlusion of their facial features, the successful extraction of the face region is not guaranteed. The CNN backbone for the face branch receives manual pre-training on the AffectNet [75] database.

Last but not least, we enrich the model’s perception of context by directly extracting the Places365 [117] scene-specific scores and the corresponding SUN [82] attributes through a pre-trained 18-layer Wide-ResNet [110]. The inclusion of all the aforementioned input streams results in a 2003-dim concatenated feature vector.

The training of the continuous emotion prediction branch, we use a standard MSE loss $\mathcal{L}_{\text{cont}}$ along the three emotional dimensions of valence, arousal and dominance. As far as categorical emotion prediction is concerned, the groundtruth targets are provided in the form of confidence scores. Therefore, we first apply a sigmoid function to the barebones extracted class scores and then impose an MSE loss between the predicted and groundtruth confidence scores. We denote this loss term as $\mathcal{L}_{\text{cat}_1}$. Secondly, after binarizing the groundtruth confidence scores with a standard threshold of 0.5, we apply a binary cross-entropy loss between the predicted confidence scores and the given target labels. We denote this term as $\mathcal{L}_{\text{cat}_2}$. Lastly, we enforce semantic congruity between the extracted deep feature representations and the categorical label word embeddings from a 300-dim GloVe model, pre-trained on Wikipedia

and Gigaword 5 data. More specifically, we first transform the concatenated visual embeddings into the same dimensionality as the word embeddings through a linear transformation and we later apply a MSE loss between the transformed visual embeddings and the average of word embeddings that correspond only to the positive labels of the given groundtruth target vector. Following the same notation as in section 4.1.3, we denote this term \mathcal{L}_{emb} . Network training is driven by a combined loss function of the form:

$$\mathcal{L} = \mathcal{L}_{\text{cat}_1} + \mathcal{L}_{\text{cat}_2} + \mathcal{L}_{\text{cont}} + \mathcal{L}_{\text{emb}} \quad (5.17)$$

The TSN-RGB is trained for 25 epochs with a batch size of 16, using the SGD optimizer with a learning rate equal to 10^{-3} , momentum equal to 0.9 and weight decay equal to 10^{-5} . The learning rate is reduced by a factor of 0.1 whenever the monitored loss on the validation set plateaus. For the current experiments, we use $K_{\text{train}} = 3$ segments during training and $K_{\text{val}} = 25$ segments during validation, while the segmental consensus function has been chosen to be average pooling. For the choices regarding the number of segments, we consult the proposed methodologies of [104]. Apart from the previously described methodologies, we experiment with the partial batch-normalization scheme, or *Partial BN* for short, as proposed by Wang et al. [104]. More specifically, after the initialization with pre-trained models, we freeze the mean and variance of parameters of all batch normalization layers, except for the first one. Table 5.3 shows a performance comparison of all TSN-RGB configurations which we have considered, relative to the BoLD validation set. For each configuration, we report performance metrics that correspond to the epoch during which the highest ERS score was obtained. The second column describes the various input streams that are being included, with ‘‘B’’ denoting the body, ‘‘C’’ denoting the context, ‘‘F’’ denoting the face, ‘‘S’’ denoting the Places365 scene-specific classification scores and ‘‘A’’ denoting the corresponding SUN attribute scores.

Model	Features	\mathcal{L}_{emb}	Partial BN	Regression	Classification		ERS
				$mR^2\uparrow$	mAP \uparrow	mRA \uparrow	
TSN-RGB	B	No	No	0.0300	0.1419	0.5910	0.1983
	BC			0.0362	0.1468	0.6021	0.2053
	BCF			0.0531	0.1615	0.6213	0.2222
	BCFS			0.0597	0.1736	0.6428	0.2347
	BCFA			0.0685	0.1763	0.6417	0.2388
	BCFSA	No	No	0.0710	0.1762	0.6435	0.2404
Yes		No	0.0713	0.1779	0.6457	0.2416	
Yes		Yes	0.0969	0.1839	0.6537	0.2579	

Table 5.3: Performance comparison of various TSN-RGB model configurations on the BoLD validation set. The second column describes the various *RGB* input streams that have been included, namely the *body*, *context*, *face*, the Places365 scene-specific classification scores and SUN attribute scores. The third column specifies whether the categorical label embedding loss \mathcal{L}_{emb} has been incorporated in the training phase, while the fourth column specifies whether the *Partial BN* scheme has been utilized. Performance metrics are presented in the range [0,1].

Both the inclusion of the context and face streams are conducive to an increase in ERS score, with the latter showcasing a more considerable boost in performance over the bare-bones body stream. These findings together with the corresponding results in EMOTIC, confirm that the face modality constitutes a rich source of affective information and its inclusion in the emotion recognition process, accompanied by a properly pre-trained feature extractor, almost definitely lead to better results. Moreover, even though the sole inclusion of the Places365 scene-specific classification scores seemingly does not improve performance, the combined usage of the latter along with the corresponding SUN attribute scores improves performance in both the categorical and continuous tasks, as expected. Lastly we experiment with the addition of the categorical embedding loss \mathcal{L}_{emb} and the application of

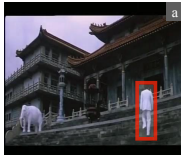
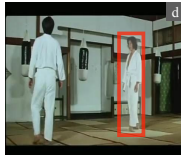




	a	Scenes Temple/Asia Pagoda Chalet Palace Hunting Lodge	Attributes Man made Natural light No horizon Open area Touring Vert. components Shingles Semi-enclosed Aged	Ground Truth Peace Annoyance	BCF Anticipation	BCFSA Peace Affection Happiness		d	Scenes Martial arts gym Clean room Locker room Artists loft Elevator lobby	Attributes No horizon Man-made Enclosed area Cloth Indoor lighting Work Vert. components Natural light Competing	Ground Truth Anticipation Confidence	BCF Anticipation	BCFSA Anticipation Confidence Engagement	
	b	Stage/Indoor Discotheque Ballroom Orchestra pit Movie theater	Enclosed area No horizon Indoor lighting Cloth Congregating Socialising Man-made Spectating Stressful	Engagement Confidence Pleasure Sensitivity	Engagement Excitement Anticipation	V: 0.5875 A: 0.6594 D: 0.6285 JC = 0.0	V: 0.6537 A: 0.5948 D: 0.6459 JC = 0.25		e	Oast house Cottage Tree farm Kashah Village	Natural light Foliage Open area Vegetation Leaves Trees No horizon Man-made Shrubbery	Affection Esteem Sympathy	Peace Happiness	Esteem Peace Engagement Happiness Pleasure V: 0.5658 A: 0.4353 D: 0.5420 JC = 0.0
	c	Beer hall Pub/Indoor Banquet hall Dining hall Bar	No horizon Enclosed area Man-made Indoor lighting Socialising Cloth Congregating Eating Stressful	Peace Anticipation Happiness Pleasure	Peace Happiness	V: 0.8234 A: 0.6079 D: 0.5651 JC = 0.25	V: 0.5601 A: 0.5537 D: 0.5086 JC = 0.33		f	Catacomb Arch. Excavation Grotto Trench Basement	No horizon Man-made Dirt Natural light Enclosed area Dry Dirty Rugged scene Aged	Anticipation Sympathy Sensitivity Sadness Suffering	Suffering Fear	Anticipation Sensitivity Sadness Suffering, Pain Engagement V: 0.4828 A: 0.5799 D: 0.6331 JC = 0.167

Figure 5.5: Top-5 predicted scene categories, top-9 predicted attributes, ground truth and predicted (regressed) emotion categories (VAD values) as well as *Jaccard similarity coefficient* (JC) on samples that have been randomly selected from the BoLD validation set. All predictions are made at video level.

Partial BN. While the addition of \mathcal{L}_{emb} leads to a trivial performance boost, the application of the *Partial BN* regularization scheme tops off our previously best performing network, reaching a maximum of 0.2579 ERS on the BoLD validation set. It seems as if the continuous re-estimation of mean and variance parameters of batch-normalization layers that are located deeper within the network, becomes obsolete and may in fact have a negative impact on generalization performance, provided that the model’s parameters have been previously initialized through a proper pre-training procedure.

