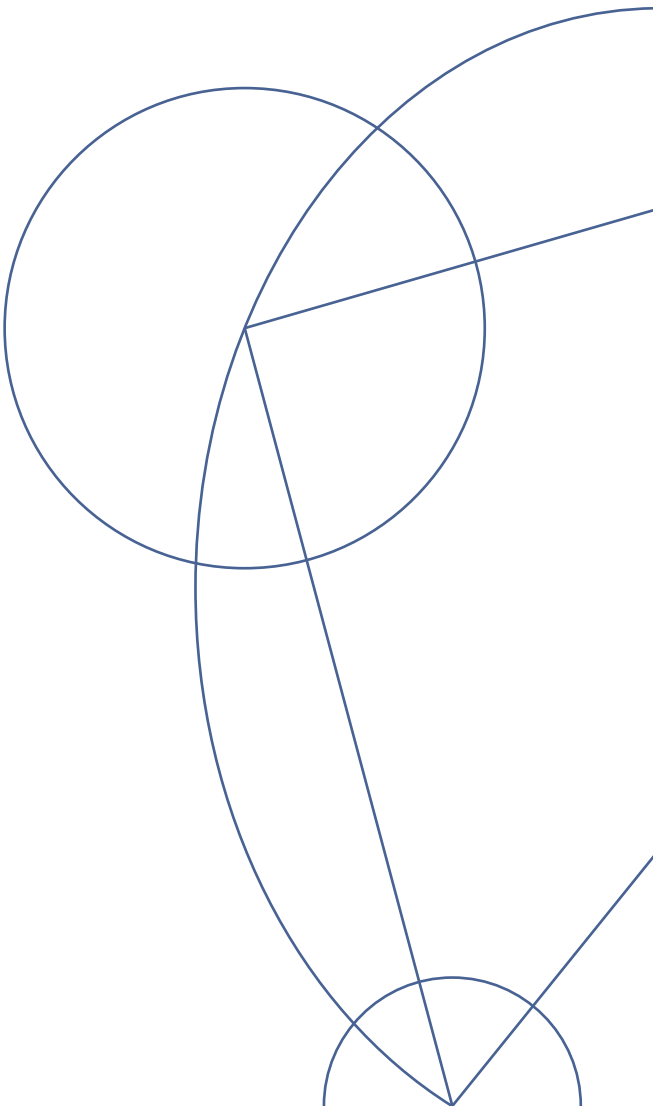
National Technical University of Athens
School of Electrical and Computer Engineering
Inter-Faculty Postgraduate Studies Programme
Data Science and Machine Learning

Self Supervised Multimodal Learning for Emotion Recognition

Master Thesis written in fulfillment of the requirements of the Data Science & Machine Learning post-graduate studies programme.

Supervisor
Dr. Alexandros Potamianos

Author
Konstantinos Poulinakis



May, 2023

Abstract

The purpose of this dissertation is to explore the utilization of self-supervised learning paradigm for multimodal deep learning on the task of emotion recognition. Our work is driven by a compelling question: Can the benefits of self-supervised learning, witnessed in vast-scale datasets, be transferred effectively to multimodal settings, and particularly for small- to medium-sized datasets? In our pursuit, we carry out a thorough investigation of self-supervised multimodal learning and contrast its performance with traditional supervised baselines.

We propose a self-supervised learning (SSL) multimodal framework that adapts existing unimodal SSL frameworks in the multimodal setting. Our framework is adapted to accommodate three modalities (text, visual, audio) and can easily expand to more modalities. The input data are sequential and come in an embedding form instead of their raw form. Our proposal relies on a "siamese networks" framework and a contrastive loss optimization based on the cross-correlation matrix of the two outputs. Our baseline multimodal architecture is based on LSTM networks and self-attention mechanisms. An augmentation module transforms the input features before feeding the two transformed views to the multimodal architecture. A projector network projects the outputs in a joint embedding space on which the contrastive loss is minimized. Through SSL our models learn transformation invariant features from unlabeled data that prove helpful in the task of emotion recognition.

A series of experimental trials, confirm that self-supervision leads to performance enhancements across all six evaluation metrics. The **absolute** metric improvements range from 0.4% to 1.3% compared to the supervised baselines. Moreover, we find out that SSL models require 20% - 40% **fewer** labeled data during fine-tuning in order to match the performance of the supervised baseline trained on the whole dataset. We conduct our experiments with the pre-extracted features of the CMU-MOSEI dataset. Our proposed methodology competes favorably with the existing state-of-the-art, yielding highly competitive results.

In order to learn transformation-invariant representations we engage three augmentation techniques adequate for multimodal pre-extracted feature inputs, namely, Gaussian noise, masking, SeqAug, and their combinations. We perform a thorough exploration to discern the most optimal combination and hyperparameters.

Finally, we propose multiple variations and provide guidance for future work that can expand upon our proposed framework.

Keywords: Self Supervised Learning, Multimodal Learning, Deep Learning, Machine Learning, Emotion Recognition, Recurrent Neural Networks, LSTM, Self-Attention

Περίληψη

Σκοπός αυτής της διατριβής είναι να διερευνήσει τη χρήση της αυτοεπιβλεπόμενης μάθησης (self-supervised learning, SSL) σε πολυτροπικά (multimodal) μοντέλα βαθιάς μάθησης, εφαρμοσμένα πάνω στο έργο της αναγνώρισης συναισθημάτων. Η εργασία μας καθοδηγείται από ένα συναρπαστικό ερώτημα: Μπορούν τα οφέλη της αυτοεπιβλεπόμενης μάθησης, που παρατηρούνται σε ευρείας κλίμακας σύνολα δεδομένων, να μεταφερθούν αποτελεσματικά σε ένα πολυτροπικό πλαίσιο, και ιδιαίτερα για σύνολα δεδομένων μικρού έως μεσαίου μεγέθους; Στο πλαίσιο της επιδίωξής μας, πραγματοποιούμε μια διεξοδική έρευνα της αυτοεπιβλεπόμενης πολυτροπικής μάθησης και αντιπαραβάλλουμε την απόδοσή της με τις παραδοσιακές τεχνικές επιβλεπόμενης μάθησης.

Προτείνουμε μια πολυτροπική μέθοδο αυτοεπιβλεπόμενης μάθησης που προσαρμόζει τις υπάρχοντες μονοτροπικές μεθόδους αυτοεπιβλεπόμενης μάθησης σε πολυτροπικό περιβάλλον. Το πλαίσιο μας είναι προσαρμοσμένο για να επεξεργάζεται τρεις μορφές δεδομένων (τροπικότητες), κείμενο, εικόνα, ήχο. Όμως μπορεί εύκολα να επεκταθεί σε ακόμα περισσότερες τροπικότητες. Τα δεδομένα εισόδου είναι διαδοχικά και έρχονται σε μορφή προεξαγόμενων χαρακτηριστικών (embeddings). Η πρότασή μας βασίζεται σε ένα πλαίσιο "σιαμέων δικτύων" και στην βελτιστοποίηση μιας αντιθετικής συνάρτησης κόστους (contrastive loss) που βασίζεται στον πίνακα ετεροσυσχέτισης των δύο εξόδων. Η βασική πολυτροπική αρχιτεκτονική μας βασίζεται σε δίκτυα LSTM και μηχανισμούς αυτοπροσοχής. Στην είσοδο ένα augmentation module μετασχηματίζει τα χαρακτηριστικά εισόδου πριν τροφοδοτήσει τις δύο μετασχηματισμένες εισόδους στην πολυτροπική αρχιτεκτονική. Ένα δίκτυο projector προβάλλει τις εξόδους σε έναν κοινό χώρο αναπαράστασης στον οποίο ελαχιστοποιείται η συνάρτηση κόστους. Μέσω της αυτοεπιβλεπόμενης μάθησης τα μοντέλα μας μαθαίνουν αμετάβλητα χαρακτηριστικά από δεδομένα χωρίς ετικέτες, που αποδεικνύονται χρήσιμα για την αναγνώριση συναισθημάτων.

Μια σειρά πειραματικών δοκιμών επιβεβαιώνει ότι η αυτοεπίβλεψη οδηγεί σε βελτιώσεις απόδοσης και στις έξι μετρικές αξιολόγησης. Οι βελτιώσεις κυμαίνονται από 0,4% έως 1,3% απόλυτες μονάδες σε σύγκριση με την επιβλεπόμενη μάθηση. Επιπλέον, ανακαλύπτουμε ότι τα μοντέλα αυτοεπιβλεπόμενης μάθησης απαιτούν 20% - 40% **λιγότερα** δεδομένα με ετικέτα, προκειμένου να ισοφαρίσουν την απόδοση της επιβλεπόμενης μάθησης που εκπαιδεύεται σε ολόκληρο το σύνολο δεδομένων. Διεξάγουμε τα πειράματά μας με χαρακτηριστικά του συνόλου δεδομένων CMU-MOSEI. Η προτεινόμενη μεθοδολογία μας ανταγωνίζεται ευνοϊκά την υπάρχουσα state-of-the-art τεχνολογία, αποδίδοντας πολύ ανταγωνιστικά αποτελέσματα.

Προκειμένου να μάθουμε αναπαραστάσεις αναλλοίωτες σε μετασχηματισμούς, χρησιμοποιούμε τρεις τεχνικές μετασχηματισμού κατάλληλες για πολυτροπικές εισόδους προεξαγόμενων χαρακτηριστικών, συγκεκριμένα, Gaussian noise, masking, SeqAug και τους συνδυασμούς τους. Πραγματοποιήσαμε ενδελεχή εξερεύνηση για να βρεθεί τόσο ο βέλτιστος μετασχηματισμός όσο και οι βέλτιστες υπερπαράμετροι.

Τέλος, προτείνουμε πολλαπλές παραλλαγές και παρέχουμε καθοδήγηση για μελλοντικές εργασίες που μπορούν να επεκτείνουν την προτεινόμενη μέθοδο.

Λέξεις Κλειδιά: Αυτοεπιβλεπόμενη Μάθηση, Πολυτροπική Μάθηση, Βαθιά Μάθηση, Μηχανική Μάθηση, Αναγνώριση Συναισθημάτων, Αναδρομικά Νευρωνικά Δίκτυα, LSTM, Αυτοπροσοχή, Ενδοπροσοχή

Acknowledgments

I would like to thank my supervisor, Professor Alexandros Potamianos, for his guidance, useful critiques of this research work, and all of his interesting lectures that motivated me to delve deeper into the field of Deep Learning.

I would also like to express my profound gratitude to Ph.D. candidate Efthymios Georgiou. His patient guidance, genuine engagement, and interest in this work have been instrumental in its fruition. His encouragement and the time he devoted to assisting me were invaluable in achieving the objectives of this work. I am also indebted to him for his excellent contributions to the field, upon which I was able to build.

Lastly, I would like to thank my loved ones for their support during this challenging year.

Athens, May 2023

Konstantinos Poulinakis

Table of Contents

Chapter 0

Εκτεταμένη Ελληνική Περίληψη

0.1 Εισαγωγή

Τα τελευταία χρόνια, γίναμε μάρτυρες μιας θεαματικής επανάστασης στον τομέα της τεχνητής νοημοσύνης (ΤΝ). Αυτή η ραδγαία ανάπτυξη δεν είναι θέμα απλώς ακαδημαϊκό. Η ΤΝ είναι πλέον ενσωματωμένη στην καθημερινή μας ζωή. Από την αναγνώριση προσώπων στις φωτογραφίες μας μέχρι την κατανόηση και συγγραφή κειμένου που μοιάζει με ανθρώπινη δημιουργία. Η ΤΝ δεν αποτελεί πλέον μια φουτουριστική ιδέα αλλά μία νέα πραγματικότητα.

Στο επίκεντρο αυτής της δραματικής αλλαγής βρίσκονται οι αλγόριθμοι Βαθιάς Μάθησης. Πλέον τα Βαθιά Νευρωνικά Δίκτυα (ΒΝΔ) αποτελούν την κινητήρια δύναμη πίσω από τις πιο εκπληκτικές εφαρμογές και ωθούν τα όρια αυτού που πιστεύαμε ότι ήταν δυνατό ακόμα πιο πέρα. Η αυτόνομη οδήγηση, η αυτόματη παραγωγή κώδικα και η δημιουργία ψηφιακής τέχνης είναι μερικά από τα εντυπωσιακά επιτεύγματα της βαθιάς μάθησης.

Όμως, υπάρχει ένα ακόμα μυστικό συστατικό που ώθησε τον τομέα της βαθιάς μάθησης τόσο μακριά. Αυτό είναι τα τεράστια ποσά δεδομένων που είναι διαθέσιμα για την εκπαίδευση αυτών των αλγορίθμων. Τα σύνολα δεδομένων αποτελούν την πηγή γνώσης για κάθε αλγόριθμο βαθιάς μάθησης, αλλά στις επικρατούμενες τεχνικές εκμάθησης είναι απαραίτητη η συμμετοχή της ανθρώπινης νοημοσύνης ώστε να δώσει την κατάλληλη μορφή και το κατάλληλο σήμα γνώσης σε αυτά τα δεδομένα. Χρειάζονται αυτό που ονομάζουμε "ετικέτες". Όμως, η επανεμφάνιση της αυτοεπιβλεπόμενης μάθησης άλλαξε ριζικά το τοπίο. Πλέον οι αλγόριθμοι μπορούν να μαθαίνουν μέσα από υπέρογκα ποσά δεδομένων με ελάχιστη ή καθόλου ανθρώπινη παρέμβαση. Η αυτοεπιβλεπόμενη μάθησης ωθεί μία νέα εποχή όπου μπορούμε να εκμεταλευτούμε όλη την ανθρώπινη γνώση που υπάρχει ελεύθερη στο διαδίκτυο σε μορφή κειμένου, κώδικα ή φωτογραφικού υλικού. Η γνώση που εκμαιεύεται μέσα από αυτά τα τεράστια δεδομένα είναι γενική και όχι απλά ειδική. Με άλλα λόγια, δημιουργεί μοντέλα που έχουν την δυνατότητα να ολοκληρώσουν με επιτυχία πολλαπλά έργα ακόμα και αν ποτέ δεν εκπαιδεύτηκαν ρητά πάνω σε αυτά. Η αυτοεπιβλεπόμενη μάθηση αποτελεί βασικό πυλώνα της παρούσας εργασίας.

Μία ακόμα συγκλονιστική τάση στην ΤΝ είναι η δημιουργία μοντέλων με την ικανότητα να επεξεργάζονται και να κατανοούν πολλαπλές μορφές δεδομένων (τροπικότητες). Μέχρι πριν λίγα χρόνια τα περισσότερα μοντέλα ήταν εκπαιδευμένα να ολοκληρώνουν ένα έργο το οποίο αφορούσε μία τροπικότητα όπως για παράδειγμα αναγνώριση αντικείμενων σε εικόνες, κατανόηση κειμένου ή κατανόηση ακουστικών σημάτων. Πλέον η τάση είναι τα μοντέλα να μπορούν να επεξεργαστούν τροπικότητες εικόνας, κειμένου, ήχου και άλλα, ταυτόχρονα. Μερικές από τις σύγκρονες εντυπωσιακές εφαρμογές πολυτροπικών μοντέλων είναι τα λεγόμενα μοντέλα κειμένουν-σε-εικόνα (text-to-image models). Μία πρόταση σε κείμενο είναι αρκετή ώστε να παραχθεί μία ακριβής απεικόνιση της περιγραφής σε εικόνα ή βίντεο! Η πολυτροπική μάθησης αποτελεί τον δεύτερο βασικό πυλώνα αυτής της εργασίας.

Η αυτοεπιβλεπόμενη εκπαίδευση πολυτροπικών μοντέλων είναι το κύριο αντικείμενο διερεύνησης

της παρούσας διατριβής. **Σκοπός μας είναι να μελετήσουμε και να σχεδιάσουμε τεχνικές αυτοεπιβλεπόμενης εκπαίδευσης κατάλληλες για πολυτροπικά δεδομένα και ιδίως για μικρού έως μεσαίου μεγέθους σύνολα.** Η μέθοδος μας βρίσκει άμεση εφαρμογή στην αναγνώριση των ανθρώπινων συναισθημάτων. Ως άνθρωποι έχουμε εκ φύσεως την ικανότητα να "διαβάζουμε" τους άλλους ανθρώπους και να κατανοούμε τα συναισθήματα τους και τις προθέσεις τους. Η εξέλιξη του είδους μας απαιτεί να είμαστε σε θέση να αντιλαμβανόμαστε αν ένας άλλος άνθρωπος είναι νευριασμένος και πιθανώς εχθρικός ή αν είναι φιλικός και συνεργάσιμος. Ακόμα και αν δεν γίνεται συνειδητά, οι άνθρωποι χρησιμοποιούμε όλες μας τις αισθήσεις για να κρίνουμε τα συναισθήματα των άλλων ανθρώπων. Πέρα από τα λεκτικά στοιχεία, η γλώσσα του σώματος και το ηχόχρωμα της φωνής αποτελούν πολύ πλούσιες πηγές πληροφορίας. Η προτεινόμενη μέθοδος μας βασίζεται σε αυτά τα στοιχεία και αποσκοπεί στην άντληση πληροφοριών από τρεις μορφές πληροφορίας ταυτόχρονα, την εικόνα, το κείμενο και τον ήχο.

0.2 θεωρητικό υπόβαθρο

Θα ξεκινήσουμε αυτήν την εκτεταμένη περίληψη μελετώντας κάποιες βασικές έννοιες, απαραίτητες για την κατανόηση της προτεινόμενης μεθόδου. Αρχικά παρουσιάζουμε σύντομα τις ένοιες της επιβλεπόμενης μάθησης, των νευρωνικών δικτύων και στην συνέχεια κάποιων πιο πολύπλοκων αρχιτεκτονικών βασισμένων σε νευρωνικά δίκτυα όπως τα αναδρομικά νευρωνικά δίκτυα, τα δίκτυα μετασχηματιστών (Transformers) και τον μηχανισμό αυτοπροσοχής (ή ενδοπροσοχής). Έπειτα αναλύεται η αυτοεπιβλεπόμενη μάθηση ώστε ο αναγνώστης να μπορέσει να κατανοήσει τις τεχνικές που χρησιμοποιούνται στην συνέχεια.

0.2.1 Βασικές Αρχές και Αρχιτεκτονικές Βαθιάς Μάθησης

Επιβλεπόμενη Μάθηση

Ένα ***σύνολο δεδομένων*** είναι μία συλλογή από στατιστικά ***δείγματα*** κάθε ένα εκ των οποίων έχει ένα σύνολο από ***χαρακτηριστικά***, είτε κατηγορικά είτε αριθμητικά. Ο όρος επιβλεπόμενη αφορά τις ***ετικέτες*** που δημιουργούν οι άνθρωποι και συνοδεύουν κάθε δείγμα. Οι ετικέτες συνήθως τοποθετούν κάθε δείγμα σε κάποια κατηγορία, που συχνά αναφέρεται ως ***κλάση***. Για παράδειγμα ένα σύνολο δεδομένων από emails θα μπορούσε να κατηγοριοποιηθεί σε spam ή no-spam. Τα χαρακτηριστικά κάθε δείγματος θα μπορούσαν να είναι ο αποστολέας του, το μέγεθός τους, το περιεχόμενό του και άλλα.

Ένας αλγόριθμος επιβλεπόμενης μηχανικής μάθησης που εκπαιδεύεται να ξεχωρίζει τα spam ή no-smap emails είναι κατουσία μία συνάρτηση η οποία χαρτογραφεί μία είσοδο x (χαρακτηριστικά) στις σωστές εξόδους $\tilde{y}$ (ετικέτες). Ο τρόπος με τον οποίο μαθαίνεται αυτή η συνάρτηση $y = \mathbf{f}(x;\theta)$ είναι μέσω της ελαχιστοποίησης μίας ***συνάρτησης κόστους*** $\mathscr{L}$ ή αλλιώς ***συνάρτησης σφάλματος***. Μία συνήθης συνάρτηση κόστους που χρησιμοποιείται σε προβλήματα παλινδρόμησης είναι το μέσο τετραγωνικό σφάλμα

$$\mathscr{L}(\tilde{y}, y) = \mathscr{L}(\tilde{y}, f(x;\theta)) = \frac{1}{N}\sum_{i=1}^{N}(\tilde{y}_i - f(x_i;\theta))^2 \tag{0.2.1}$$

Βαθιά Νευρωνικά Δίκτυα και Εκπαίδευση

Τα Βαθιά Νευρωνικά Δίκτυα, ή απλά Νευρωνικά Δίκτυα, αποτελούν τη βάση της σύγχρονης βαθιάς μάθησης. Εμπνευσμένα από τη δομή του ανθρώπινου εγκεφάλου, τα τεχνητά νευρωνικά δίκτυα

αποτελούνται από διασυνδεδεμένους κόμβους, γνωστούς και ως νευρώνες ή μονάδες, δομημένοι σε στρώματα. Κάθε στρώμα μετασχηματίζει την είσοδό του σε μια πιο αφηρημένη αναπαράσταση, οδηγώντας σε μια ιεραρχία αυξανόμενα σημαντικών χαρακτηριστικών καθώς προχωράμε βαθύτερα στο δίκτυο.

Τα βαθιά νευρωνικά δίκτυα αποτελούνται από πολλαπλά στρώματα (ή επίπεδα) που είναι δεμένα μεταξύ τους. Στην πιο απλή τους μορφή, τα ΒΝΔ αποτελούνται μόνο από γραμμικά στρώματα και μη γραμμικές συναρτήσεις ενεργοποίησης που βοηθούν το μοντέλο να απεικονίζει μη γραμμικότητες και να αποφεύγει την κατάρρευση. Κάθε επίπεδο αποτελείται από πολλαπλούς νευρώνες κάθε ένας εκ των οποίων ορίζεται ως

$$y = \sigma(\mathbf{w}^T x + b) \tag{0.2.2}$$

όπου σ είναι μία μη-γραμμική συνάρτηση ενεργοποίησης, $\mathbf{w}$ είναι ένα διάνυσμα βαρών και b μία βαθμωτή τιμή που ονομάζεται πόλωση.

Τα νευρωνικά βαθιά νευρωνικά δίκτυα αποτελούνται από πολλαπλά επίπεδα κάθε ένα από τα οποία έχει πολλαπλούς νευρώνες με δική τους συνάρτηση ενεργοποίησης. Το πρώτο επίπεδο ονομάζεται επίπεδο εισόδου, το τελευταίο επίπεδο είναι συνήθως το επίπεδο εξόδου που παράγει τις προβλέψεις ενώ όλα τα ενδιάμεσα επίπεδα ονομάζοντια κρυφά επίπεδα. Μαθηματικά ένα νευρωνικό δίκτυο αναπαρίσταται ως

$$y = f_L(W^{(L)} f(W^{(L-1)} f(...f(W^{(1)} x + b^{(1)})...) + b^{(L-1)}) + b^{(L)}) \,, \tag{0.2.3}$$

Η εκπαίδευση ενός νευρωνικού δικτύου ώστε να εκτελεί ένα έργο έχεις ως σκοπό τον εντοπισμό των κατάλληλων παραμέτρων θ^* που ελαχιστοποιούν την συνάρτηση κόστους $\mathscr{L}$. Η ελαχιστοποίηση επιτυγχάνεται με την μέθοδο Κατάβασης Κλίσης (Cauchy, 1847) η οποία επαναληπτικά ανανεώνει τις παραμέτρους θ της συνάρτησης $f(x;\theta)$ έως ώτου επιτευχθεί σύγκλιση.

$$\theta = \theta - \alpha \nabla \mathscr{L}(\theta) \tag{0.2.4}$$

όπου α είναι μία παράμετρος που ονομάζεται ***ρυθμός εκμάθησης***.

Οι μερικές παράγωγοι υπολογίζονται με τη μέθοδο της Οπισθοδιάδοσης (Rumelhart et al., 1986) η οποία πρώτα εκτελεί ένα πρόσθιο πέρασμα, παράγει μία έξοδο και υπολογίζει το σφάλμα της εξόδου. Στην συνέχεια εκτελεί ένα οπίσθιο πέρασμα όπου υπολογίζεται η συνεισφορά κάθε παραμέτρου στο σφάλμα μέσω της μερικής του παραγώγου και ανανεώνεται βάση της εξίσωσης 0.2.4. Η συνεισφορά κάθε παραμέτρου στο σφάλμα υπολογίζεται βάση τον κανόνα της αλυσίδας

$$\frac{\partial \mathscr{L}}{\partial z} = \frac{\partial \mathscr{L}}{\partial a} \cdot \frac{\partial a}{\partial z} \tag{0.2.5}$$

Αναδρομικά Νευρωνικά Δίκτυα

Τα αναδρομικά νευρωνικά δίκτυα δημιουργήθηκαν ώστε να γίνει δυνατή η επεξεργασία σειριακών δεδομένων, δηλαδή δεδομένα τα οποία διακατέχονται από κάποια μορφή χρονικής εξάρτησης. Τέτοια παραδείγματα είναι οι χρονοσειρές δεδομένων (π.χ. καρδιογράφημα, τιμές μετοχών) αλλά και η φυσική γλώσσα η οποία αν και δεν διακατέχεται από χρονική εξάρτηση η θέση κάθε λέξης σε μία πρόταση καθώς και οι γειτονικές τις λέξεις έχουν μεγάλη επίδραση στην σημασία της ίδιας αλλά και του ευρύτερου συνόλου. Σκοπός είναι μια σειριακή είσοδος $(x1, ..., xn)$ να χαρτογραφηθεί πάνω σε μία συνήθως σειριακή έξοδο $(y1, ..., yn)$.

Για να μοντελοποιηθούν αποτελεσματικά τέτοιες μορφές δεδομένων είναι απαραίτητο τα νευρωνικά δίκτυα να έχουν κάποια μορφή μνήμης. Η μνήμη αυτή δημιουργείται μέσω ενός διανύσματος $\mathbf{h}$ που ονομάζεται κρυφή κατάσταση (Elman, 1990). Μαθηματικά τα αναδρομικά δίκτυα περιγράφονται από τις εξισώσεις

$$\mathbf{h}_t = \sigma_h(\mathbb{U}_h h_{t-1} + \mathbb{W}_h \mathbf{x}_t + \mathbf{b}_h), \tag{0.2.6}$$
$$\mathbf{o}_t = \sigma_o(\mathbb{W}_o h_t + \mathbf{b}_o), \tag{0.2.7}$$

, $\mathbf{h}_t$ είναι η κρυφή κατάσταση την χρονική στιγμή t, $\mathbf{x}_t$ είναι η είσοδος την χρονική στιγμή t, $\mathbf{o}_t$ είναι η έξοδος του δικτύου, $\mathbb{U}_h, \mathbb{W}_h, \mathbb{W}_o$ είναι πίνακες βαρών προς εκμάθηση, και $\mathbf{b}_h, \mathbf{b}_o$ είναι η πολώσεις. Τέλος, σ_h και σ_o είναι μη-γραμμικές συναρτήσεις ενεργοποίησης (Goodfellow et al., 2016).

Όμως, αποδείχθηκε ότι τα απλά αναδρομικά δίκτυα είναι δύσκολο να εκπαιδευτούν διότι πάσχουν από τα προβλήματα εξασθενούμενων και εκτοξευόμενων παραγώγων (vanishing and exploding gradients) (Hochreiter, 1998). Σε αυτές τις περιπτώσεις οι μερικές παράγωγοι που υπολογίζονται κατά την οπισθοδιάδοση γίνονται είτε υπορβολικά μικρές οδηγώντας σε σχεδόν μηδενικές τιμές είτε εκτοξεύονται σε τεράστιες τιμές με αποτέλεσμα να είναι αδύνατη η σύκλιση του αλγορίθμου ελαχιστοποίησης. Η λύση βρέθηκε μερικά χρόνια αργότερα με την δημιουργία των δικτύων μακροπρόθεσμης-βραχυπρόθεσμης μνήμης (LSTM) (Hochreiter and Schmidhuber, 1997).

Τα δίκτυα LSTM είναι παρόμοια με τα αναδρομικά αλλά προστέθηκε ένα ακόμα στοιχείο μνήμης, η κατάσταση κυττάρου c. Η αλλαγή της τιμής της κατάστασης κυττάρου περιορίζεται από τις συναρτήσεις πύλης οι οποίες επιλέγουν ποιες καινούριες πληροφορίες θα γραφτούν ή θα διαγραφούν από το κύτταρο μνήμης. Η εικόνα 0.2.1 παρουσιάζει ένα σχεδιάγραμμα αυτών των δικτύων.

Ένα δίκτυο LSTM περιγράφεται με έξι εξισώσεις και αποτελεί ένα από τα πιο πετυχημένα δίκτυα με πολλές εφαρμογές στην έπεξεργασία φυσικής γλώσσας και την πρόβλεψη χρονοσειρών. Οι εξισώσεις που περιγράφουν μία LSTM είναι οι παρακάτω

$$\mathbf{f}_t = \sigma(\mathbb{U}_f \mathbf{h}_{t-1} + \mathbb{W}_f \mathbf{x}_t + \mathbf{b}_f), \tag{0.2.8}$$
$$\mathbf{i}_t = \sigma(\mathbb{U}_i \mathbf{h}_{t-1} + \mathbb{W}_i \mathbf{x}_t + \mathbf{b}_i), \tag{0.2.9}$$
$$\tilde{\mathbf{C}}_t = \tanh(\mathbb{U}_c \mathbf{h}_{t-1} + \mathbb{W}_c \mathbf{x}_t + \mathbf{b}_c), \tag{0.2.10}$$
$$\mathbf{C}_t = \mathbf{f}_t \odot \mathbf{C}_{t-1} + \mathbf{i}_t \odot \tilde{\mathbf{C}}_t, \tag{0.2.11}$$
$$\mathbf{o}_t = \sigma(\mathbb{U}_o \mathbf{h}_{t-1} + \mathbb{W}_o \mathbf{x}_t + \mathbf{b}_o), \tag{0.2.12}$$
$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{C}_t), \tag{0.2.13}$$
$$\tag{0.2.14}$$

όπου $\mathbf{C}_t$ είναι η τιμή του κυττάρου μνήμης την χρονική στιγμή t; $\mathbf{f}_t$, $\mathbf{i}_t$; και $\mathbf{o}_t$ είναι οι συναρτήσεις πύλεων λήθης, εισόδου και εξόδου την χρονική στιγμή t, αντίστοιχα. $\mathbb{W}_k, \mathbb{U}_k$ είναι πίνακες βαρών προς εκμάθηση, ενώ $\mathbf{b}_f, \mathbf{b}_i, \mathbf{b}_C, \mathbf{b}_o$ είναι διανύσματα πόλωσης προς εκμάθηση. Το σύμβολο $\odot$ παριστάνει το Hadamard product, ενώ οι σ και *tanh* είναι η σιγμοειδής συνάρτηση ενεργοποίησης και της υπερβολικής εφαπτομένης αντίστοιχα.

Μηχανισμός Αυτοπροσοχής και Δίκτυα Μετασχηματιστών

Παρότι τα δίκτυα LSTM είναι πιο ικανά από τα απλά αναδρομικά δίκτυα, πάσχουν και αυτά από τα ίδια προβλήματα όταν το μήκος της εισόδου μεγαλώσει αρκετά. Το 2017 η δουλειά των (Vaswani et al., 2017b) παρουσίασε μία νέα αρχιτεκτονική η οποία είναι πολύ ικανότερη στην επεξεργασία σειριακών δεδομένων όπως η φυσική γλώσσα, αλλά ακόμα και των εικόνων με κάποιες μετατροπές.

Η αρχιτεκτονική αυτή ονομάζεται Μετατροπέας (Transformer) και χρησιμοποιεί έναν ***μηχανισμό αυτοπροσοχής (ή ενδοπροσοχής)*** ο οποίος εντοπίζει συσχετίσεις στα σειριακά δεδομένα εισόδου. Ένα επίπεδο αυτοπροσοχής (self-attention layer) επεξεργάζεται την σειριακή είσοδο τμηματικά. Μέσω της ενδοπροσοχής η σειριακή είσοδος μοντελοποιείται με πιο περίπλοκο τρόπο που επιτρέπει τον εντοπισμό σημαντικών κρυφών συσχετίσεων μεταξύ των δεδομένων εισόδου.

Κάθε στοιχείο της σειριακής εισόδου μπορεί να λάβει τρεις διαφορετικούς ρόλους:

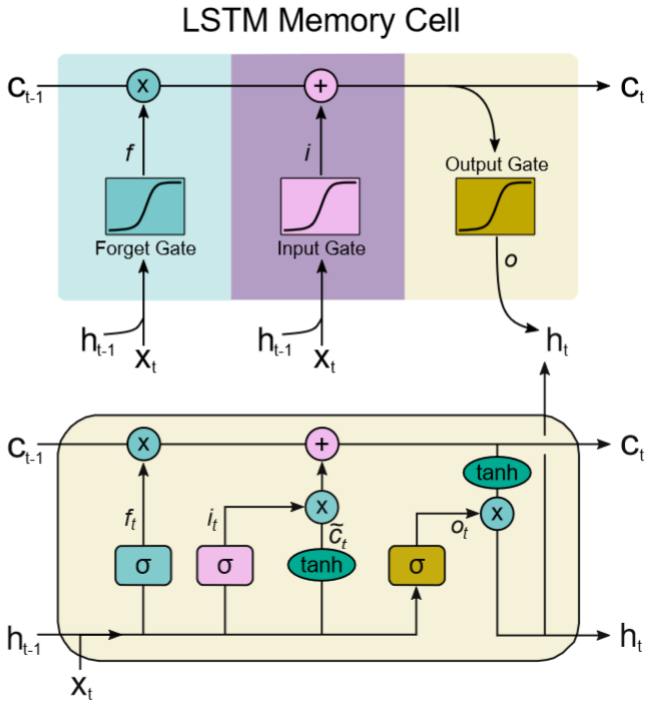


Figure 0.2.1: Μια αναπαράσταση υψηλού επιπέδου του δικτύου LSTM (άνω σχήμα) και μια σχηματική απεικόνιση χαμηλού επιπέδου του μπλοκ ενός δικτύου LSTM (κάτω σχήμα). Πηγή (Poulinakis et al.)

- **Ως Ερώτημα (Query)**, όταν αποτελεί το τρέχον επίκεντρο της προσοχής και συγκρίνεται με όλα τα προηγούμενα στοιχεία της σειριακής εισόδου.
- **Ως Κλειδί (Key)**, όταν είναι ένα από τα προηγούμενα στοιχεία της εισόδου που συγκρίνεται με την τρέχουσα εστίαση της προσοχής (το ερώτημα).
- **Ως Τιμή (Value)**, όταν χρησιμοποιείται για τον υπολογισμό της εξόδου που σχετίζεται με την τρέχουσα εστίαση της προσοχής.

Δίνοντας τρεις ρόλους σε κάθε στοιχείο, ο μηχανισμός ενδοπροσοχής του μετασχηματιστή διαφοροποιεί τη σημασιολογία μιας λέξης στο γενικότερο πλαίσιο, ανάλογα με τον ρόλο που έχει κατά τη διάρκεια μιας σύγκρισης. Για να το πετύχει αυτό, χρησιμοποιεί τρεις πίνακες βαρών

$$q_i = W_Q x_i,$$
$$k_i = W_K x_i,$$
$$v_i = W_V x_i.$$

Για να υπολογιστεί μία έξοδος αρχικά υπολογίζεται η ομοιότητα μεταξύ όλων των δυνατών ζευγών της εισόδου $x_1, x_2, ..., x_n$. Η ομοιότητα αυτή μπορεί να υπολογιστεί με διάφορους τρόπους αλλά το εσωτερικό γινόμενο δύο διανυσμάτων είναι μια συχνή μετρική.

$$s(x_i, x_j) = \frac{q_i . k_j}{\sqrt{d_k}} \tag{0.2.15}$$

Στην συνέχεια μέσω κανονικοποίησης υπολογίζεται η ενδοπροσοχή (attention score) μεταξύ δύο στοιχείων. Όσο μεγαλύτερη είναι αυτή η μετρική τόσο μεγαλύτερη συσχέτιση υπάρχει μεταξύ αυτών

των δύο στοιχείων και η σχέση τους είναι σημαντική.

$$\alpha_{ij} = \text{softmax}(\text{s}(x_i, x_j)) \quad \forall j \leq i \tag{0.2.16}$$

$$= \frac{\exp\big(\text{s}(x_i, x_j)\big)}{\sum_{k=1}^{i} \exp(\text{s}(x_i, x_k))} \quad \forall j \leq i \tag{0.2.17}$$

Τέλος, αφού υπολογιστούν οι ενδοπροσοχές μεταξύ όλων των στοιχείων τότε η έξοδος y_i υπολογίζεται μέσω του παρακάτω σταθμισμένου αθροίσματος.

$$y_i = \sum_{j<i} \alpha_{ij} v_j \tag{0.2.18}$$

όπου d_k είναι η διαστατικότητα των q, v, k διανυσμάτων και ο παρανομαστής χρησιμοποιείται για λόγους κανονικοποίησης.