The beneficial influence of scene and attribute related features in human emotion understanding becomes more evident in cases where the facial characteristics and poses of the depicted agents are occluded. This is further highlighted in Fig. 5.5 which includes instances that were randomly selected from the validation set. Each instance is accompanied by its top-5 predicted scene categories, top-9 predicted attributes, ground truth and predicted (regressed) emotion categories (VAD values) as well as the corresponding *Jaccard similarity coefficient* (JC), for each model configuration. Correct category recognition is indicated in green. In all cases, the incorporation of scene and attribute characteristics, on top of the existing bodily, contextual and facial features, results in more emotions being correctly recognised. In addition, emotions that are semantically related, i.e. peace-happiness-pleasure (e) and sadness-suffering-pain (f), are jointly predicted, even though some of them have not been included by the annotators.

Subsequently, we pick our best model configuration and further evaluate it on the validation set while varying the number of segments K from 1 to 11. In this case, we have also

Model	K	Regression	Classification		ERS
		$mR^2 \uparrow$	mAP \uparrow	mRA \uparrow	
TSN-RGB	1	0.0387	0.1634	0.6229	0.2159
	3	0.0808	0.1811	0.6498	0.2481
	5	0.0834	0.1834	0.6533	0.2509
	7	0.0878	0.1836	0.6527	0.2530
	9	0.0892	0.1851	0.6569	0.2551
	11	0.0877	0.1834	0.6571	0.2539

Table 5.4: Performance comparison of different TSN-RGB configurations on the BoLD validation set, relative to the number of segments K . All configurations utilize average pooling as the segmental consensus function. Performance metrics are presented in the range [0,1].

used equal number of segments in both training and validation phases. Table 5.4 summarizes the results. In the special case of $K = 1$, the TSN framework degenerates into a simple convolutional network and the model exhibits poor performance as it cannot effectively capture the long-range temporal structure of video sequences. As K increases, we notice a gradual performance improvement up to $K = 9$. This constitutes a confirmation of the fact that a higher number of segments helps in obtaining a more descriptive encoding of the temporal structure of video sequences. For $K = 11$ we notice that the monitored performance improvement slightly saturates, indicating that a further increase in the number of segments will likely not have any additional positive effect. We do not explore configurations with a greater number of segments due to GPU memory limitations. We also ought to mention that the utilization of a high number of segments proves to be more crucial during inference than during training. A higher number of training iterations can potentially counter a lack of segments during the training phase. However, during inference each video instance is examined only once and therefore a higher number of segments is required with the aim of obtaining more representative prediction scores.

Lastly, we experiment with different aggregation functions. Apart from average pooling which has already been examined, we investigate whether a linear weighting or attention weighting consensus function would lead to a performance improvement. For the following experiments we use $K = 7$ segments for both the training and validation phases. Table 5.5 summarizes the results. Average pooling clearly outperforms all other methodologies. The application of linear and attention weighting leads to a deterioration in performance, despite the theoretical advantages that they provide in comparison with the seemingly basic average pooling function. Presumably, a dataset which featured a more complex temporal structure would make better use of the latter advanced techniques.

Model	Consensus Function	Regression	Classification		ERS
		$mR^2\uparrow$	mAP \uparrow	mRA \uparrow	
TSN-RGB	Average Pooling	0.0878	0.1836	0.6527	0.2530
	Linear Weighting	0.0809	0.1723	0.6341	0.2418
	Attention Weighting	0.0604	0.1849	0.6529	0.2396

Table 5.5: Performance comparison of different TSN-RGB configurations on the BoLD validation set, relative to the type of segmental consensus function that has been utilized, with $K = 7$ segments for both the training and validation phases. Performance metrics are presented in the range [0,1].

TSN-Flow

Similarly to the case of temporal stream ConvNets in the original two-stream convolutional architecture, we experiment with training a TSN on stacked optical flow fields. Optical flow extraction is carried out via the TVL1 algorithm [109]. This form of dense optical flow is known to effectively encode motion between consecutive frames. We denote this model as TSN-Flow. Subsequently, we attempt to expand the single-stream temporal ConvNet architecture by incorporating multiple input streams in parallel which are later combined through an early fusion scheme, following our previous implementation regarding the TSN-RGB.

In all our subsequent experiments, we stack bidirectional optical flow fields from $L = 5$ consecutive frames for each snippet. After decomposing each displacement vector into its horizontal and vertical components, we end up with a 10-channel input volume per segment, per input stream. To begin with, we train a standard TSN using the *Optical Flow* modality and the body crops of each frame instance. The usage of body joint coordinates for the localization and extraction of the necessary bounding boxes remains the same as in the case of the RGB modality. Body-oriented dense optical flow encodes the movement of the primary agent depicted in each instance. Subsequently, we incorporate a context stream, in the form

of stacked optical flow fields whose primary depicted agents have been masked out. The optical flow input stream effectively encodes the motion of any occasional secondary agent, object and the surrounding environment in general. Lastly, we introduce an input stream that focuses solely on the head and face movements of the primary agent. This is achieved by training an additional temporal ConvNet on small fragments of dense flow that correspond to the head region of each agent. The inclusion of all the aforementioned input streams results in a 1536-dim concatenated feature vector.

The features extracted using optical flow streams have distributions that greatly differ from their RGB counterparts. As optical flow values are discretized in the interval $[0, 255]$, therefore sharing the same range with RGB images, we use RGB models to initialize the parameters of the temporal ConvNets. Consequently, the weights of the first convolutional layer are modified so as to handle the input of optical flow fields. More specifically, the weights are averaged across the RGB channels and replicated by the number of channels of the temporal stream inputs, hence the activation values of the first convolutional layer will also have different distributions. This is where the *Partial BN* scheme comes in, taking care of the re-estimation of the first batch-normalization layer mean and variance parameters. As noted by Wang et al. [104], the aforementioned method works well with temporal networks, reducing the effect of over-fitting.

For all of our following experiments, we use the average pooling aggregation function due to its proven superiority over other methods that have been discussed in the previous section. Additionally, in order to cut down on training times for the TSN-Flow configuration, we chose to use $K_{\text{train}} = 3$ during training and $K_{\text{val}} = 25$ segments during validation. The reduction in segments for the training phase has been found to have a negligible effect in actual performance of TSN-Flow, while the choice in number of segments for the validation phase is inspired by the original testing scheme used with *two-stream convolutional networks*, as proposed in [94]. As far parameter initialization is concerned, we pre-train the TSN-Flow body stream on ImageNet. Furthermore, we consider the Places365 and AffectNet datasets as alternatives to ImageNet for the pre-training of the context and face streams respectively. Tables 5.6 and 5.7 provide a performance comparison for the two TSN-Flow configurations based on the pre-training method that is being used. All other training settings have been kept the same as the ones used for the TSN-RGB.

Model	Body Stream Pre-training	Context Stream Pre-training	Regression	Classification		ERS
			$mR^2\uparrow$	mAP \uparrow	mRA \uparrow	
TSN-Flow-BC	ImageNet	ImageNet	0.0661	0.1415	0.5882	0.2155
		Places365	0.0657	0.1472	0.5857	0.2161

Table 5.6: Performance comparison of various TSN-Flow-BC model configurations, on the BoLD validation set, relative to the datasets used for pre-training each one of the input streams. Performance metrics are presented in the range $[0,1]$.

Model	Body Stream Pre-training	Face Stream Pre-training	Regression	Classification		ERS
			$mR^2\uparrow$	mAP \uparrow	mRA \uparrow	
TSN-Flow-BF	ImageNet	ImageNet	0.0650	0.1491	0.5971	0.2190
		AffectNet	0.0661	0.1449	0.5933	0.2176

Table 5.7: Performance comparison of various TSN-Flow-BF model configurations, on the BoLD validation set, relative to the datasets used for pre-training each one of the input streams. Performance metrics are presented in the range $[0,1]$.

In the first case we notice that pre-training of the context stream temporal ConvNet in the scene-centric Places365 dataset offers a rather trivial boost in performance compared to an object-centric pre-training on ImageNet. In addition, pre-training the face stream temporal

Model	Features	\mathcal{L}_{emb}	Partial BN	Regression	Classification		ERS
				$mR^2\uparrow$	mAP \uparrow	mRA \uparrow	
TSN-Flow	B	No	No	0.0560	0.1431	0.5778	0.2082
	BC			0.0661	0.1415	0.5882	0.2155
	BF			0.0649	0.1497	0.5971	0.2190
	BCF	No	No	0.0795	0.1524	0.6054	0.2292
		Yes	No	0.0888	0.1563	0.6135	0.2369
		Yes	Yes	0.0947	0.1554	0.6149	0.2398

Table 5.8: Performance comparison of various TSN-Flow model configurations, on the BoLD validation set. The second column describes the various *Optical Flow* input streams that have been included, namely the *body*, *context* and *face*. The third column specifies whether the categorical label embedding loss \mathcal{L}_{emb} has been incorporated in the training phase, while the fourth column specifies whether the *Partial BN* scheme is being utilized. Performance metrics are presented in the range [0,1].

ConvNet on AffectNet actually leads to a deterioration in performance. Therefore, in all of our subsequent experiments we initialize all temporal ConvNets using the ImageNet dataset.

Table 5.8 provides a complete performance analysis among all TSN-Flow configurations which we have considered. For each configuration, we report performance metrics that correspond to the epoch during which the highest ERS score was obtained. We notice that the introduction of either the context or face streams leads to a marginal improvement over the barebones temporal body stream. A more considerable boost in performance is achieved through the inclusion of all three input streams. As mentioned in the case of TSN-RGB, the addition of the categorical label embedding loss \mathcal{L}_{emb} improves the network’s performance in both categorical and continuous tasks, while with the application of the *Partial BN* scheme, the resulting model tops off all previous configurations reaching a maximum of 0.2398 ERS.

TSN-RGBDiff

The difference between two consecutive RGB frames essentially describes changes in appearance which correspond to the motion of salient regions. Following the example of Wang et al. [104], we experiment with training a TSN on stacked RGB difference as an alternative to optical flow for describing motion. Figure 5.6 illustrates examples for the *body*, *context* and *face* streams for *RGB Difference* as well as the other two aforementioned modalities.

Similarly to the case of optical flow, the distribution of RGB difference values differs from that of typical RGB images. We again choose to initialize the ConvNets of all input streams of the TSN, using RGB pre-trained models. Based on our experiments with the *Optical Flow* input modality, datasets such as Places365 and AffectNet have no considerable effect on performance when used for pre-training in place of ImageNet. Therefore, all ConvNets in RGB difference input streams are pre-trained on ImageNet. Moreover, the weights of the first convolutional layer of each ConvNet are averaged across the RGB channels and replicated by the number of channels of the RGB difference input streams. All training settings have been kept the same as in the case of TSN-Flow.

Model	Features	\mathcal{L}_{emb}	Partial BN	Regression	Classification		ERS
				$mR^2\uparrow$	mAP \uparrow	mRA \uparrow	
TSN-RGBDiff	B	No	No	0.0347	0.1274	0.5682	0.1912
	BC			0.0401	0.1357	0.5823	0.1995
	BF			0.0450	0.1358	0.5799	0.2014
	BCF			0.0249	0.1398	0.5874	0.1943

Table 5.9: Performance comparison of various TSN-RGBDiff model configurations, on the BoLD validation set. The second column describes the various *RGB Difference* input streams that have been included, namely the *body*, *context* and *face*. Neither categorical label embedding loss \mathcal{L}_{emb} nor *Partial BN* has been utilized. Performance metrics are presented in the range [0,1].

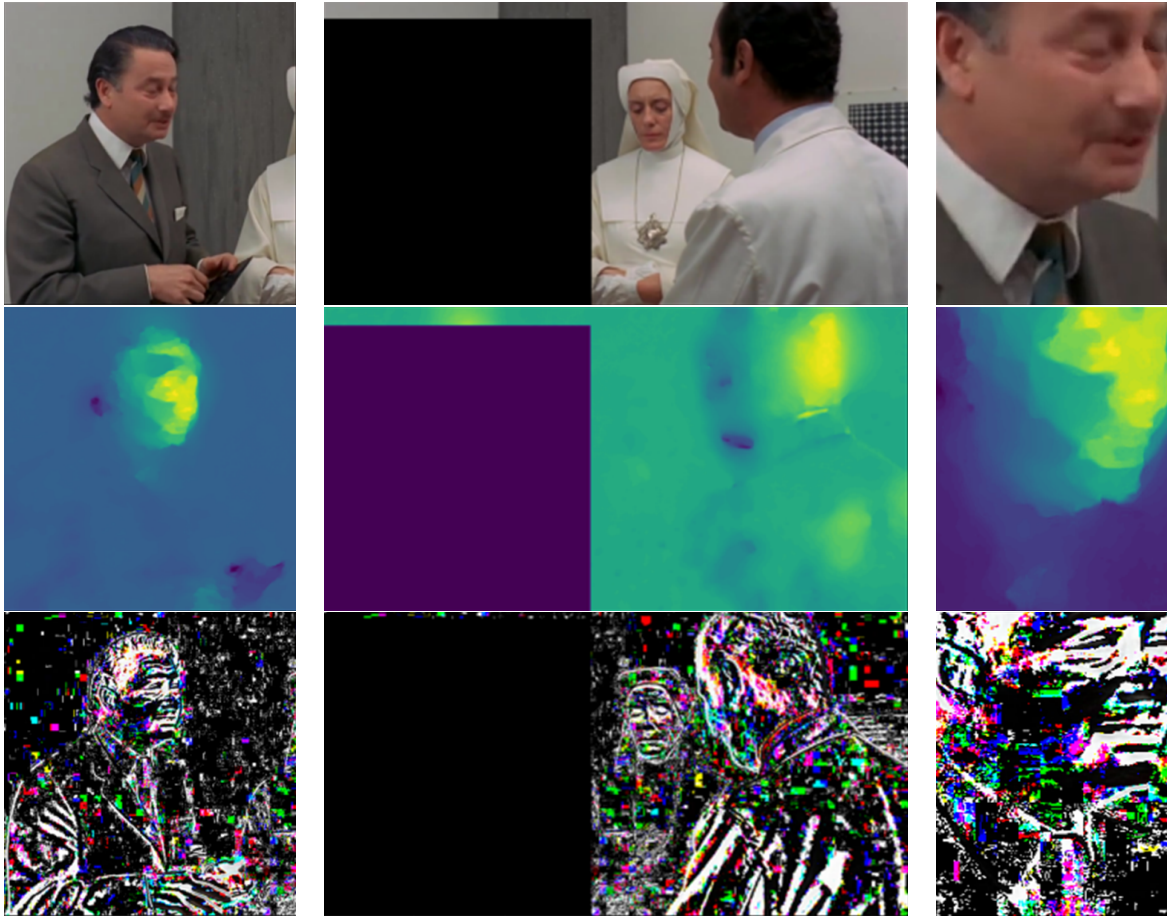


Figure 5.6: Input examples for the *body* (first column), *context* (second column) and *face* (third column) streams, as obtained from the same single snippet of a video sequence. The first row corresponds to *RGB*, the second row corresponds to the vertical component of *Optical Flow* and the third row corresponds to the *RGB Difference* modality.

We immediately notice that the descriptive capacity of the TSN-RGBDiff model is inferior to its *Optical Flow* counterpart. Even though the addition of the either the *context* or *face* stream provide a slight improvement in performance over the barebones *body* stream, reaching a maximum of 0.2014 ERS, the inclusion of all three streams actually leads to a deterioration in ERS. More specifically, classification scores do increase but the dimensional emotion regression branch fails to converge. We presume that this issue is associated with the fact that the *RGB Difference* modality appears to be significantly more noisy compared to the other two, especially in low resolution videos. We also considered adding the categorical label embedding loss \mathcal{L}_{emb} and applying *Partial BN*, but all our attempts resulted in lower ERS.

Skeleton-Based Learning

The current section is dedicated to the exploration of skeleton-based learning methodologies for the task of emotion recognition in context. To this end, we shift our attention to the rather promising LMA features and the widely used ST-GCN mechanism.

Firstly, we experiment with training an ST-GCN on BoLD. We deploy the vanilla ST-GCN consisting of 9 layers of spatial-temporal graph convolution operators (ST-GCN units). The first three layers have 64 output channels, followed by three layers with 128 channels

while the last three layers have 256 channels. The temporal kernel size is fixed to 9 for all layers. All ST-GCN units utilize residual connections and are followed by dropout layers with dropout probabilities equal to 0.5. The stride of the 4th and 7th ST-GCN units is set to 2. The features extracted from the last ST-GCN unit undergo average pooling and after a pair of 1×1 convolutions, the final categorical and continuous predictions are produced.