0.2.2 Αυτοεπιβλεπόμενη Μάθηση

Η αυτοεπιβλεπόμενη μάθηση δεν αποτελεί κάποια ιδιαίτερα καινούρια ιδέα καθώς πρώιμες ιδέες μπορούν να εντοπιστούν ακόμα και στα πρώτα χρόνια έρευνας της βαθιάς μάθησης (Balestriero et al., 2023). Όμως το 2020 περίπου αναγεννήθηκε χάρη στην ευρεία διαθεσιμότητα μεγάλων δεδομένων και ισχυρών καρτών γραφικών (GPU) με μεγάλα ποσά ενσωματομένης μνήμης RAM.

Η αυτοεπιβλεπόμενη μάθηση αποτελεί μία φιλοσοφία εκμάθησης η οποία βρίσκεται στο ενδιάμεσο μεταξύ επιβλεπόμενης και μή-επιβλεπόμενης μάθησης. Μέσω τεχνικών αυτοεπιβλεπόμενης μάθησης μπορούμε να δημιουργήσουμε επιβλεπόμενα σενάρια εκπαίδευσης από δεδομένα χωρίς ετικέτες. Αυτό επιτυγχάνεται μέσω της χρήσης ***ψευδο-ετικετών*** δηλαδή αυτόματα παραγμένων ετικέτων που απαιτούν ελάχιστη ή καθόλου ανθρώπινη παρέμβαση. Μέσω αυτών των ψευδο-ετικετών, τα μοντέλα μπορούν να εκπαιδευτούν σε ***έργα προ-εκπαίδευσης*** γνωστά ως "pretext tasks" τα οποία βοηθούν τα δίκτυα να μάθουν γενικής φύσεως αναπαραστάσεις των δεδομένων που αποδεικνύονται πολύ χρήσιμες σε πολλαπλές εφαρμογές. Τα pretext tasks συχνά δεν σχετίζονται με την τελική εφαρμογή, αυτό που συχνά ονομάζεται ***"downstream task"***.

Μέσω της προεκπαίδευσης στα pretext tasks, τα βάρη των δικτύων δεν είναι πλέον τυχαία αλλά κρύβουν ήδη κάποια γνώση. Μετέπειτα, μπορούμε να προσαρμόσουμε τα μοντέλα μας στις τελικές τους εφαρμογές είτε εκπαιδεύοντας έναν ταξινομητή (π.χ. ένα γραμμικό επίπεδο) πάνω στις αυτοεπιβλεπόμενες αναπαραστάσεις που έχει ήδη μάθει το δίκτυο, είτε επανεκπαιδεύοντας ολόκληρο το δίκτυο. Κατά την προσαρμογή στην τελική εφαρμογή απαιτούνται δεδομένα με ετικέτες αλλά ο αριθμός που απαιτείται είναι συχνά (αρκετά) μικρότερος σε σχέση με την επιβλεπόμενη εκπαίδευση.

Στην αυτοεπιβλεπόμενη εκπαίδευση υπάρχουν δύο κατηγορίες μεθόδων (Lilian Weng, 2021), αυτές είναι οι

- Αντιθετική Μάθηση (Contrastive Learning)
- Ενδοπρόβλεψη (Self-prediction)

Αντιθετική Μάθηση

Οι αντιθετικές μέθοδοι μάθησης στοχεύουν στη δημιουργία ενός χώρου αναπαράστασης των δεδομένων μέσω χειρισμού και συντονισμού σχέσεων μεταξύ των δειγμάτων εκπαίδευσης. Η φιλοσοφία πίσω από την αντιθετική μάθηση είναι τα παρόμοια δείγματα να τοποθετούνται πλησιέστερα μεταξύ τους στο χώρο αναπαράστασης ενώ απομακρύνονται μεταξύ τους τα ανόμοια δείγματα. Ο σχηματισμός του χώρου αναπαράστασης επιτυγχάνεται με τη χρήση ***συναρτήσεων Κόστους Αντίθεσης*** (Contrastive Loss). Όμως πως μετριέται η ομοιότητα όταν δεν υπάρχουν ετικέτες; Εδώ είναι που γίνεται κατανοητή

η σημασία των pretext tasks και των ψευδοετικετών. Μέσω αυτών μπορούμε να καθοδηγήσουμε τα δίκτυα ως προς το τι θεωρείται όμοιο και τι ανόμοιο.

Ένα παράδειγμα που βοηθάει στην κατανόηση αυτής της έννοιας είναι το εξής. Φανταστείτε ότι έχουμε ένα βίντεο με ήχο. Έστω ότι χωρίζουμε αυτό το βίντεο σε ίσα διαστήματα και έχουμε σε ξεχωριστά αρχεία των ήχο και την εικόνα. Γνωρίζοντας ότι ο ήχος με το βίντεο είναι συγχρονισμένα μπορούμε να δημιουργήσουμε τις ψευδοετικέτες όμοια/ανόμοια με βάση το συγχρονισμό μεταξύ ενός αποσπάσματος ήχου και ενός αποσπάσματος εικόνας. Αν αυτά τα δύο προέρχονται από το ίδιο διάστημα τότε είναι όμοια, διαφορετικά (αν ειναι ασυγχρόνιστα) είναι ανόμοια. Αυτό απαιτεί ελάχιστη ανθρώπινη παρέμβαση και μπορεί να χρησιμοποιηθεί για εκατομμύρια ώρες βίντεο.

Μπορούμε λοιπόν να εκπαιδεύσουμε ένα δίκτυο με αυτοεπίβλεψη να ταξινομεί τυχαία ζεύγη ήχων-εικόνων σε συγχρονισμένα ή μη-συγχρονισμένα. Για να επιτευχθεί η εκπαίδευση χρησιμοποιείται κάποια αντιθετική συνάρτηση κόστους με μορφή παρόμοια με την παρακάτω (Hadsell et al., 2006)

$$\mathscr{L}(W) = \sum_{i=1}^{P} \mathscr{L}(W, (y, X_1, X_2))^i \tag{0.2.19}$$

$$\mathscr{L}(W, (y, X_1, X_2)^i = (1-y)L_p(D_w^i) + yL_n(D_w^i) \tag{0.2.20}$$

όπου L_p είναι το μερικό σφάλμα μεταξύ των θετικών ζευγαριών (συγχρονισμένες είσοδοι), ενώ L_n είναι το σφάλμα μεταξύ των αρνητικών ζευγών (μη-συγχρονισμένες είσοδοι). Η ακριβής μορφή που πρωτοπροτάθηκε από τους Hadsell et al. (2006) ήταν

$$\mathscr{L}(W, (y, X_1, X_2)^i = (1-y)\frac{1}{2}D_w^i + (y)\frac{1}{2}\{\max(0, m - D_w^i)\}^2 \tag{0.2.21}$$

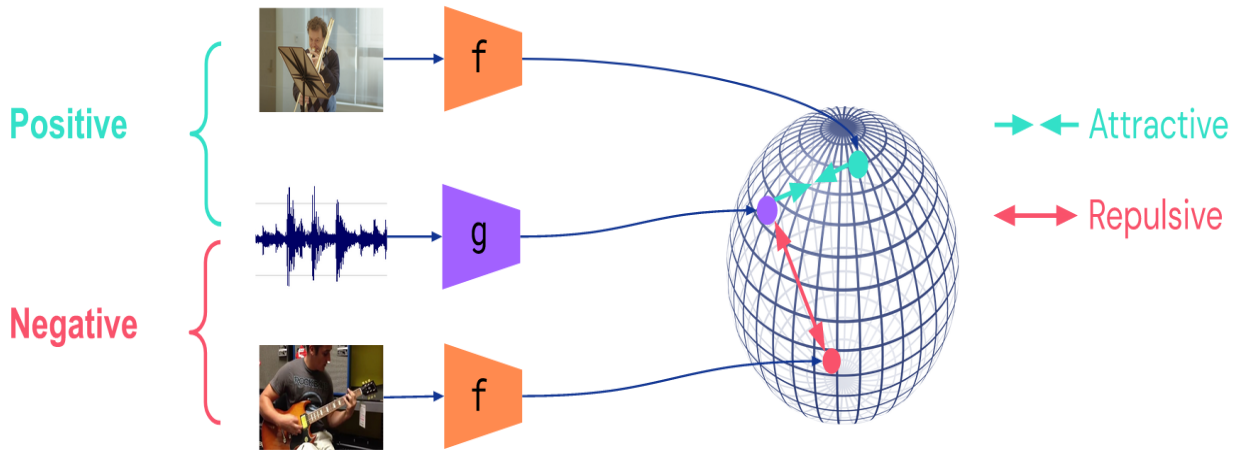


Figure 0.2.2: Οπτική απεικόνιση σχηματισμού ενός χώρου αναπαράστασης μέσω αντιθετικής μάθησης. Πηγή

Το μοντέλο εκπαίδευσης που προτείνουμε στην εργασία αυτή ανήκει στην κατηγορία της αντιθετικής μάθησης αλλά δεν απαιτεί τη χρήση αρνητικών ζευγών.

Μέθοδοι Ενδοπρόβλεψης

Οι μέθοδοι ενδοπρόβλεψης (self-prediction) είναι η δεύτερη μεγάλη κατηγορία. Αυτές οι μέθοδοι έχουν ως σκοπό να προβλεφθεί ένα μέρος της εισόδου από ένα άλλο ή κάποια σημαντική ιδιότητα της εισόδου. Για παράδειγμα, το δίκτυο μπορεί να εκπαιδευτεί να προβλέπει τις τιμές των pixel που έχουν

επίτηδες μηδενιστεί σε μία εικόνα με βάση τα διαθέσιμα pixel. Ένα τέτοιο έργο βοηθάει το δίκτυο να μάθει συσχετίσεις μεταξύ των pixel και να αποκτήσει κάποια γενική γνώση για τις εικόνες που βλέπει. Μία άλλη συχνή τακτική είναι να τροποποιείται η εικόνα εισόδου και το δίκτυο να πρέπει να προβλέψει σωστά την τροποποίηση που έγινε. Για παράδειγμα, πριν δωθεί στο δίκτυο ως είσοδος, μια εικόνα περιστρέφεται κατά 0, 90, 180 ή 275 μοίρες. Το δίκτυο θα πρέπει να προβλέψει ποια περιστροφή έχει συμβεί. Αν μάθει να υλοποιεί αυτό το έργο τότε θα έχει εγγενώς κατανοήσει σημαντικά χαρακτηριστικά των αντικειμένων που βρίσκονται στις εικόνες.

Η αυτοεπιβλεπόμενη μάθηση επιτρέπει την εκπαίδευση των δικτύων σε υπέρογκα ποσά δεδομένων χωρίς να απαιτείται έντονη ανθρώπινη παρέμβαση, παρά μόνο ο σχεδιασμός των pretext-tasks. Έχει φανεί ότι τα αυτοεπιβλεπόμενα δίκτυα είναι χρήσιμα σε εφαρμογές που τα ετικετοποιημένα δεδομένα είναι σπάνια ή ακριβά (π.χ. ιατρικές εφαρμογές) (Ciga et al., 2021), (Krishnan et al., 2022). Έχει ακόμα δειχτεί ότι τα αυτοεπιβλεπόμενα δίκτυα είναι πιο ανεκτικά σε θόρυβο, λάθοι στις ετικέτες και επειδυκνύουν μεγαλύτερη δικαιότητα σε σχέση με τα δίκτυα επιβλεπόμενης μάθησης (Hendrycks et al., 2019),(Goyal et al., 2022).

0.3 Το σύνολο δεδομένων CMU-MOSEI

Το σύνολο δεδομένων CMU Multimodal Opinion Sentiment and Emotion Intensity (CMU-MOSEI) (Zadeh et al., 2018b) είναι ένα δημοσίως διαθέσιμο πολυτροπικό σύνολο δεδομένων που έχει γίνει σημείο αναφοράς στην αναγνώριση πολλαπλών συναισθημάτων. Το CMU-MOSEI αποτελείται από τρεις βασικούς τύπους δεδομένων (***τροπικότητες***): βίντεο, ήχο και κείμενο. Όλα τα δείγματα έχουν ετικέτες με σχολιασμούς συναισθημάτων και έντασης αισθημάτων.

Το σύνολο δεδομένων έχει σχεδιαστεί για να συλλαμβάνει ένα ευρύ φάσμα αισθημάτων και συναισθημάτων που εκφράζονται σε φυσικές συνθήκες. Περιλαμβάνει 23.500 YouTube βίντεο από 1000 διαφορετικούς εκφωνητές , συνολικής διάρκειας άνω των 65 ωρών, καλύπτοντας περισσότερα από 250 θέματα, όπως κριτικές ταινιών, ομιλίες TED και vlog. Αυτό το ποικίλο σύνολο ομιλητών, θεμάτων, σχολιασμών και δειγμάτων επιτρέπει γενικεύσιμες μελέτες για την πολυτροπική γλώσσα των ανθρώπων.

Τα συναισθήματα (emotion) ορίζονται ως η έκφραση της κατάστασης του νου του ομιλητή κατά την εκφορά της πρότασης. Το αίσθημα (sentiment) ορίζεται ως η στάση του ομιλητή απέναντι στο θέμα της συζήτησής του. Ζητήθηκε από τους σχολιαστές να τοποθετήσουν το αίσθημα των ομιλητών σε μια κλίμακα Likert επτά βημάτων [-3: εξαιρετικά αρνητικό, -2: αρνητικό, -1: ασθενώς αρνητικό, 0: ουδέτερο, 1: ασθενώς θετικό, 2: θετικό, 3: πολύ θετικό].

Τα συναισθήματα που επιλέχθηκαν είναι τα έξι βασικά συναισθήματα του Ekman (Ekman et al., 1992) της ευτυχίας, της λύπης, του θυμού, του φόβου, της αηδίας και της έκπληξης. Κάθε ένα από τα συναισθήματα σχολιάζεται σε μια κλίμακα Likert τεσσάρων βημάτων για την ένταση ενός συναισθήματος x : [0: καμία ένδειξη x, 1: ασθενώς x, 2: x, 3: πολύ x].

Το σύνολο δεδομένων μπορεί να προσπελαστεί με πολλές μορφές. Το σύνολο δεδομένων μπορεί να χρησιμοποιηθεί στην ακατέργαστη μορφή του, π.χ. βίντεο, κείμενο και ηχητικά κύματα. Ωστόσο, για τη διευκόλυνση των ερευνητών, το σύνολο δεδομένων παρέχεται και με προεξαγόμενα χαρακτηριστικά (embeddings) αντί για ακατέργαστα δεδομένα.

Τα δεδομένα είναι σε μορφή ακολουθιών. Κατά τη διάρκεια των πειραμάτων μας χρησιμοποιήσαμε την ήδη ευθυγραμμισμένη έκδοση του συνόλου δεδομένων σε μορφή προεξαγόμενων χαρακτηριστικών. Συγκεκριμένα χρησιμοποιήθηκαν τα embeddings Glove για το κείμενο, FACET για την εικόνα και COVAREP για τον ήχο.

0.4 Η πρόταση μας

Η πρόταση μας χρησιμοποιεί μία μέθοδο Αντιθετικής Μάθησης εμπνευσμένη από το έργο των Zbontar et al. (2021) που υλοποίησαν μία αντιθετική συνάρτηση κόστους βασισμένη στον πίνακα ετεροσυσχέτισης και το εφάρμοσαν στο πλαίσιο Barlow Twins. Εμείς, επεκτείνουμε το άνωθεν έργο προσαρμόζοντας τη μέθοδο για πολυτροπικό περιβάλλον. Η εικόνα 0.4.1 παρουσιάζει σχηματικά την ιδέα μας. Να αναφερθεί ότι για την αυτοεπιβλεπόμενη μάθηση η μέθοδος μας δεν απαιτεί καμία χρήση ετικετών αλλά ούτε και αρνητικά ζεύγη δειγμάτων όπως είδαμε στην εξίσωση 0.2.20.

Η σκέψη πίσω από την μέθοδο μας είναι το δίκτυο να μάθει να εντοπίζει όμοια παραδείγματα μέσω μιας αντιθετικής συνάρτησης κόστους. Μαθαίνοντας αναπαραστάσεις

Η προτεινόμενη μέθοδος περιλαμβάνει 3 μέρη: μια πολυτροπική επαύξηση δεδομένων (MM-AUG), μια πολυτροπική αρχιτεκτονική κωδικοποιητή E και ένα δίκτυο προβολέων (Projector). Η συνάρτηση κόστους $\mathscr{L}_{BT}$ του μοντέλου Barlow Twins εφαρμόζεται στην έξοδο του προβολέα, στο χώρο των embeddings. Η ιδέα πίσω από την μέθοδο είναι ότι η μονάδα επαύξησης δημιουργεί δύο διαφορετικές αλλά όμοιες εκδοχές της πολυτροπικής εισόδου. Οι κωδικοποιητές στη συνέχεια κωδικοποιούν τις πολυτροπικές πληροφορίες, οι οποίες στη συνέχεια προβάλλονται σε έναν κοινό χώρο (joint embedding space). Πάνω σε αυτόν τον χώρο εφαρμόζεται η συνάρτηση κόστους, και μέσω της οπισθοδιάδοσης οι κωδικοποιητές E ανανεώνουν τις παραμέτρους τους και μαθαίνουν πολύτιμες αναπαραστάσεις γενικής φύσεως, δίνοντάς τους τη δυνατότητα να αντιμετωπίσουν αποτελεσματικά στη συνέχεια το τελικό έργο (π.χ. αναγνώριση συναισθημάτων).

Η μελέτη μας επικεντρώθηκε στην επεξεργασία τριών τύπων πληροφορίας, της οπτικής τροπικότητας, τροπικότητας κειμένου και τροπικότητας ήχου. Ωστόσο, η επεξεργασία περισσοτέρων ή λιγότερων τροπικοτήτων είναι εξίσου εύκολη να επιτευχθεί.

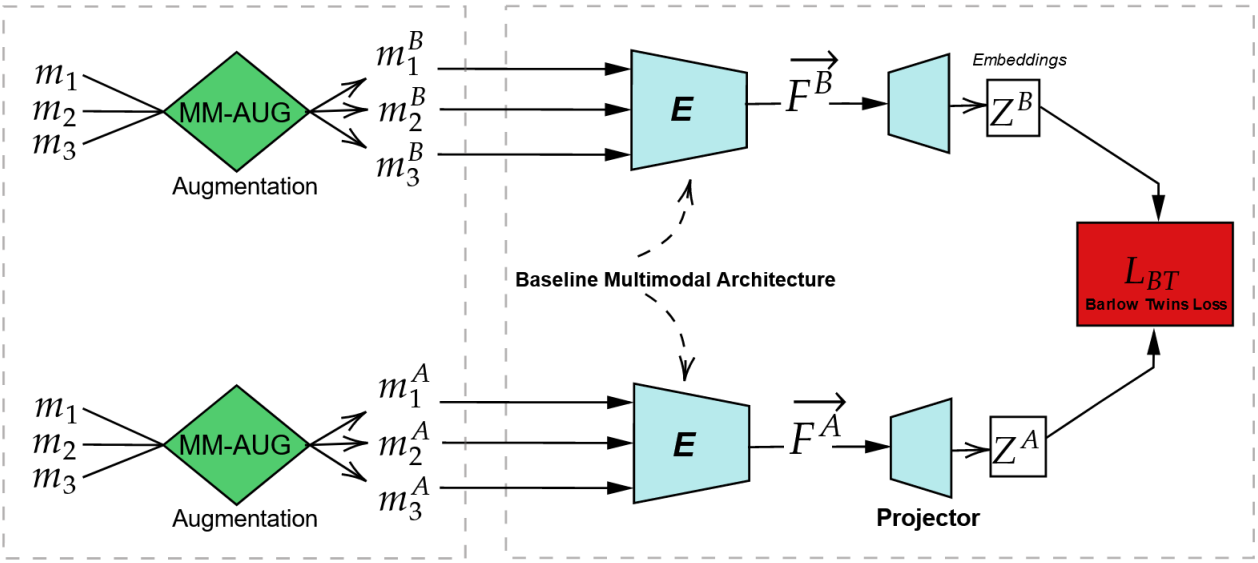


Figure 0.4.1: Προτεινόμενη Αρχιτεκτονική. Μια μέθοδος επαύξησης χαρακτηριστικών δημιουργεί δύο νέες εκδοχές A, B για κάθε είσοδο m_i. Κάθε τριπλέτα m_1, m_2, m_3 επεξεργάζεται από την βασική πολυτροπική αρχιτεκτονική και εξάγει τα συγχωνευμένα χαρακτηριστικά $\vec{F}$ τα οποία προβάλλονται σε έναν ενιαίο χώρο αναπαράστασης. Τέλος, η αντιθετική συνάρτηση κόστους Barlow Twins $\mathscr{L}_{BT}$ εφαρμόζεται στον χώρο αναπαράστασης.

Η αρχική είσοδος έρχεται με τη μορφή τριών διανυσμάτων, ένα για κάθε τροπικότητα m_i. Η μονάδα επαύξησης είναι υπεύθυνη για την αλλοίση της αρχικής πολυτροπικής εισόδου με τρόπο που να δημιουργούνται δύο διαφορετικές εκδοχές του ίδιου αντικειμένου. Σε κάθε είσοδο m_i εφαρμόζονται δύο μετασχηματισμοί T^A, T^B δημιουργώντας δύο εξόδους m_i^A, m_i^B. Αυτές είναι ελαφρώς ή έντονα

τροποποιημένες εκδοχές της m_i ανάλογα με τον μετασχηματισμό που τους επιβλήθηκε.

Οι τεχνικές επαύξησης χαρακτηριστικών εφαρμόζονται χωριστά σε κάθε διαφορετική τροπικότητα. Η μονάδα επάυξησης δέχεται M εισόδους και παράγει $2 * M$ εξόδους, ένα ζεύγος (m_i^A, m_i^B) για κάθε διάνυσμα τροπικότητας. Οι τριπλέτες παρέχονται σε batchses μεγέθους B, επομένως οι συνολικές έξοδοι είναι $2 * B$ τριπλέτες.

Η μελέτη μας, εστιάζει σε επαυξήσεις που είναι κατάλληλες για εισροές που έρχονται με τη μορφή προεξαγμένων χαρακτηριστικών (embeddings). Συγκεκριμένα, πειραματιστήκαμε με τρεις τεχνικές επαύξησης, Γκαουσιανό θόρυβο, masking, και SeqAug (Georgiou and Potamianos, 2023) μια νέα μέθοδο επαύξησης χαρακτηριστικών κατάλληλη για πολυτροπικά δίκτυα.

Συνάρτηση Κόστους Barlow Twins $\mathscr{L}_{BT}$

Η συνάρτηση απώλειας είναι αναπόσπαστο μέρος της μεθόδου. Ο υπολογισμός της συνάρτησης απώλειας απαιτεί τον υπολογισμό του πίνακα ετεροσυσχέτισης μεταξύ των πινάκων εξόδου των προβολέων. Η $\mathscr{L}_{BT}$ εφαρμόζεται στον πίνακα ετεροσυσχέτισης και δεν ορίζεται για μεμονωμένες εξόδους. Η λειτουργία της συνάρτησης κόστους στοχεύει στην εκμάθηση ενός ενιαίου χώρου αναπαραστάσεων και για τις 3 τροπικότητες. Μαθηματικά, στοχεύουμε στην ελαχιστοποίηση της συνάρτησης

$$\mathscr{L}_{BT} = \sum_{i=1}^{N}(1 - C_{ii})^2 + \lambda \sum_{i=1}^{N}\sum_{j\neq 1}^{N}(C_{ij})^2 \tag{0.4.1}$$

όπου C είναι ο πίνακας ετεροσυσχέτισης και τα στοιχεία του υπολογίζονται ως

$$C_{ij} = \frac{\sum_{b=1}^{N} Z_{bi}^A Z_{bj}^B}{\sqrt{\sum_{b=1}^{N}(Z_{bi}^A)^2}\sqrt{\sum_{b=1}^{N}(Z_{bi}^B)^2}} \tag{0.4.2}$$

όπου το N υπονοεί το batch size και Z^A, Z^B είναι οι πίνακες των αναπαραστάσεων κοινού χώρου, δηλαδή οι προβολές των συγχωνευμένων χαρακτηριστικών $\vec{F}$, ορισμένες ως $\vec{Z_b^K}$.

Η συνάρτηση απώλειας $\mathscr{L}_{BT}$ τιμωρεί το δίκτυο ανάλογα με την τετραγωνική διαφορά μεταξύ του πίνακα ετερουσχέτισης **C** και του ταυτοτικού πίνακα **I**. **Εξυπηρετεί δύο στόχους:**

1. Τραβάει τις αναπαραστάσεις των δύο όψεων κοντά μεταξύ τους στον κοινό χώρο. Ο πρώτος όρος της εξίσωσης 0.4.1 ονομάζεται *invariance όρος* και βοηθά το δίκτυο να μάθει αμετάβλητα χαρακτηριστικά μετασχηματισμού. Το C_{ii} υποδηλώνει τα διαγώνια στοιχεία του πίνακα **C** τα οποία ωθούνται να εξισωθούν με 1. Με άλλα λόγια, το δίκτυο πρέπει να κατανοήσει ότι οι δύο επαυξημένες εκδοχές προέρχονται από το ίδιο αντικείμενο, επομένως πρέπει να τοποθετηθούν κοντά μεταξύ τους στο χώρο αναπαράστασης.
2. Αναγκάζει το δίκτυο να απορρίψει τις περιττές πληροφορίες που είναι ήδη κωδικοποιημένες στα βάρη του. Ο δεύτερος όρος, είναι ο *redundancy* όρος. Τιμωρεί τα μη διαγώνια στοιχεία C_{ij} με την τετραγωνική τους τιμή, αναγκάζοντάς τα να μηδενιστούν. Το λ είναι μια παράμετρος που ελέγχει την επίδραση αυτού του όρου στη συνολική απώλεια. Με άλλα λόγια, ο δεύτερος όρος βοηθά στον αποσυσχετισμό των διαστάσεων των embedding, μειώνοντας έτσι τις περιττές πληροφορίες που κωδικοποιούνται.

0.5 Αποτελέσματα

Για να αξιολογήσουμε τα πλεονεκτήματα που προσφέρει η πολυτροπική αυτοεποπτευόμενη αρχιτεκτονική μας, απαιτείται η ύπαρξη ενός σωστού και δίκαιου μέτρου σύγκρισης. Για αυτό εκτελούμε μια

σειρά πειραμάτων εποπτευόμενης εκπαίδευσης στην βασική (baseline) αρχιτεκτονική (κωδικοποιητής E). Να τονιστεί ότι κάθε πείραμα διεξάγεται τουλάχιστον 5 φορές και τα τελικά αποτελέσματα βγαίνουν ως ο μέσος όρος των 5 επαναλήψεων. Έτσι, μειώνουμε την επίδραση της στοχαστικότητας που χαρακτηρίζει την εκπαίδευση των νευρωνικών δικτύων και αποκτούμε μεγαλύτερη εμπιστοσύνη στα αποτελέσματα μας.

Κατά τη διάρκεια της εκπαίδευσης του βασικού πολυτροπικού μοντέλου, διατηρούμε τις παραμέτρους της αρχικής εργασίας του μοντέλου Georgiou et al. (2021), με μικρές τροποποιήσεις όπως εξηγούνται αναλυτικά στην ενότητα 7.3. Η πιο ουσιώδης διαφορά είναι πως αντί να χρησιμοποιούμε δικατευθυντήρια LSTM δίκτυα χρησιμοποιούμε το κλασσικό μονοκατευθυντήριο μοντέλο, καθώς μας απέδωσε καλύτερη επίδοση.

Αποτελέσματα Επίδοσης

Τα αποτελέσματα μας δείχνουν ότι η μέθοδος μας επιτυγχάνει σαφώς αυξημένη επίδοση σε σχέση με την εποπτευόμενη μάθηση. Όταν προεκπαιδεύουμε τα μοντέλα μας με αυτοεπιβλεπόμενο τρόπο και στην συνέχεια εφαρμόζουμε επιβλεπόμενη μάθηση με ετικέτες τότε επιτυγχάνεται βελτίωση σε όλες τις μετρικές αξιολόγησης $F1$, $Acc2$, $Acc5$, $Acc7$, Cor, MAE.

Συγκεκριμένα, παρατηρούμε ότι ο μετασχηματισμός SeqAug Georgiou and Potamianos (2023) παρέχει κατά μέσο όρο έως και 1,3% απόλυτη αύξηση στην μετρική $F1$. Η δυαδική ακρίβεια $Acc2$ επίσης αυξάνεται έως και 1,2% κατά απόλυτη τιμή. Τα κέρδη είναι επίσης υψηλά για την ακρίβεια, με τον μετασχηματισμό Masking να κυριαρχεί, παρέχοντας έως και 0,37% αύξηση για το $Acc7$ και αύξηση 0,64% για το $Acc5$ (ίδιο με το SeqAug). Οι βελτιώσεις είναι επίσης αξιοσημείωτες για τη συσχέτιση Pearson (Cor) και το MAE, με το masking να κυριαρχεί στις μετρήσεις ακολουθούμενο από το SeqAug. Τέλος, ο Γκαουσιανός θόρυβος ξεπερνά σημαντικά το εποπτευόμενο μοντέλο στα $F1$, $Acc2$ και Cor αλλά πετυχαίνοντας ελαφρώς μικρότερη επίδοση στις υπόλοιπες μετρικές. Ο πίνακας 0.5.1 παρουσιάζει την μέση απόδοση κάθε τεχνικής.

Table 0.5.1: Επιβλεπόμενα και αυτοεπιβλεπόμενα μοντέλα εκπαιδευμένα στο πλήρες επιβλεπόμενο σύνολο δεδομένων CMU-MOSEI. **Μέση απόδοση 5 επαναλήψεων.**

Mode	Augmentations	F1(%)	Acc2(%)	Acc5(%)	Acc7(%)	Corr	MAE
Supervised	-	81.48 ±0.55	81.31	52.58	51.41 ±0.35	0.688	0.5952
Self-Supervised	Gaussian Noise	82.12 ±0.12	81.55	52.37	51.28 ±0.75	0.696	0.5969
Self-Supervised	Masking	82.32 ±0.31	82.22	**53.22**	**51.78** ±0.22	**0.702**	**0.5887**
Self-Supervised	SeqAug	**82.77** ±0.13	**82.48**	**53.22**	51.69 ±0.18	0.696	0.5909

Ο Πίνακας 0.5.2 εστιάζει στα μοντέλα με τις καλύτερες επιδόσεις στις 5 επαναλήψεις. Προκύπτει μία παρόμοια εικόνα. Τα προεκπαιδευμένα μοντέλα αποδίδουν σταθερά καλύτερα σε όλες τις μετρήσεις. Η βελτίωση για το $F1$ είναι περίπου 0,9% με το SeqAug ενώ το Masking επιτυγχάνει έως και 0,47% αύξηση απόλυτης τιμής για το $Acc7$. Τα καλύτερα μοντέλα μας επιτυγχάνουν απόδοση συγκρίσιμη με προηγούμενα μοντέλα SOTA όπως το MuLT Tsai et al. (2019), M3 Georgiou et al. (2021), SeqAug Georgiou and Potamianos (2023) και MMLatch Paraskevopoulos et al. (2022).

Ένα άλλο θετικό υποπροϊόν είναι η μειωμένη διακύμανση στην απόδοση μεταξύ των μετρήσεων. Μοντέλα που έχουν προεκπαιδευτεί με αυτοεπίβλεψη έχουν σημαντικά πιο σταθερές επιδόσεις σε σύγκριση με τα επιβλεπόμενα μοντέλα. Αυτό σημαίνει ότι η μέθοδος μας παρέχει μια πιο αξιόπιστη μεθοδολογία εκπαίδευσης.

Table 0.5.2: Επιβλεπόμενα και αυτοεπιβλεπόμενα μοντέλα εκπαιδευμένα στο πλήρες επιβλεπόμενο σύνολο δεδομένων CMU-MOSEI. **Τα καλύτερα μοντέλα των 5 επαναλήψεων.**

Mode	Augmentations	F1(%)	Acc2(%)	Acc5(%)	Acc7(%)	Corr	MAE
Supervised	-	81.91	81.83	52.87	51.76	0.691	0.5902
Self-Supervised	Gaussian Noise	82.2	81.8	52.82	51.78	0.696	0.5969
Self-Supervised	Masking	82.75	82.48	**53.41**	**52.23**	**0.700**	**0.5886**
Self-Supervised	SeqAug	**82.83**	**82.62**	53.33	51.74	0.699	0.5916

Εκπαίδευση με Μειωμένα Δεδομένα

Το πιο πολλά υποσχόμενο πλεονέκτημα της αυτοεπιβλεπόμενης μάθησης είναι η ικανότητά της να μαθαίνει αναπαραστάσεις μέσα από μεγάλα σύνολα δεδομένων χωρίς επίβλεψη, μειώνοντας έτσι σημαντικά την ανάγκη για ετικέτες. Για να αξιολογήσουμε την ικανότητα της μεθόδου μας να μαθαίνει χρήσιμες αναπαραστάσεις από μη εποπτευόμενα σύνολα δεδομένων και περιορισμένες ποσότητες ετικετοποιημένων δεδομένων πραγματοποιήσαμε μια σειρά πειραμάτων με εκδόσεις του CMU-MOSEI με μειωμένο μέγεθος. Αρχικά εκπαιδεύσαμε το δίκτυό μας με καθαρά επιβλεπόμενο τρόπο σε όλες τις μειωμένες εκδόσεις του αρχικού συνόλου δεδομένων. Στη συνέχεια, εκπαιδεύσαμε το δίκτυο χρησιμοποιώντας αυτοεπίβλεψη στα ίδια αντίστοιχα σύνολα δεδομένων.

Τα αποτελέσματα που λάβαμε είναι ενθαρρυντικά και δείχνουν ότι η αυτοεπίβλεψη μπορεί να βοηθήσει τα μοντέλα να επιτύχουν καλύτερη επίδοση όταν το σύνολο των ετικετών είναι μικρό. Συγκεκριμένα, φάνηκε ότι τα αυτοεπιβλεπόμενα μοντέλα μπορούν να επιτύχουν ίση ή καλύτερη επίδοση χρησιμοποιώντας 20%-40% λιγότερα δεδομένα από ότι τα επιβλεπόμενα μοντέλα. Η εικόνα 0.5.1 προσφέρει περισσότερες λεπτομέρειες για το συγκεκριμένο πείραμα.

0.6 Συμπεράσματα και Μελλοντικές Κατευθύνσεις

Τα συμπεράσματα που πηγάζουν από την παρούσα διατριβή είναι ενθαρρυντικά ως προς τον ρόλο της αυτοεπιβλεπόμενης μάθησης σε πολυτροπικά δίκτυα εκπαιδευμένα σε δεδομένα μικρού έως μεσαίου μεγέθους. Γίνεται σαφές ότι η προεκπαίδευση με αυτοεπίβλεψη βελτιώνει την επίδοση των νευρωνικών δικτύων στο τελικό τους έργο. Παράλληλα, φάνηκε ότι η αυτοεπίβλεψη παρέχει μία καλή λύση όταν το σύνολο των ετικετών είναι πολύ μικρό, αφού μέσω της αυτοεπίβλεψης μπορεί να επιτευχθεί καλύτερη απόδοση με 20% έως 40% λιγότερα δεδομένα.

Όμως, παρατηρήθηκε ότι η προσαρμογή των παραμέτρων είναι ένα αρκετά δύσκολο έργο στα πολυτροπικά μοντέλα ιδίως όταν χρησιμοποιείται αυτοεπίβλεψη. Φάνηκε ότι η μέθοδος είναι ευαίσθητη στην επιλογή των παραμέτρων μετασχηματισμού που εφαρμόζεται στην είσοδο. Μικρές αλλαγές μπορούν να έχουν μεγάλες αρνητικές επιπτώσεις στην επίδοση, οπότε απαιτείται πολύ προσεχτική επιλογή του κατάλληλου μετασχηματισμού. Η εργασία μας όμως δείχνει ότι τόσο οι μετασχηματισμοί SeqAug, Masking όσο και ο συνδυασμός *SeqAug + Θόρυβος + Masking* αποτελούν μία καλή επιλογή μετασχηματισμού για υψηλή επίδοση.

Αναγνωρίζοντας ότι κάθε μικρή ανακάλυψη και διορατικότητα, κάθε σταδιακό βήμα συμβάλλει στη θεμελίωση της γνώσης που ωθεί τον τομέα μας εμπρός νιώθουμε υπερήφανοι που συνεισφέρουμε, έστω και σε μικρό βαθμό, σε αυτό το συναρπαστικό πεδίο.