Joint coordinates are normalized in the range $[0, 1]$ using the largest joint bounding box within each sequence. As far as data augmentation is concerned, we follow the proposed methodologies of Yan et al. [108]. Firstly, we find the maximum sequence length T within our dataset and pad every clip with zeros until it reaches that specified length. During the training phase, padding is applied randomly within the sequence while during inference the paddings are placed always at the end for consistency. Additionally, during training we perform random affine transformations on the skeleton sequences of all frames. The affine transformation is produced by first selecting a few fixed angle, translation and scaling factors for the first and last frame of the sequence as candidates and then by randomly sampling two combinations of three factors that are going to be applied and interpolated for all intermediate frames. This transformation scheme is meant to simulate camera movement and is denoted as *random moving*.

After augmentation, input data is represented by tensors of size (C, T, V) where C denotes the number of input channels, V denotes the number of nodes in each graph sequence and T is the maximum sequence length. For each frame of a sequence, BoLD provides 18 tuples that contain 2D joint coordinates plus a detection confidence score associated with each joint, therefore in our case $C = 3$ and $V = 18$. Moreover, in order to reduce the effect of overfitting, we experiment with pre-training the ST-GCN model on the Kinetics dataset which was introduced by Kay et al. [16] and has been extensively used for skeleton based action recognition.

The ST-GCN is trained for 25 epochs with a batch size of 16, using the SGD optimizer with a learning rate of $5 \cdot 10^{-3}$, momentum equal to 0.9 and weight decay equal to 10^{-5} . The learning rate is reduced by a factor of 0.1 whenever the monitored loss on the validation set plateaus. Training is driven by the same combined loss that was used during TSN training, excluding the categorical label embedding loss component. The main variable setting of the ST-GCN configurations is the joint labeling strategy that is being used during the construction of the graph adjacency matrix, namely *uniform*, *distance* or *spatial*. Table 5.10 provides a performance comparison among all ST-GCN configurations which we have considered.

We notice that the *spatial* labeling strategy leads to better results compared to the others, confirming the findings of Yan et al. [108]. More importantly, pre-training the ST-GCN on Kinetics provides a significant performance boost in both categorical and continuous tasks over all of its counterparts that have been trained on BoLD from scratch, reaching a maximum ERS of 0.2237 on the validation set. As confirmed through our experiments, pre-training plays a crucial role in the performance of the network and presumably constitutes the main

Set	Model	Pre-training	Labeling Strategy	Regression	Classification		ERS
				$mR^2 \uparrow$	mAP \uparrow	mRA \uparrow	
Valid.	ST-GCN (ours)	None	Uniform	0.0245	0.1295	0.5701	0.1871
			Distance	0.0323	0.1383	0.5841	0.1967
			Spatial	0.0383	0.1387	0.5838	0.1998
		0.0652		0.1542	0.6103	0.2237	
Test	Luo et al. [67]	N/A	N/A	0.0440	0.1263	0.5596	0.1940
	ST-GCN (ours)	Kinetics [16]	Spatial	0.0908	0.1694	0.6268	0.2444

Table 5.10: Performance comparison of various ST-GCN model configurations, on the BoLD validation and test sets. The third column specifies the joint labeling strategy that has been utilized during the construction of the graph adjacency matrix. Performance metrics are presented in the range $[0, 1]$.

LMA Representation	Method	Regression	Classification		ERS
		$mR^2\uparrow$	mAP \uparrow	mRA \uparrow	
Raw Sequence	Conv1D	0.0183	0.1263	0.5467	0.1774
	LSTM	0.0192	0.1187	0.5428	0.1749
Descriptive Statistics	Conv1D	0.0098	0.1204	0.5517	0.1729
	Random Forest	0.0285	0.1043	0.5009	0.1655

Table 5.11: Performance comparison of various LMA-based methods on the BoLD validation set. The first column specifies the LMA feature representation, i.e., raw sequences or descriptive statistics. Performance metrics are presented in the range [0,1].

differentiating factor between the reported results of Yuo et al. [67] and ours.

Furthermore, we experiment with the 54 LMA features, as described in section 5.1.4, in the form of both raw sequences as well as their 216-dim descriptive statistics. Feature extraction, in the case of raw sequences is carried out through a two-layer bidirectional LSTM with 1024 hidden units. The output of the last time step is passed through a FC layer so as to obtain the categorical and continuous emotion predictions. Another methodology involves the application of 1D convolutions with the aim of extracting features in both cases of raw sequences and LMA descriptive statistics. The corresponding network is comprised of three consecutive 1D convolutional layers, with kernels of size 5 and 3, necessary padding so as to retain input tensor lengths and 256-512-1,024 filters. The last convolutional layer is followed by an average pooling layer and 3 consecutive FC layers with 1,024-512-256 hidden units. Both convolutional layers and FC layers are followed by batch-normalization and ReLU nonlinearities. We denote this method as ‘‘Conv1D’’. Last but not least, we try to replicate the initial approach of Luo et al. [67] who applied a Random Forest Classifier and Regressor on the 216-dim descriptive statistics. Table 5.11 summarizes the results obtained on BoLD validation set with each one of the aforementioned methodologies.

We immediately notice that the LMA approach to skeleton-based emotion recognition performs significantly worse than the more refined and complex ST-GCN mechanism. The best result was obtained using the raw sequences of the 54 LMA features combined with our proposed 1D convolutional network, reaching a maximum 0.1774 ERS. The LMA feature representation of raw sequences seems to perform marginally better than that of descriptive statistics. The worst ERS was obtained using the Random Forest Classifier and Regressor, even though the latter provides the highest mean coefficient of determination mR^2 , equal to 0.0285. More specifically, the Random Forest Regressor, when operated on the 216-dim descriptive statistics, produced a high *coefficient of determination* R^2 , equal to 0.0794 for the arousal emotion dimension, confirming the findings of Luo et al. [67] regarding the strong correlation between arousal and the LMA features.

5.2.2 Proposed Method

The proposed methodology constitutes a late fusion scheme among the best performing models from all modalities, namely *RGB*, *Optical Flow*, *RGB Difference* and *Human Skeleton*. A complete schematic diagram of our proposed network ensemble is shown in figure 5.7. The score fusion methods which will be considered include: maximum, simple average and weighted average. Table 5.12 summarizes the results.

We notice that score fusion among TSN-RGB, TSN-Flow and TSN-RGBDiff actually produces worse results compared to when we just exclude the *RGB Difference* modality. The maximum operator consistently performs the worst among the three fusion schemes. Eventually, a weighted average of the *RGB*, *Optical Flow* modalities as well as the ST-GCN mechanism, with a weight ratio of 2:2:1 respectively, leads to the best result of 0.2897 ERS on the validation set. More importantly, our implementation surpasses the current state-of-the-

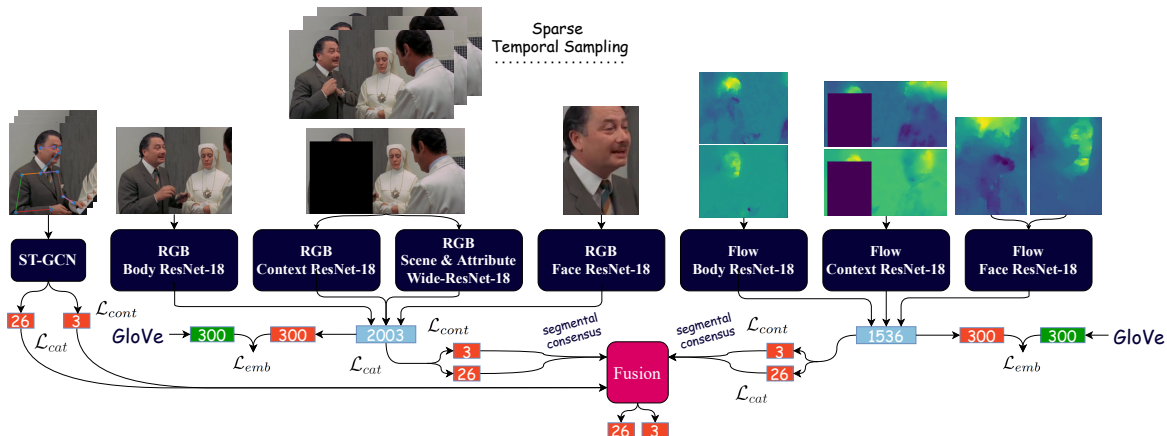


Figure 5.7: Complete schematic diagram of the proposed network ensemble, featuring an ST-GCN module and three TSN input streams (*body*, *context*, *face*) for both the *RGB* and *Optical Flow* modalities, plus a *scene & attribute* related stream, especially for the *RGB* modality. Concatenated feature vectors are depicted in cyan, fully-connected layers are depicted in orange and GloVe word embeddings are depicted in green, along with their dimensionality or number of hidden units. The ST-GCN inherently produces video-level predictions, while in the case of the TSN-*RGB* and TSN-*Flow*, this requires the prior application of segmental consensus upon the corresponding snippet-level predictions (26 confidence scores for discrete emotions, 3 regressed values for VAD dimensions). Final predictions are obtained through late score fusion.

Network Ensembles	Score Fusion	Regression	Classification		ERS
		$mR^2 \uparrow$	mAP \uparrow	mRA \uparrow	
All TSN Modalities	Maximum	0.0632	0.1804	0.6483	0.2388
	Average	0.1283	0.1852	0.6624	0.2761
TSN- <i>RGB</i> +TSN- <i>Flow</i>	Maximum	0.0939	0.1840	0.6543	0.2566
	Average	0.1444	0.1883	0.6661	0.2858
TSN- <i>RGB</i> +TSN- <i>Flow</i> +ST-GCN	Maximum	0.0652	0.1809	0.6493	0.2402
	Average	0.1438	0.1933	0.6658	0.2867
	Weighted Average	0.1489	0.1929	0.6682	0.2897
Filntisis et al. [37]	Average	0.0917	0.1656	0.6266	0.2439

Table 5.12: Performance comparison of various network ensembles on the BoLD validation set, utilizing a late fusion scheme. The second column specifies the score fusion method, i.e., maximum, simple or weighted averaging. Performance metrics are presented in the range [0,1].

art of 0.2439 ERS on the BoLD validation set by a large margin, as it was recently achieved by Filntisis et al. [37].

Figs. 5.8 and 5.9 summarize the results for the 26 discrete emotion categories and the continuous VAD dimensions relative to the AP and R^2 performance metrics, respectively. In addition, Fig. 5.10 shows the *Jaccard similarity coefficient* (JC) for all samples in the validation set. As a detection threshold for each emotion category, we pick the *equal error rate* (EER) point, i.e. the value at which *precision* equals *recall*. Notice that even our best model is incapable of predicting a single correct emotion for almost half of the samples in the validation set. Fig 5.11 shows the *absolute error* (AE) across the VAD dimensions, for all samples in the validation set. In the case of continuous emotion regression, the achieved performance boost is far less noticeable, indicating the increased difficulty of the task compared to multi-label discrete emotion classification.

Subsequently, we use our best performing network ensemble so as to tackle the task of emotion recognition on the official BoLD test set. The predictions for our final submission are produced using $K = 25$ segments for both TSN-*RGB* and TSN-*Flow*. A comparative study regarding the performance and complexity of our proposed model and earlier published works is presented in Table 5.13. The proposed network ensemble manages to surpass the

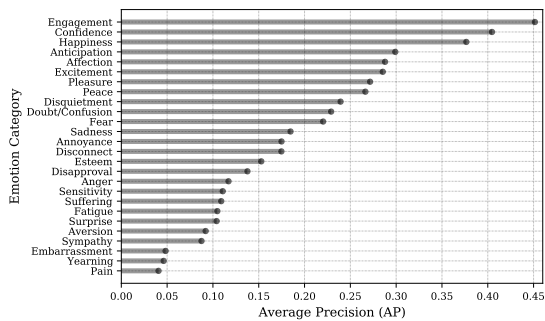


Figure 5.8: Average precision (AP) scores per emotion category, as obtained on the BoLD validation set, using our proposed network ensemble.

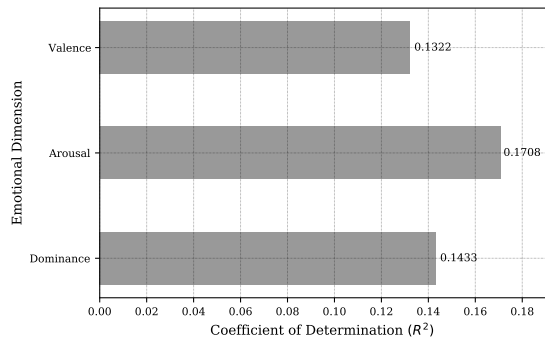


Figure 5.9: Coefficient of determination (R^2) per emotional dimension, as obtained on the BoLD validation set, using our proposed network ensemble.

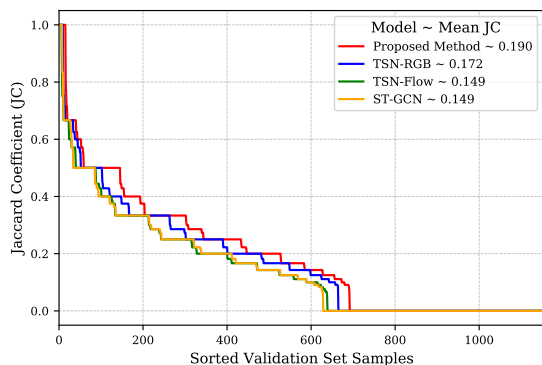


Figure 5.10: Jaccard similarity coefficient (JC) per sample in the BoLD validation set, sorted in descending order.

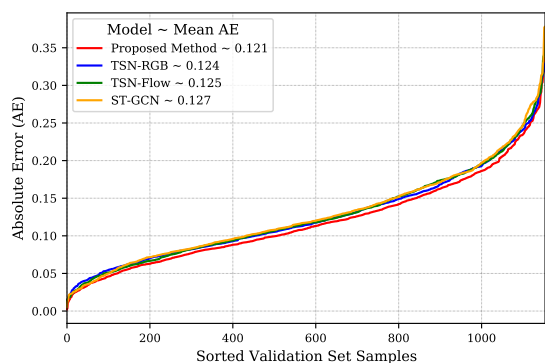


Figure 5.11: Absolute error (AE) across the VAD dimensions, per sample in the BoLD validation set, sorted in ascending order.

Model	# Parameters ($\times 10^6$)	Regression	Classification		ERS
		$mR^2 \uparrow$	mAP \uparrow	mRA \uparrow	
Luo et al. [67]	N/A	0.1030	0.1714	0.6352	0.2530
Filntisis et al. [37]	111.5	0.1141	0.1796	0.6416	0.2624
Ours	71.4	0.1597	0.2185	0.6826	0.3051

Table 5.13: Quantitative results on the BoLD test set regarding our proposed network ensembles and other published works. Performance metrics are presented in the range [0,1].

current state-of-the-art of 0.2624 ERS, as achieved in [37], by a considerable margin on all metrics, thus verifying the superiority of our technique. As far as complexity is concerned, our model does a good job at maintaining a lower number of trainable parameters through the efficient utilization of multiple input streams and shallow feature extractors (ResNet-18, Wide-ResNet-18), in comparison with previous implementations that made use of deeper ConvNet backbones [37] (ResNet-50 & 101) and disentangled the various input streams all together [67].

Chapter 6

Conclusion and Future Work

6.1 Conclusion

In this thesis we addressed the problem of visual emotion recognition in context, on the basis of images as well as video sequences. More importantly, we pushed the boundaries of the automated emotion recognition process to fully uncontrolled, “in-the-wild” settings with the conduction of extensive experiments on two newly assembled and challenging databases, namely the EMOTions In Context (EMOTIC) and Body Language Dataset (BoLD).

Our ablation studies on EMOTIC revealed that the joint utilization of multiple input streams such as the *body*, *context*, *face* and *pose* results in an cumulative boost in emotion recognition performance. This multimodal approach to the problem proves to be essential when emotion recognition is performed in “in-the-wild” environments and the unpredictability of real-world scenarios can render one or more of the aforementioned sources of affective information temporarily inaccessible. As the various input streams function in a complementary manner to each other, the model gains the option of relying on the occasionally most reliable modalities while neglecting others, depending on the availability of each source of affective information.

Furthermore, we noticed that the scene and surrounding environment as depicted in an image can be closely related with the emotions shared by the people who are present. Therefore, we directly infused scene classification scores and attributes as additional features in the emotion recognition process and to our knowledge we are the first to do so. Our experiments confirm the validity of our initial intuition.