Στις σελίδες που ακολουθούν ο αναγνώστης μπορεί να βρει περισσότερες λεπτομέρειες, στην Αγγλική γλώσσα, τόσο για τα υποπεδία που εξετάζονται όσο και για την προτεινόμενη μέθοδο.

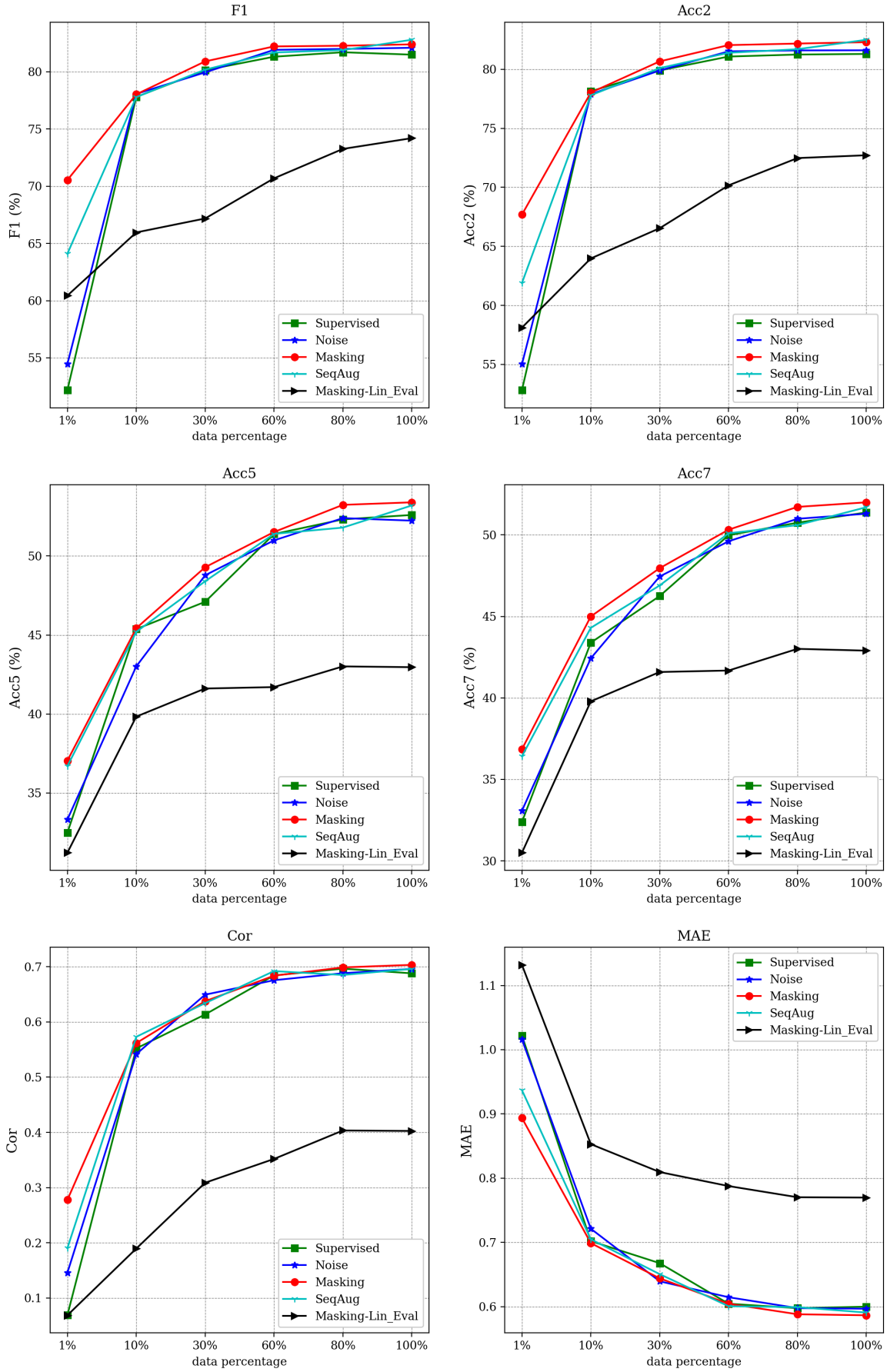


Figure 0.5.1: Απόδοση των επιβλεπόμενων και των αυτοεπιβλεπόμενων μοντέλων όταν εκπαιδεύονται σε μειωμένες εκδόσεις του αρχικού συνόλου δεδομένων. Οι μετρήσεις αντιπροσωπεύουν τα **καλύτερα μοντέλα από 3 επαναλήψεις**. Η μαύρη γραμμή αντιπροσωπεύει ένα μοντέλο που έχει εκπαιδευτεί σε λειτουργία SSL αλλά οι ετικέτες χρησιμοποιήθηκαν μόνο για ανανέωση των παραμέτρων του τελικού ταξινομητή στην κορυφή του δικτύου.

Chapter 1

Introduction

1.1 Background and Motivation

Over the last few years, we've borne witness to a spectacular revolution in the field of artificial intelligence (AI). This isn't merely academic; this transformation is rapidly embedding itself into every facet of our day-to-day lives. From recognizing faces in our photos to understanding and generating human-like text and digital art, AI is no longer a futuristic concept but an integral part of our present.

At the heart of this dramatic shift are deep learning algorithms. They are the silent engines driving AI forward, continually pushing the boundaries of what we thought was possible. They are becoming more sophisticated, more potent, and more intuitive. Today, they underpin a vast range of applications, from autonomous driving and coding assistants to artistic creation and beyond.

But if there's one area in deep learning that has genuinely captivated me, it's self-supervised learning. It is, in my opinion, one of the most exciting developments in AI research. What sets self-supervised learning apart is its ability to learn from unlabelled data and complete tasks it was never explicitly trained to. This is a game-changer! It means we're no longer tied down by the need for vast amounts of human-annotated data. Instead, we can now train models on vast reservoirs of unlabelled data – think all of the available internet text, the billions of Instagram images, or the entirety of GitHub repositories. It's an astonishing prospect!

And we're only just scratching the surface. We're now seeing the rise of large-scale language models, often referred to as 'LLMs.' These behemoths can understand and generate human-like text, making them incredibly powerful tools. The progress in this area is truly awe-inspiring. But more importantly, these models will soon be able to interact and understand other forms of data such as images or sounds, in other words, they are turning multimodal. This will set the tone for a new era beyond our imagination!

In essence, the world of AI and deep learning is undergoing a seismic shift. The progress we're making is nothing short of extraordinary. This thesis explores the conjecture of two of my favorite deep learning fields, two fields that when combined, provide the first glimpses of what artificial general intelligence, AGI, may look like.

This thesis will explore the fascinating field of **self-supervised multimodal learning**, the field that studies how self-supervision can be employed in the teaching of multimodal models.

1.2 Problem Statement and Study Objectives

While the power and potential of self-supervised learning are evident, its application within multimodal settings, particularly for emotion recognition, presents a novel and challenging problem space.

Can we train our models to intelligently learn from multiple unlabeled sources of data and detect human emotions, when the amount of data is not in the scale of millions or billions? This is the question that fuels our inquiry.

This thesis will examine four key research questions in this context.

- Can the methodologies developed for unimodal self-supervised learning be effectively adapted to a multimodal setting with small-medium sized datasets?
- How does the performance of self-supervised learning methodologies compare to traditional supervised techniques in the context of multimodal emotion recognition with small-medium sized datasets?
- Can self-supervised learning provide an adequate solution when labeled data scarcity becomes an issue?
- Where does self-supervised learning fall short in practice, and under what circumstances does it prove worthy of implementation for multimodal emotion recognition?

By assessing the effectiveness of self-supervision in smaller datasets, we venture into largely uncharted territory. The prevailing notion is that self-supervised learning thrives on vast amounts of data, in the scale of the internet. By challenging this assumption, we strive to unlock the potential of self-supervision in scenarios where data is scarce. This can democratize the benefits of self-supervision, making it accessible to data-constrained applications.

By contrasting self-supervised learning with traditional supervised techniques, we aim to gain a more nuanced understanding of the pros and cons of each approach in the context of multimodal emotion recognition.

By identifying the limitations of self-supervised learning, we contribute to a more realistic and grounded understanding of this promising paradigm. We shed light on the scenarios where self-supervision shines and those where it falls short, promoting a balanced and informed adoption of this methodology.

We hope that this knowledge could act as a guide to practitioners, helping them choose the appropriate methodology for their specific applications, and optimizing for accuracy, efficiency, and practicality.

1.3 Thesis Structure

The structure of the Thesis first aims to provide the reader with all the necessary background knowledge. Afterward, we introduce the reader to our niche field, describe our proposal and present our experimental results.

In greater detail, chapters 2,3,4 provide the background knowledge. Chapter 2 begins with an introduction to the general field of machine learning and then sprouts into deep learning and neural networks. Afterwards a detailed explanation of recurrent neural networks, the self-attention mechanism and transformer networks is provided since we later give a lot of references to them. Chapter 3 focuses on self-supervised learning, a key concept in this study. This chapter aims to introduce the reader into this new and vast field by providing multiple examples and explaining important mathematical concepts. We separated the chapter into 'Early Work' that provided the foundational ideas, 'Self-prediction methods' and 'Contrastive Learning Methods', a taxonomy used by many highly respected researchers. Moving on, chapter 4 discusses multimodal learning. First, we provide an overview of the challenges multimodal learning aims to solve and delve deeper into one of them - 'Multimodal Representation'. Afterwards, important prior work on the field is discussed and we also provide an introduction to the field of Self-Supervised Multimodal Learning, first providing a brief explanation and then important

research on the field. Finally, we present important research datasets for multimodal learning, giving the reader a better overview and understanding of the capabilities of this field.

Chapter 5 delves deeper into the core subject of this study. In chapter 5 we discuss specific applications of multimodal learning in emotion recognition. First, The dataset we use for our study is thoroughly discussed. Then all of the most important prior works on this dataset are discussed and explained in detail before providing a comprehensive overview of the state-of-the-art.

In chapter 6 we formally present our own proposal that aims to solve the same task presented in chapter 5. Our approach is presented in a hierarchical manner by first explaining the general framework on a higher level and then diving into lower-level details and mathematical explanations. The loss function, the architecture and augmentation methods are thoroughly elaborated upon. Finally, some possible variations of our work are presented that we aim at testing in the future.

Chapter 7 provides all the technical details required for understanding the proposed methodology on a deeper level and reproducing it. First, we lay out the setting and vocabulary, that we will use in the 'Results', chapter by elaborating on our 'Training Methodologies'. Then, specific technical details are provided for understanding a key ingredient in our proposal - augmentations. Finally, we provide an overview of all the parameters, hyperparameters, software, and hardware configurations necessary to effectively reproduce this study.

Chapter 8 is highly technical and quantitative. Here we present all of the experimental results and data gathered throughout all of our technical experiments. First, a set of baselines is provided and the comparison between the supervised and self-supervised methodology is made. Second, we present our discoveries regarding the combination of multiple transformations during the self-supervised training phase. Afterward, we provide our discoveries and answer one of the key questions of this study 'How data scarcity affects performance'. Finally, some ablation and hyperparameter tuning studies are presented along with their conclusions. This chapter is accompanied by numerous quantitative tables and graphs.

Finally, in chapter 9 we lay out the conclusions made out of this study. We try to give answers to the questions initially posed by our research but also present unexpected discoveries made along the way. The possible future steps, that we or other interested parties, could follow to advance this work are also discussed.

In the end, an appendix section holds extra material and results that we include for the sake of completeness and for those interested. Since including those in the main sections would harm the flow of reading we decided to create this section for storing such extra material.

1.4 Summary

This study explores the ability of self-supervised deep learning methods in recognizing human emotions by analyzing multiple data sources such as images, text and audio. Our aim is to improve and blend existing methodologies in an innovative way and contribute to the advancement of this exciting field.

We hope that our study will act both as an introduction and a guide to the fascinating fields of deep learning, self-supervision, and multimodal learning. We expect that our study provides a comprehensive analysis of prior work and a detailed presentation of our proposed methodology. We provide answers to some open questions in the field of self-supervised multimodal learning but also pose new questions along the way.

After studying this research one will understand how and why self-supervised learning can enhance deep learning models and help them provide more accurate predictions when detecting human emotions. The reader will be able to answer how data scarcity affects both supervised and self-supervised

methods and to what extent. Moreover, the readers should be able to reproduce this study and have a set of guidelines in their hands if they chose to embark on similar research.

Chapter 2

Deep Learning

> "Machines of this character can behave in a very complicated manner when the number of units is large."
>
> Alan Turing (1948) "Intelligent Machines", page 6

2.1 Introduction

What is learning? How can a machine learn? How do we quantify the resources needed to learn a given concept? Is learning always possible? Can we know if the learning process succeeded or failed? These are some of the questions the field of Machine Learning (ML) tries to answer.

Machines learn by analyzing data, learning (un)known probability distributions and creating boundary plains in very high dimensional spaces. It's actually not the machines that learn, but rather human-designed algorithms that manage to model the world around them through vast amounts of data. Statistics, machine learning, big data, data science, deep learning. These are all terms often used to describe the science of modeling the world through data.

This study focuses on a subfield of machine learning, deep learning. This chapter serves as a brief introduction to this fascinating field. First, we explain some foundational concepts related to machine learning. Then we explore the field of deep learning in particular, presenting the basic building blocks of modern deep learning models, the feedforward linear networks. Then, we dive deeper into more complex structures such as recurrent neural networks. long-short term memory (LSTM) networks, and the more recent Transformer networks. After studying this chapter, the reader should be equipped to continue into the next more advanced chapters of this study.

2.2 Supervised and Unsupervised Learning

Supervised learning and unsupervised learning are the two primary methods used in machine learning. These are two strategies humans have employed in order to help algorithms learn from data. Learning is however a very wide domain. Consequently, the field of machine learning has branched into various learning paradigms such as reinforcement learning (Sutton and Barto, 2018) and self-supervised learning, a hybrid method between supervised and unsupervised learning. Self supervised learning

will be studied in greater detail in the next chapter, however, reinforcement learning is not relevant to this study and shall not be discussed. Here we explore the two most prominent methods of designing machine learning algorithms, supervised and unsupervised learning.

2.2.1 Supervised Learning

Supervised learning is a process in which a machine learning model learns to make predictions or decisions based on a labeled dataset. A ***dataset*** is a collection of data samples, each of which has several attributes either numeric or categorical, usually referred to as ***features***. The term '***supervised***' refers to the presence of a 'supervisor' in the form of ***labeled*** data, which provides the model with correct answers during the training phase.

As an illustrative example, consider a binary classification problem where we aim to separate emails into "spam" and "not spam". In this case, the labels are "spam" and "not spam". The features the algorithm has access to could be different attributes of the emails such as the subject line, the sender, the length or content of the email.

Mathematically, a supervised learning algorithm seeks to learn a function $y = \mathbf{f}(x;\theta)$ that maps the inputs x (features) to the correct outputs $\tilde{y}$ (labels). This is usually achieved by minimizing a loss function $\mathscr{L}$, which quantifies the discrepancy between the model's predictions $\mathbb{f}(x;\theta)$ and the true labels $\tilde{y}$. Here, θ denotes the parameters of the model, which modern networks enumerate in the billions and trillions.

The loss function $\mathscr{L}$ is a mathematical function that quantifies the error between the model's predictions and the correct answers. The loss function is one of the most important ingredients in deep learning. A very simple example of a loss function, that is however still relevant, is the mean squared error loss function (MSE) used in regression problems.

$$\mathscr{L}(\tilde{y}, y) = \mathscr{L}(\tilde{y}, f(x;\theta)) = \frac{1}{N}\sum_{i=1}^{N}(\tilde{y}_i - f(x_i;\theta))^2 \tag{2.2.1}$$

Another very common loss function used in classification tasks is the Cross-Entropy (CE) loss.

$$\mathscr{L}(\tilde{y}, y) = \mathscr{L}(\tilde{y}, f(x;\theta)) = -\sum_{i=1}^{N}\tilde{y}_i \log(f(x;\theta)) \tag{2.2.2}$$

2.2.2 Unsupervised Learning

In contrast to supervised learning, unsupervised learning operates without labels. The goal in unsupervised learning is to identify inherent structures or patterns in the input data. Because there are no labels to guide the learning process, these algorithms learn from the structure of the data itself.

A classic example of unsupervised learning is clustering, where the goal is to group data points based on their similarity. For instance, in a dataset of news articles, we might want to group together articles that are about the same topic, e.g. an interview with a singer and an article about a new art gallery (art).

Unsupervised learning can be described mathematically as finding parameters θ that maximize the likelihood of observing the given data x.

$$\theta^* = \arg\max_{\theta} \log P(x;\theta) \tag{2.2.3}$$

Here, $P(x;\theta)$ is the probability of observing the data x given the parameters θ. This objective function is often used in algorithms such as k-means clustering or Gaussian mixture models.

In both supervised and unsupervised learning, the objective is to learn meaningful representations of the data that can be used for making predictions or understanding the structure of the data. While supervised learning is often more straightforward due to the presence of labels, unsupervised learning can uncover hidden patterns that may not be apparent when labels are used. As a side note, this positive "side-effect" is also encountered in self-supervised learning as well, and it's one of the reasons these models achieve higher generalization and can complete many different tasks without explicit training.

2.3 Neural Networks

Deep Neural Networks, or simply Neural Networks, serve as the foundation of modern deep learning. Inspired by the human brain's structure, artificial neural networks are composed of interconnected nodes, also known as neurons or units, structured in layers. Each layer transforms its input into a more abstract representation, leading to a hierarchy of increasingly meaningful features as we go deeper into the network.

Deep neural networks (DNNs) consist of multiple layers chained together. In their simplest form, DNNs consist only of feedforward linear layers and non-linear activation functions that help model non-linearities and avoid collapse. However, more complex layers have also been developed, like convolutional layers used in CNNs, recurrent layers used in RNNs, e.g. LSTM, and the more recent self-attention layers used in Transformer networks.

2.3.1 Neurons and Activation Functions

A neuron is the fundamental computational unit of a neural network. It receives inputs, applies specific operations to these inputs, and produces an output. The operations applied usually involve calculating the weighted sum of the inputs, adding a bias term, and finally applying an activation function σ.

Formally a single neuron's output in matrix notation is described as

$$y = \sigma(w^T x + b) \tag{2.3.1}$$

where $x = [x_1, x_2, ..., x_n]^T$ is the input vector, $w = [w_1, w_2, ..., w_n]^T$ is the synapse weight vector, b is a scalar bias term, and σ is a non-linear activation function. The synapse weights w and bias term b are learnable through the backpropagation of error Rumelhart et al. (1986), executed during the training phase of a neural network (see **??**).

The ***activation function*** σ plays a critical role by introducing non-linearity into the model. This allows the neural network to model complex, non-linear relationships between inputs and outputs. A commonly used activation function is the Rectified Linear Unit (ReLU), which is defined as:

$$\sigma(x) = \max(0, x) \tag{2.3.2}$$

There are also variations of ReLU, that aim to solve some of its problems. A common variation is the LeakyReLU function (Maas, 2013) defined as:

$$\sigma(x) = \max(-0.1x, x) \tag{2.3.3}$$

Another great activation function that greatly improves upon ReLU is GELU. The Gaussian Error Linear Unit (GELU) (Hendrycks and Gimpel, 2020). The GELU function has found great success over recent years, proving a very suitable activation function for Transformer networks used in Large Language Models (LLMs). GELU is approximated as

$$f(x) = 0.5x\left(1 + \tanh\left[\sqrt{\frac{2}{\pi}}\left(x + 0.044715x^3\right)\right]\right) \tag{2.3.4}$$

but is defined as

$$\text{GELU}(x) = xP(X \leq x) \tag{2.3.5}$$

where x is the input to the GELU function and X is a random variable following a standard normal distribution. $P(X <= x)$ is the cumulative distribution function of X. This interpretation connects the GELU function to probabilities and allows an interesting perspective into its behavior and properties.

A historically very important activation function is the sigmoid function. Even though it has fallen out of favor due to multiple shortcomings, sigmoid remains one of the most important activations and a key ingredient for LSTM networks (see 2.5). Sigmoid function is defined as:

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{2.3.6}$$

2.3.2 Layers and Deep Neural Networks

Neurons are typically organized into layers in a neural network. Neural networks usually have an input layer, an output layer and multiple intermediate layers called hidden layers. ***Hidden layers*** are the layers between the input and output layers. If a neural networks has 1 or more hidden layers then it is defined as a ***Deep Neural Network.*** **(DNN)**.

The features learned by the hidden layers are often referred to as ***hidden representations***. These layers extract features and patterns from the input data, which are then used by the final output layer for making predictions. Each layer's output becomes the input to the next layer, creating a chain of transformations. This process can be mathematically described as:

$$y = f_L(W^{(L)} f(W^{(L-1)} f(...f(W^{(1)}x + b^{(1)})...) + b^{(L-1)}) + b^{(L)}) \,, \tag{2.3.7}$$

where x is the input vector, y is the output vector, $W^{(l)}$ is the weight matrix for layer l, $b^{(l)}$ is the bias vector for neurons of layer l, and f is the activation function applied element-wise to the outputs of each neuron. Note that the activation function f can differ among different layers and even among different neurons.

The output layer often acts as a ***classifier***. In a classification problem, the output layer commonly uses the ***softmax*** activation function, which transforms its inputs into a probability distribution over the possible classes. Given inputs z, the hidden representations in the case of a DNN, the softmax function is defined as:

$$\sigma(z)_i = \frac{e^{z_i}}{\sum_j e^{z_j}} \tag{2.3.8}$$

2.4 Gradient Descent and Backpropagation

2.4.1 Gradient Descent

Once we have defined the neural network architecture and the corresponding loss function, the goal of learning is to find the optimal set of parameters, often denoted as θ^*, that minimise the loss function $\mathscr{L}$. This optimization problem is typically solved using an iterative method called Gradient Descent, a method invented by the legendary French mathematician Augustin-Louis Cauchy back in 1847 (Cauchy, 1847).

Gradient Descent iteratively adjusts the parameters θ, of a function f e.g. a DNN, in the direction that decreases the loss function the most. This direction is given by the negative gradient of the loss

function evaluated at the current parameters. Formally, an update step of gradient descent can be written as:

$$\theta = \theta - \alpha \nabla \mathscr{L}(\theta) \tag{2.4.1}$$

In this equation,α denotes the learning rate, a hyperparameter that controls the step size of the updates, and $\nabla \mathscr{L}(\theta)$ denotes the gradient of the loss function with respect to the parameters. This gradient is a vector that points in the direction of the steepest increase of the loss function.

2.4.2 Backpropagation

However, computing the gradient of the loss function in a deep neural network can be challenging due to the large number of parameters and the complexity of the functions involved. The backpropagation algorithm is used to efficiently compute these gradients and train deep neural networks (Rumelhart et al., 1986).

Backpropagation works by first performing a forward pass through the network to compute the output and the loss. It then traverses the network in reverse order, from the output layer to the input layer, to compute the gradient of the loss with respect to each parameter.

The chain rule of calculus forms the backbone of the backpropagation algorithm. For a neuron in the l-th layer of the network with activation function h, input z, and output a = h(z), the chain rule gives us:

$$\frac{\partial \mathscr{L}}{\partial z} = \frac{\partial \mathscr{L}}{\partial a} \cdot \frac{\partial a}{\partial z} \tag{2.4.2}$$

The term $\partial \mathscr{L} / \partial$ a is computed from the gradients of the loss with respect to the outputs of the next layer (closer to the output of the network), which have already been computed in the reverse pass of backpropagation. The term $\partial a / \partial z$ is computed from the derivative of the activation function.

In this way, backpropagation efficiently computes the gradients of the loss with respect to all the parameters of the network, which are then used to update the parameters using gradient descent.

In the next sections, we will delve into specific types of neural network architectures, including LSTM networks and Transformer networks, and how they build on these fundamental concepts.

2.5 Recurrent Neural Networks and LSTM

2.5.1 Recurrent Neural Networks

Recurrent Neural Networks (RNNs) are a class of artificial neural networks designed for processing sequential data. They maintain a hidden state that can encode information from an arbitrary point in the sequence (Elman, 1990).

Formally, an RNN computes the hidden vector sequence and the output vector sequence by iterating the following equations over $t = 1, \ldots, T$:

$$\mathbf{h}_t = \sigma_h(\mathbb{U}_h h_{t-1} + \mathbb{W}_h \mathbf{x}_t + \mathbf{b}_h), \tag{2.5.1}$$
$$\mathbf{o}_t = \sigma_o(\mathbb{W}_o h_t + \mathbf{b}_o), \tag{2.5.2}$$

where $\mathbf{h}_t$ is the hidden state at time t, $\mathbf{x}_t$ is the input at time t, $\mathbf{o}_t$ is the output, $\mathbb{U}_h, \mathbb{W}_h, \mathbb{W}_o$ are weight matrices to be learned, and $\mathbf{b}_h, \mathbf{b}_o$ are bias vectors. σ_h and σ_o are activation functions, usually chosen to be nonlinear (Goodfellow et al., 2016).

Despite their expressivity, RNNs are notoriously difficult to train effectively. This is largely due to the so-called vanishing and exploding gradients problems, which make it difficult for an RNN to learn to maintain and use information in its hidden state over long periods of time (Hochreiter, 1998).

2.5.2 Long Short-Term Memory Networks

To mitigate the issues with training RNNs, Hochreiter and Schmidhuber introduced the Long Short-Term Memory (LSTM) architecture (Hochreiter and Schmidhuber, 1997). LSTM networks have a similar structure to standard RNNs, with the addition of memory ***cells*** and ***gate*** mechanisms that control the flow of information through the network. A cell operates similarly to the hidden state, but its update is regulated by three learned gates: an input gate i_t, a forget gate f_t, and an output gate o_t.

LSTMs divide the context management problem into two subproblems: removing information no longer needed from the context and adding information likely needed for later decision-making. The memory cells allow the network to selectively remember or forget information over time, while the gating mechanisms enable the network to regulate the flow of new information.

The LSTM is described by six equations that define the update of information inside the LSTM cell. These are given by:

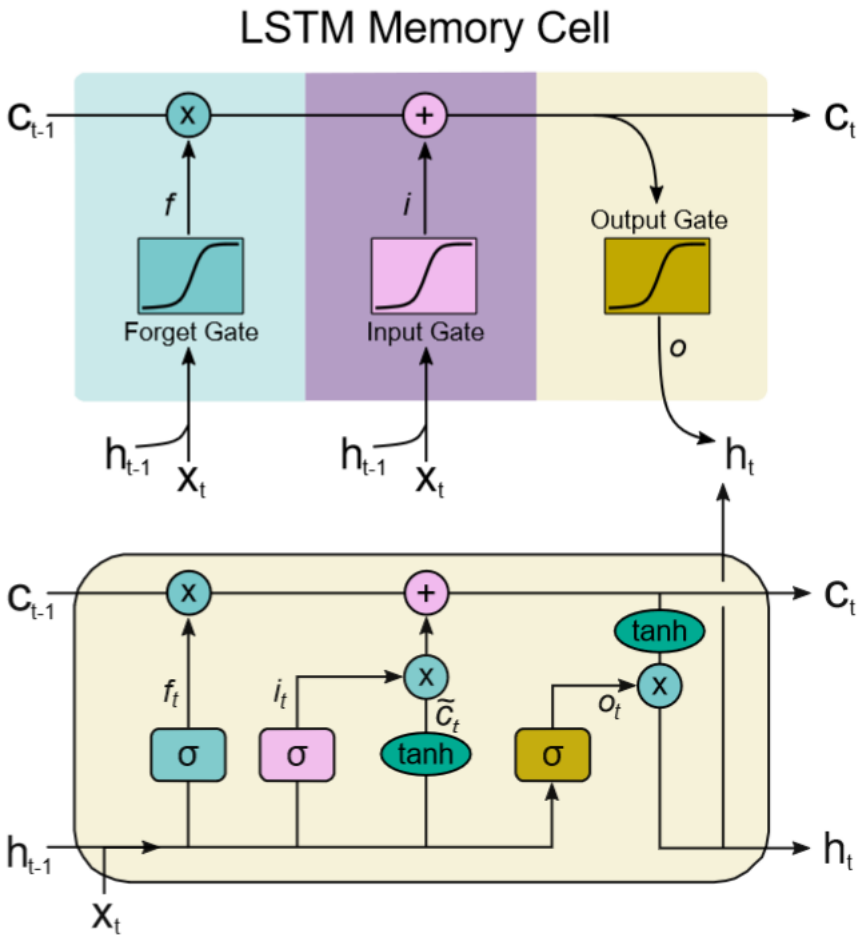


Figure 2.5.1: A high level conceptualization of the LSTM network (top figure) and a low level schematic of the LSTM block (bottom figure). Source Poulinakis et al.

$$\mathbf{f}_t = \sigma(\mathbb{U}_f \mathbf{h}_{t-1} + \mathbb{W}_f \mathbf{x}_t + \mathbf{b}_f), \tag{2.5.3}$$
$$\mathbf{i}_t = \sigma(\mathbb{U}_i \mathbf{h}_{t-1} + \mathbb{W}_i \mathbf{x}_t + \mathbf{b}_i), \tag{2.5.4}$$
$$\tilde{\mathbf{C}}_t = \tanh(\mathbb{U}_c \mathbf{h}_{t-1} + \mathbb{W}_c \mathbf{x}_t + \mathbf{b}_c), \tag{2.5.5}$$
$$\mathbf{C}_t = \mathbf{f}_t \odot \mathbf{C}_{t-1} + \mathbf{i}_t \odot \tilde{\mathbf{C}}_t, \tag{2.5.6}$$
$$\mathbf{o}_t = \sigma(\mathbb{U}_o \mathbf{h}_{t-1} + \mathbb{W}_o \mathbf{x}_t + \mathbf{b}_o), \tag{2.5.7}$$
$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{C}_t), \tag{2.5.8}$$
$$\tag{2.5.9}$$

where $\mathbf{x}_t$ is the input at time step t; $\mathbf{h}_t$ is the hidden state at time step t; $\mathbf{C}_t$ is the cell state at time step t; $\mathbf{f}_t$, $\mathbf{i}_t$; and $\mathbf{o}_t$ are the forget gate, input gate, and output gate vectors at time step t, respectively. $\mathbb{W}_k$ and $\mathbb{U}_k$ are learnable weight matrices, while $\mathbf{b}_f$, $\mathbf{b}_i$, $\mathbf{b}_C$, and $\mathbf{b}_o$ are learnable bias vectors. The symbol $\odot$ denotes the Hadamard product, and σ and *tanh* are the sigmoid and hyperbolic tangent activation functions, respectively.

The forget gate $\mathbf{f}_t$ controls which information to discard from the cell state $\mathbf{C}_{t-1}$, based on the current input $\mathbf{x}_t$ and the previous hidden state $\mathbf{h}_{t-1}$. The input gate $\mathbf{i}_t$ controls which new information to add to the cell state, and the candidate values $\tilde{\mathbf{C}}_t$ are computed using a hyperbolic tangent function to ensure that the values are in the range of $[-1, 1]$. Combining the forget gate and input gate allows the LSTM to selectively retain or discard information over time, which is useful for tasks that require remembering long-term dependencies

The LSTM network has been a revolutionary architecture for deep learning. It allowed networks to have memory and accomplish tasks that were previously poorly handled like timeseries forecasting, video analysis, and natural language processing (NLP). Especially in the field of NLP, RNNs provided a huge boost and dominated the field for many years. However, over recent years a new transformative architecture entered the field and shifted the dominance away from RNNs - the Transformer architecture and its self-attention mechanism. The next section discusses this revolutionary model.

2.6 Self-Attention and Transformers

Even though attention mechanisms are not a new invention Schmidhuber (1992); Bahdanau et al. (2015), they have gained widespread popularity after the work of Vaswani et al. (2017a) was published in 2017, where an innovative more sophisticated form of attention; self-attention was introduced. In this section we introduce the architecture of transformers. Like the LSTMs introduced in the previous section, transformers can handle distant information. But unlike LSTMs, transformers are not based on recurrent connections (which can be hard to parallelize) which means that transformers can be more efficient to implement at scale.

Transformers map sequences of input vectors $(x1, ..., xn)$, to output vectors $(y1, ..., yn)$ that maintain the same length. Transformer networks consist of stacks of transformer blocks, each of which is mainly made up of feedforward layers and self-attention mechanisms. The ***self-attention*** mechanism is the key innovation of transformers which allows the extraction of information from arbitrarily large contexts without the need for recurrent connections. First, we will explain how self-attention functions and subsequently elaborate on how it integrates into the more comprehensive transformer blocks.

The self-attention mechanism enables the network to directly assimilate and utilize information from an extensively large context. Unlike RNNs, it accomplishes this without the need for transferring it through any intermediary recurrent connections. Initially, we will explain the simple attention mechanism, then expand to how self-attention functions and subsequently elaborate on how it integrates into the more comprehensive transformer blocks.

2.6.1 Attention mechanism

An attention layer (fig. 2.6.1) maps input sequences ($x1, ..., xn$) to output sequences of the same length ($y1, ..., yn$). During processing, the model has access to all of the inputs up to and including the one under consideration, but no access to inputs beyond it. The computation of each input element is independent of all the other computations. The first point ensures that we can use this approach to create language models and use them for autoregressive generation, and the second point means that we can easily parallelize both inference and training of such models.

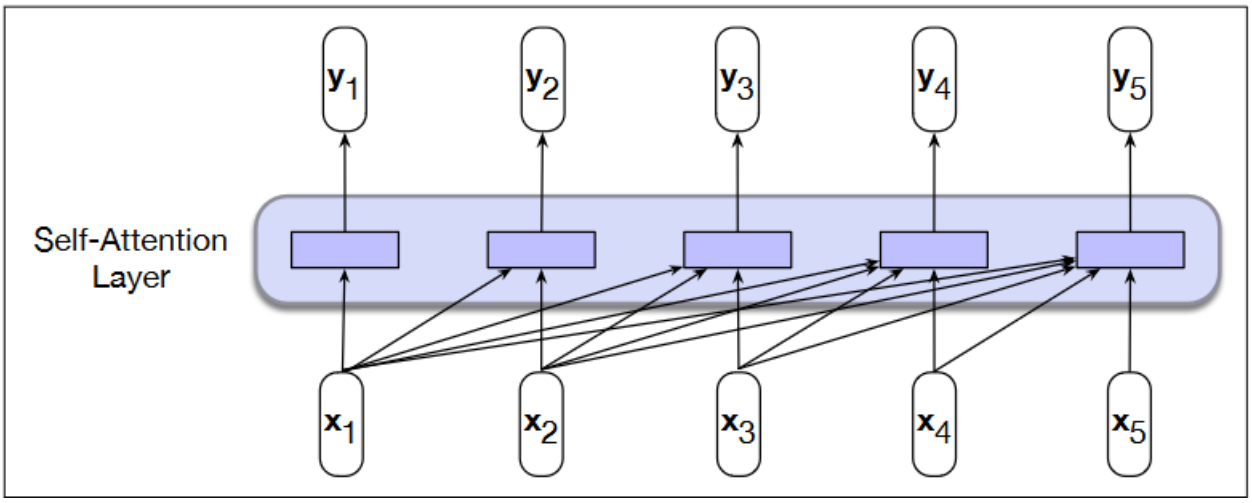


Figure 2.6.1: Information flow in a causal (or masked) self-attention model. In processing each element of the sequence, the model attends to all the inputs up to, and including, the current one. Unlike RNNs, the computations at each time step are independent of all the other steps and therefore can be performed in parallel. Source Jurafsky and Martin (2023) chapter 10 pg. 3

At the core of the attention layer is the ability to compare an item of interest with a collection of other items, the context. This allows the model to understand the relevance and interaction between two inputs in the current context. The result of these comparisons is then used to compute an output for the current input. For example (see fig 2.6.2) the output y_3 is the result of the comparisons between x_3, its preceeding inputs x_1, x_2 and with x_3 itself. The simplest form of comparison between elements in a self-attention layer is a dot product and the result is often refered to as score.

$$s(x_i, x_j) = x_i.x_j \tag{2.6.1}$$

The resulting scores are then normalized, with a softmax function, to create a vector of attention weights α_{ij}, defined as:

$$\alpha_{ij} = \text{softmax}(\text{s}(x_i, x_j)) \quad \forall j \leq i \tag{2.6.2}$$

$$= \frac{\exp\big(\text{s}(x_i, x_j)\big)}{\sum_{k=1}^{i} \exp(\text{s}(x_i, x_k))} \quad \forall j \leq i \tag{2.6.3}$$

Once these weights are available generating an output y_i is the result of a weighted sum of all inputs $\leq i$, computed as

$$y_i = \sum_{j<i} \alpha_{ij} x_j \tag{2.6.4}$$

2.6.2 Self-Attention mechanism

Equation 2.6.4 shows how a simple attention mechanism is used to generate outputs. This attention mechanism can also be used in LSTM networks. However, the transformers go a step further and model context in a more complex and sophisticated way with ***self-attention***. Consider the three different roles that each input embedding plays during the course of the attention process.