In addition, we aspired to exploit the categorical emotion label dependencies that reside within the datasets in an attempt to leverage visual-semantic information. We examined two separate methodologies, one based on Graph Convolutional Networks (GCN) and the other based on metric-learning, with the utilization of GloVe word embeddings constituting the common denominator between the two. Both approaches led to improvements in categorical emotion prediction. By combining all of the aforementioned proposed methodologies, we managed to achieve a competitive score of 31.29% mAP in the EMOTIC test set, falling behind of the current state-of-the-art by a mere 0.2%, provided the exclusion of depth data.

Subsequently, we aimed at extending the concept of context-based visual emotion recognition in the dynamic setting of video sequences. Our goal was to create a unified model that shared a similar architecture with our previous implementations for the EMOTIC dataset while being able to process videos. We straightforwardly chose Temporal Segment Networks (TSN) as the backbone of our network while we also considered using Laban Movement Analysis (LMA) features and Spatial-Temporal Graph Convolutional Networks (ST-GCN) as means of skeleton-based learning.

Following the same architectural principles as in our previous implementations, we extended the original TSN framework with the inclusion of multiple input streams that effectively encode bodily, contextual, facial as well as generic scene-related features that enhance the model’s perception of visual context and emotion in general. The combination of a properly pre-trained ST-GCN and our modified TSN results in significant improvements over the state-of-the-art techniques with relation to the Body Language Dataset (BoLD), reaching a maximum of 0.3051 Emotion Recognition Score on the official test set. Additionally, as we utilized shallow CNN feature extractors, the achieved performance boost comes with no additional increase in computational cost relative to all previous published implementations.

6.2 Future Work

Many different adaptations, tests, and experiments have been left for the future due to lack of time (i.e. the experiments with real data are usually very time consuming, requiring even days to finish a single run). Future work concerns deeper analysis of particular mechanisms, new proposals to try different methods, or simply curiosity.

Firstly, during our experiments in EMOTIC, we failed to match the performance of the baseline model [59] on the continuous emotion regression task, even though our implementation utilized more input streams and deeper CNN backbones compared to the latter. The actual reasons behind this unordinary behavior still remain unclear and require further research.

Additionally, we faced difficulties in replicating the results of related literature regarding the inclusion of depth data in the emotion recognition process. Even though we followed every detail in the description of the provided implementations, we did not receive any additional improvement and actually noticed a degradation in performance in contrast to the significant gains reported by Mittal et al. [71]. We presume that the successful incorporation of the depth modality would require manually pre-training of the CNN feature extractor on additional RGB-D data.

Lastly, a possible future research direction might be proposals for further exploitation of the categorical label dependencies and visual-semantic relationships that reside within video frames and may lead to an additional improvement in categorical emotion prediction. Despite the fact that the ML-GCN mechanism had been successfully adapted for the static version of the problem during our experiments on EMOTIC, we did not manage to properly configure it so as to function as a supplementary component to the TSN framework.

Bibliography

- [1] T. Ahonen, A. Hadid, and M. Pietikainen, “Face description with local binary patterns: Application to face recognition”, *IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 28, pp. 2037–2041, 2006.
- [2] T. R. Almaev and M. F. Valstar, “Local Gabor binary patterns from three orthogonal planes for automatic facial expression recognition”, in *2013 Humaine Assoc. Conf. on Affective Computing and Intelligent Interaction (ACII)*, 2013.
- [3] Z. Ambadar, J. W. Schooler, and J. F. Cohn, “Deciphering the enigmatic face: The importance of facial dynamics in interpreting subtle facial expressions”, *Psychological Science*, vol. 16, pp. 403–410, 2005.
- [4] A. Angelova, A. Krizhevsky, and V. Vanhoucke, “Pedestrian detection with a Large-Field-Of-View deep network”, in *2015 IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2015.
- [5] O. Bălan, G. Moise, L. Petrescu, A. Moldoveanu, M. Leordeanu, and F. Moldoveanu, “Emotion classification based on biophysical signals and machine learning techniques”, *Symmetry*, vol. 12, 2020.
- [6] T. Baltrusaitis, A. Zadeh, Y. C. Lim, and L. Morency, “OpenFace 2.0: Facial behavior analysis toolkit”, in *2018 13th IEEE Int. Conf. on Automatic Face Gesture Recognition (FG)*, 2018.
- [7] T. Bänziger, M. Mortillaro, and K. R. Scherer, “Introducing the Geneva multimodal expression corpus for experimental research on emotion perception”, *Emotion*, vol. 12, pp. 1161–1179, 2012.
- [8] L. Barrett, B. Mesquita, and M. Gendron, “Context in emotion perception”, *Current Directions in Psychological Science*, vol. 20, pp. 286–290, 2011.
- [9] L. F. Barrett and E. A. Kensinger, “Context is routinely encoded during emotion perception”, *Psychological Science*, vol. 21, pp. 595–599, 2010.
- [10] I. Bartenieff, *Effort-shape Analysis of Movement: The Unity of Expression and Function*. Albert Einstein College of Medicine, Yeshiva University, 1965.
- [11] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.
- [12] A. Bosch, A. Zisserman, and X. Munoz, “Representing shape with a spatial pyramid kernel”, in *Proc. 6th ACM Int. Conf. on Image and Video Retrieval (ICIVR)*, 2007.
- [13] J. K. Burgoon, L. K. Guerrero, and K. Floyd, *Nonverbal Communication*, English. Allyn & Bacon, 2010.
- [14] J. Canny, “A computational approach to edge detection”, *IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 8, pp. 679–698, 1986.

-
- [15] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh, “OpenPose: Realtime multi-person 2D pose estimation using part affinity fields”, *IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 43, pp. 172–186, 2021.
- [16] J. Carreira and A. Zisserman, “Quo vadis, action recognition? A new model and the Kinetics dataset”, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017.
- [17] G. Castellano, L. Kessous, and G. Caridakis, “Emotion recognition through multiple modalities: Face, body gesture, speech”, in *Affect and Emotion in Human-Computer Interaction*, Springer, 2008, pp. 92–103.
- [18] G. Castellano, S. D. Villalba, and A. Camurri, “Recognising human emotions from body movement and gesture dynamics”, in *Int. Conf. on Affective Computing and Intelligent Interaction (ACII)*, 2007.
- [19] G. Chen, *A gentle tutorial of recurrent neural network with error backpropagation*, 2018. arXiv: [1610.02583 \[cs.LG\]](https://arxiv.org/abs/1610.02583).
- [20] S. Chen, Y. Tian, Q. Liu, and D. N. Metaxas, “Segment and recognize expression phase by fusion of motion area and neutral divergence features”, in *2011 IEEE Int. Conf. on Automatic Face Gesture Recognition (FG)*, 2011.
- [21] S. Chen, Y. Tian, Q. Liu, and D. N. Metaxas, “Recognizing expressions from face and body gesture by temporal normalized motion and appearance features”, *Image and Vision Computing*, vol. 31, pp. 175–185, 2013.
- [22] Z. Chen, X. Wei, P. Wang, and Y. Guo, “Multi-label image recognition with graph convolutional networks”, in *2019 IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019.
- [23] R. Cowie, E. Douglas-Cowie, N. Tsapatsoulis, G. Votsis, S. Kollias, W. Fellenz, and J. Taylor, “Emotion recognition in human-computer interaction”, *IEEE Signal Processing Magazine*, vol. 18, pp. 32–80, 2001.
- [24] G. Cybenko, “Approximation by superpositions of a sigmoidal function”, *Mathematics of Control, Signals and Systems*, vol. 2, pp. 303–314, 1989.
- [25] N. Dalal and B. Triggs, “Histograms of oriented gradients for human detection”, in *2005 IEEE Computer Society Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2005.
- [26] J. Deng, W. Dong, R. Socher, L. Li, Kai Li, and Li Fei-Fei, “ImageNet: A large-scale hierarchical image database”, in *2009 IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2009.
- [27] A. Dhall, R. Goecke, S. Lucey, and T. Gedeon, “Static facial expression analysis in tough conditions: Data, evaluation protocol and benchmark”, in *2011 Int. Conf. on Computer Vision Workshops (ICCVW)*, 2011.
- [28] A. Dhall, R. Goecke, S. Lucey, and T. Gedeon, “Acted facial expressions in the wild database”, *Australian National University, Canberra, Australia, Technical Report TR-CS-11*, 2011.
- [29] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell, “Long-term recurrent convolutional networks for visual recognition and description”, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2015.

- [30] J. Duchi, E. Hazan, and Y. Singer, “Adaptive subgradient methods for online learning and stochastic optimization”, *Journal of Machine Learning Research (JMLR)*, vol. 12, pp. 2121–2159, 2011.
- [31] B. Dudzik, M. Jansen, F. Burger, F. Kaptein, J. Broekens, D. K. J. Heylen, H. Hung, M. A. Neerinx, and K. P. Truong, “Context in human emotion perception for automatic affect detection: A Survey of audiovisual databases”, in *2019 8th Int. Conf. on Affective Computing and Intelligent Interaction (ACII)*, 2019.
- [32] P. Ekman and W. Friesen, “Measuring facial movement”, *Environmental Psychology and Nonverbal Behavior*, vol. 1, pp. 56–75, 1976.
- [33] P. Ekman, “Strong evidence for universals in facial expressions: A reply to Russell’s mistaken critique”, *Psychological Bulletin*, vol. 115, pp. 268–287, 1994.
- [34] P. Ekman and W. V. Friesen, “Constants across cultures in the face and emotion”, *Journal of Personality and Social Psychology*, vol. 17, pp. 124–129, 1971.
- [35] P. Ekman, W. V. Friesen, and J. C. Hager, “Facial Action Coding System: The manual on CD ROM”, *A Human Face, Salt Lake City*, pp. 77–254, 2002.
- [36] P. Ekman and W. V. Friesen, “The repertoire of nonverbal behavior: Categories, origins, usage, and coding”, *Semiotica*, vol. 1, pp. 49–98, 1969.
- [37] P. P. Filntisis, N. Efthymiou, G. Potamianos, and P. Maragos, “Emotion understanding in videos through body, context, and visual-semantic embedding loss”, in *Proc. 16th Eur. Conf. on Computer Vision Workshops (ECCVW) - Workshop on Bodily Expressed Emotion Understanding (BEEU)*, 2020.
- [38] H. Fukui, T. Hirakawa, T. Yamashita, and H. Fujiyoshi, “Attention Branch Network: Learning of attention mechanism for visual explanation”, in *2019 IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019.
- [39] K. Fukushima, “A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position”, *Biological Cybernetics*, vol. 36, pp. 193–202, 1980.
- [40] B. Gelder, “Why bodies? Twelve reasons for including bodily expressions in affective neuroscience”, *Philosophical Trans. of the Royal Society of London. Series B, Biological Sciences*, vol. 364, pp. 3475–3484, 2009.
- [41] R. Girshick, “Fast R-CNN”, in *Proc. Int. Conf. on Computer Vision (ICCV)*, 2015.
- [42] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.
- [43] I. J. Goodfellow, D. Erhan, P. L. Carrier, A. Courville, M. Mirza, B. Hamner, W. Cukierski, Y. Tang, D. Thaler, D.-H. Lee, *et al.*, “Challenges in representation learning: A report on three machine learning contests”, in *Int. Conf. on Neural Information Processing (ICONIP)*, 2013.
- [44] A. Graves, A.-r. Mohamed, and G. Hinton, “Speech recognition with deep recurrent neural networks”, in *2013 IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, 2013.
- [45] H. Gunes and M. Piccardi, “A bimodal face and body gesture database for automatic analysis of human nonverbal affective behavior”, in *18th Int. Conf. on Pattern Recognition (ICPR)*, 2006.
- [46] H. Gunes and M. Piccardi, “Automatic temporal segment detection and affect recognition from face and body display”, *IEEE Trans. on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 39, pp. 64–84, 2009.

-
- [47] H. Gunes and M. Piccardi, “Bi-modal emotion recognition from expressive face and body gestures”, *Journal of Network and Computer Applications*, vol. 30, pp. 1334–1345, 2007.
- [48] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition”, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [49] U. Hess and S. Hareli, “The influence of context on emotion recognition in humans”, in *2015 11th IEEE Int. Conf. and Workshops on Automatic Face and Gesture Recognition (FG)*, 2015.
- [50] C. H. Hjortsjö, *Man’s Face and Mimic Language*. Lund: Studentlitteratur, 1970.
- [51] S. Hochreiter and J. Schmidhuber, “Long short-term memory”, *Neural Computation*, vol. 9, pp. 1735–1780, 1997.
- [52] K. Hornik, “Approximation capabilities of multilayer feedforward networks”, *Neural Networks*, vol. 4, pp. 251–257, 1991.
- [53] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, “Densely connected convolutional networks”, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017.
- [54] S. Ioffe and C. Szegedy, “Batch Normalization: Accelerating deep network training by reducing internal covariate shift”, in *Proc. 32nd Int. Conf. on Machine Learning (ICML)*, 2015.
- [55] S. Ji, W. Xu, M. Yang, and K. Yu, “3D convolutional neural networks for human action recognition”, *IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 35, pp. 221–231, 2013.
- [56] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization”, in *Int. Conf. on Learning Representations (ICLR)*, 2015.
- [57] T. N. Kipf and M. Welling, “Semi-supervised classification with Graph Convolutional Networks”, in *Int. Conf. on Learning Representations (ICLR)*, 2017.
- [58] R. Kostić, J. M. Alvarez, A. Recasens, and A. Lapedriza, “EMOTIC: Emotions in Context Dataset”, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2017.
- [59] R. Kostić, J. M. Alvarez, A. Recasens, and A. Lapedriza, “Context based emotion recognition using EMOTIC dataset”, *IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 42, no. 11, pp. 2755–2766, 2019.
- [60] R. Laban and L. Ullmann, *The Mastery of Movement*. Boston, Plays, 1971.
- [61] Y. LeCun, C. Cortes, and C. Burges, “MNIST handwritten digit database”, 2010.
- [62] J. Lee, S. Kim, S. Kim, J. Park, and K. Sohn, “Context-aware emotion recognition networks”, in *Proc. Int. Conf. on Computer Vision (ICCV)*, 2019.
- [63] Z. Li and N. Snavely, “MegaDepth: Learning single-view depth prediction from internet photos”, in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018.
- [64] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, “Microsoft COCO: Common objects in context”, in *Eur. Conf. on Computer Vision (ECCV)*, 2014.
- [65] D. G. Lowe, “Object recognition from local scale-invariant features”, in *Proc. 7th Int. Conf. on Computer Vision (ICCV)*, 1999.

- [66] B. D. Lucas and T. Kanade, “An iterative image registration technique with an application to stereo vision”, in *Proc. 7th Int. Joint Conf. on Artificial Intelligence*, 1981.
- [67] Y. Luo, J. Ye, R. B. Adams, J. Li, M. G. Newman, and J. Z. Wang, “ARBEE: Towards automated recognition of bodily expression of emotion in the wild”, *Int. Journal of Computer Vision (IJCV)*, vol. 128, pp. 1–25, 2019.
- [68] T. Masuda, P. C. Ellsworth, B. Mesquita, J. Leu, S. Tanida, and E. Van de Veerdonk, “Placing the face in context: Cultural differences in the perception of facial emotion”, *Journal of Personality and Social Psychology*, vol. 94, pp. 365–381, 2008.
- [69] W. S. McCulloch and W. Pitts, “A logical calculus of the ideas immanent in nervous activity”, *The Bulletin of Mathematical Biophysics*, vol. 5, pp. 115–133, 1943.
- [70] A. Mehrabian, J. Russell, J. Russell, and J. Russell, *An Approach to Environmental Psychology*. M.I.T. Press, 1974.
- [71] T. Mittal, P. Guhan, U. Bhattacharya, R. Chandra, A. Bera, and D. Manocha, “EmotiCon: Context-aware multimodal emotion recognition using Frege’s principle”, in *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020.
- [72] T. Mittal, U. Bhattacharya, R. Chandra, A. Bera, and D. Manocha, “M3ER: Multiplicative multimodal emotion recognition using facial, textual, and speech cues”, in *Proc. AAAI Conf. on Artificial Intelligence*, 2020.
- [73] D. Mobbs, N. Weiskopf, H. C. Lau, E. Featherstone, R. J. Dolan, and C. D. Frith, “The Kuleshov Effect: The influence of contextual framing on emotional attributions”, *Social Cognitive and Affective Neuroscience*, vol. 1, pp. 95–106, 2006.
- [74] P. Molchanov, S. Gupta, K. Kim, and J. Kautz, “Hand gesture recognition with 3D convolutional neural networks”, in *2015 IEEE Conf. on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2015.
- [75] A. Mollahosseini, B. Hasani, and M. H. Mahoor, “AffectNet: A database for facial expression, valence, and arousal computing in the wild”, *IEEE Trans. on Affective Computing*, vol. 10, pp. 18–31, 2019.
- [76] Y. Nesterov, “A method for unconstrained convex minimization problem with the rate of convergence $O(1/k^2)$ ”, in *Doklady Akademii Nauk SSSR*, vol. 269, 1983, pp. 543–547.
- [77] V. Nitsch and M. Popp, “Emotions in robot psychology”, *Biological Cybernetics*, vol. 108, pp. 621–629, 2014.
- [78] F. Noroozi, C. A. Corneanu, D. Kamińska, T. Sapiński, S. Escalera, and G. Anbarjafari, “Survey on emotional body gesture recognition”, *IEEE Trans. on Affective Computing*, vol. 12, pp. 505–523, 2021.
- [79] J. Päävärinta, E. Rahtu, and J. Heikkilä, “Volume local phase quantization for blur-insensitive dynamic texture classification”, in *Scandinavian Conf. on Image Analysis (SCIA)*, 2011.
- [80] M. Pantic, “Machine analysis of facial behaviour: Naturalistic and dynamic behaviour”, *Philosophical Trans. of the Royal Society B: Biological Sciences*, vol. 364, pp. 3505–3513, 2009.
- [81] R. Pascanu, T. Mikolov, and Y. Bengio, “On the difficulty of training recurrent neural networks”, in *Int. Conf. on Machine Learning (ICML)*, 2013.