- As the current focus of attention when being compared to all of the other preceding inputs. We'll refer to this role as a **query**.
- In its role as a preceding input being compared to the current focus of attention. We'll refer to this role as a **key**.
- As a value used to compute the output for the current focus of **value** attention.

The transformer self-attention mechanism differentiates a word's semantics in the context depending on the role it has during a comparison. To achieve this, it uses three weight matrices W_Q, W_K, W_V. These matrices are used to project input vectors x_i into a different space depending on their role as key, query, or value.

$$q_i = W_Q x_i,$$
$$k_i = W_K x_i,$$
$$v_i = W_V x_i.$$

Once the inputs have been projected, the score values and output generation slightly change. So equations 2.6.1 2.6.4 become :

$$score(x_i, x_j) = \frac{q_i.k_j}{\sqrt{d_k}} \tag{2.6.5}$$

$$y_i = \sum_{j<i} \alpha_{ij} v_j \tag{2.6.6}$$

where d_k is the dimensionality of q, v, k vectors and the denominator in 2.6.5 is used for normalization reasons. The softmax normalization of α_{ij} still follows equation 2.6.2.

2.6.3 Transformer block

The core component of a transformer block is the self-attention calculation. Each transformer block consists not only of the self-attention layer, but also includes other components such as feedforward layers, residual connections, and normalization layers (fig. 2.6.3). In order for these blocks to be stacked in a similar fashion to stacked RNNs, their input and output dimensions are designed to match.

Deep neural networks often make use of residual connections. These connections bypass certain layers, enabling information to flow from lower layers to higher ones without passing through an intermediary layer. Residual connections serve to enhance learning by providing higher-level layers with direct access to information from lower layers, thus improving the efficiency of both forward activation and backward gradient propagation (He et al., 2015).

In the context of transformers, residual connections are realized by adding the input vector of a layer to its output vector before it is forwarded. This occurs in both the attention and feedforward sublayers within a transformer block. The vectors produced from these sums are then subject to layer normalization (Ba et al., 2016). Layer normalization, or layer norm, is one of several normalization techniques that can be leveraged to enhance training performance in deep neural networks. It operates

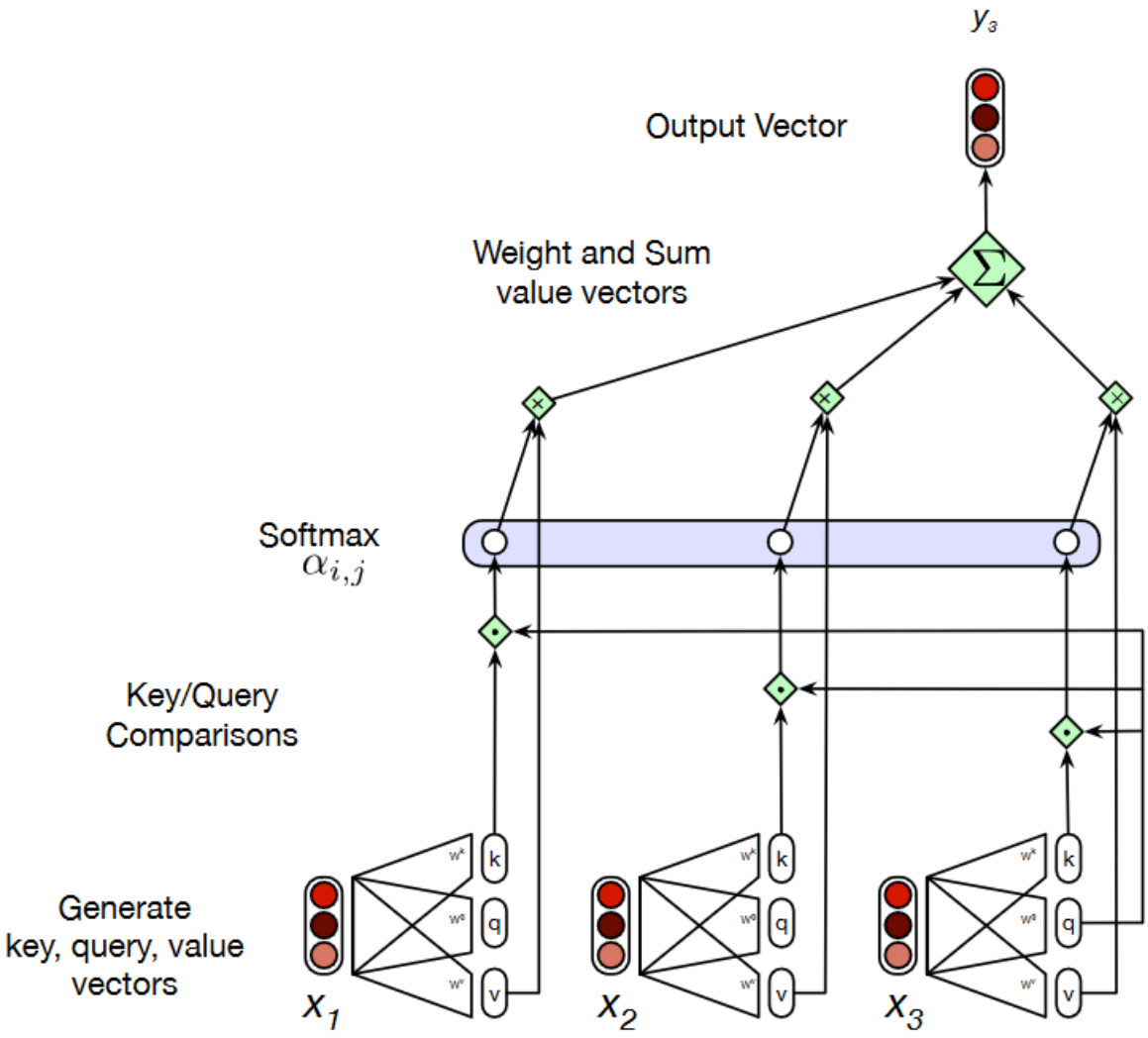


Figure 2.6.2: Calculating the value of y3, the third element of a sequence using causal (left-to-right) self-attention. Source Jurafsky and Martin (2023) chapter 10 pg. 5

by ensuring the values of a hidden layer remain within a range that is conducive to gradient-based training.

The operations carried out in a transformer block and its final output y can be represented as:

$$z = \text{LayerNorm}(x + \text{SelfAttention}(x)),$$
$$y = \text{LayerNorm}(z + \text{FFN}(z)).$$

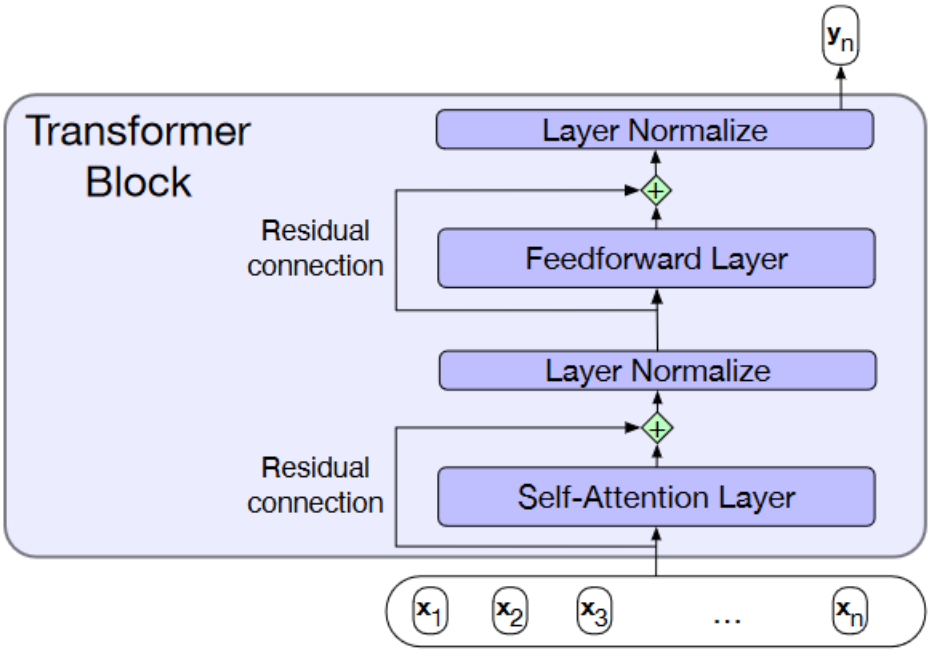


Figure 2.6.3: A vanilla transformer block with all of its layers. After computing the attention weights, the results are passed through a normalization layer, a feedforward layer, then again through a normalization layer. Two residual connections are also present. Source Jurafsky and Martin (2023) chapter 10 pg. 6

Chapter 3

Self Supervised Learning

> "Self-supervised learning is the dark matter of intelligence."
>
> Yann LeCun

Over the last few years AI is experiencing an important shift. After years of developing task specific models we are now entering the era of the so called *"Foundational Models"*. Foundational models are trained on a broad set of unlabeled data that can be used for different tasks, with minimal fine-tuning. Self Supervised Learning (SSL) is a key ingredient to this success.

We've seen the first glimmers of the potential of foundation models in the worlds of imagery and language. Early examples of models, like GPT-3 (Brown et al., 2020), BERT (Devlin et al., 2019), or DALL-E 2 (Ramesh et al., 2021), have shown what's possible. Input a short prompt, and the system generates an entire essay, or a complex image, based on your parameters, even if it wasn't specifically trained on how to execute that exact argument or generate an image in that way.

All of these models exploit self-supervision, a new learning paradigm that seems to open the doors to new extreme possibilities. Some may even argue that these new models show signs of inherent creativity, thanks to their self-supervised training that allows them to mine knowledge out of documented human knowledge and creativity.

SSL lies in the heart of this study and this chapter explores this booming and extremely interesting field.

3.1 Introduction

SSL methods have enjoyed a renaissance since 2020, thanks in large part to the availability of extremely large datasets and high-memory GPUs. However, the origins of SSL go back to the very beginning of the deep learning era (Balestriero et al., 2023).

Self-supervised learning is a learning paradigm through which neural networks learn without human-made labels. Unlike unsupervised learning whose methods just exploit the training data's inherent statistics and structure, self-supervised learning transforms unsupervised tasks into supervised tasks while also exploiting the dataset's inherent structure. **They employ pre-training tasks, referred to as *"pretext tasks"*.**

Pretext tasks serve as a means of converting unsupervised datasets into supervised tasks, facilitating representation learning. These tasks are often unrelated to the final ***downstream tasks***. Pretext tasks are carefully crafted by human designers and leverage autogenerated labels, also known as ***pseudo-labels***.

There are two methods of self-supervised training (Lilian Weng, 2021)

- Contrastive Learning
- Self-prediction

Contrastive learning methods learn representations by learning similarities (dissimilarities) between different views of the same (different) object. Various contrastive learning losses have been proposed which will be discussed in greater detail. Data augmentation plays a vital role in contrastive learning and heavily depends on the modality of the input. Manipulating inputs to an extend that it is challenging but not impossible for the network to find similarities between the augmented view and the original is essential for success.

With contrastive learning a network has to detect relationships between multiple data samples. For example, it needs to find which samples are most similar or if an audio clip is in sync with a video. Based on these relationships, representation of the samples are manipulated and placed adequately on a high dimensional representation space.

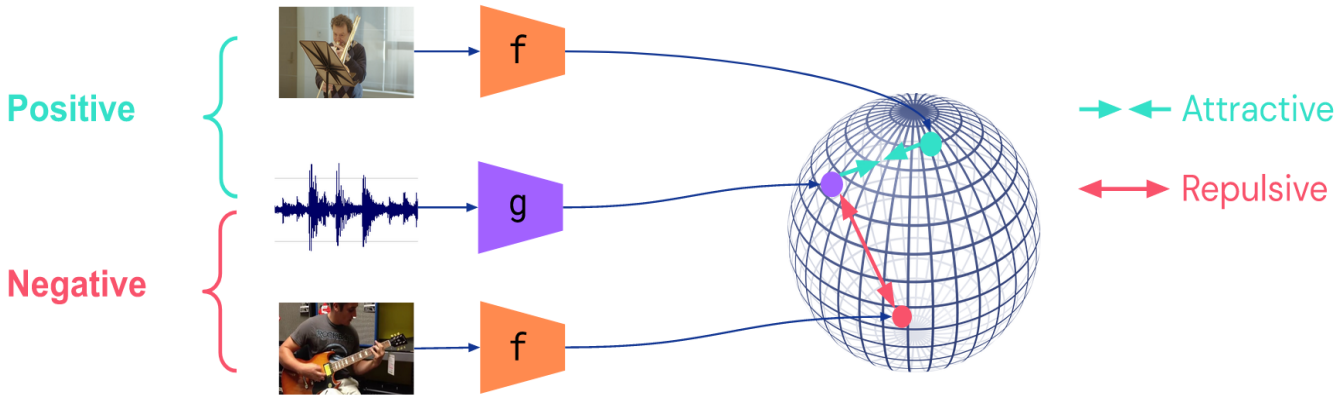


Figure 3.1.1: Visual illustration of the contrastive learning framework. Source

In contrast, **self-prediction** methods construct *prediction tasks* within every individual data sample. Given a data sample, we try to predict parts that are intentionally missing, while the missing part acts as the pseudo-label. Predicting masked words in a sentence, colorizing grayscale images, or predicting future parts from the current ones are all examples self-prediction tasks.

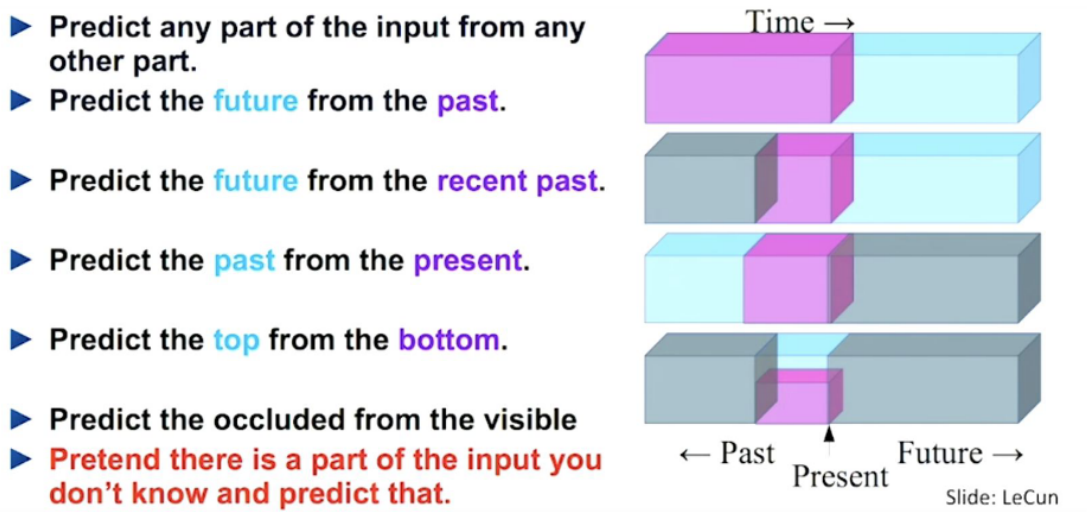


Figure 3.1.2: Visual illustration of self-prediction methods (famous illustration from Yann LeCun). Source

3.2 Early Work

Autoencoders (Rumelhart and McClelland, 1987), are one of the oldest forms of self-supervised learning since the input itself acts as the pseudo label. However, when the dimensionality of the autoencoder is high, representations may collapse into the identity function. Vincent et al. (2010) partially corrupted the input image before feeding it to the autoencoder, which is asked to recover the undistorted version. Such and similar techniques have been employed in order to resolve the collapse issue. In a more recent work, Zhang et al. (2019) proposed to autoencode transformations by training on the objective

$$\min_{E,D} \mathop{\mathbb{E}}_{t\sim T, x\sim X} \ell(t, \hat{t}) \tag{3.2.1}$$

where E, D denote the encoder and decoder networks we try to learn, T a set of transformation operations, t is the transformation applied on the input and

$$\hat{t} = D(E(x), E(t(x))) \tag{3.2.2}$$

is the estimation of the original transformation t applied. The loss function ℓ is a regression loss applied on the parameterized matrices that represent the affine and projective transformations t.

Such works have been critical in paving the way towards unsupervised training and exploitation of pseudo-labels.

3.3 Self-prediction methods

Self-prediction methods aim at constructing and solving tasks that require no human made labels but act educationally to the network. The philosophy behind these tasks is to guess a missing part of the input given its context.

Many pretext tasks for self-prediction methods involve transforming an input I, computing a representation of the transformed input, and predicting properties of transformation T from that representation. Such approaches include colorizing greyscale images (Zhang et al., 2016b) and predicting the rotation applied in training images (Gidaris et al., 2018). Learning is associated with determining model parameters θ that maximize the likelihood of the latent variables given the observations. The objective function usually aims at maximizing the likelihood by minimizing a loss similar to

$$loss(X_i, \theta) = -\frac{1}{K}\sum_{y=1}^{K} log(F^y(X^{y^*}|\theta)) \tag{3.3.1}$$

where $F^y(X^{y^*}|\theta)$ is the predicted probability for the geometric transformation with label y, X^{y^*} is the transformed input and θ are the learnable parameters of model F (.) .

The works of Mikolov et al. (2013b,a) in natural language processing showcased that learning from context is very promising. This inspired researchers to adapt the method to other modalities as well. Doersch et al. (2015) tried to predict the spatial relationship between image patches and Noroozi and Favaro (2016) to solve visual jigsaw puzzles, as methods of context learning adapted to computer vision. Techniques similar to CBOW and Skip-Gram introduced by Mikolov et al. (2013b) were also applied to the audio modality. Speech2Vec (Chung and Glass, 2018) aims at reconstructing a spectrogram slice from just past and future slices, similar to the Skip-gram model, or reconstructing past and future slices from just one target spectrogram slice, similar to C-BOW model. Both techniques minimize the

reconstruction L_2 loss. Moreover, Speech2Vec (**?**) can also be considered an audio version of Word2Vec that expanded on the prior work of Chung and Glass (2018) to non-speech related tasks.

Another adaptation of learning through context on the audio modality includes auto-regressive models. In Auto-regressive Predictive Coding (APC) (Chung et al., 2019) an encoder network encodes each wave sample x_t and a RNN operates as a context network that provides a representation of the signal up to the current timestep. The context network accepts the current encoded wave sample z_t and the previous context c_{t-1}. Its output is used to predict the signal's representation τ steps ahead by minimizing $L_1(z_t, z_{t+\tau})$ loss. The step τ is a hyperparameter that controls what the representations learn. In a similar work, van den Oord et al. (2018) proposed the Contrastive Predictive Coding (CPC) model. Even though it shares many similarities with APC, it employs a contrastive objective function based on negative cross entropy. During learning CPC does not only compare to the positive sample $z_{t+\tau}$ but also several negative samples $\bar{X}$.

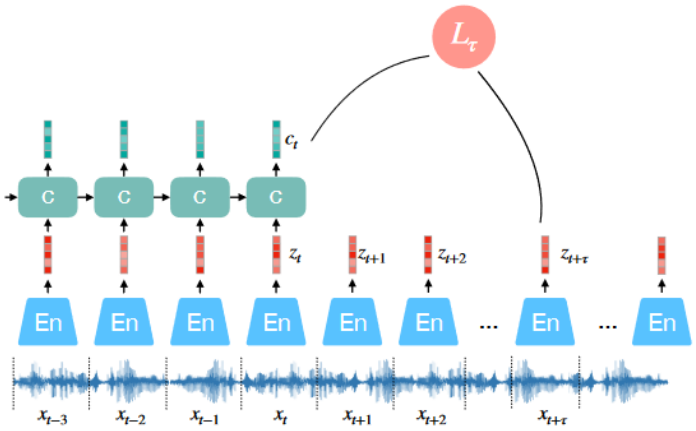


Figure 3.3.1: Auto-regressive Predictive Coding (APC) architecture. The Context (C) network is usually implemented with a RNN. The Loss function L_τ is an L_1 loss.

3.4 Contrastive Learning methods

Contrastive learning methods aim at creating an embedding space by manipulating and coordinating relationships between input samples. The philosophy behind contrastive learning is to place similar samples close together in the embedding space while distancing disimilar samples. Formation of the embedding space is achieved through the use of the so called *"**Contrastive Losses**"*.

Contrastive learning loss functions were introduced by Hadsell et al. (2006). These are loss functions whose minimizations produce mappings that map similar input vectors to nearby points on the output manifold and dissimilar vectors to distant points. For a pair of input vectors X_1, X_2 and a binary label y (0 for similar inputs, 1 otherwise) the contrastive loss function can be formulated as

$$L(W) = \sum_{i=1}^{P} L(W, (y, X_1, X_2))^i \tag{3.4.1}$$

$$\mathscr{L}(W, (y, X_1, X_2)^i = (1-y)L_p(D_w^i) + yL_n(D_w^i) \tag{3.4.2}$$

where L_p is the partial loss that computes the loss between positive pairs (i.e. similar input vectors), while L_n the loss between negative pairs (i.e. dissimilar input vectors). The exact form of the loss

introduced by Hadsell et al. (2006) was

$$L(W,(y,X_1,X_2)^i = (1-y)\frac{1}{2}D_w^i + (y)\frac{1}{2}\{\max(0,m-D_w^i)\}^2 \tag{3.4.3}$$

One of the most important works in natural language processing that successfully used contrastive losses are the works of Mikolov et al. (2013a,b) that produced the Word2Vec model (CBOW, Skip-gram). They learn word representations by utilizing their context. Skip-gram maximizes the probability

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq +c, j\neq 0}\log p(w_{t+j}|w_t) \tag{3.4.4}$$

where c is a hyperparameter that adjusts the size of the training context. In their work, they tested Noise-Contrastive Estimation (NCE) loss first introduced by Gutmann and Hyvärinen (2010), but also Negative Sampling (NEG) Mikolov et al. (2013b), a simplification of NCE.

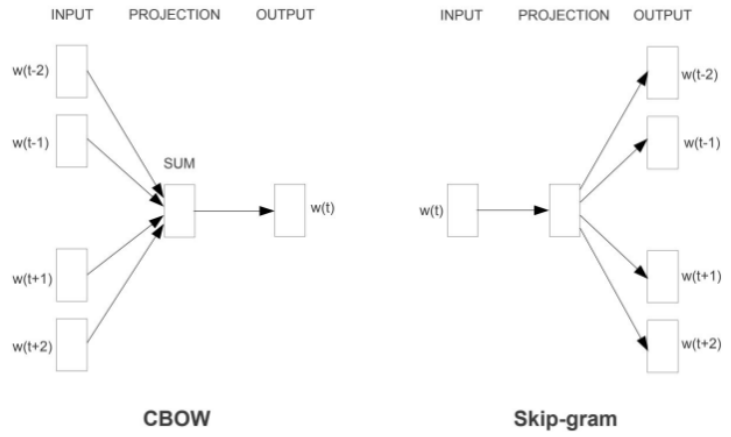


Figure 3.4.1: Word2Vec algorithms visually illustrted.

Another successful example of contrastive learning models is **Siamese networks**, first introduced by Bromley et al. (1993). They refer to a family of architectures consisting of two, usually identical, neural networks. This is a "two tower" architecture where each "tower" processes a view of a training data sample (fig. 3.4.2) and their representations are usually manipulated using contrastive objective functions. When the two views form a positive pair (i.e. are augmentations of the same object) the representations are pulled together. In contrast, when the two views form a negative pair (i.e. are augmentations of different objects) then their representations are pushed away.

Misra and van der Maaten (2019) in their work PIRL, instead of applying a transformation T on an input image I and predicting a property of T, proposed to use Siamese networks and compare representations of input pairs produced by identical encoders. Input pairs may be positive or negative. They minimize an NCE loss and use a memory bank of negative samples to store representations of negative examples. The memory bank allows for a high number of negative samples, as required for the convergence of NCE loss to a non-constant solution, without increasing the batch size to an infeasibly large number.

A similar approach is introduced by Chen et al. (2020). SimCLR is a simple framework for contrastive self-supervised learning. The siamese networks architecture is again used and a series of different Transformation methods T are investigated to create new views of the original input images. During training, they minimize the NT-Xent (Normalized Temperature-scaled Cross Entropy) objective function formulated as

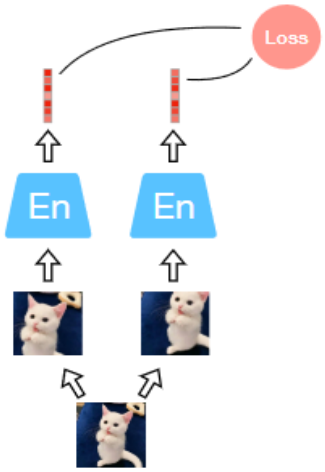


Figure 3.4.2: A typical siamese-networks architecture.

$$\ell_{i,j} = -log \frac{exp(sim(z_i, z_j)/\tau)}{\sum_{k=1}^{2*N} \mathbb{1}_{k \neq i} exp(sim(z_i, z_k)/\tau)}, \tag{3.4.5}$$

where the $sim(.)$ operator is a cosine similarity operator.

Furthermore, SimCLR (Chen et al., 2020) introduced a very crucial component, that is nowadays used by most SSL methods with joint embeddings, ***the projector***. The projector is usually a MLP with 2-3 linear layers and ReLU activation. The SSL loss is applied to the projector's output (embeddings), and the projector is usually discarded after training. The projector has been shown to provide significant improvements in performance. For example, in a 100-epochs training, the projector adds around 20% of top-1 accuracy in SimCLR and VICReg (Bardes et al., 2022) (from around 50% to 68% and 48% to 68% respectively).

The projector component has been shown to be highly beneficial even for supervised learning settings, when there is misalignment between the training and downstream tasks (Bordes et al., 2022). When looking from a lens of transfer learning it's easy to understand that using a projector to "absorb" the training task's data bias and then discarding it, can help to reduce overfitting, improve generalization and create more robust representations.

The ***Barlow Twins*** model (Zbontar et al., 2021) is another successful self-supervised method. Even though it bears similarities to SimCLR **it does not require negative pair samples**, hence they do not use a contrastive loss function. Their aim is to learn transformation invariant representations by pulling the representations of similar samples closer together while also decorrelating different embedding dimensions in order to reduce redundancies. Their suggested objective function is a function of the cross correlation matrix $\mathbf{C}$ computed on a batch of output embedding vectors.

$$L_{BT} = \sum_{i=1}^{N}(1 - C_{ii})^2 + \lambda \sum_{i=1}^{N} \sum_{j \neq 1}^{N} (C_{ij})^2 \tag{3.4.6}$$

where C_{ii} are the diagonal terms of matrix $\mathbf{C}$, C_{ij} the non-diagonal and λ a parameter that controls the influence of the redundancy reduction term. The first term of the loss is the *invariance term* that helps the network learn transformation invariant features. The second term, the *redundancy term*, helps decorrelate the embeddings' dimensions, hence reducing redundant information encoded in them.

The Barlow Twins architecture lies at the heart of this study. Building on this self-supervised idea

we extend the barlow twins framework to a multimodal architecture and aim at evaluating its efficacy on the domain of emotion recognition.

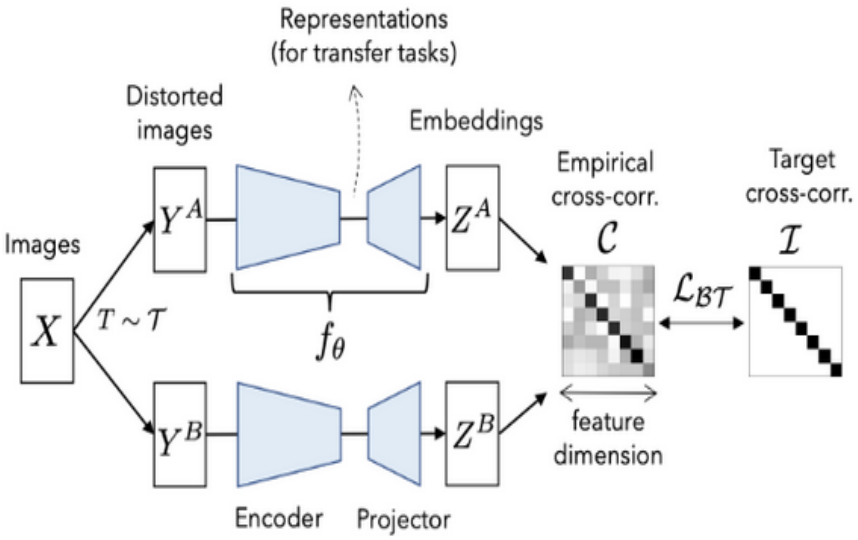


Figure 3.4.3: The Barlow Twins model's architecture. Two augmented versions Y^A, Y^B of an input image X are created. An encoder network E produces representations for each augmented view and a projection network projects the representation into an embedding space of fixed dimensions. The cross-correlation matrix of the output embedding vectors Z^A, Z^B is calculated and compared to the identity matrix **I**.

3.5 Conclusion

The ability to train on extensive unlabeled data brings numerous advantages. Unlike traditional supervised learning methods that rely on labeled data for a specific known task, Self-Supervised Learning enables the learning of generic representations that can be applied across multiple tasks. SSL proves particularly valuable in domains like medicine, where obtaining labels is expensive or where the specific task is not known in advance (Ciga et al., 2021), (Krishnan et al., 2022). Additionally, research suggests that SSL models can acquire more robust representations against adversarial examples, label corruption, and input perturbations, while also exhibiting increased fairness compared to their supervised counterparts (Hendrycks et al., 2019),(Goyal et al., 2022). As a result, SSL has garnered increasing interest within the field.

SSL lies at the heart of this study. Understanding the foundations is important in order to understand the proposed method. This chapter aimed at explaining the foundational knowledge and provide some relevant examples. For a more detailed overview on the subject we highly suggest reading the work of Balestriero et al. (2023).

Chapter 4

Multimodal Learning

4.1 Introduction

The world surrounding us involves multiple modalities – we see objects, hear sounds, feel texture, smell odors, and so on. In general terms, a modality refers to the way in which something happens or is experienced. Most people associate the word modality with the sensory modalities which represent our primary channels of communication and sensation, such as vision or touch. A research problem or dataset is therefore characterized as multimodal when it includes multiple such modalities. In order for Artificial Intelligence to make progress in understanding the world around us, it needs to be able to interpret and reason about multimodal messages. Multimodal machine learning aims to build models that can process and relate information from multiple modalities.

Recent advances in artificial intelligence have sparked a wave of generative AI. Foundational AI models beginning with large pre-trained language models, from BERT (Devlin et al., 2019) all the way to GPT3 (Brown et al., 2020) power revolutionary AI applications such as ChatGPT. It wasn't long before **generative AI sprouted into multimodal applications.** Multimodal foundational models started with visual-language models pioneered by works like CLIP (Radford et al., 2021) and later evolved into milestone generative models DALL-E (Ramesh et al., 2021) and Flamingo(Alayrac et al., 2022). Now, multimodal models have expanded to include audio as well, with MUSIC-LM (Agostinelli et al., 2023) being an extraordinary example of music generation from text, audio, and even visual queues.

These multimodal foundational models may be the first sparks of Artificial General Intelligence (AGI). Multimodal research paves the way into a new era of artificial intelligence with limitless applications to real life.

This section acts as an introduction to the revolutionary and extremely interesting field of Multimodal Deep Learning. We first explore the core challenges of multimodal learning, and move on to some of the foundational early works and recent milestone studies. We then explore how self-supervision is being used to train multimodal networks and finally explore multimodal datasets that accelerate research in the field.

4.2 Challenges of multimodal learning

Multimodal learning is a field of machine learning that aims to build models that can process and integrate information from multiple types of data or modalities. These modalities can include, but are not limited to, images, text, audio, and video data. The main goal of multimodal learning is to leverage the complementary information present in different modalities to improve the performance

of machine learning tasks, and to enable models to understand and interact with the world in a more human-like way by processing a rich variety of sensory inputs.

Training multimodal networks comprises many unique challenges, not found in unimodal models. According to Baltrušaitis et al. (2019), multimodal machine learning has 5 unique challenges that research needs to tackle in order for the field to advance further. These are representation, translation, alignment, fusion, and co-learning.

Representation

A primary, critical hurdle involves determining a method to characterize and condense multimodal data. The objective is to utilize the supplementary nature and repetitiveness of diverse modalities. Owing to the diverse nature of multimodal data, creating these depictions becomes complex. For instance, while language is frequently expressed in symbols, auditory and visual modalities are typically presented as signals. A good representation of data should capture emotional cues that can generalize over different speakers, background conditions, and semantic contents.

Translation

Another obstacle involves figuring out how to convert data from one type of modality to another (translation). This process becomes complex due to the heterogeneous nature of the data and the often ambiguous or subjective relationship between different modalities. For instance, there could be multiple accurate descriptions for a single image, meaning a unique ideal translation may not always be achievable.

Alignment

Alignment is a third challenge that involves establishing clear correlations between elements or sub-elements from two or more distinct modalities. An example might be correlating the steps in a culinary recipe to a corresponding video demonstrating the preparation of the dish. Addressing this issue requires quantifying similarities across different modalities, and managing potential long-distance dependencies and ambiguities.

Fusion

Fusion constitutes a fourth challenge, where the task is to merge information from two or more modalities to make a prediction. An example can be seen in audio-visual speech recognition, where visual data of lip movements is fused with auditory signals to predict spoken words. The data sourced from diverse modalities may exhibit differing predictive strength and noise structures, and there could potentially be incomplete data in at least one of the modalities.A good fusion mechanism should be able to combine input modalities effectively while respecting each modality's specificities.

Co-Learning

Co-learning is a fifth challenge, which involves transferring knowledge between modalities, their representations, and their predictive models. This can be seen in co-training algorithms, conceptual grounding, and zero-shot learning. Co-learning investigates how knowledge derived from one modality can assist a model trained on a different modality. This challenge becomes especially pertinent when one of the modalities has limited resources (e.g. annotated data).

In the following section we delve deeper into the challenge of representation and explain the two most prominent models of representation.

4.3 Multimodal Representation

The research field of Multimodal Machine Learning brings some unique challenges for computational researchers given the heterogeneity of the data. Learning from multimodal sources offers the possibility of capturing correspondences between modalities and gaining an in-depth understanding of natural phenomena.

One of the primary challenges to overcome is the ability to effectively represent and condense multimodal information by leveraging the strengths and overlaps of various modalities. Given the heterogeneous nature of multimodal data, creating such representations can be particularly difficult. For instance, language is frequently symbolic, while audio and visual data are often represented as signals.

While there has been a huge amount of work on unimodal representation, up until recently most multimodal representations involved simple concatenation of unimodal ones, but this has been rapidly changing. According to Baltrušaitis et al. (2019) the multimodal representation strategies can be categorized in two groups ***joint*** and ***coordinated*** representations. Joint representations combine the unimodal signals into the same representation space, while coordinated representations process unimodal signals separately, but enforce certain similarity constraints on them to bring them to what we term a coordinated space.

4.3.1 Joint Representations

$$x_m = f(x_1, \dots, x_n), \tag{4.3.1}$$

where the multimodal representation x_m is the output of a function with unimodal inputs $x_1, \dots, x_n$. This function f is commonly a deep neural network, a recurrent neural network, a transformer, or restricted Boltzmann machine.

The effect of joint representation is that all unimodal representations are projected together into a multimodal space, $x_m \in \mathbf{R}^M$.

The simplest form of joint representation is concatenation of the individual modalities, often referred to as *early fusion*. In the case of fixed length data feedforward layers are mostly used to create joint representations. However, in the case of varying length sequential data RNNs are used instead.

Neural networks are generally composed of a series of elements, including inner products followed by non-linear activation functions. When we want to use a neural network to depict data, it's first tasked with a specific function, such as identifying objects within images. Due to the layering in deep neural networks, each subsequent layer is hypothesized to abstractly represent the data at an increasingly higher level, making it commonplace to use the final or near-final layers for data representation.

When building a multimodal representation using neural networks, each modality begins with multiple individual neural layers. These are followed by a hidden layer that merges the modalities into a shared space. Concatenation, cross-attention ((Lu et al., 2019)), or any other technique may be used during merging. This combined multimodal representation is then passed through several hidden layers or used directly for prediction. Such models can be trained from beginning to end, learning both data representation and the performance of a specific task. This methodology forms a close relationship between multimodal representation learning and multimodal fusion within the framework of neural networks

One significant advantage of joint representations is their capacity to pre-train with unlabeled data when there isn't sufficient labeled data for supervised learning. It's also common to fine-tune the resulting representation on a specific task, as the representation built with unsupervised data is generic and not necessarily optimal for a particular task. One drawback is the model's inability to naturally handle missing data, though there are methods to mitigate this issue. Finally, training deep networks

can often be challenging, but progress is being made in the field with techniques such as improved regularization, batch normalization, and adaptive gradient algorithms.