- [82] G. Patterson and J. Hays, “SUN attribute database: Discovering, annotating, and recognizing scene attributes”, in *2012 IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2012.
- [83] A. Pease and B. Pease, *The Definitive Book of Body Language: How to read others’ attitudes by their gestures*. Orion, 2016.
- [84] A. Pease and B. Pease, *The Definitive Book of Body Language / Allan & Barbara Pease*. Pease Int. Buderim, Qld, 2004.
- [85] J. Pennington, R. Socher, and C. D. Manning, “GloVe: Global vectors for word representation”, in *Proc. 2014 Conf. on Empirical Methods in Natural Language Processing (EMNLP)*, 2014.
- [86] I. Pikoulis, P. P. Filntisis, and P. Maragos, *Leveraging semantic scene characteristics and multi-stream convolutional architectures in a contextual approach for video-based visual emotion recognition in the wild*, 2021. arXiv: [2105.07484](https://arxiv.org/abs/2105.07484) [cs.CV].
- [87] R. Plutchik, “The nature of emotions: Human emotions have deep evolutionary roots, a fact that may explain their complexity and provide tools for clinical practice”, *American Scientist*, vol. 89, pp. 344–350, 2001.
- [88] R. Righart and B. De Gelder, “Recognition of facial expressions is influenced by emotional scene gist”, *Cognitive, Affective, & Behavioral Neuroscience*, vol. 8, pp. 264–272, 2008.
- [89] F. Rosenblatt, “The Perceptron: A probabilistic model for information storage and organization in the brain”, *Psychological Review*, vol. 65, pp. 386–408, 1958.
- [90] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, “Learning representations by back-propagating errors”, *Nature*, vol. 323, pp. 533–536, 1986.
- [91] J. Russell and A. Mehrabian, “Evidence for a three-factor theory of emotions”, *Journal of Research in Personality*, vol. 11, pp. 273–294, 1977.
- [92] M. Schuster and K. K. Paliwal, “Bidirectional recurrent neural networks”, *IEEE Trans. on Signal Processing*, vol. 45, pp. 2673–2681, 1997.
- [93] C. Shan, S. Gong, and P. W. McOwan, “Beyond facial expressions: Learning human emotion from body gestures”, in *Proc. British Machine Vision Conf. (BMVC)*, 2007.
- [94] K. Simonyan and A. Zisserman, “Two-stream convolutional networks for action recognition in videos”, in *Proc. 27th Int. Conf. on Neural Information Processing Systems (NIPS)*, 2014.
- [95] H. Singh and Y. A. Lone, “Artificial Neural Networks”, in *Deep Neuro-Fuzzy Systems with Python: With Case Studies and Applications from the Industry*. Apress, 2020, pp. 157–198.
- [96] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: A simple way to prevent neural networks from overfitting”, *Journal of Machine Learning Research (JMLR)*, vol. 15, pp. 1929–1958, 2014.
- [97] J. M. Susskind, A. K. Anderson, and G. E. Hinton, “The Toronto face database”, *Dept. of Computer Science, University of Toronto, Toronto, ON, Canada, Tech. Rep.*, vol. 3, 2010.
- [98] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, “On the importance of initialization and momentum in deep learning”, in *Int. Conf. on Machine Learning (ICML)*, 2013.

- [99] T. Tieleman and G. Hinton, *Lecture 6.5-RMSprop: Divide the gradient by a running average of its recent magnitude*, COURSERA: Neural Networks for Machine Learning, 2012.
- [100] A. Toshev and C. Szegedy, “DeepPose: Human pose estimation via deep neural networks”, in *2014 IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2014.
- [101] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, “Learning spatiotemporal features with 3D convolutional networks”, in *2015 Int. Conf. on Computer Vision (ICCV)*, 2015.
- [102] M. F. Valstar, H. Gunes, and M. Pantic, “How to distinguish posed from spontaneous smiles using geometric features”, in *Proc. 9th Int. Conf. on Multimodal Interfaces (ICMI)*, 2007.
- [103] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. Van Gool, “Temporal Segment Networks for action recognition in videos”, *IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 41, pp. 2740–2755, 2019.
- [104] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. Van Gool, “Temporal Segment Networks: Towards good practices for deep action recognition”, in *Eur. Conf. on Computer Vision (ECCV)*, 2016.
- [105] Z. Wei, J. Zhang, Z. Lin, J.-Y. Lee, N. Balasubramanian, M. Hoai, and D. Samaras, “Learning visual emotion representations from web data”, in *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020.
- [106] M. Wieser and T. Brosch, “Faces in context: A review and systematization of contextual influences on affective face processing”, *Frontiers in Psychology*, vol. 3, 2012.
- [107] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba, “SUN database: Large-scale scene recognition from abbey to zoo”, in *2010 IEEE Computer Society Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2010.
- [108] S. Yan, Y. Xiong, and D. Lin, “Spatial temporal graph convolutional networks for skeleton-based action recognition”, in *Proc. AAAI Conf. on Artificial Intelligence*, 2018.
- [109] C. Zach, T. Pock, and H. Bischof, “A Duality Based Approach for Realtime TV- L^1 Optical Flow”, in *29th DAGM Symp. on Pattern Recognition*, 2007.
- [110] S. Zagoruyko and N. Komodakis, “Wide residual networks”, in *Proc. British Machine Vision Conf. (BMVC)*, 2016.
- [111] Z. Zeng, M. Pantic, G. I. Roisman, and T. S. Huang, “A survey of affect recognition methods: Audio, visual, and spontaneous expressions”, *IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 31, pp. 39–58, 2009.
- [112] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, “Joint face detection and alignment using multitask cascaded convolutional networks”, *IEEE Signal Processing Letters*, vol. 23, pp. 1499–1503, 2016.
- [113] M. Zhang, Y. Liang, and H. Ma, “Context-aware affective graph reasoning for emotion recognition”, in *2019 IEEE Int. Conf. on Multimedia and Expo (ICME)*, 2019.
- [114] W. Zhang, S. Shan, W. Gao, X. Chen, and H. Zhang, “Local Gabor binary pattern histogram sequence (LGBPHS): A novel non-statistical model for face representation and recognition”, in *10th Int. Conf. on Computer Vision (ICCV)*, 2005.

- [115] G. Zhao and M. Pietikainen, “Dynamic texture recognition using local binary patterns with an application to facial expressions”, *IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 29, pp. 915–928, 2007.
- [116] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, “Learning deep features for discriminative localization”, in *2016 IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [117] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba, “Places: A 10 million image database for scene recognition”, *IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 40, pp. 1452–1464, 2018.
- [118] B. Zhou, H. Zhao, X. Puig, T. Xiao, S. Fidler, A. Barriuso, and A. Torralba, “Semantic understanding of scenes through the ADE20K dataset”, *Int. Journal of Computer Vision (IJCV)*, vol. 127, pp. 302–321, 2019.