4.3.2 Coordinated Representations

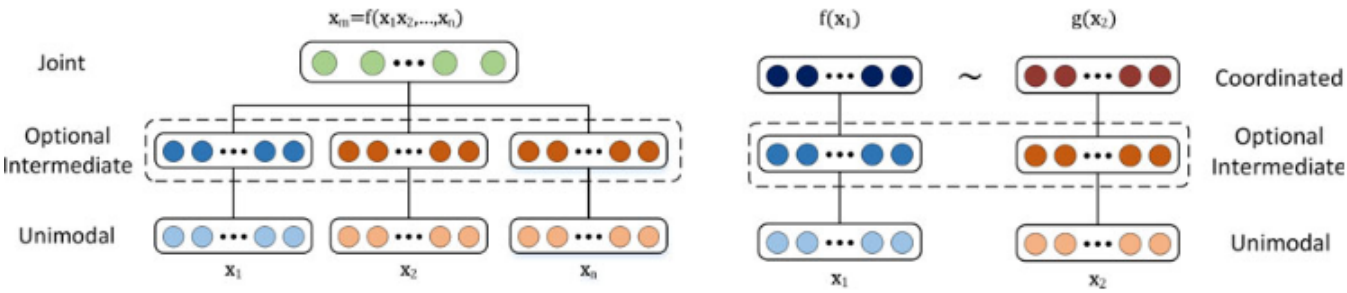
Coordinated representations are an alternative to joint representations. A separate representation for each modality x_1, x_2 is learned through functions f, g (e.g. neural networks) . Subsequently these different representation spaces are coordinated together through some form of constraint resulting a new joint space.

$$f(x_1) \sim g(x_2), \tag{4.3.2}$$

Coordination of these representations is usually achieved via enforcing similarity. Similarity models will minimize the distance between the individual representations in the common space. This is a concept similar to contrastive learning Hadsell et al. (2006). For example the representations of a pool image, the sentence "She dived into the pool" and that of a splash sound are motivated to end up very close together in the coordinated space. In contrast the sentence "The land is very dry" ending up close to the pool image should inflict a "penalty" motivating the coordination function to place them further apart.

An example of the above coordinated representation technique for visual and textual data is the work of Frome et al. (2013) Deep Visual Semantic Embedding (DeViSE) which uses an inner product and a ranking loss function to coordinate the visual and textual representations. Both works aimed at creating a representation "map" for words and images.

The work of Kiros et al. (2014) extended the above technique to coordinate sentences with images by exploiting the inherent effectiveness of LSTM networks to process sequential data such as text sentences.



(a) Joint representation (b) Coordinated representations

Figure 4.3.1: Structure of joint and coordinated representations. Joint representations are projected to the same space using all of the modalities as input. Coordinated representations, on the other hand, exist in their own space, but are coordinated through a similarity (e.g., Euclidean distance) or structure constraint (e.g., partial order). Source (Baltrušaitis et al., 2019)

4.4 Prior work

Many foundational works in multimodal machine learning have eplored the idea of learning one modality (e.g. vision) with supervision from another (e.g. text). One of the earliest works of multimodal machine learning dates as far as 1999, Mori et al. (1999) explored the enhancement of content-based image retrieval through the training of a model that predicts nouns and adjectives in text documents

coupled with images. Later, Quattoni et al. (2007) successfully demonstrated the possibility of achieving more data-efficient image representations by employing manifold learning in the weight space of classifiers trained to predict words in image captions. Srivastava and Salakhutdinov (2012) further investigated deep representation learning by training multimodal Deep Boltzmann Machines using low-level image and text tag features.

While exciting as proofs of concept, natural language supervision for image representation learning is still rare, likely because performance on common benchmarks (e.g. ImageNet) is much lower than alternative approaches in zero-shot settings. More narrowly scoped and precisely targeted uses of weak supervision from text have provided much better results. Building upon this line of research, Joulin et al. (2015), showcased that convolutional neural networks trained to predict words in image captions, from image inputs, acquire valuable image representations.

Later, Mahajan et al. (2018) trained image classifiers to predict ImageNet related hastags on Instagram images. This approach is different from the traditional pipeline of pretraining a convolutional network on a large, manually annotated and highly curated image classification dataset and then fine-tuning the network on a smaller, task-specific dataset. Instead, Mahajan et al. (2018) used data found "in the wild" containing only weak supervisory signals (hastags), possibly error prone, and pretrained their network on this task. With this technique, they were able to gather a dataset orders of magnitudes larger than ImageNet. They pretrained standard convolutional networks on up to 3.5 billion instagram images and then fine-tuned the networks on ImageNet. They were able to outperform all the previous SOTA results by more than 5%. This paved the way for research that relied more on smart techniques that can create massive datasets with weak supervisory signals instead of smaller, highly curated human-annotated ones.

4.5 Self Supervised Multimodal Learning

The fields of multimodal learning and SSL are closely related, with self-supervised learning being a foundational technique for building very large and powerful multimodal models. With self-supervision, researchers have managed to train models on massively larger datasets than was previously possible. Since, results prove to be massively superior compared to those trained on smaller, well-curated human annotated datasets, Self Supervised Multimodal Learning (SSML) research has flourished over the last years. Moreover, self-supervised models are astonishingly better in zero-shot settings, meaning that they can perform tasks they were not explicitly trained for with high precision. If fine-tuned with just a few samples (few-shot learning) these models can even match the performance of supervised models, explicitly trained on a single task.

SSML enables a new era of AI models that already provide value to millions of people and is undoubtedly disrupting many industries like filmmaking, marketing, graphics and gaming design forever.

Some of the most impressive applications of multimodal deep learning, trained via self-supervision, that captured the masses' attention, are the recent text-to-image and text-to-video models like DALL-E 2 (Ramesh et al., 2021) and Make-A-Video (Singer et al., 2022). These models are trained on large datasets of images, videos, and corresponding textual descriptions, using a self-supervised approach to learn representations that capture the relationships between the two modalities. The resulting models can then be used to generate realistic visual content from textual descriptions, demonstrating the power of multimodal deep learning and self-supervised learning.

4.5.1 Prior Work

A milestone multimodal model trained with self-supervision is CLIP (Radford et al., 2021). CLIP did not introduce any novel idea, since (weak) supervision from language is an old idea. However,

they were able to implement their technique on a very vast scale and also used a contrastive loss that allowed the networks to learn useful and generic representations. After constructing a new dataset of 400 million (image, text) pairs collected from a variety of publicly available sources on the Internet they jointly trained an image encoder and a text encoder to predict the correct pairings of a batch of (image, text) training examples.

Let's denote the image encoder as $f(.)$ and the text encoder as $g(.)$. Given an image I and its corresponding text T, the encoders will produce representations $v_I = f(I)$ and $v_T = g(T)$. The similarity between the image and text is then calculated as the dot product of their representations, divided by a temperature parameter τ: $s_{IT} = \frac{v_I \cdot v_T}{\tau}$.

Alayrac et al. (2020) proposed versatile multimodal networks trained using text, video and audio solely using self-supervised training methods. Their results beat the other SSL SOTA methods at the time. The representations learnt could be used in multiple downstream tasks across different modalities. Instead of designing unimodal pretext tasks or relying on contrastive training applied on augmented views of unimodal inputs, they argue that using multiple modalities as different views of the same object is simpler, more natural and potentially more effective.

Their novelty was they tried to learn multiple embedding spaces at once. Instead of processing different modalities on the same joint embedding space, a fine-grained space is used for visual and audio inputs while a coarse-grained space is used for text. Moreover, they abstained from using pretext tasks that combined solely text and audio but instead used the visual modality as their common ground.

Wang et al. (2021) combined raw audio signal, video frame, and log-mel spectrograms (images) in order to train encoders in learning audio representations. They use 3 encoders, 1 for each modality, and a single projector network g to create the embedding space upon which the loss functions are calculated (fig. 4.5.1). Each input modality is augmented before being processed by the encoders. They use a contrastive loss inspired by CPC (van den Oord et al., 2018) to manipulate the embedding space and train their networks. Their final loss is the sum of the pairwise losses

$$L_{tot} = L_{us} + L_{uw} + L_{su} \tag{4.5.1}$$

where u denotes the video modality, w the audio modality, and s the spectrogram modality. It's important to note that each pairwise loss L_{ab} is the sum of the symmetric counterparts $L^{a \to b}$ and $L^{b \to a}$. Their results closed the gap between SSL and supervised techniques on multiple tasks such as speaker identification, keyword spotting, language identification, and music instrument classification. All SSL networks were pre-trained on the AudioSet dataset. Their results suggest that the video modality provides strong supervisory signals to train audio networks even without labels.

The mutual complementarity between different modalities, treated as different views, representing one unique object, is beneficial for the representation learning of each considered modality. Many works rely on audio-visual processing. The correspondence between video and audio frames may be used to create positive and negative pairs, which are later used as inputs of a Siamese network. In this case, each modality of the two can be seen as the supervisory signal for the other.

Arandjelović and Zisserman (2017a,b) used audio-visual correspondence (AVC) as a self-supervised pretext task. Video and audio segments are either taken synchronized from the same video (positive pair) or each one from a different video (negative pair). The AVC task is to decide whether the audio sequence matches the video sequence. They proved that such a pre-text task can help networks learn useful and generic representations both for audio and visual downstream tasks. Pixel Player (Zhao et al.) proposed audio source separation as a pretext task to learn audio-visual representations in a self-supervised manner.

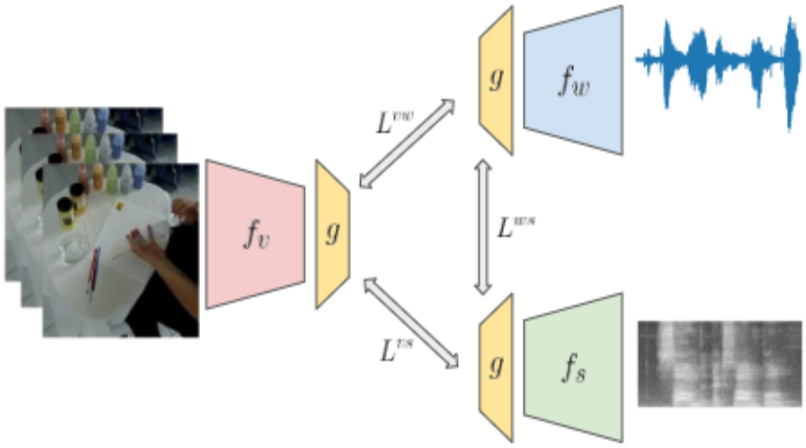


Figure 4.5.1: The multimodal contrastive framework of Wang et al. (2021). Video, audio, and mel-spectograms are encoded by their respective encoders and projected into a joint embedding space by projector g. Three pair-wise contrastive losses are applied to manipulate the joint embedding space.

4.6 Multimodal Research Datasets

Multimodal datasets are collections of data that contain information from multiple sources, such as text, images, video, audio, and sensor data. These datasets are of particular value to researchers in the field of multimodal deep learning. Such public datasets allow the study of problems that lie in the intersection of multiple modalities e.g. vision and language.

Tasks like visual question answering, a skill inherent to humans, is only possible to be researched and advanced if specific datasets are developed that provide an opportunity to explore the relationship between different modalities. Such research has the opportunity to lead to model capable of integrating information from multiple sources.

An example of the above coordinated representation technique is the work of Deep Visual Semantic Embedding (DeViSE)

Common examples of popular multimodal datasets used in research include:

- **VQA**: The Visual Question Answering (VQA) dataset (Antol et al., 2015) is a collection of images with corresponding questions and answers, making it a useful resource for exploring the relationship between vision and language. The dataset contains over 250,000 images, more than 750,000 questions and more than 10 million possible answers.

Figure 4.6.1: Examples of visual question answering from the VQA dataset (Antol et al., 2015).

- **Flickr30k Entities**: The Flickr30k entities dataset (Plummer et al., 2016) is a collection of over 31,000 images with 158,000 captions and 276,000 bounding boxes linked to the mentions of entities in the captions, making it a useful resource for tasks such as image captioning, localization

of textual entities in images, language understanding and multimodal retrieval. The captions are annotated with reference to specific regions in the images, allowing researchers to explore the relationship between language and visual attention.

Figure 4.6.2: Examples from the multimodal flickr30 entities dataset (Plummer et al., 2016). Multiple captions describe images in detail. Textual entities are localized in the image via bounding boxes

- **MSR-VTT**: The Microsoft Research Video-to-Text (MSR-VTT) dataset (Xu et al., 2016) is a large-scale video captioning dataset that contains over 10,000 videos and more than 200,000 captions. The videos cover a wide range of topics, from sports and cooking to news and documentaries. The novelty of this dataset lies in its complexity, something previous datasets lack. Each video clip is annotated with about 20 natural language sentences by 1327 amazon turk annotators, offering a detailed description of what takes place in the video.
- **MusicCaps** : The MusicCaps dataset (Agostinelli et al., 2023) contains 5,521 music examples, each of which is labeled with an English aspect list and a free text caption written by musicians. An aspect list is for example, "pop, tinny wide hi hats, mellow piano melody, high pitched female vocal melody, sustained pulsating synth lead". The caption consists of multiple sentences about the music, e.g.,

  > "A low sounding male voice is rapping over a fast paced drums playing a reggaeton beat along with a bass. Something like a guitar is playing the melody along. This recording is of poor audio-quality. In the background a laughter can be noticed. This song may be playing in a bar." The text is solely focused on describing how the music sounds, not the metadata like the artist name.

This is a non-exhaustive list of the many multimodal datasets that are available for research. By leveraging these datasets and developing new models and algorithms, researchers can work towards creating more advanced AI systems capable of integrating information from multiple modalities, imitating human intelligence, and performing complex tasks in real-world settings.

Chapter 5

Multimodal Emotion Recognition and Sentiment Analysis

5.1 Introduction

Multimodal human Emotion Recognition and Sentiment Analysis play a pivotal role in various domains, including customer service, healthcare, human resources and education among other. The advancements in Multimodal Deep Learning techniques have significantly enhanced the accuracy and effectiveness of emotion recognition systems. By leveraging information from multiple modalities such as video, audio, and text, these systems gain access to a rich set of cues that collectively provide a more comprehensive understanding of a person's emotional state compared to using a single modality.

The combination of different modalities allows for a holistic assessment of emotions, leading to more accurate predictions and a deeper understanding of individuals' emotional experiences. For example, while visual information from videos captures facial expressions, body language, and gestures, audio data captures vocal tone, pitch, and intensity. Simultaneously, text data can provide additional contextual information through the analysis of written language. By combining these modalities, the system gains access to a broader range of cues leading to improved performance and a nuanced understanding of human emotions.

The utilization of multimodal inputs in human Emotion Recognition and Sentiment Analysis has emerged as a significant advancement in the field. In this section, we delve into this fascinating domain, providing an overview of the current state of the art to enlighten and inform the reader. First, we make a thorough mention of the CMU-MOSEI dataset, one of the most prominent datasets in the field of multimodal emotion recognition. This dataset has been widely utilized by previous studies, and we specifically direct our research efforts toward this task. Then we provide a comprehensive overview of the current state of the art, shedding light on the latest advancements and trends in the field. Finally, we present the current state-of-the-art results, of various previous works, on the CMU-MOSEI task.

5.2 CMU-MOSEI Dataset

The CMU Multimodal Opinion Sentiment and Emotion Intensity (CMU-MOSEI) dataset (Zadeh et al., 2018b) is a publicly available multimodal dataset that has become a standard benchmark in multimodal emotion recognition. CMU-MOSEI consists of the three main modalities, video, audio, and text. All samples are corresponded by sentiment and **emotion intensity annotations**. This dataset was developed by researchers at Carnegie Mellon University (CMU) to facilitate research on sentiment and emotion analysis in multimodal data.

The dataset is designed to capture a wide range of emotions and sentiments expressed in naturalistic contexts. It includes 23,500 utterance videos from 1000 different YouTube speakers, with a total duration of over 65 hours, covering over 250 topics such as movie reviews, TED talks, and vlogs. This diverse set of speakers, topics, annotations, and samples allows for generalizable studies of human multimodal language

Figure 5.2.1: The diversity of topics of videos in CMU-MOSEI, displayed as a word cloud. Larger words indicate more videos from that topic. The 5 most frequent topics are: reviews (16.2%), debate (2.9%), consulting (1.8%), financial (1.8%) and speech (1.6%). The remaining topics are almost uniformly distributed at around 0.5%-1.5% each. Source Zadeh et al. (2018b)

Emotions are defined as the speaker's expression of state of mind and feeling while uttering the sentence. Sentiment is defined as the speaker's attitude towards the topic of his/her discussion. The annotators were asked to annotate sentiment on a seven-step Likert scale of [-3: highly negative, -2: negative, -1: weakly negative, 0: neutral, 1: weakly positive, 2: positive, 3: highly positive]. The Emotions selected are the six basic Ekman emotions (Ekman et al., 1992) of happiness, sadness, anger, fear, disgust, surprise. Each of the emotions is annotated at a four-step Likert scale for the presence of an emotion x: [0: no evidence of x, 1: weakly x, 2: x, 3: highly x].

The dataset can be accessed in multiple forms. The dataset may be used in its raw format, i.e. video, text transcripts, and audio waves. However, for the facilitation of researchers, the dataset comes in both aligned and unaligned formats, and it's also possible to use a version of the dataset that comes with pre-extracted features (embeddings) instead of the raw data.

All videos come with manually annotated transcripts. Glove word embeddings (Pennington et al., 2014) were used to extract word vectors from transcripts. Words and audio are aligned at phoneme level using P2FA forced alignment model (Yuan and Liberman, 2008). Following this, the visual and acoustic modalities are aligned to the words by interpolation. Since the utterance duration of words in English is usually short, this interpolation does not lead to substantial information loss

In the visual modality, a specific approach for feature extraction is followed. First, frames are extracted from the full videos at a frequency of 30Hz. To locate the facial region of interest, they utilized the MTCNN face detection algorithm (Zhang et al., 2016a). Next, the Facial Action Coding System (FACS) was used to extract facial action units (Ekman et al., 1992). These action units enable accurate tracking and understanding of facial expressions (Baltrušaitis et al., 2016). Furthermore, the authors utilized the Emotient FACET software (iMotion, 2017) to extract a set of six basic emotions solely

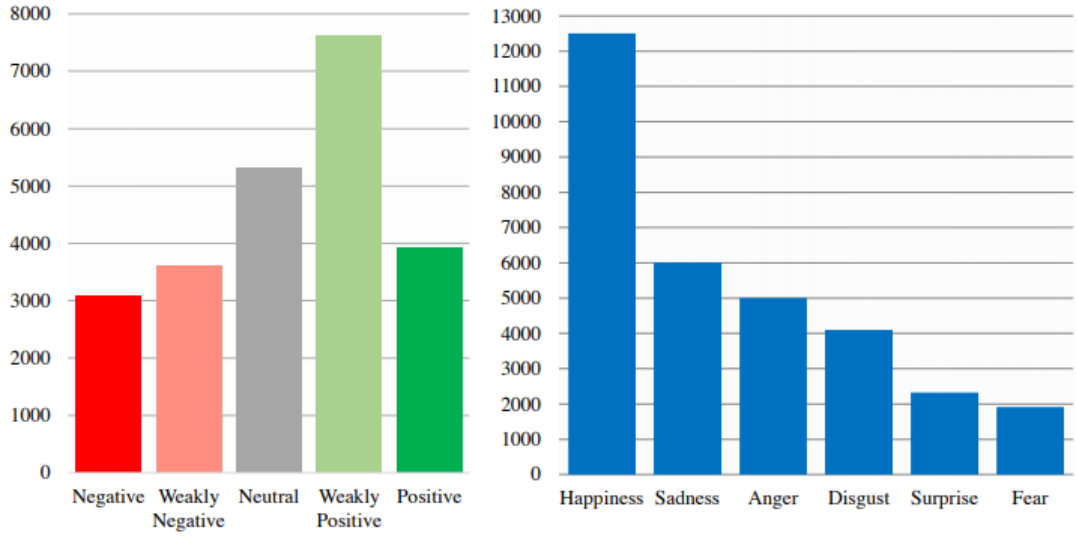


Figure 5.2.2: Distribution of sentiment and emotions in the CMU-MOSEI mega corpus. The distribution shows a natural skew towards more frequently used emotions. However, the least frequent emotion, fear, still has 1,900 data points which is an acceptable number for machine learning studies. Source

from static facial images. Finally, multiple face embeddings were obtained from well-established facial recognition models such as DeepFace (Taigman et al., 2014), FaceNet (Schroff et al., 2015), and SphereFace (Liu et al., 2018).

For the acoustic modality, the COVAREP software (Degottex et al., 2014b) was utilized to extract a range of acoustic features. These features include 12 Mel-frequency cepstral coefficients, pitch information, and multiple other features (Zadeh et al., 2018b). All of these extracted features are specifically related to emotions and the tone of speech.

During our experiments we used the already aligned version of the dataset. The data comes in the form of sequences and extracted features (embeddings) rather than raw video or audio signals. We used Glove embeddings for the text modality, FACET visual embeddings for the visual modality, and COVAREP embeddings for the audio modality.

Figures 5.2.2 and 5.2.1 provide a visual presentation of the distribution of emotions in the samples as well as the frequency of each topic discussed in the video samples. The dataset is considered well representative with a lot of data diversity, both in terms of emotions and discussion topics.

5.3 Prior Work

As already discussed, CMU-MOSI and CMU-MOSEI are the most prominent datasets on emotion recognition. Another notable dataset is IEMOCAP (Busso et al., 2008). Most prior work related to multimodal emotion recognition and sentiment analysis evaluated their performance on these datasets. We focus our attention on works that utilize these datasets.

One of the earliest works that attempted the MOSEI challenge is MFN (Zadeh et al., 2018a)(same authors). Their work challenged the problem of Multi-view sequential learning, which focuses on how to leverage multiple perspectives or representations of the same data for efficient learning. Their proposal was a network with 3 components, each serving a different purpose (figure 5.3.1). The first is the system of LSTMs, where each view (or modality) is assigned to an LSTM function to learn its intra dependencies. Second, the Delta Memory Attention Network (DMAN) is tasked with discovering interactions and dependencies across the memories of the LSTM system of the first stage. The goal is to identify cross-view interactions in sequences. Finally, a Multi-view Gated Memory updates its con-

tents based on the outputs of DMAN and the previous inputs (memory). This serves the purpose of identifying interactions and relationships across multiple timesteps. This multistep pipeline of encoding unimodal sequences, then identifying cross-modal interactions and finally merging them across time inspired many subsequent works that iterated, extended, and improved upon similar ideas.

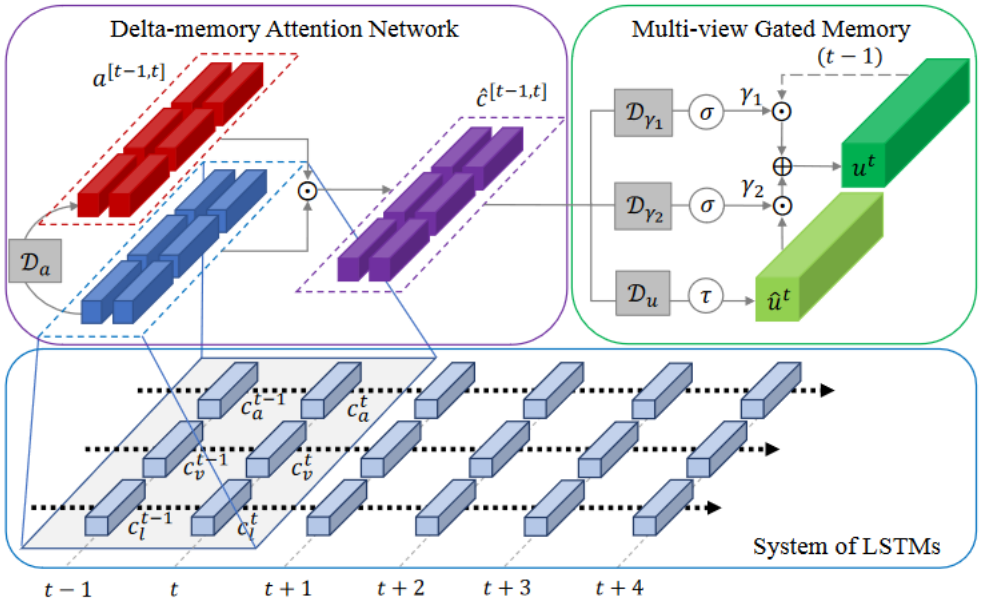


Figure 5.3.1: Overview figure of Memory Fusion Network (MFN) pipeline. Each LSTM encodes information from one view such as language (l), video (v) or audio (a). Source Zadeh et al. (2018b)

Speaker intentions often vary dynamically depending on different nonverbal contexts, such as vocal patterns and facial expressions.Wang et al. (2018) attempted to model the variation in word representations by learning multimodal-shifted word representations with their Recurrent Attended Varioation Embedding Network (RAVEN).

Pham et al. (2020) experimented with learning joint multimodal representations through cycle-consistency cross modal mappings by translating between modalities. The method is based on the key insight that translation from a source modality S to a target modality T results in an intermediate representation that captures joint information between modalities S and T.

Tsai et al. (2019) observed that multimodal "language" sequences often exhibit "unaligned" nature and require inferring long-term dependencies across modalities. Hence, the usual word-level alignment of modalities may not be optimal. They proposed Multimodal Transformer (MulT), an end-to-end model that extended the standard Transformer network (Vaswani et al., 2017a) to learn representations directly from unaligned multimodal streams. Prior work in multimodal language analysis focused on early fusion, hierarchical attention strategies, and cross-modal autoencoders. However, these approaches assumed aligned multimodal sequences at the word level and mostly considered short-term multimodal interactions. In contrast, MulT does not rely on alignment assumptions and defines cross-modal interactions at the sequence level. The architecture is capable of learning representations from both aligned and unaligned multimodal streams, beating all previous state-of-the-art.

Unlike the traditional transformer architecture used in neural machine translation, MulT does not have an encoder-decoder structure. Instead, it directly attends to low-level features of different modalities. First, to ensure each element of the input sequences is sufficiently aware of its neighborhood elements, the input sequences are passed through a 1D temporal convolutional layer denoted as

$$X_{L,V,A} = \text{Conv1D}(X_{L,V,A}, k_{L,V,A}) \in \mathbb{R}^{T_{L,V,A} \times d} \tag{5.3.1}$$

where kL,V,A are the sizes of the convolutional kernels for modalities L, V, A, and d is a common dimension. After passing the convolved inputs through positional embedding (Vaswani et al., 2017b), the sequences are fed to crossmodal transformers. Based on the crossmodal attention blocks, the designed crossmodal transformer enables one modality to receive information from another. Assuming two modalities α and β, attempting to fuse information from β to α through crossmodal attention will formally be defined as follows (Tsai et al., 2019).

We define

- Queries as $Q_\alpha = X_\alpha W_{Q_\alpha}$,
- Keys as $K_\beta = X_\beta W_{K_\beta}$,
- Values as $V_\beta = X_\beta W_{V_\beta}$,

where

- $W_{Q_\alpha} \in \mathbb{R}^{d_\alpha \times d_k}$,
- $W_{K_\beta} \in \mathbb{R}^{d_\beta \times d_k}$,
- $W_{V_\beta} \in \mathbb{R}^{d_\beta \times d_v}$

are weight matrices. The latent adaptation from β to α is presented as the crossmodal attention $Y_\alpha := CM_{\beta\to\alpha}(X_\alpha, X_\beta) \in \mathbb{R}^{T_\alpha \times d_v}$:

$$Y_\alpha = CM_{\beta\to\alpha}(X_\alpha, X_\beta) = \text{softmax}\left(\frac{Q_\alpha K_\beta^\top}{\sqrt{d_k}}\right)V_\beta = \text{softmax}\left(\frac{X_\alpha W_{Q_\alpha} W_{K_\beta}^\top X_\beta}{\sqrt{d_k}}\right)X_\beta W_{V_\beta} \tag{5.3.2}$$

In a similar work, Delbrouck et al. (2020) (TBJE) also implemented a multimodal network based on Transformers. Three unimodal transformers process the unimodal inputs. The inputs are not raw data, but rather use Glove embeddings for text, mel-spectrograms for audio and a pre-trained 3D CNN on Sports-1M and Kinetics datasets was used to extract embeddings for the video input. Fusion of modalities is achieved by using cross-modal transformers. In another work, Siriwardhana et al. (2020) proposed a similar transformer based architecture, similar to Delbrouck et al. (2020) but used input embeddings created from independently SOTA pre-trained SSL models (SSE-FT). They used Wav2Vec inputs for audio, RoBERTa embeddings for text (Liu et al., 2019) and Fabnet embeddings for video input. Due to their complex SOTA input embeddings, they achieve much higher performance across metrics.

Wang et al. (2019) showed that multimodal inputs might sometimes degrade performance, compared to unimodal, due to the dominance of one modality over the others. Georgiou et al. (2021) proposed a modality masking mechanism, M^3, that attempts to solve the problem of "dominant modality". During training, the embeddings of one or more modalities are masked to zero with a probability p, effectively not contributing to the fused output vector and final prediction. This enforces the network to depend on the remaining modalities for its predictions, hence introducing a barrier to overdependency on a single modality. This masking mechanism was shown to significantly increase performance compared to the baseline. In their work, Georgiou et al. (2021) designed an LSTM based architecture that utilizes both self-attention and cross attention. Figure 6.4.1 provides a visual overview of the architecture. This is a two stage architecture. In the first stage, a unimodal encoder based on Bi-LSTMs and self attention identifies intra dependencies for each unimodal sequence. Then, in the second stage, four cross attention blocks are utilized to discover inter dependecies between all modalities. A final Bi-LSTM is used for the final fusion and the output is given to a linear classifier. The architecture surpassed or achieved comparable performance across all metrics with previous SOTA on the CMU-MOSEI task.

Paraskevopoulos et al. (2022) (MM-Latch) proposed another multimodal architecture based on LSTMs and attention mechanisms that also incorporates masking input modalities. The novelty lies in an early "fusion" mechanism that attempts to mask unimodal inputs with masks derived from high-level representations of the other modalities (weighted sum), before continuing with the main fusion mechanism. This top-down interaction offers an alternative to the usual bottom-up approach. Their model uses pre-trained embeddings (Glove, FACET visual, COVAREP audio). The work achieves comparable or better performance to previous SOTA.

Finally, a work that we build upon and describe thoroughly in another chapter is SeqAug (Georgiou and Potamianos, 2023). The work presented an augmentation method suitable for multimodal architectures. They tested the augmentation technique on both RNN based and Transformer based architectures and discovered that in both cases the augmentation provided a boost in performance. Moreover, the augmentation is designed with embedding input features in mind, making it an excellent choice for datasets that come in the embedding form.

5.4 Previous SOTA

As discussed, previous works on multimodal sentiment analysis on the cmu-mosei dataset have experimented with many diverse architectures ranging from LSTM to Transformers and attention mechanisms. Though similar in many aspects, the details of how fusion is achieved make these works different. Tables 5.4.1, 5.4.2 provide an overview of the previous state-of-the-art results on the cmu-mosei dataset. Not all works present their results as the average over multiple runs so two separate tables are used to present the results. Table 5.4.1 presents the average values only for the papers that made them available, while table 5.4.2 presents the best models for all the previous works. Results are adopted from the original papers.

Model	**Text-Embeddings**	**F1** (%)	**Acc2**	**Acc7**	**Cor**	**MAE**
M3	Glove	82.64 ±0.5	82.46 ±0.5	51.78 ±0.6	0.700 ±0.004	0.586 ±0.004
MM-Latch	Glove	82.5 ±0.3	82.4 ±0.3	52. ±0.2	0.700 ±0.002	0.585 ±0.002
SeqAug (RNN)	Glove	-	82.48 ±0.38	51.90 ±0.49	-	0.585 ±0.38

Table 5.4.1: Performance on CMU-MOSEI as the average over 5 runs. All results are adopted from the original papers. Metrics that weren't reported are filled with '-'.

Model	**Text-Embeddings**	**F1** (%)	**Acc2**	**Acc7**	**Cor**	**MAE**
RAVEN	Glove	79.5	79.1	50.0	0.662	0.614
MCTN	Glove	80.6	79.8	49.6	0.670	0.609
TBJE	Glove	-	82.4	45.5	-	-
MuLT	Glove	82.3	82.5	51.8	0.703	**0.580**
M3	Glove	**82.92**	82.5	51.9	0.700	0.586
MM-Latch	Glove	82.9	**82.8**	**52.1**	**0.704**	0.582
SSE-FT	RoBERTa	87.0	87.3	55.7	0.792	0.529

Table 5.4.2: Performance on CMU-MOSEI. All results are adopted from the original papers. Metrics that weren't reported are filled with '-'.

Chapter 6

Our Proposal

6.1 Introduction

In this section, we thoroughly describe our proposal. We explain all the important details, what makes it innovative, and highlight its distinctions from the prior work discussed over previous sections.

Our proposal focuses on utilizing Self Supervised Learning techniques to train a Multimodal Network. Unlike previous approaches, we aim to simplify the integration of self-supervision into multimodal networks. Our method does not rely on intricate pretext tasks or multiple siamese networks. Instead, we employ a single "siamese network" structure that processes augmented multimodal inputs. Subsequently, we employ joint representation learning to establish a shared embedding space from which models can acquire valuable insights.

Our method fall under the category of contrastive learning self-supervised techniques and does not necessitate negative samples, as is often the case with many other contrastive learning methods. We process inputs that are already in embedding form, but the framework can also readily accommodate raw data.

Firstly, we will provide a high-level overview of our proposed framework, elucidating crucial mathematical equations such as the loss function. Secondly, we will delve into a detailed discussion of the underlying architecture. Subsequently, we will present pertinent information concerning the augmentation techniques we incorporate into our framework, an integral part of its operation. Lastly, we will explore potential variations and extensions of our proposed framework.

6.2 The Framework

Our framework is heavily inspired and builds upon the Barlow Twins framework, which uses a siamese network structure and processes different views of the same input. The original work was limited to the visual modality, but we have modified the framework to make it suitable for multimodal applications.

Our framework, presented in figure 6.2.1, comprises 3 modules: a multimodal augmentation (denoted as MM-AUG), a multimodal encoder architecture E, and a projector network. The Barlow Twins loss function (Zbontar et al., 2021) is applied on the projector's output. **The central idea behind this framework** is that the augmentation module generates two additional views of the input. The encoders then encode the multimodal information, which is subsequently projected onto a shared embedding space. The loss function operates on this joint embedding space, and through error backpropagation, the encoders E acquire valuable and accurate representations, enabling them to effectively address the given task.

Our study focused on processing three modalities, visual, textual, and audio. However, processing

more or fewer modalities is equally easy to perform. We will now provide a short overview of each module.

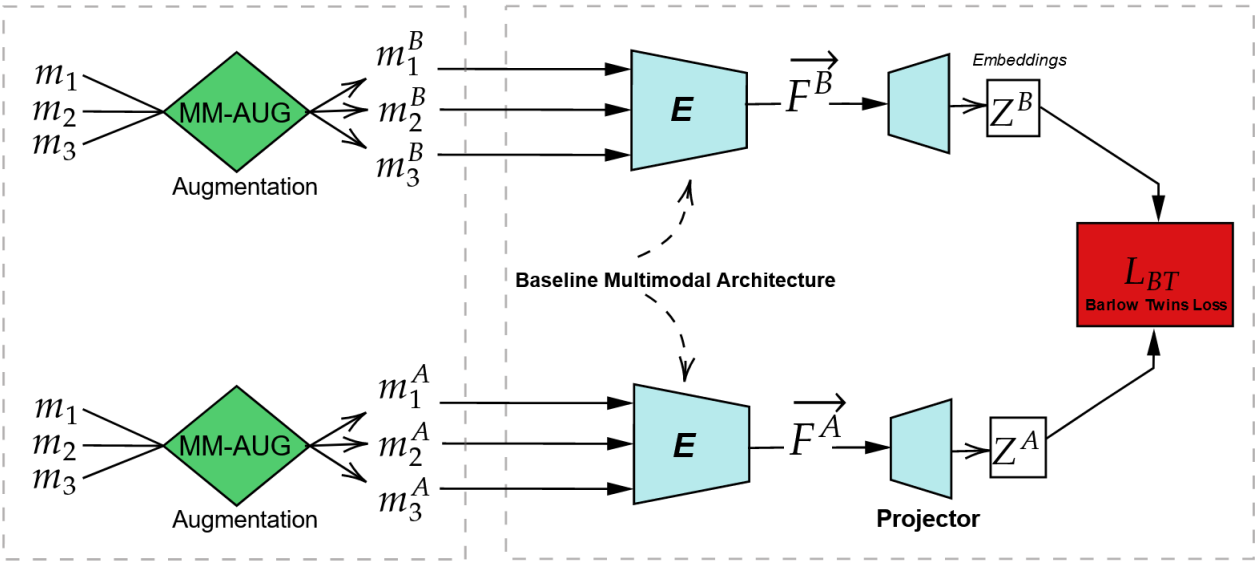


Figure 6.2.1: Proposed Architecture. A feature augmentation method creates two new views A, B for every modality m_i. Each triplet pair is processed by the baseline multimodal architecture and extracts the fused features $\vec{F}$ which are projected to a single joint embedding space. Finally, the Barlow Twins loss $\mathscr{L}_{BT}$ is applied to the embeddings.

Multimodal Augmentation

The original input comes in the form of three vectors, one for each modality m_i. The augmentation module is responsible for altering the original multimodal input in a way that two different views of the same object are created. Transformations T^A and T^B are applied on the input m_i creating two outputs m_i^A and m_i^B. These are slightly or heavily modified versions of m_i, depending on the transformation imposed on them.

The feature augmentation techniques are applied separately on each modality. The augmentation module accepts M inputs and produces $2*M$ outputs, one pair (m_i^A, m_i^B) for each input modality vector. The triplets are provided in batches of size B, hence the generated outputs are $2 * B$ triplets.

In our study, we focus on augmentations that are suitable for inputs that come in the form of embeddings. Specifically, we experimented with three augmentation techniques, namely Gaussian noise, masking, and SeqAug (Georgiou and Potamianos, 2023) a novel feature augmentation method suitable for multimodal networks. Each transformation is further explained in section 6.5 and their implementation details are explained in section 7.2.

Encoders - Baseline Architecture

The encoder module E is a multimodal network that encodes the given inputs. Despite us focusing on sequential inputs and working with LSTM based encoders, the choice of encoder is not restricted by the framework. We will later present our encoding architecture in detail.

Projector

The projectors are stacked fully connected layers. After receiving the encoded information they project it into a fixed-size dimensional space. This projected space is what we call the embedding space. The

loss function is directly applied to this space. The size of the projectors can be tuned as a hyperparameter.

The projectors are useful during backpropagation but are discarded after training is complete. We refer to the **projectors' outputs as embeddings** and denote them by Z^A, Z^B. The outputs are collected in batches. We represented them with matrices Z_b^A,Z_b^Aas:

$$Z_b^A = g(E(aug_A(m_1^b, m_2^b, m_3^b))) \tag{6.2.1}$$

$$Z_b^B = g(E(aug_B(m_1^b, m_2^b, m_3^b))) \tag{6.2.2}$$

where $aug_A(.), aug_B(.)$ denote different augmentation operations, E denotes the multimodal encoding architecture, $g(.)$ is the projector, and m_j^b denotes input of modality j for a batch sample b.

6.3 Barlow Twins Loss Function $\mathscr{L}_{BT}$

The loss function is an integral part of the framework. The computation of the loss function necessitates the computation of the cross-correlation matrix between embedding matrices Z_b^A and Z_b^B. The loss function is applied on the cross-correlation matrix and cannot be defined for single outputs.

Our loss function aims at learning a single joint embedding space for all 3 modalities. Formally, we aim at minimizing the loss function $\mathscr{L}_{BT}$

$$\mathscr{L}_{BT} = \sum_{i=1}^{N}(1 - C_{ii})^2 + \lambda \sum_{i=1}^{N}\sum_{j\neq 1}^{N}(C_{ij})^2 \tag{6.3.1}$$

where matrix C is the cross-correlation matrix and its elements are calculated as

$$C_{ij} = \frac{\sum_{b=1}^{N} Z_{bi}^A Z_{bj}^B}{\sqrt{\sum_{b=1}^{N}(Z_{bi}^A)^2}\sqrt{\sum_{b=1}^{N}(Z_{bi}^B)^2}} \tag{6.3.2}$$

where N denotes the batch size and Z^A, Z^B are matrices that contain the embeddings, i.e. the projected in the joint embedding space fused output vectors $\vec{Z_b^K}$.

The loss function $\mathscr{L}_{BT}$ penalizes the network depending on the squared difference between the cross-correlation matrix **C** and the identical matrix **I**. It serves two goals:

1. It pulls the embeddings of the two augmented views close together in the joint space. The first term of equation 6.3.1 is called the *invariance term* and helps the network learn transformation invariant features. C_{ii} denotes the diagonal elements of matrix **C** which are pushed to equal 1. In other words, the network must understand that the two augmented views originate from the same object, hence they must be placed close together in the embedding space.
2. It forces the network to discard redundant information that is already encoded in its weights. The second term, is the *redundancy term*. It penalizes non-diagonal elements C_{ij} by their square value, forcing them to zero. λ is a parameter that controls the influence of the redundancy reduction term. In other words, the second term helps decorrelate the embeddings' dimensions, hence reducing redundant information encoded in them.

Equation 6.3.1 does not include any negative sample terms. Negative examples are not necessary for successfully learning representations, in contrast to other contrastive loss functions.

6.4 Baseline Architecture

The choice of the baseline architecture of encoders E is not restricted by the framework. However, we require an architecture suitable for sequential data. We decided to build on top of the multimodal LSTM-based architecture proposed in M3 (Georgiou et al., 2021) due to its SOTA performance.

The baseline multimodal architecture comprises three building blocks (fig. 6.4.1). The first is a Bidirectional LSTM block **which we convert into one-directional LSTM block**. The second is the scaled dot product attention mechanism (Self Attention) Vaswani et al. (2017b), and the third is a cross-modal attention module (Cross Attention) inspired by the one proposed in Lu et al. (2019) (ViLBERT).

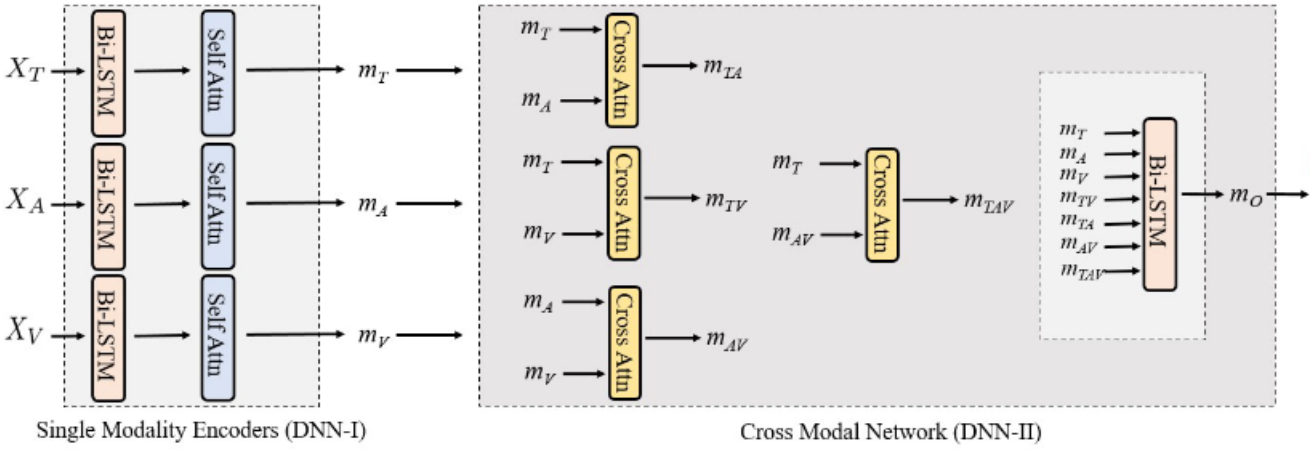


Figure 6.4.1: Baseline Multimodal Architecture. The source is Georgiou et al. (2021) original paper.

The architecture is composed of two major sub-networks as depicted in 6.4.1. These are the Single Modality Encoders (DNN-I) and the Cross Modal Network (DNN-II). DNN-I processes the unimodal inputs while DNN-II extracts fused representations that can be used for prediction.

The Single Modality Encoder processes the input features X_A, X_T, X_V for each modality and uses three Bi-LSTMs (one directional in our case) for encoding, each pertaining to a modality. The hidden states are then reweighted through Self Attention blocks, resulting in the unimodal encodings m_T, m_A, m_V.

The Cross Modal Network captures cross-modal interactions. It consists of four Cross Attention modules and a Bi-LSTM (one directional in our case) for the final fusion. This network receives single modality representations as input and employs symmetric cross attention to capture interactions between T, A, and V. It produces 4 pairs m_{TA}, m_{TV}, m_{AV} capturing all symmetric combinations of m_T, m_A, m_V, and also m_{TAV} which fuses all 3 modalities together. **The symmetric attention** is defined by summing the two assymetrical cross modal attentions as

$$m_{kl} = \alpha_{kl} + \alpha_{lk} \tag{6.4.1}$$

Finally, all symetric attention pairs along with the self attented m_T, m_A, m_V are concatanated into $\tilde{m}$ and fed into the final LSTM network which produces the output m_o, later fed into our projector networks.

$$\tilde{m} = m_T^k \,||\, m_A^k \,||\, m_V^k \,||\, m_{TAV}^k \,||\, m_{AV}^k \,||\, m_{TV}^k \,||\, m_{TA} \tag{6.4.2}$$

Paraphrasing from the original paper (Georgiou et al., 2021), we now discuss the symmetric cross-modal attention mechanism employed. Let's take $m_k, m_l \in \mathbb{R}^{B \times N \times d}$, where $k \neq l$ different modalities. Given the input modality representations, we can create

- keys $K_l = W_K^k \, m_l$
- queries $Q_k = W_Q^k \, m_k$
- values $V_l = W_V^k \, m_l$,

which are learnable owing to their respective projection matrices $W_{K,Q,V_{k,l}}$.
The cross-modal attention layer can now be defined as:

$$a_{kl} = s\left(\frac{K_l^T Q_k}{\sqrt{d}}\right) V_l + m_k, \tag{6.4.3}$$

where $s(.)$ denotes the softmax function and B, N, d are the batch size, sequence length, and hidden size respectively.

6.5 Augmentations

In order for the framework to successfully learn useful representations, suitable feature augmentation is required. In our study we work directly with extracted features instead of raw data representations. Hence, we are limited to feature augmentation methods suitable for extracted features. In our study we experiment with Gaussian Noise, Masking, and SeqAug. Further implementation details are provided in section 7.2, but a short theoretical presentation of the augmentation techniques takes place here.

6.5.1 SeqAug

At a high level, SeqAug is a modality agnostic data augmentation method that operates on sequential data. It generates new training examples by randomly permuting the time steps of existing examples. This is done in two steps: first, a subset of feature dimensions are randomly selected from the set of all possible features. Then, a permutation on the temporal axis takes place replacing original values with existing features of the same sequence (fig. 6.5.2). The chosen feature dimensions are assumed to be drawn from an uknown feature distribution (fig. 6.5.1).

The idea behind this approach is that it creates new views of the same underlying data without affecting the underlying semantics since resampling occurs within the sequence itself. Additionally, because SeqAug only requires permuting existing data rather than generating entirely new data points, it can be computationally efficient and reduce the need for large amounts of training data.

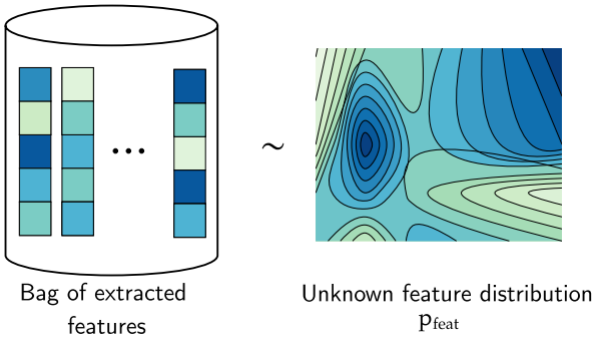
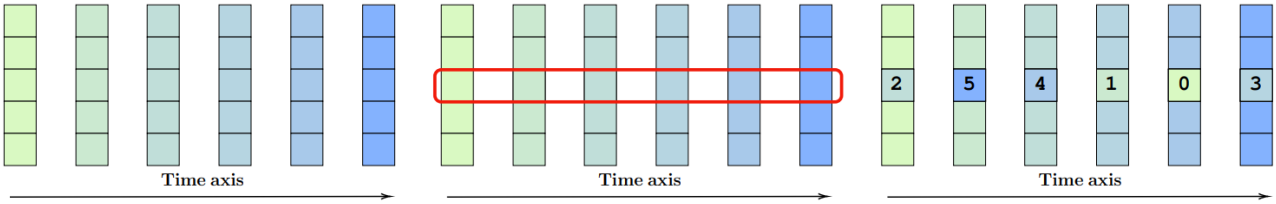


Figure 6.5.1: The bag of all extracted features. It's assumed that sampling the bag of features approximates the unknown feature disrtibution p_{feat}. Source (Georgiou and Potamianos, 2023)

Let $x_{t=0}^T$ be a sequence of input features with length T, where x_t lies in $\mathbb{R}^{\mathbb{d}}$ and d is the feature dimensionality. We define $\mathbb{D}$ as the set of all possible feature addresses in x_t.

In Step 1 of SeqAug, we randomly select a subset S of feature addresses from $\mathbb{D}$. These addresses are preserved across all timesteps within the sequence. In Step 2, we perform a random permutation π on the ordered time indices 0, 1, ..., T-1. The feature addresses sampled in Step 1 are permuted along the time-axis according to pi. The remaining feature addresses not in S remain unchanged.



(a) *A sequence of features. Different colors indicate different timesteps. Time axis arrow shows time progression.*

(b) ***Feature Address Sampling**: We randomly pick some feature dimensions across all timesteps (red rectangle).*

(c) ***Time Resampling**: We resample each feature value from the sequence itself. Can be modeled as permutation.*

Figure 6.5.2: Multimodal feature augmentation (SeqAug) technique proposed in SeqAug. The first step (b) involves randomly selecting some feature dimensions. The second step (c) involves a temporal permutation of the feature dimensions selected during step one. Source (Georgiou and Potamianos, 2023)

6.5.2 Masking

In the context of deep learning, a mask is typically a sequence of Boolean values (0s and 1s) that indicate whether a particular input element should be considered or ignored. For example, in a sentence of 10 words, if the sentence is shortened to 7 words, the mask for the shortened sentence may look like this: [1, 1, 1, 1, 1, 1, 1, 0, 0, 0], where 1s represent active words and 0s represent padded or inactive words.

The mask can be generated by sampling from a Bernoulli distribution. The Bernoulli distribution is a discrete distribution that takes value 1 with probability p and value 0 with probability $1 - p$.

$$M_i \sim \text{Bernoulli}(p) \tag{6.5.1}$$

Once a mask is sampled it is applied by element-wise multiplication.

$$X' = X \odot M \tag{6.5.2}$$

where X' is the transformed feature vector, X is the original feature vector and M is the sampled mask. Masking can be performed either in the temporal dimensions inactivating whole timesteps (Time Masking), or in the feature dimension inactivating only features (Feature Masking). We experimented with both options, but decided to continue with Time Masking since it provided slightly better results.

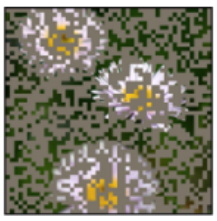


Figure 6.5.3: A masking example applied in an image. A random mask is sampled from a Bernouli distribution and is multiplied element-wise with the original image. Source (Xie et al., 2022)

6.5.3 Gausian Noise

Gaussian noise feature transformation is a data augmentation technique where random Gaussian (or normal) noise is added to the original data. The noise helps to generate new training examples and thereby possibly allowing the model to learn more generalizable patterns in the data.

The Gaussian noise is defined by two parameters: mean (typically 0) and standard deviation. The noise is generated by sampling from a Gaussian distribution with the specified mean and standard deviation, and then added to the original data. This process is often applied to each data point individually, and can be written as:

$$N \sim \mathcal{N}(\mu, \sigma^2)$$

$$X' = X + N$$

6.6 Variations

The first variation we propose is on an architectural level. A baseline architecture based on Transformers instead of RNNs can be employed. MuLT (Tsai et al., 2019) is a suitable multimodal architecture that achieves sota results on the cmu-mosei. Training MuLT under our framework can possibly results in performance improvements, similarly to the LSTM based architecture we tested.

The second variation can be on the framework level. The heterogeneous nature of input modalities may undermine performance. Visual and audio modalities are represented as signals and are fine-grained. On the other hand, language is symbolic and coarse-grained. Inspired by Alayrac et al. (2020) we can learn two different embedding spaces a fine-grained and a coarse-grained. Vision and audio will be compared in the fine-grained space (*Sva*), while text will be jointly compared with vision and audio in the lower dimensional coarse-grained space (*Svat*).

However, such a modification would require including a second siamese-network module. The newly added module will be responsible for learning the bimodal fine-grained embedding space for audio and video. A projector network g' of higher dimension is required to learn the fine-grained space. Moreover, a third projector network that $g_{av->avt}$ that projects the visual and audio modalities from the fine-grained to the coarse-grained space is also necessary and must be learned. Two loss functions will be computed, one for the trimodal space and one for the bimodal.

Another option is to learn 3 disjoint bimodal spaces, one for every pair of modalities. The best multimodal representation approach for our case can only be discovered by experimenting with different embedding spaces and modifying our architecture accordingly.

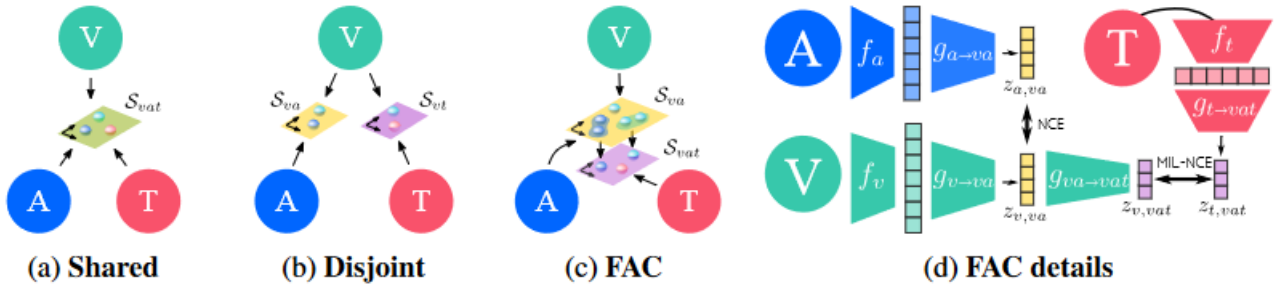


Figure 6.6.1: Different approaches to multimodal representation. Shared (joint) spaces project all modalities in a single space. Disjoint representation uses different embedding spaces for some/each modality. Fine and Coarse FAC space uses two disjoint spaces but imposes a restriction between them (e.g. representation of fine-grained modalities in the coarse-grained space must be similar to the fine-grained). Source (Alayrac et al., 2020)

Chapter 7

Implementation

This chapter describes all the technical details required for understanding the proposed methodology on a deeper level and reproducing it. First, we lay out the setting and vocabulary, that we will use in the 'Results', chapter by elaborating on our 'Training Methodologies'. Then, specific technical details are provided for understanding a key ingredient in our proposal - augmentations. Finally, we provide an overview of all the parameters, hyperparameters, software, and hardware configurations necessary to effectively reproduce this study.

7.1 Training Methodologies

Throughout the experiments, four distinct training methodologies were employed, each with the objective of providing a fair and unbiased evaluation of the model's performance. It is crucial to describe the differences between these methodologies in detail. The following section will outline the step-by-step process for each of the four methodologies.

Fully Supervised Training

During fully supervised training (FS) we train the network using the full or reduced supervised dataset. We follow the standard procedures of supervised training. The weight initilization of the network is completely random and we do not employ any pre-training technique.

1. Randomly initialize the network's weights and biases.
2. Fine-tune **all** network parameters through supervised training using the full or a reduced version of the supervised dataset.

The objective of the FS training methodology is to establish a robust baseline performance evaluation. The baseline is obtained as the average over multiple runs to take stochasticity into account. It is crucial to obtain this baseline because it allows for a comparison with the self-supervised methodologies. This comparison will inform us whether we are achieving any improvements or not.

Supervised Linear Evaluation

This methodology is similar to FS. However, during training we update **only** the classification layer and the rest of the network remains frozen to its initial random state. In coding terms, we freeze the gradients of all network layers except for the final one; the classification layer.

1. Randomly initialize the network's weights and biases.
2. Freeze all network parameters except for the final classification layer.

3. Fine tune **only** the final classification layer through supervised training using the full or a reduced version of the supervised dataset.

This training methodology aims to provide a fair baseline performance to evaluate the self-supervised representations. Its objective is to be compared against the Linear Evaluation methodology(see below). By doing so, we can isolate and assess the effectiveness of the self-supervised learned representations, without interference from supervised fine-tuning.

Self Supervised Learning Pre-training

During SSL Pre-training we begin by training the network with our self-supervised technique. Then we proceed to fine-tune the network via supervised training.

1. Train all network layers in a self-supervised manner using the full **unlabeled** dataset.
2. Fine-tune **all** network parameters through supervised training using the full or a reduced version of the supervised dataset.

The key difference between this training methodology and FS methodology is that instead of using random initialization, we utilize the weights obtained through self-supervised learning.

The objective of SSL pre-training is to leverage large unsupervised datasets to provide an informed non-random initialization of the network's parameters. This approach is comparable to transfer learning techniques commonly used in deep learning practice, but with the distinction that transfer learning requires prior training on a similar supervised dataset (Yang et al., 2020; Weiss et al., 2016).

Linear Evaluation (of Self-Supervised Learning Representations)

During Linear Evaluation we begin by training the network with our self-supervised technique. Then we proceed to fine-tune the classifier via supervised training.

1. Train all network layers in a self-supervised manner using the full **unlabeled** dataset.
2. Freeze all network parameters except for the final classification layer.
3. Fine tune **only** the final classification layer through supervised training using the full or a reduced version of the supervised dataset.

The aim of Linear Evaluation is to have minimal impact on the SSL representations learned during pre-training. The objective is to evaluate the "pure" SSL representations in solving the tasks without intervening through fine-tuning. The tuning of the classifier is necessary to adapt it to the target classes.

In future work, we will attempt to solve the task using clustering (e.g. KNN) of the SSL representations, which should be the least intervening evaluation.

7.2 Self-Supervised Transformations

We implemented and tested three different augmentation techniques applied to the entrance of our multimodal Barlow Twins model. Augmentations are an integral part of any contrastive learning framework and can significantly impact performance. The augmentations were performed on an embedding level, and we examined Gaussian-Noise, Masking, and a feature resampling method inspired by Geo (SeqAug).

This section will provide important implementation details regarding each augmentation. Moreover, you can find the relevant Python code in the appendix section A.1.1. There, the Transformator class is presented, which should clarify the details presented here.

Gaussian Noise

This is a unimodal implementation of Gaussian noise applied to multimodal inputs. A probability p_{noise} determines if random Gaussian noise will be induced to a modality's embeddings. For every input batch B_i, the triplet input $M = (m_1, m_2, m_3)$ is separated and each one of the three modalities m_i has a probability p_{noise} to be augmented. The characteristics μ and σ of the Gaussian noise distribution can vary, but values of $\mu = 1$ and $\sigma = 0.1$ were found to be effective. The noise is applied on the feature dimension F, of the $M \times B \times L \times F$ input.

Since each modality is processed separately, depending on the value of p_{noise} it is possible that either all three, two, one, or none of the modalities m_i are augmented during a forward pass. For higher values of p_{noise} it is most probable that all or two of the modalities will undergo augmentation. In contrast, for lower values, it's more probable that a single or none of the modalities will undergo augmentation.

The probability p_{noise} of applying the augmentation, as well as the noise distribution's parameters μ and σ are all hyperparameters to be adjusted.

During Self-Supervised learning, it makes sense to apply the transformation on either one or both twin input streams. We apply it to both inputs with probabilities p_{noise1}, p_{noise2}. Hence we avoid identical inputs entering the network as much as possible. For more details see the ablation studies in section 8.5.

Masking

Similar to Gaussian Noise, the masking augmentation transforms unimodal inputs rather than multimodal. For every input batch B_i, the triplet input $M = (m_1, m_2, m_3)$ is separated and each one of the three modalities m_i has a probability $p_{masking}$ to be augmented.

The masking implementation is a soft-masking implementation. It randomly masks a percentage *mask_percentage* of the input in the desired dimension. The transformation may either be applied 'timestep' or 'feauture' wise on an input batch, by masking the L or F dimensions of the $M \times B \times L \times F$ input, respectively. By nature the 'timestep' masking is a more aggressive transformation.

Since each modality is processed separately, it is possible that during a timestep all 3 modalities, 2 modalities, one modality, or none of them have been masked simultaneously. For small values of $p_{masking}$, it's most probable that one or none of the modalities will be masked. In contrast, for a higher value of $p_{masking}$ it's most probable that all or two modalities will be masked simultaneously

The masking percentage *mask_percentage* , the probability of application $p_{masking}$ as well as the *masking_mode* (timestep or feature wise) are all hyperparameters to be adjusted. See section 8.5 for more information.

During augmentation, the transformation can be applied to one or both twin input streams of the Barlow-Twins architecture. This effect can be controlled by tweaking the values of the application probabilities $p_{masking\_1}$ and $p_{masking\_2}$. Different modes and different masking percentages may be used for each stream.

SeqAug

In accordance with the previous transformations, SeqAug is also applied on each different modality separately. For every input batch B_i, the triplet input $M = (m_1, m_2, m_3)$ is separated and each one of the three modalities m_i has a probability p_{seqaug} to be augmented.

The SeqAug transformation is a feature resampling technique that is applied on the feature level; dimension F. Firstly, a spatial sampling determines which feature indices D', of the F dimension, will remain invariant. Then, feature distributions (i.e. bags of features) are created for each remaining index, using the feature values across all timesteps of the L dimension. Finally, the time reshuffling takes place, mixing the features in the F dimension with ones from different timesteps i.e. L dimension.

The feature distributions created are half-normal distributions with a standard deviation $\sigma = 0.01$ but different mean values for each modality. We employ the mean values $\alpha = (0.15, 0.10, 0.20)$ for the textual, audio and visual modality respectively.

The probability of application p_{seqaug}, as well as the distributions' mean values $seqaug_{alpha}$ are hyperparameters to be adjusted.

During augmentation, the transformation can be applied to one or both twin input streams of the Barlow-Twins architecture. This effect can be controlled by tweaking the values of the application probabilities p_{seqaug_1} and p_{seqaug_2}.

7.3 Architecture Hyperparameters

In the realm of deep learning, selecting the ideal combination of hyperparameters is crucial for achieving optimal performance from deep neural networks. Factors such as learning rate, dropout regularization, and the choice of optimizer can greatly impact the final performance, often in unforeseen ways.

The process of selecting the best hyperparameters is both time-consuming and under-documented. The machine learning research community widely acknowledges that hyperparameter tuning is predominantly experimental, with little theoretical foundation to rely on. Consequently, practitioners and researchers frequently lack a well-documented, scientific approach to hyperparameter selection, making the tuning process even more computationally demanding.

In our research, we strive to conduct hyperparameter tuning using a scientific methodology, allowing us to draw robust conclusions and minimize computational costs as much as possible. Given the limited computational budget for this study, it is crucial to maximize its potential in order to achieve the best possible results.

During the course of a training experiment, three distinct types of parameters are typically encountered: scientific, nuisance, and fixed parameters. Each category serves a unique purpose and should be carefully considered when designing experiments and drawing conclusions.

1. **Scientific** parameters are the primary variables under investigation, and the objective is to elucidate their influence on the model's performance. By systematically adjusting these parameters, researchers can gain a deeper understanding of their impact on model outcomes. Selecting an appropriate range of values that cover the possible spectrum efficiently is important. Searching algorithms such as Bayesian Optimization (Shahriari et al., 2016) or Hyperband search (Li et al., 2018) may also be employed for discovering the most optimal values.
2. **Nuisance** parameters, in contrast, are highly interdependent with scientific parameters and require tuning in conjunction with each other. For instance, when examining the effect of different optimizers on model performance, the learning rate acts as a nuisance parameter. To facilitate a fair comparison between optimizers, it is crucial to identify an appropriate learning rate for each optimizer before drawing conclusions about their relative performance.
3. **Fixed** parameters encompass all other variables not classified as scientific or nuisance parameters. These parameters are kept constant throughout the experiment to prevent them from confounding the results. Altering multiple parameters simultaneously can obscure the true effects of the parameters under investigation and render the experiment's findings inconclusive or uninterpretable.

Following the above framework we have come to select a multitude of different hyperparameters and extract conclusions for their effects on performance. However, it must be noted that our particular choice of hyperparameters is by no means the most optimal one as that would requires an almost infinite computational budget.

After experimentation, we chose to use the following setup. We employ **one-directional LSTMs** instead of bidirectional compared to the original work. Each LSTM has 1 layer with $hidden_{dim} = 100$. All projection sizes for the attention modules are set to 100. We use dropout $drop = 0.2$. The projector of the Barlow Twins framework (complete SSL architecture) consists of 3 feedforward layers with 1000 neurons each.

For the supervised training phase, we use Adam (Kingma and Ba, 2017) with learning rate $lr = 3e{-}4$ and halve the learning rate if the validation loss does not decrease for 2 epochs. All models are trained using a batch size $B = 32$. We use early stopping on the validation loss (patience 10 epochs).

During the self-supervised training phase, we use Adam with a learning rate $lr_{ssl} = 3e - 06$ and train for 20 epochs without early stopping enabled. The Barlow Twins loss alpha parameter is set at $\alpha_{BT} = 2e - 2$ and the batch size used is $B_{ssl} = 170$, restricted by the hardware rather than choice.

Models are trained for regression on sentiment values using Mean Absolute Error (MAE) loss. We use standard evaluation metrics: 7-class (i.e. classification in $Z \cap [-3, 3]$), binary accuracy and F1-score (negative in $[-3, 0)$, positive in $(0, 3]$), MAE and Pearson correlation coefficient between model and human predictions.

Parameter_name	**SSL_arch**	**Baseline_arch**
RNN type	one-directional LSTM	one-directional LSTM
Number of layers	1	1
Hidden size (h_{dim})	100	100
Attention proj. size	100	100
Dropout ($drop$)	0.2	0.2
Projector layers	3	-
Projector neurons	1000	-

Table 7.3.1: Architecture parameter values for SSL and supervised training.

Parameter_name	**SSL_training**	**Supervised_training**
Optimizer	Adam	Adam
Learning rate (lr)	3×10^{-6}	3×10^{-4}
Epochs	20	100
Early stopping	False	True
α_{BT}	2×10^{-2}	-
Batch size (B)	170	32

Table 7.3.2: Training parameter values for SSL and supervised training.

7.4 Hardware and Software Configuration

In order to implement the proposed experiments, a variety of tools, technologies, and hardware were utilized. The following subsection provides an overview of the software and hardware used for the development of the model.

Python programming language version 3.8.10 was used as the primary programming language for the project. Python is a widely used programming language with a large community, and a variety of libraries for scientific computing. Python has become the de-facto language for deep learning.

PyTorch version 1.7.1 and PyTorch-Lightning version 1.2.0 were used for the implementation of the deep learning model. PyTorch is an open-source machine learning framework that provides support

for tensor computations and automatic differentiation. PyTorch-Lightning is a lightweight PyTorch wrapper for high-performance AI research.

NumPy version 1.20 and Pandas version 1.2.4 were used for data processing and analysis. NumPy is a library for numerical computing in Python, while Pandas is a library used for data manipulation and analysis.Scipy version 1.6.1 was used for scientific computing and statistical analysis. Scipy is a Python library that provides functions for optimization, integration, interpolation, eigenvalue problems, etc.

The SLP library version 1.1.6 was used Paraskevopoulos (2020), as it provided a lot of glue code for training, logging and evaluation proccedures. SLP is a framework for fast and reproducible development of multimodal models, with an emphasis on NLP models.

Finally, Wandb version 0.10.31 was used for tracking experiments and logging metrics. Wandb is a library for experiment tracking, dataset versioning, and machine learning visualization.

The project was executed on a computer with the following specifications: 2 NVIDIA GeForce GTX 1080 Ti graphics cards with 11 GB of VRAM. Intel(R) Xeon(R) CPU E5-1620 v3 3.50 GHz processor with 94 GB of CPU RAM. The operating system used was Ubuntu 20.04 LTS.

Overall, the aforementioned tools, technologies, and hardware provided a solid foundation for the implementation of the proposed methodology and enabled the efficient development of the deep learning model.

Table 7.4.1: Software Tools and Technologies used for the Experiment

Tool/Technology	**Version**
Python	3.8.10
PyTorch	1.7.1
PyTorch-Lightning	1.2.0
SLP	1.1.6
Wandb	0.10.31
NumPy	1.20
Pandas	1.2.4
Scipy	1.6.1

Table 7.4.2: Hardware used for the Experiment

Hardware Component	**Specification**
Graphics Card	2 NVIDIA GeForce GTX 1080 Ti with 11 GB of VRAM
Processor	Intel(R) Xeon(R) CPU E5-1620 v3 3.50 GHz
RAM	94 GB

Chapter 8

Results

8.1 Introduction

This chapter details the outcomes of our experiments that aimed to investigate the efficacy of self-supervised methods on multimodal architectures using small-medium sized datasets for emotion recognition. Our experiments were conducted to test our research hypotheses and determine if the proposed methodology is effective.

The analysis focuses on the performance of the proposed methodology, measured by 6 metrics, compared to the supervised baseline. Any significant trends or patterns observed in relation to certain hyperparameter values are also presented. The results are presented in a clear and concise manner and are supported by statistical analysis where appropriate. As long as our computation budget allowed us, we performed multiple iterations and used the average values to drive our conclusions. However, we do also give emphasis to the best models out of all runs as we believe it's important.

First, we present the baseline supervised performance and compare our self-supervised methodology against it for all three transformations. Both SSL pretraining and SSL linear evaluation training methodologies are examined. Second, we examine the results obtained by combining various transformations during self-supervised learning and draw conclusions. Then we proceed to examine our methodology against the supervised when only limited labeled data are used for training.

Following our comprehensive investigation, we conducted ablation studies to isolate specific augmentation hyperparameters and examine their impact on performance. These experiments enabled us to derive valuable insights concerning the augmentation requirements of our framework and identify scenarios where certain parameters were not as crucial as anticipated.

8.2 Baselines

To establish a comparative baseline performance and evaluate the advantages offered by our multimodal self-supervised architecture we run a series of experiments on the baseline supervised architecture. During training, we adhere to the parameters introduced in the original paper Georgiou et al. (2021) with slight modifications as explained in the Implementation section 7.

For our self-supervised training, we first ran experiments to help us discover suitable values for the self-supervised training related hyperparameters (e.g. epochs, optimizer, learning rate), presented in the implementation section 7. The rest of the hyperparameters remained stable, as in the supervised case. These baselines help us evaluate performance alterations introduced by parameter tuning, altering of augmentation steps, or altering the amount of supervised data used during training.

Table 8.2 shows the exact transformation hyperparameters used to obtain the results presented in

table 8.2.2.

Table 8.2.1: **Selected hyperparameters for each transformation.**

Transformation	P_{apply}	μ	σ	Mode	Mask %
Gaussian Noise	[0.9, 0.1]	[0.0, 0.0]	[0.1, 0.1]	-	-
SeqAug	[0.5, 0.5]	[0.15, 0.10, 0.20]	-		
Masking	[0.6, 0.6]	-	-	Timestep	[70%, 70%]

The μ in SeqAug represents the mean values of the half-normal distributions used to sample each modality (see 7.2). We used $\mu_T = 0.15$, $\mu_A = 0.20$ and $\mu_V = 0.10$ for the textual, audio and visual modalities respectively. Not to be confused with Gaussian Noise's μ value that characterizes the normal distribution used to sample noise values for each input stream (same across modalities).

Table 8.2.2: Supervised and Self-supervised models trained/tuned on the full supervised CMU-MOSEI dataset. **Average values over 5 runs.**

Mode	Augmentations	F1(%)	Acc2(%)	Acc5(%)	Acc7(%)	Corr	MAE
Supervised	-	81.48 ±0.55	81.31	52.58	51.41 ±0.35	0.688	0.5952
Self-Supervised	Gaussian Noise	82.12 ±0.12	81.55	52.37	51.28 ±0.75	0.696	0.5969
Self-Supervised	Masking	82.32 ±0.31	82.22	**53.22**	**51.78** ±0.22	**0.702**	**0.5887**
Self-Supervised	SeqAug	**82.77** ±0.13	**82.48**	**53.22**	51.69 ±0.18	0.696	0.5909

For more details on the transformation implementations see section 7.2

It quickly becomes evident that **Self-Supervised Pre-training is a potent method for increasing performance across all six metrics** used for evaluation. Results are significantly better when self-supervised mode is used, with a slight exception on *Acc*5 and *Acc*7 when it comes to using the simplest transformation ; Gaussian Noise.

We observe that SeqAug provides on average up to 1.3% increase in F1-score. Binary accuracy is also increased up to 1.2%. Gains are also high for accuracy, but Masking tends to dominate in this metric, providing up to 0.37% increase for *Acc*7 and 0.64% increase for *Acc*5 (on par with SeqAug). Improvements are also notable for Pearson correlation and MAE, with Masking dominating the metrics followed by SeqAug. Finally, Gaussian Noise significantly outperforms the supervised model in the $F1$, *Acc*2 and Person correlation but misses on the other metrics.

Another positive byproduct is a **decreased variance in performance across metrics**. Models that have been pre-trained with self-supervision have significantly more persistent performance compared to supervised ones. This means that our framework provides a more robust and trustworthy training methodology.

Table 8.2.3 focuses on the best performant models out of 5 runs. A similar picture emerges. SSL pre-trained models consistently perform better across metrics. The improvement for $F1$ stands at about 0.9% while for masking achieves up to 0.47% for *Acc*7. Our best models achieve performance that **is comparable or surpasses the previous SOTA models** such as MuLT (Tsai et al., 2019), M^3 (Georgiou et al., 2021), SeqAug (Geo), and MMLatch (Paraskevopoulos et al., 2022) (see table 5.4.1, 5.4.2).

Based on our results, we can conclude that SSL pretraining using the Masking transformation results in the highest *Acc*5, *Acc*7, correlation, and MAE while SeqAug results in the highest $F1$ score and *Acc*2.

Table 8.2.3: Supervised and Self-supervised models trained/tuned on the full supervised CMU-MOSEI dataset. **Best models over 5 runs.**

Mode	Augmentations	F1(%)	Acc2(%)	Acc5(%)	Acc7(%)	Corr	MAE
Supervised	-	81.91	81.83	52.87	51.76	0.691	0.5902
Self-Supervised	Gaussian Noise	82.2	81.8	52.82	51.78	0.696	0.5969
Self-Supervised	Masking	82.75	82.48	**53.41**	**52.23**	**0.700**	**0.5886**
Self-Supervised	SeqAug	**82.83**	**82.62**	53.33	51.74	0.699	0.5916

For more details on the transformation implementations see section 7.2

It's very interesting, how the metrics each transformation optimizes are reversed when Linear Evaluation is used instead of fine-tuning (table 8.2.4). The picture is reversed, with Masking providing the best results for F1 and *Acc*2, while SeqAug better optimizes *Acc*5 and *Acc*7.

Despite that, again we do observe improved performance for self-supervision compared to the supervised baseline. However, improvements are lesser and not always consistent across all metrics. Variance still remains smaller for SSL models providing a more robust training method.

Table 8.2.4: **Linear evaluation** of Supervised and Self-supervised models trained on the full supervised dataset. **Average values over 5 runs.**

Mode	Augmentations	F1 (%)	Acc2 (%)	Acc5 (%)	Acc7 (%)	Corr	MAE
Linear Evaluation	-	72.58 ± 0.76	71.67	42.61	42.60 ± 0.73	0.371	0.781
Self-Supervised	Noise	71.16 ± 0.63	70.28	42.71	42.34 ± 0.57	0.3953	0.7766
Self-Supervised	Masking	**73.18** ± 0.36	**72.02**	42.55	42.55 ± 0.26	**0.407**	**0.771**
Self-Supervised	SeqAug	72.89 ± 0.50	70.5	**42.9**	**42.88** ± 0.25	0.3652	0.7847

8.3 Combining Transformations

In our research, we test the hypothesis that the use of multiple transformations on the model's inputs may provide enhanced performance compared to a single transformation during SSL training. This section thoroughly examines the results obtained by chaining the three transformations in various combinations.

Chaining (combining) multiple transformations in a suitable order has been shown to substantially improve performance in previous works Zbontar et al. (2021); Arandjelović and Zisserman (2017a); Zhao et al.. However, degradation of performance has also been observed, hence it's more about the suitability of the combined transformations than their number. It's also worth noting that previous works operated on raw data while we operate on the embedding level.

The hypothesis is that since certain transformations provide better results on specific metrics (e.g. SeqAug optimizes *F*1, Masking optimizes *Acc*7), combining them could possibly provide the "best of both worlds", hence further increasing both metrics.

The hyperparameters used for each transformation throughout these experiments are summarized in table 8.3.1. These values are not picked randomly, but rather are the ones that exhibit the best performance across most metrics.

Table 8.3.1: Transformations and their Parameters

Transformation	P_{apply}	μ	σ	Mode	Mask %
Gaussian Noise	[0.9, 0.1]	[0.0, 0.0]	[0.1, 0.1]	-	-
SeqAug	[0.5, 0.5]	[0.15, 0.10, 0.20]	-		
Masking	[0.6, 0.6]	-	-	Timestep	[70%, 70%]

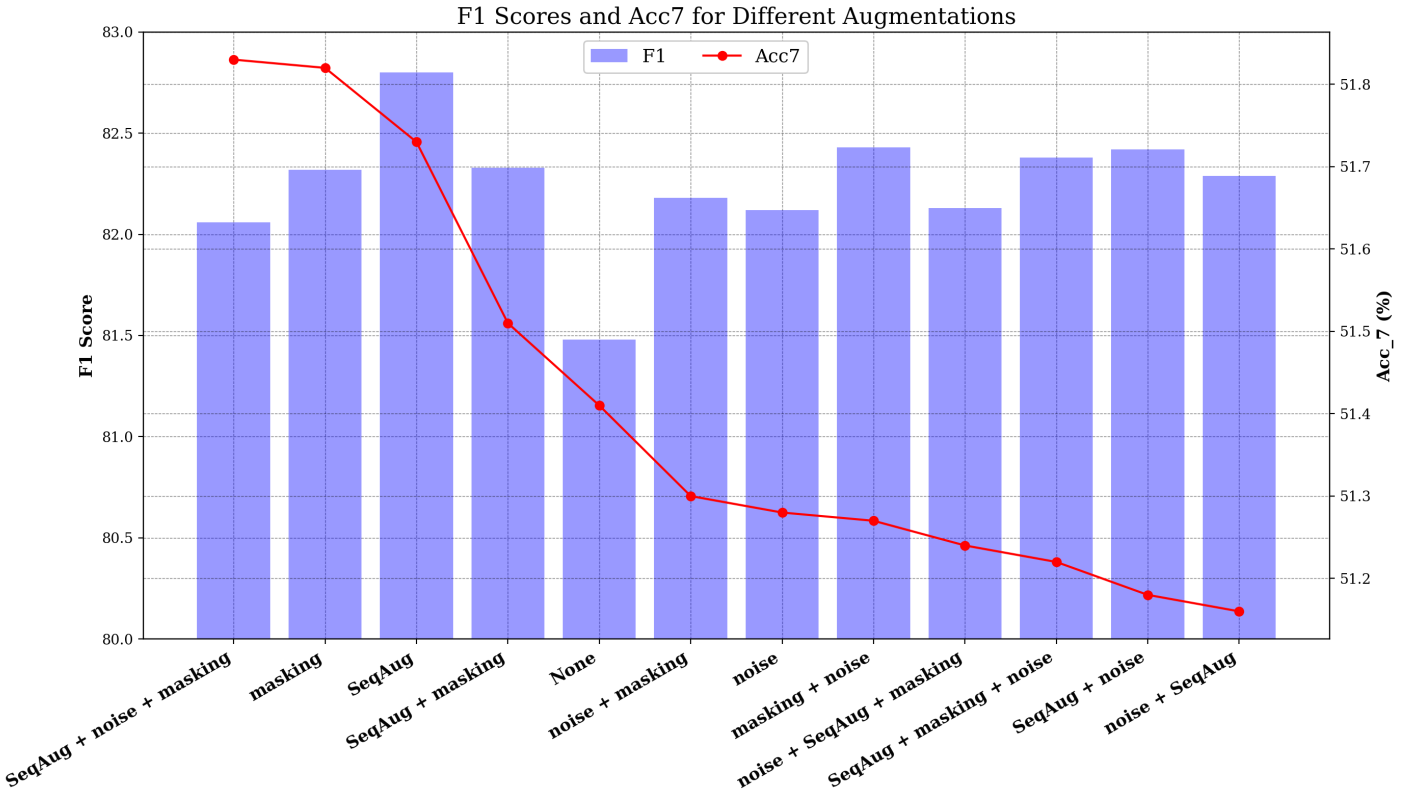


Figure 8.3.1: F1 score and *Acc*7 for different transformation chainings. The plot shows a comparison of the *F*1 score (bars - left y axis) and *Acc*7 (red line - right y axis) metric values for different combinations (x-axis) during SSL Pre-training. The supervised model is also included as "None" for reference.

Table 8.3.2: Evaluation of Supervised and SSL pretrained models trained on the full supervised dataset. **Average values over 5 runs.**

Augmentations	F1	Acc2 (%)	Acc5 (%)	Acc7 (%)	Corr	MAE
none	81.48	81.31	52.58	51.41	0.688	0.5952
noise	82.12	81.55	52.37	51.28	0.696	0.5969
noise + SeqAug	82.29	82.03	52.39	51.16	0.6938	0.594
noise + masking	82.18	81.85	52.60	51.30	0.6949	0.5913
noise + SeqAug + masking	82.13	81.88	52.48	51.24	0.6952	0.5918
masking	82.32	82.22	**53.22**	51.78	**0.702**	0.5887
masking + noise	82.43	82.21	52.74	51.27	0.6981	0.5924
SeqAug	**82.77**	**82.48**	**53.22**	51.69	0.696	0.5909
SeqAug + noise	82.42	82.25	52.48	51.18	0.6976	0.5894
SeqAug + masking	82.33	82.16	52.78	51.51	0.697	0.5889
SeqAug + noise + masking	82.06	82.00	53.12	**51.83**	0.6968	**0.5882**
SeqAug + masking + noise	82.38	82.17	52.60	51.22	0.6981	0.5895

A quick observation of figure 8.3.1 or table 8.3.2, makes it evident that not only the existence but even the order in which the transformations are chained plays a significant role in the performance of the SSL pretrained models. An iconic example are combinations [*SeqAug*, *noise*, *masking*] and [*SeqAug*, *masking*, *noise*] where just the order of the last two transformations is reversed. It's surprising that the former exhibits the highest *Acc*7 while the latter is among the lowest! Similarly, coupling *SeqAug* with *noise* proves disastrous.

Compared to our supervised model most transformations are performing significantly better in relation to F1 score but many of them still struggle in Accuracy.

Overall, we observe that **our chosen transformation chainings did not perform better compared to single transformations.** In some cases, coupling transformations provides an improvement compared to single ones but usually comes as a tradeoff between different metrics.

Combining even the best-performing transformations in a single chaining [*SeqAug* + *masking*] significantly reduces performance. This possibly signifies that the combined transformations become very hard for the model to solve, resulting in worse representation learning. However, it must also be noted that **optimally fine-tuning such chains of transformations is extremely hard** as the hyperparameter space becomes very large and the model seems to be quite sensitive to them.

Further research is required to effectively answer whether such combinations are fruitful and under which circumstances.

8.4 Training with Reduced Supervised Datasets

The most promising potential of self-supervised learning is its ability to learn representations out of large unsupervised datasets, thus significantly reducing the need for labeled data. To evaluate the capability of our framework to learn useful representations out of unsupervised datasets and limited amounts of labeled datasets we performed a series of experiments with reduced versions of the CMU-MOSEI labeled dataset.

First, in order to establish accurate baselines, we trained our network in a purely supervised manner on multiple reduced versions of the original dataset. Subsequently, we trained the network using self-supervision on the **same** respective datasets.

Note that during the self-supervised phase, we train with the full unsupervised dataset, without making use of labels in any way. Only after that, the available labels are used to fine-tune the network. Both SSL Pretraining and Linear Evaluation (see 7.1) were evaluated. Figures 8.4.1, 8.4.2 sum up our findings. The former plots the best models out of 3 runs while the latter plots the average measurements over the same 3 runs.

We note that SSL models, especially Masking and SeqAug models, lend themselves very well for smaller datasets. The SSL models perform impressively better on tiny datasets (1%). The difference tends to shrink as as we increase the percentages, and even completely disappear in certain situations. However, the overall trend seems to favor SSL models across all sizes.

Overall, we can see that SSL models require about 80%**, and even** 60% **in extreme cases, to match or outperform the performance of the Supervised model trained on the full dataset (fig. 8.4.1).** This smaller reliance on the size of the dataset can lead to many practical advantages and applications such as automatic labeling tools and faster annotation cycles.

Performance saturation is observed around the $60 - 80\%$ mark. The saturation seems to be more prevalent for the Gaussian Noise model and Supervised models compared to the other two. It's also more visible for $F1$, $Acc2$, Cor and MAE. Surprisingly enough there are limited cases where the same model on less data marginally outperformed itself compared to when it was trained with more data. Such behaviors are worth further research, but the stochasticity of training may be a possible root cause and more experiments may correct it.

The linearly evaluated model (black line, SSL with Masking) performs, as expected, substantially worse compared to its fine-tuned counterparts. Its rate of increase (slope) is much smaller, something we anticipate since only the final layer is enhanced by the excess data.

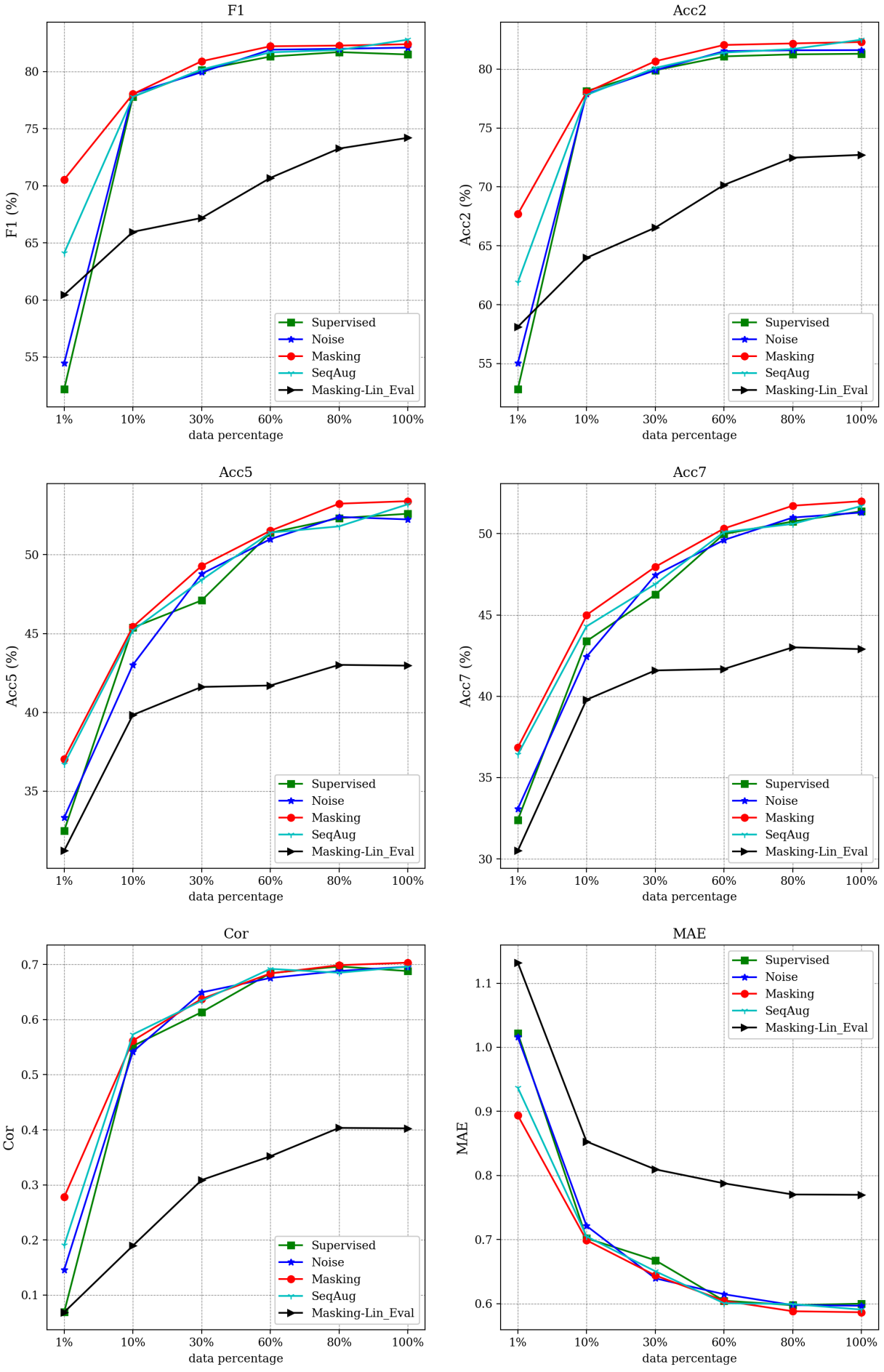


Figure 8.4.1: Performance of the supervised and self-supervised models when trained on reduced versions of the original dataset. The measurements represent the **best models out of 3 runs**. The black line represents a model trained in SSL mode with Masking and Linearly Evaluated by fine-tuning a classifier on top.

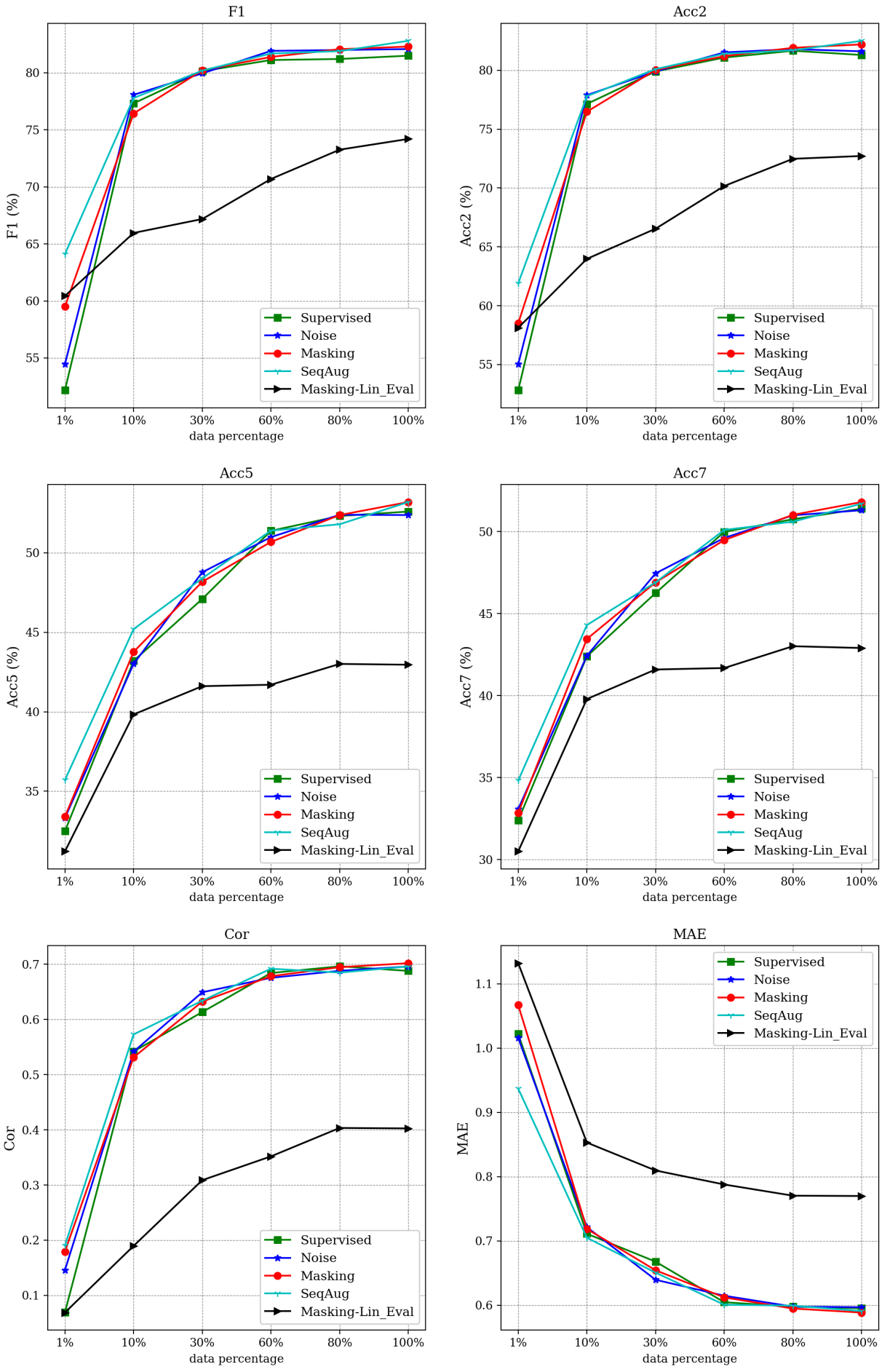


Figure 8.4.2: Performance of the supervised and self-supervised models when trained on reduced versions of the original dataset. The measurements represent the **average values over 3 runs**. The black line represents a model trained in SSL mode with Masking and Linearly Evaluated by fine-tuning a classifier on top.

8.5 Ablation studies

In this section, we present results gathered from multiple experiments dedicated to the discovery of suitable parameter values for our transformations. First, we explore the parameter space for masking when the augmentation is applied to just one of the two input streams. Subsequently, we explore the parameter space for double-masking, i.e. the application of Masking to both input streams of our Multimodal Barlow Twins model. Finally, we explore the parameter space for SeqAug.

The original work on Barlow-Twins followed the strategy of applying strong augmentation on the first stream and a very mild augmentation on the second stream. However, the architectural differences between our work and the original paper prompt us to test this experimental choice and discover new guidelines.

The goal of these experiments is to discover patterns in our tuning process that can help us guide both current and future experiments. In some cases, we do extract interesting conclusions, while in others we discover that no apparent trends exist or that the parameter values do not exert a big influence on performance. In the latter cases, a "brute force" approach must be followed to discover suitable parameter values.

Single-Stream Masking

The implementation of Masking transformation has already been discussed in section 7.2. In this case, we explore masking of just one input stream. This means that $P_{apply2} = 0$. Picking the best values for either the masking percentage or the probability of application was a trial and error procedure. By conducting multiple experiments we gathered the results presented in figure 8.5.1. The figure plots both the SSL pre-trained models and the linearly evaluated ones. The supervised baselines are also plotted as grey dotted lines for reference.

Most combinations already fare well above the supervised baselines (for fine-tuned models). Moreover, the difference in performance between the best and worst models is not high except for *Acc*7 and *MAE* metrics. In contrast, the linearly evaluated models are less potent, and fewer combinations provide appropriate performance.

At first glance, the lines' behavior seems stochastic. Different metrics are optimized by different parameter combinations, which can sometimes appear "opposite". Trends often appear non-monotonic and the different behavior observed between the SSL Pre-trained models (stars) and the linearly evaluated ones (triangles) make it increasingly more complex to discover patterns.

However, a careful examination of the results helped us establish some **soft rules of thumb**. First, **a higher probability of application** $P_{apply} \geq 0.6$ **is best combined with a smaller to medium masking percentage** $[20\%, 50\%]$**.** Our claim is supported by a slight downward trend (upward for *MAE*) observed on the blue lines. A possible explanation for our soft rule is that as the probability of application increases more modalities will be masked in each batch. This on its own makes the self-supervised objective more difficult to minimize. Further increasing the percentage of input values that are masked makes the transformation even harder to solve. If the transformation becomes extremely difficult, the self-supervised training is harder to converge to a minimum of the Barlow Twins Loss function. Having said all that, some well-performing combinations are $(P_{apply} = 0.3, mask\_percent = 100\%)$, $(P_{apply} = 1.0, mask\_percent = 50\%)$ and $(P_{apply} = 0.6, mask\_percent = 20\%)$.

The opposite is also true to an extent, even if rule exceptions appear. **A low** $P_{apply} = 0.3$ **is better combined with a higher masking percentage.** This is easier detected by the linearly evaluated models where a clear ascending trend of the green-triangled line is observed. Similar trends can be observed in the fine-tuned models (green-starred), with the most typical example being *MAE*. The rule also holds for the other metrics but the trend is non-monotonic, making it harder to observe at first glance. A

logical explanation for this rule is that since, statistically, fewer modalities are masked in each batch, the transformation solved by the Barlow Twins objective function is easier if small percentages of the inputs are masked. As an effect, the network detects basic similarities between the two streams but is not challenged enough to learn more difficult similarities. This may lead to poorer representation learning.

Double-Stream Masking

During Double-Stream masking both P_{apply1} and P_{apply2} are non zero. This means that both input streams may be masked during SSL training. Our experimental results can be summarized in the three plots that follow, figures A.2.1, A.2.3 and 8.5.4. Plots for the rest of the metrics are also provided in the appendix section A.2.1.

We have tested multiple combinations for the application probabilities as well as the masking percentages. The subtitles 'Low-Low', 'Medium-Medium', 'High-High', and 'High-Low' above each subfigure characterize the probabilities of applying the transformation on each input stream i.e. $P1$, $P2$. Through these 4 combinations, we test 4 different augmentation strategies. Previous works have mostly followed the "High-Low" strategy, applying a strong and frequent augmentation on the first input stream and a weak, less frequent one on the second. The color bars in the graphs indicate the value of the metrics. Note that for MAE a lower value is better, hence lighter colors mean worse performance.

A common behavior, detected in most of our experiments is that different hyperparameters optimize different metrics and it is hard to discover a single combination that outperforms across all metrics. Hence, the best combinations are not necessarily the ones that achieve the highest performance on a single metric but rather score among the highest across all metrics. Having said that, studying the aforementioned figures helped us discover some trends and establish some rough guidelines.

First, if optimizing the $F1$ score is more important then 'High-High' and 'High-Low' augmentation strategies provide higher performance. In contrast, for $Acc7$ a 'Low-Low' augmentation strategy better optimizes for accuracy. MAE on the other hand, does not seem to be highly affected by the application probabilities. It's the values of the masking percentages that have a high impact on MAE. It's easy to observe how performance remains mostly stable regardless of P_{apply_i} and that the pair $(mask\_perc_1, mask\_perc_2) = (70\%, 70\%)$ dominates.

Detecting patterns in relation to the masking percentages is much harder if any exist at all. However, we can see that for unexplainable reasons certain combinations fare better than others across all metrics, providing more robust models. The most successful combination is $[70\%, 70\%]$, as it proves quite flexible and is what we used in most cases.

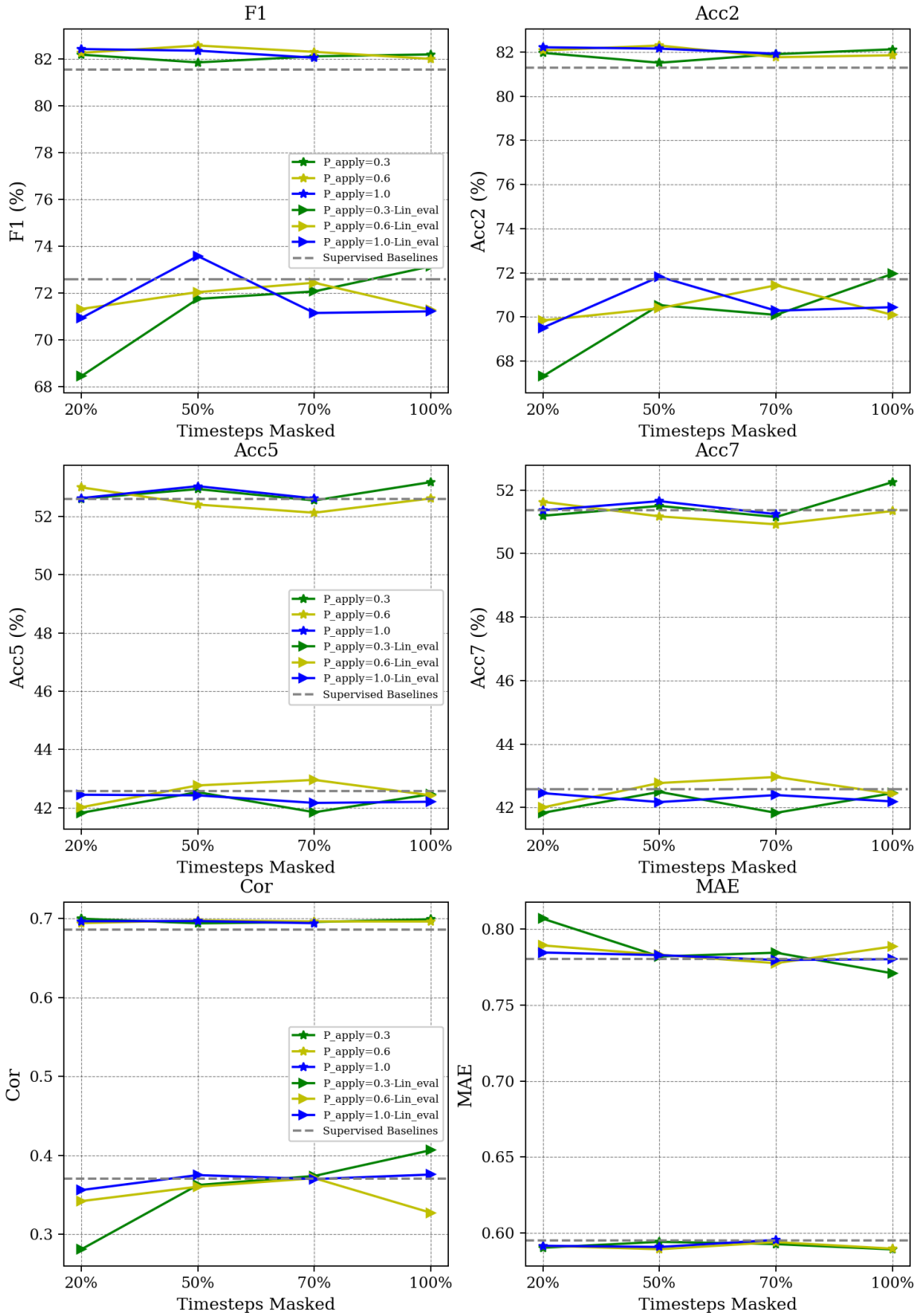


Figure 8.5.1: Fine-tuning study of the single-masking augmentation. Masking is applied only on the first input stream with probability P_{apply}. The second input stream is not affected. The graph presents the effects of *mask_percent* and P_{apply} on performance for various metrics. The star-styled lines denote SSL Pre-trained models while the triangle styles lines denote the Linearly Evaluated SSL models as discussed in section 7.1.

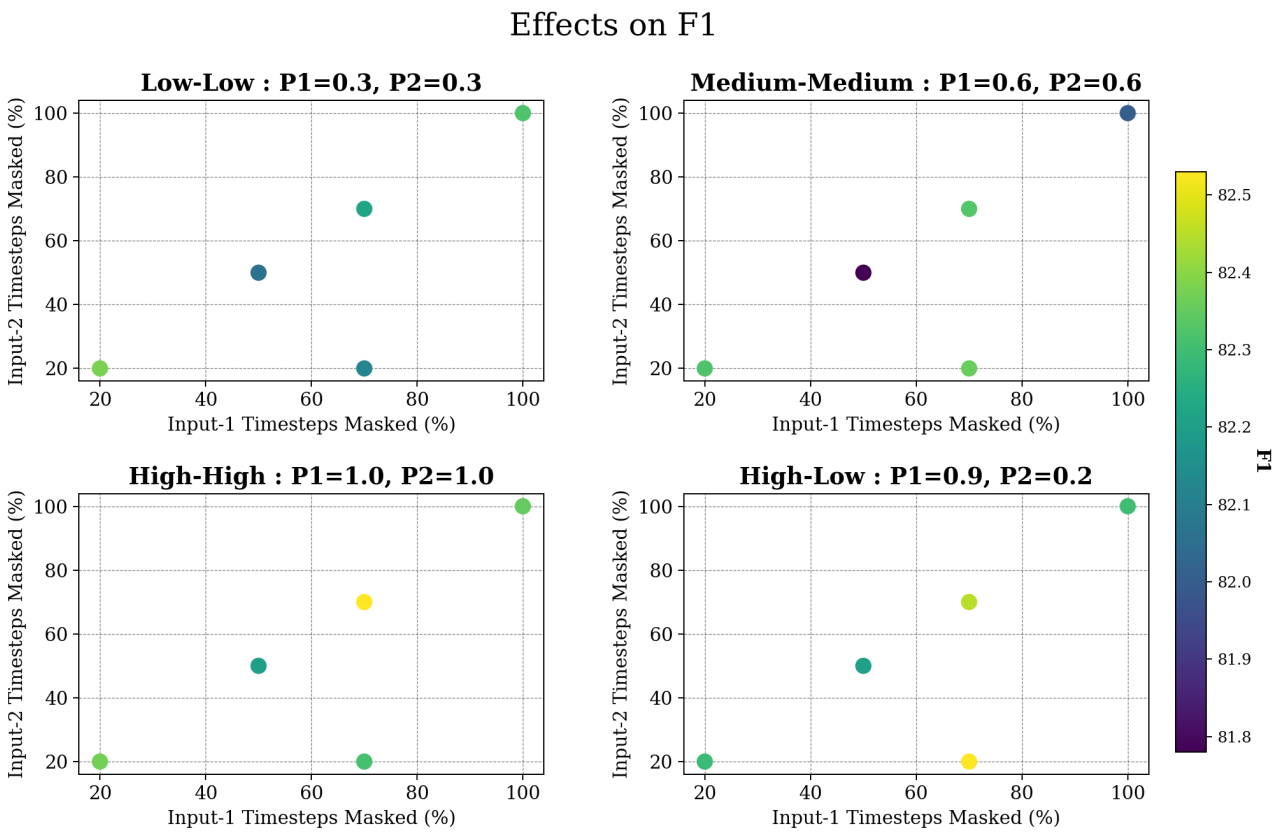


Figure 8.5.2: Fine-tuning Masking parameters in relation to $F1$. Effects of *mask_percent* and P_{apply} of each input stream. The subtitles 'Low-Low', 'Medium-Medium', 'High-High', and 'High-Low' characterize the probabilities of applying the transformation on each input stream i.e. $P1$, $P2$. The x-axis and y-axis denote the percentage of input masked in stream1 and stream2 respectively, when the transformation is applied. The color shows the metric performance. The results correspond to **SSL Pre-trained** models as discussed in section 7.1. **Averages over 5 runs.**

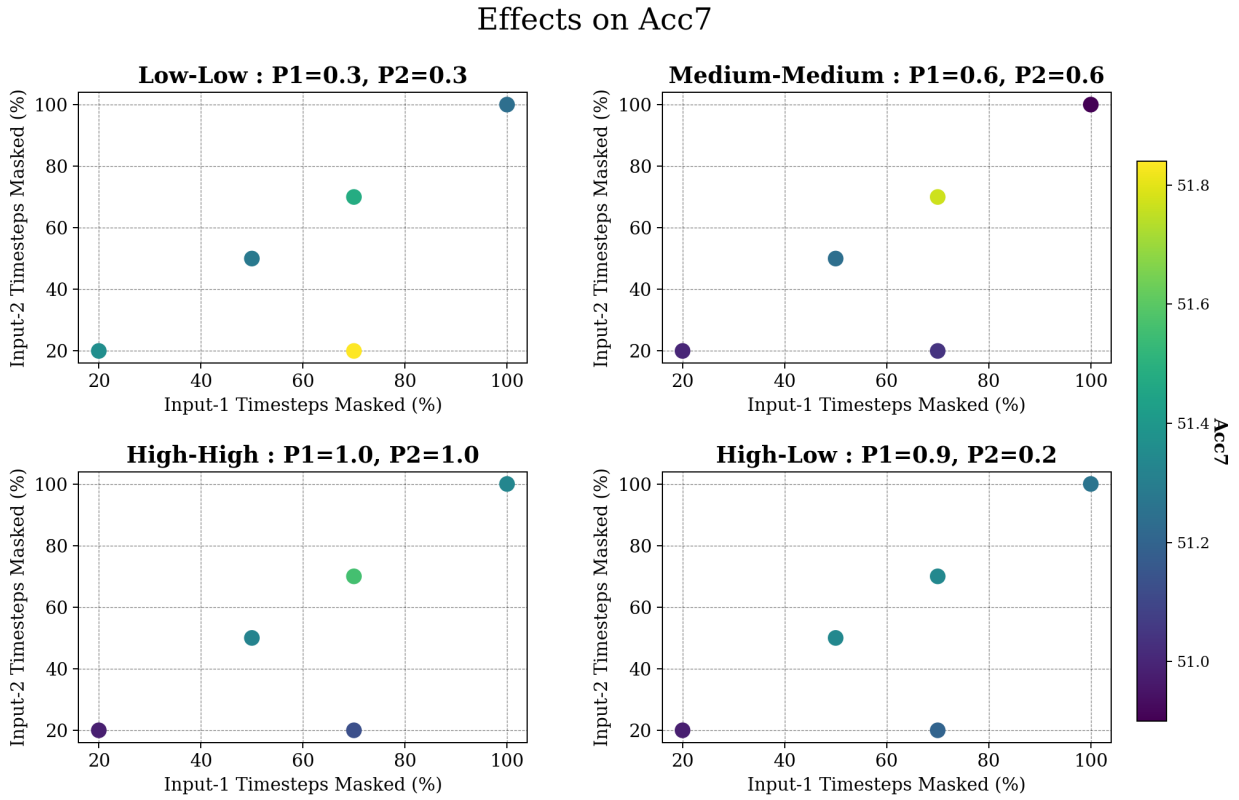


Figure 8.5.3: Fine-tuning Masking parameters in relation to *Acc7*. Averages over 5 runs.

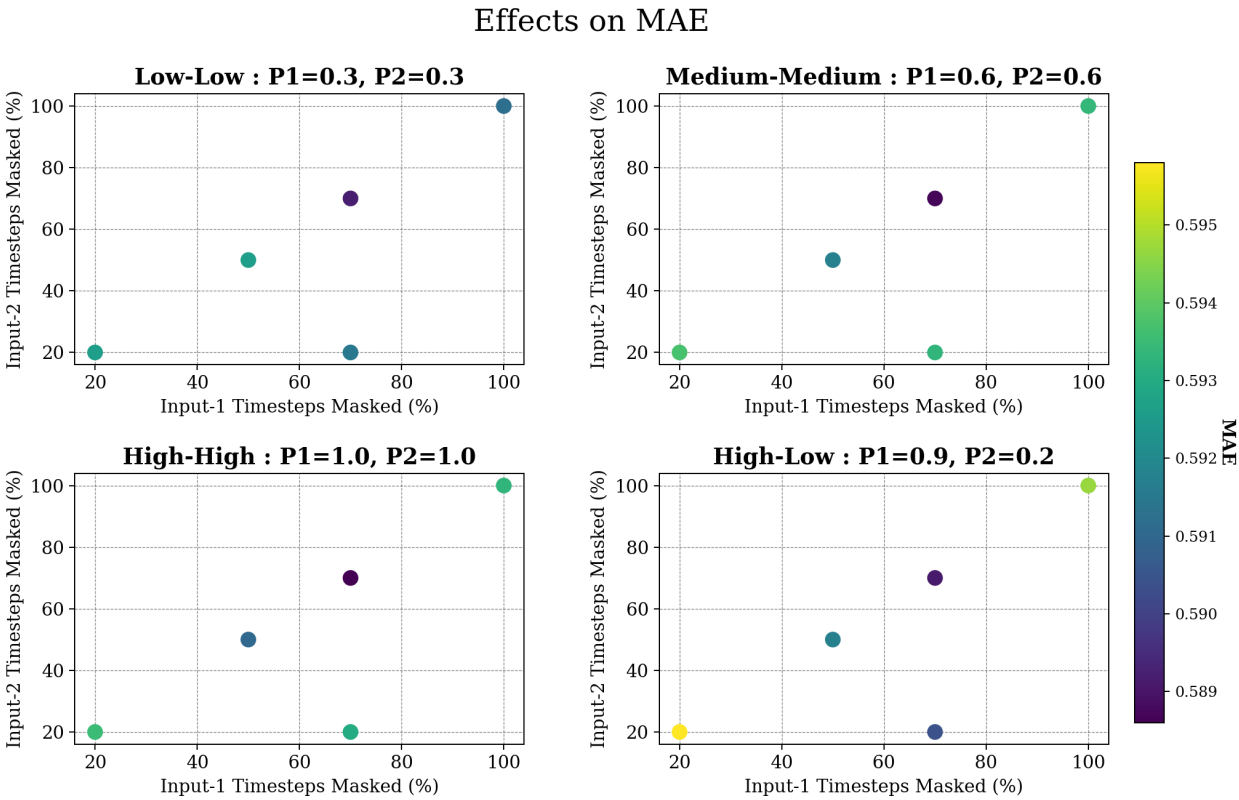


Figure 8.5.4: Fine-tuning Masking parameters in relation to *MAE*. Dark color means better performance. Averages over 5 runs.

Tuning of SeqAug Augmentation

Tuning the SeqAug transformation required following a trial and error approach. The transformation is generally very well suited for optimizing $F1$ and $Acc2$. This can be observed in figure A.2.4, where the lowest values for these metrics are still quite higher compared to the supervised baseline (81.48%, 81.31% respectively). However, not all combinations fare well in terms of $Acc5$ and $Acc7$.

A strategy augmentation of 'Medium-Medium' probabilities of application performs the best across all metrics. The combination $(P_{apply1}, P_{apply2}) = (0.5, 0.5)$ clearly dominates all other combinations. Although, a noticeable pattern is not visible, hence we cannot draw any useful conclusion from this ablation study. This ablation experiment helped us pick the best-performing combination which we used in the rest of our experiments.

Chapter 9

Conclusions and Future Research

9.1 Introduction

In this final chapter, we reflect on our journey exploring the realm of self-supervised multimodal learning for emotion recognition. The overarching goal of this study was to explore the uncharted waters of self-supervision for small and medium sized multimodal datasets.

Self-supervision has proved a very powerful technique and is fueling many of the impressive advances in deep learning we see today. However, these models are trained on vast reservoirs of (unlabeled) data; think of all the available text on the internet, all of GitHub repositories, or all the available images on social media. Our aim was to explore the effectiveness of self-supervision when the pool of unlabeled data is small, by modern standards, and when the available labeled data may be even fewer.

Emotion recognition is a task that humans are inherently good at. It's something that is rarely taught yet pretty much all humans are capable of doing, even if they are not consciously aware of it. In a dialogue, our receptors and classifiers process what is being said (text), the body language of the other person (visual), and the tone of voice (audio). We are by nature multimodal creatures.

Integrating artificial intelligence into our everyday life demands that machines are capable of detecting such emotions and adapting their behavior accordingly. Using multiple sources of data (video, text, audio) is a more effective way of solving the task of emotion recognition compared to a single modality. However, multimodal deep learning comes with its own set of challenges Baltrušaitis et al. (2019); Wang et al. (2019). Representing and fusing heterogeneous data is an active field of research.

9.2 Summary of findings

Before beginning our study we posed 4 questions we wished to answer. We present our derived conclusions based on the question they answer. However, some conclusions may answer more than one question at once. The most important conclusion is that our proposal provides competitive SOTA performance on the task of multimodal emotion recognition. This means that self-supervision has successfully been adapted to our multimodal setting with a small-medium dataset.

1. Can the methodologies developed for unimodal self-supervised learning be effectively adapted to a multimodal setting with small-medium sized datasets?
    - Unimodal self-supervised frameworks such as Barlow Twins (Zbontar et al., 2021) can be adapted to multimodal settings with success. **Our proposal is either on par or outperforms** the previous SOTA in the field, which all use supervised frameworks.
    - The augmentation strategies followed in unimodal frameworks may need to be adapted and the original parameters may not work as effectively.

- Both masking and SeqAug (Georgiou and Potamianos, 2023) are effective augmentation techniques for multimodal data with pre-extracted features (embeddings). They can be incorporated to SSL frameworks with success.

2. How does the performance of self-supervised learning methodologies compare to traditional supervised techniques in the context of multimodal emotion recognition with small-medium sized datasets?

   - Self-Supervised Learning has the potential to substantially improve performance compared to supervised baselines. We observed **absolute metric improvements** ranging from 0.40% to 1.3%.
   - Self-supervision may be used both as a pre-training method or as the main training method. Both provide similar percental improvements compared to the supervised baselines.

3. Can self-supervised learning provide an adequate solution when labeled data scarcity becomes an issue?

   - SSL pre-trained models have the potential to match the performance of supervised models using **as few as 60% - 80% of the full labeled dataset**.
   - The number of labeled multimodal samples available for supervised training has a positive correlation with performance but a **plateau is reached quicker for supervised models.**

4. Where does self-supervised learning fall short in practice, and under what circumstances does it prove worthy of implementation for multimodal emotion recognition?

   - Self-supervised frameworks designed around transformation-invariant representation learning can be **very sensitive to the augmentations used**. A non-suitable augmentation methodology can harm performance instead of improve it.
   - Fine-tuning architectural hyperparameters and augmentation hyperparameters is **very challenging** for multimodal networks.
   - SSL training **adds a lot of complexity** to the training pipeline. If improvements are not high then it may not be worth the investment from an engineering point of view.

9.3 Future Steps

Our research has provided promising signs that self-supervision is an adequate paradigm of learning for multimodal datasets and emotion recognition. Having validated our initial hypotheses we feel encouraged to expand upon it. Our future steps our guided by the objective to further increase performance and the quality of predictions.

We propose the following future steps for us and other interested parties.

1. Scale up and diversify the pool of unlabeled data by merging multiple similar datasets (e.g. IEMOCAP Busso et al. (2008)). We suspect that by increasing the amount of unlabeled data our self-supervised models can draw knowledge from, the learnt representations will be of higher quality.
2. Experiment and evaluate the efficacy of our framework with Transformer based architectures.
3. Experiment with other multimodal representation techniques like fine and coarse-grained spaces (Alayrac et al., 2020) or 3 bimodal disjoint spaces that require more than one loss functions to operate. We already expanded on these ideas during section 6.6

9.4 Final Words

Our proposed methodology has proved capable of tackling this complex task and provides the groundwork for further contributions. We are both excited and humbled to have contributed, even if modestly, to this fascinating field. Recognizing that each small discovery and insight, each incremental step forward, contributes to the foundation of knowledge that propels this field onward, we take great pride in our work.

However, we also feel a deep sense of humility, fully aware of the immense scope of what remains unexplored and the collective effort required to continue to push the boundaries of what we currently understand. We are just one piece of this vast intellectual endeavor, and we are truly honored to be part of this journey.

Appendix A

Appendix

A.1 Code

A.1.1 Transformator Code

The code below defines the Transformator class. The Transformator class is responsible for defining and applying stochastically each different augmentation. The code acts as extra material and complements the theoretical and implementation details provided in sections 6.5, 7.2 respectively.

```
class Transformator:
    def __init__(
        self,
        transformation_order: List[str] = ["masking"],
        noise_p1=0.7,
        noise_p2=0.2,
        noise_mean1=0.0,
        noise_mean2=0.0,
        noise_std1=0.1,
        noise_std2=0.1,
        masking_p1=0.5,
        masking_p2=0.0,
        masking_percentage_1=0.5,
        masking_percentage_2=0.5,
        masking_mode = 'timestep',
        seqaug_p1= 1.0,
        seqaug_p2= 0.0,
        seqaug_alpha= [0.15, 0.1, 0.2],
        seqaug_p1_t= 1.0,
        seqaug_p2_t= 0.0,

        p_mod1: Optional[List[float]] = None,
        p_mod2: Optional[List[float]] = None,
        m3_sequential: bool = True,
    ) -> None:
        """ Wrapper class for other augmentation functions. It combines the
            augmentations in the desired order (dictionary), controls the
            probability of application, and calls the transformation functions.
        """
        assert all(
            isinstance(x, str) and x in ["noise", "masking", "seqaug"]
            for x in transformation_order
        ), "Allowed modes for transformation_order are ['noise' | 'masking' | '
            seqaug']"
```

```
        # Define transformation objects in a dict to use afterwards.
        self.instructions = {
            "noise": [
                transforms.RandomApply(
                    [Gaussian_noise(noise_mean1, noise_std1)], noise_p1
                ),
                transforms.RandomApply(
                    [Gaussian_noise(noise_mean2, noise_std2)], noise_p2
                ),
            ],
            "masking": [
            transforms.RandomApply(
                [Masking(masking_percentage_1, masking_mode)], masking_p1
                ),
            transforms.RandomApply(
                [Masking(masking_percentage_2, masking_mode)], masking_p2
                )
            ],
            "seqaug": [
            transforms.RandomApply(
                [SeqAug(alpha=seqaug_alpha, p_t_mod=[seqaug_p1_t])], seqaug_p1
                ),
            transforms.RandomApply(
                [SeqAug(alpha=seqaug_alpha, p_t_mod=[seqaug_p2_t])], seqaug_p2
                )
            ],
        }

        # Define the first transformator (using dict key [0])
        self.transform = transforms.Compose(
            [self.instructions[order][0] for order in transformation_order]
        )
        # Define the second transformator`prime` (using dict key [1])
        self.transform_prime = transforms.Compose(
            [self.instructions[order][1] for order in transformation_order]
        )

    def __call__(self, *mods):
        # B x L x Feats. Modality shapes for MOSEI:
        y1 = self.transform(mods)
        y2 = self.transform_prime(mods)

        return y1, y2

    def __repr__(self):
        return (
            self.__class__.__name__
            + "Transformation Order: {0}, (transform_1={1}, transform_2={2})".
                format(
                self.instructions, self.transform, self.transform_prime
            )
        )
```

A.2 Extra results

A.2.1 Double-Stream Masking

During section 8.5 we explored the fine-tuning process of our masking augmentation. Here we present 3 more plots, for the metrics not presented in the main section *Acc*2, *Acc*5, and *Corr*.

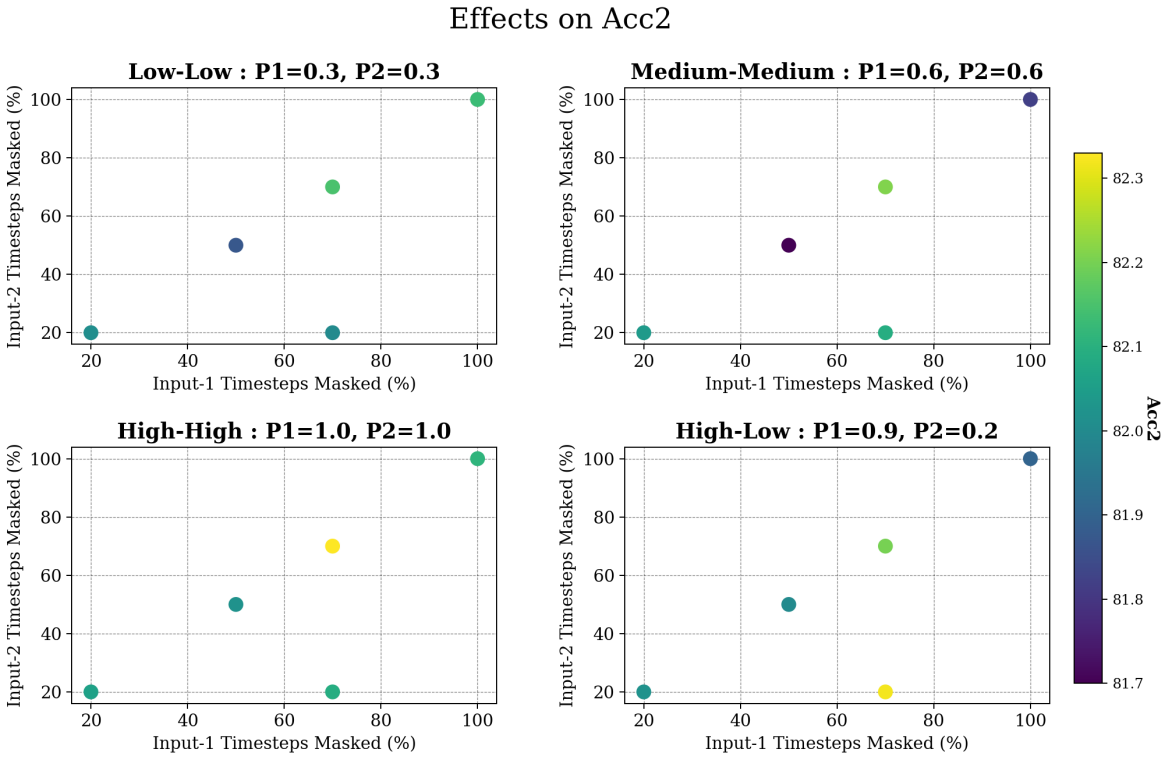


Figure A.2.1: Fine-tuning Masking parameters in relation to *Acc*2. Effects of *mask_percent* and P_{apply} of each input stream. The subtitles 'Low-Low', 'Medium-Medium', 'High-High', and 'High-Low' characterize the probabilities of applying the transformation on each input stream i.e. *P*1, *P*2. The x-axis and y-axis denote the percentage of input masked in stream1 and stream2 respectively, when the transformation is applied. The color shows the metric performance. The results correspond to **SSL Pre-trained** models as discussed in section 7.1. **Averages over 5 runs.**

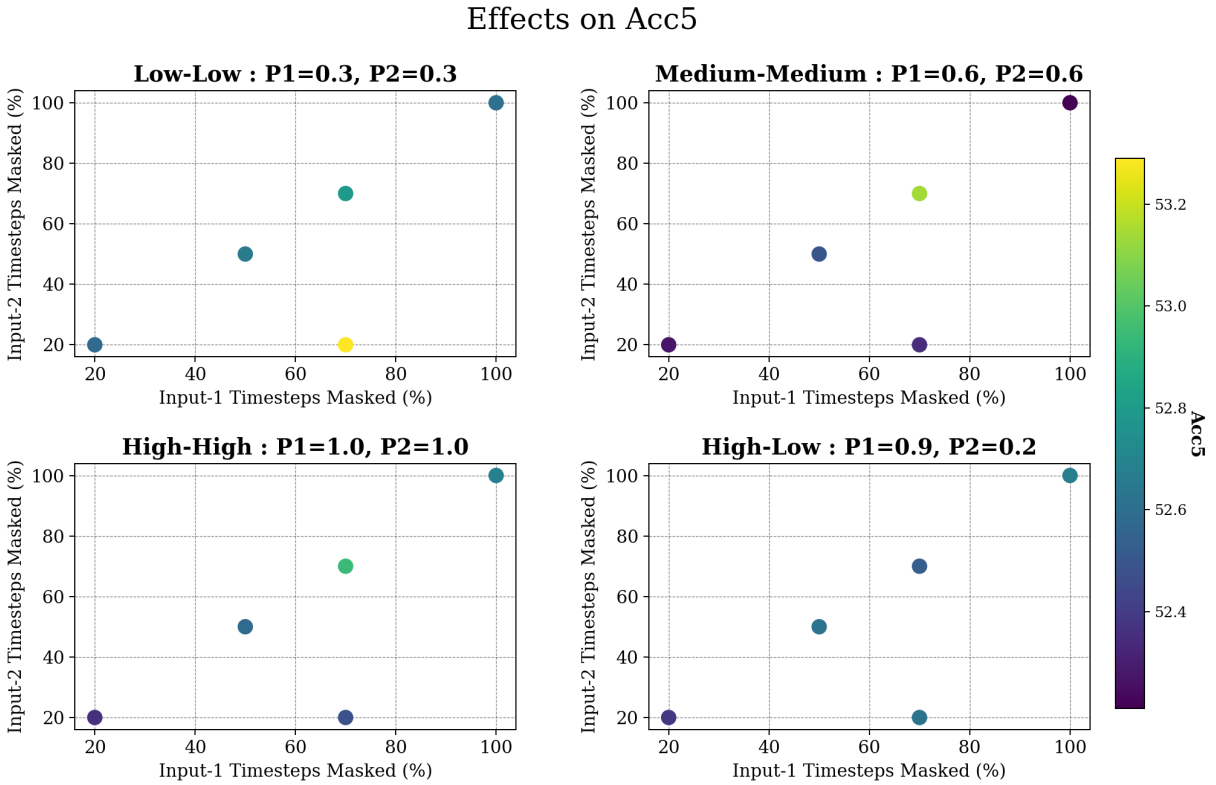


Figure A.2.2: Fine-tuning Masking parameters in relation to *Acc*5. Averages over 5 runs.

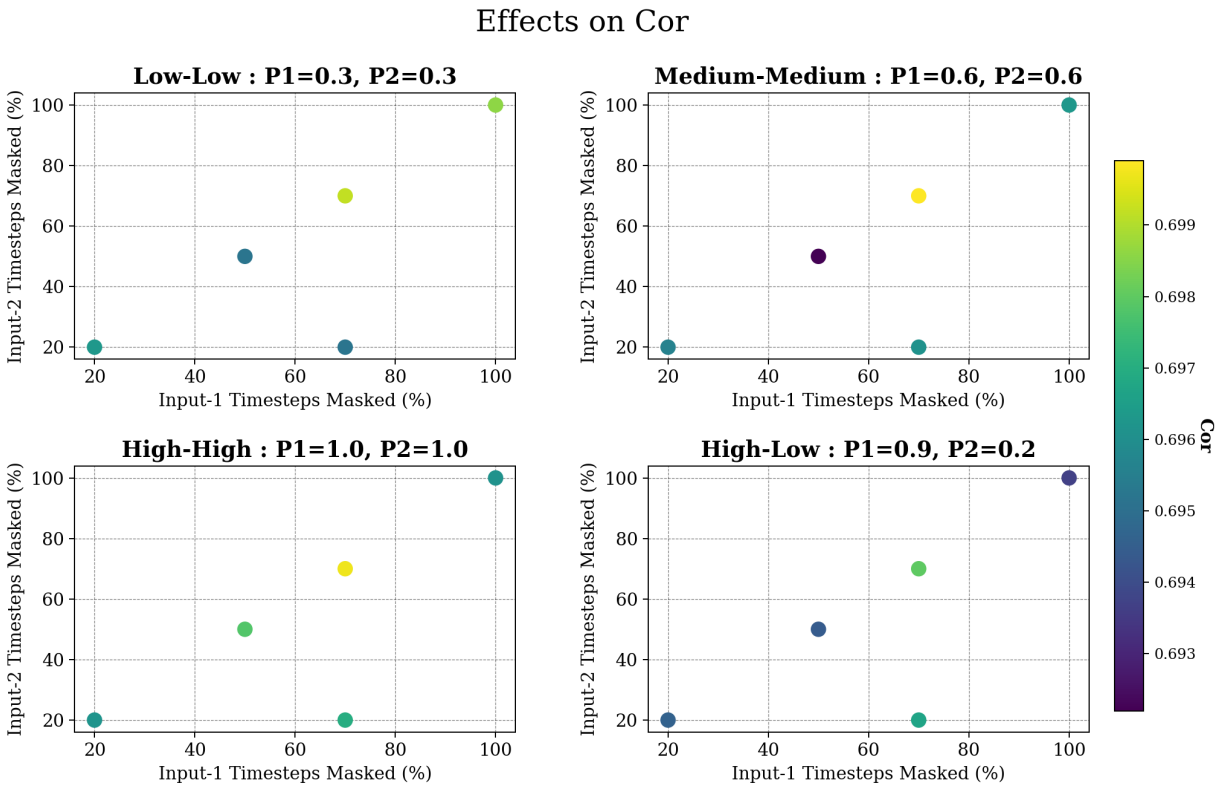


Figure A.2.3: Fine-tuning Masking parameters in relation to *Cor* (Pearson Correlation). Averages over 5 runs.

A.2.2 SeqAug

Graph created after the ablation study performed in order to fine-tune the SeqAug transformation.

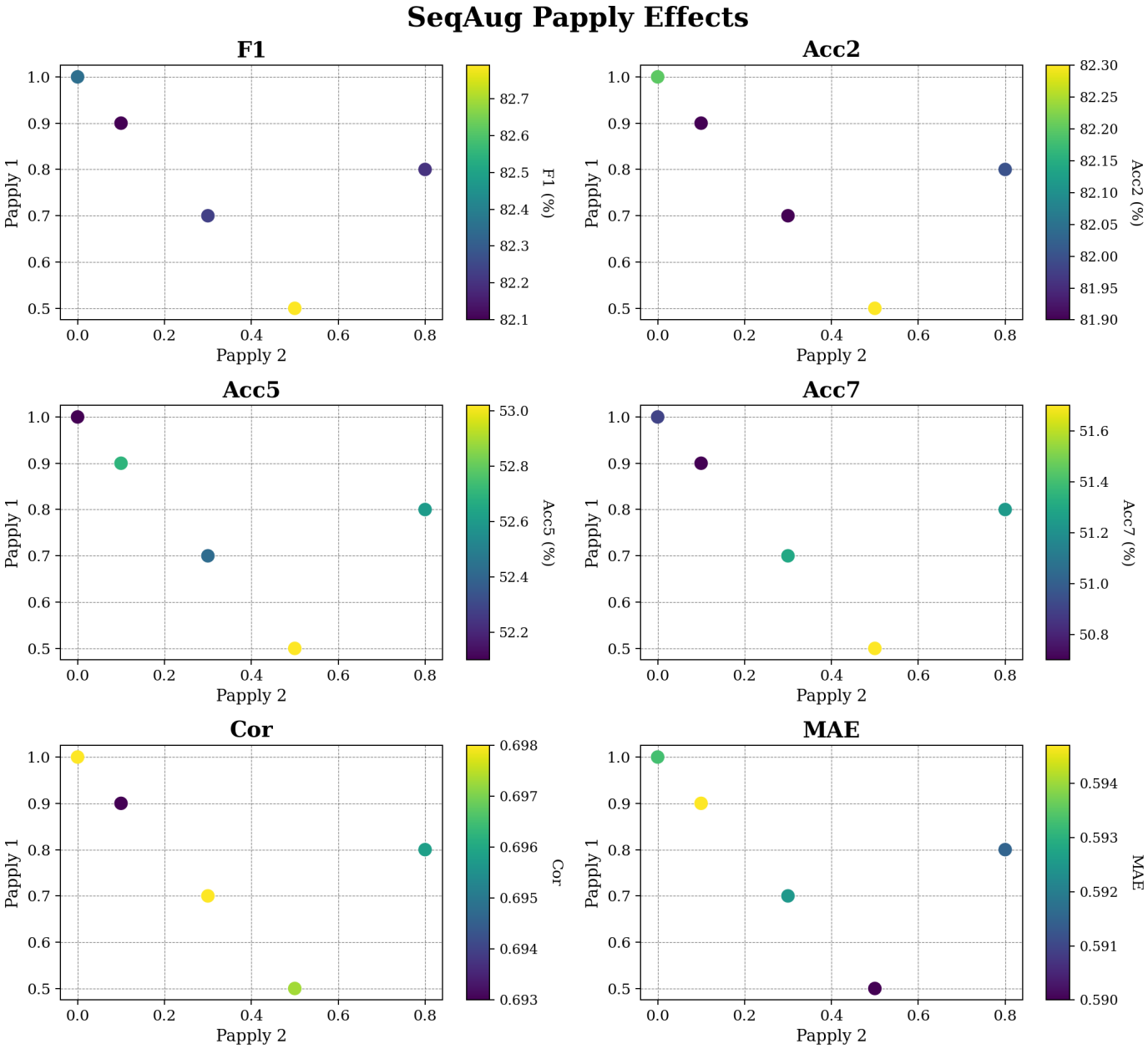


Figure A.2.4: Fine-tuning study of the SeqAug augmentation. Studying the effects of P_{apply} on each input stream. This graph studies the **SSL Pre-trained** models as discussed in section 7.1. **Averages over 5 runs.**

Bibliography

A. Agostinelli, T. I. Denk, Z. Borsos, J. Engel, M. Verzetti, A. Caillon, Q. Huang, A. Jansen, A. Roberts, M. Tagliasacchi, M. Sharifi, N. Zeghidour, and C. Frank. Musiclm: Generating music from text, 2023.

H. Akbari, L. Yuan, R. Qian, W.-H. Chuang, S.-F. Chang, Y. Cui, and B. Gong. Vatt: Transformers for multimodal self-supervised learning from raw video, audio and text, 2021. URL https://arxiv.org/abs/2104.11178.

J. Alayrac, A. Recasens, R. Schneider, R. Arandjelovic, J. Ramapuram, J. D. Fauw, L. Smaira, S. Dieleman, and A. Zisserman. Self-supervised multimodal versatile networks. *CoRR*, abs/2006.16228, 2020. URL https://arxiv.org/abs/2006.16228.

J.-B. Alayrac, J. Donahue, P. Luc, A. Miech, I. Barr, Y. Hasson, K. Lenc, A. Mensch, K. Millican, M. Reynolds, R. Ring, E. Rutherford, S. Cabi, T. Han, Z. Gong, S. Samangooei, M. Monteiro, J. Menick, S. Borgeaud, A. Brock, A. Nematzadeh, S. Sharifzadeh, M. Binkowski, R. Barreira, O. Vinyals, A. Zisserman, and K. Simonyan. Flamingo: a visual language model for few-shot learning, 2022.

S. Antol, A. Agrawal, J. Lu, M. Mitchell, D. Batra, C. L. Zitnick, and D. Parikh. VQA: Visual Question Answering. In *International Conference on Computer Vision (ICCV)*, 2015.

R. Arandjelović and A. Zisserman. Look, listen and learn, 2017a. URL https://arxiv.org/abs/1705.08168.

R. Arandjelović and A. Zisserman. Objects that sound, 2017b. URL https://arxiv.org/abs/1712.06651.

J. L. Ba, J. R. Kiros, and G. E. Hinton. Layer normalization, 2016.

D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2015.

R. Balestriero, M. Ibrahim, V. Sobal, A. Morcos, S. Shekhar, T. Goldstein, F. Bordes, A. Bardes, G. Mialon, Y. Tian, A. Schwarzschild, A. G. Wilson, J. Geiping, Q. Garrido, P. Fernandez, A. Bar, H. Pirsiavash, Y. LeCun, and M. Goldblum. A cookbook of self-supervised learning, 2023.

T. Baltrušaitis, P. Robinson, and L.-P. Morency. Openface: An open source facial behavior analysis toolkit. In *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1–10, 2016. doi: 10.1109/WACV.2016.7477553.

T. Baltrušaitis, C. Ahuja, and L.-P. Morency. Multimodal machine learning: A survey and taxonomy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(2):423–443, 2019. doi: 10.1109/TPAMI.2018.2798607.

A. Bardes, J. Ponce, and Y. LeCun. Vicreg: Variance-invariance-covariance regularization for self-supervised learning, 2022.

F. Bordes, R. Balestriero, Q. Garrido, A. Bardes, and P. Vincent. Guillotine regularization: Improving deep networks generalization by removing their head, 2022.

J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah. Signature verification using a "siamese" time delay neural network. In J. Cowan, G. Tesauro, and J. Alspector, editors, *Advances in Neural Information Processing Systems*, volume 6. Morgan-Kaufmann, 1993. URL https://proceedings.neurips.cc/paper/1993/file/288cc0ff022877bd3df94bc9360b9c5d-Paper.pdf.

T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language models are few-shot learners, 2020.

C. Busso, M. Bulut, C.-C. Lee, A. Kazemzadeh, E. Mower, S. Kim, J. N. Chang, S. Lee, and S. S. Narayanan. Iemocap: interactive emotional dyadic motion capture database. *Language Resources and Evaluation*, 42(4):335–359, Dec 2008. ISSN 1574-0218. doi: 10.1007/s10579-008-9076-6. URL https://doi.org/10.1007/s10579-008-9076-6.

A.-L. Cauchy. Méthode générale pour la résolution des systèmes d'équations simultanées. *Comptes rendus de l'Académie des sciences*, 25:536–538, 1847.

T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for contrastive learning of visual representations, 2020.

Y. Chung and J. R. Glass. Speech2vec: A sequence-to-sequence framework for learning word embeddings from speech. *CoRR*, abs/1803.08976, 2018. URL http://arxiv.org/abs/1803.08976.

Y.-A. Chung, W.-N. Hsu, H. Tang, and J. Glass. An unsupervised autoregressive model for speech representation learning, 2019. URL https://arxiv.org/abs/1904.03240.

O. Ciga, T. Xu, and A. L. Martel. Self supervised contrastive learning for digital histopathology, 2021.

G. Degottex, J. Kane, T. Drugman, T. Raitio, and S. Scherer. Covarep — a collaborative voice analysis repository for speech technologies. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 960–964, 2014a. doi: 10.1109/ICASSP.2014.6853739.

G. Degottex, J. Kane, T. Drugman, T. Raitio, and S. Scherer. Covarep — a collaborative voice analysis repository for speech technologies. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 960–964, 2014b. doi: 10.1109/ICASSP.2014.6853739.

J.-B. Delbrouck, N. Tits, M. Brousmiche, and S. Dupont. A transformer-based joint-encoding for emotion recognition and sentiment analysis. In *Second Grand-Challenge and Workshop on Multimodal Language (Challenge-HML)*. Association for Computational Linguistics, 2020. doi: 10.18653/v1/2020.challengehml-1.1. URL https://doi.org/10.18653%2Fv1%2F2020.challengehml-1.1.

J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.

C. Doersch, A. Gupta, and A. A. Efros. Unsupervised visual representation learning by context prediction, 2015. URL https://arxiv.org/abs/1505.05192.

P. Ekman, E. T. Rolls, D. I. Perrett, and H. D. Ellis. Facial expressions of emotion: An old controversy and new findings [and discussion]. *Philosophical Transactions: Biological Sciences*, 335(1273):63–69, 1992. ISSN 09628436. URL http://www.jstor.org/stable/55476.

J. L. Elman. Finding structure in time. *Cognitive science*, 14(2):179–211, 1990.

A. Frome, G. S. Corrado, J. Shlens, S. Bengio, J. Dean, M. A. Ranzato, and T. Mikolov. Devise: A deep visual-semantic embedding model. In C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013. URL https://proceedings.neurips.cc/paper_files/paper/2013/file/7cce53cf90577442771720a370c3c723-Paper.pdf.

E. Georgiou and A. Potamianos. Seqaug: Sequential feature resampling as a modality agnostic augmentation method, 2023.

E. Georgiou, G. Paraskevopoulos, and A. Potamianos. M3: MultiModal Masking Applied to Sentiment Analysis. In *Proc. Interspeech 2021*, pages 2876–2880, 2021. doi: 10.21437/Interspeech.2021-1739.

S. Gidaris, P. Singh, and N. Komodakis. Unsupervised representation learning by predicting image rotations, 2018. URL https://arxiv.org/abs/1803.07728.

I. Goodfellow, Y. Bengio, and A. Courville. *Deep learning*. MIT press, 2016.

P. Goyal, Q. Duval, I. Seessel, M. Caron, I. Misra, L. Sagun, A. Joulin, and P. Bojanowski. Vision models are more robust and fair when pretrained on uncurated images without supervision, 2022.

M. Gutmann and A. Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In Y. W. Teh and M. Titterington, editors, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 297–304, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. PMLR. URL https://proceedings.mlr.press/v9/gutmann10a.html.

R. Hadsell, S. Chopra, and Y. Lecun. Dimensionality reduction by learning an invariant mapping. pages 1735 – 1742, 02 2006. ISBN 0-7695-2597-0. doi: 10.1109/CVPR.2006.100.

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition, 2015.

D. Hendrycks and K. Gimpel. Gaussian error linear units (gelus), 2020.

D. Hendrycks, M. Mazeika, S. Kadavath, and D. Song. Using self-supervised learning can improve model robustness and uncertainty. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/a2b15837edac15df90721968986f7f8e-Paper.pdf.

S. Hochreiter. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 06(02):107–116, 1998. doi: 10.1142/S0218488598000094.

S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

iMotion. Emotient facet, 2017.

A. Joulin, L. van der Maaten, A. Jabri, and N. Vasilache. Learning visual features from large weakly supervised data. *CoRR*, abs/1511.02251, 2015. URL http://arxiv.org/abs/1511.02251.

D. Jurafsky and J. H. Martin. *Speech and Language Processing*. Prentice Hall, 3 edition, 2023. URL https://web.stanford.edu/~jurafsky/slp3/.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization, 2017.

R. Kiros, R. Salakhutdinov, and R. S. Zemel. Unifying visual-semantic embeddings with multimodal neural language models, 2014.

R. Krishnan, P. Rajpurkar, and E. J. Topol. Self-supervised learning in medicine and healthcare. *Nat Biomed Eng*, 6(12):1346–1352, Aug. 2022.

L. Li, K. Jamieson, G. DeSalvo, A. Rostamizadeh, and A. Talwalkar. Hyperband: A novel bandit-based approach to hyperparameter optimization, 2018.

J. W. K. Lilian Weng. Self-supervised learning self-prediction and contrastive learning, 2021. URL https://nips.cc/media/neurips-2021/Slides/21895.pdf.

W. Liu, Y. Wen, Z. Yu, M. Li, B. Raj, and L. Song. Sphereface: Deep hypersphere embedding for face recognition, 2018.

Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. Roberta: A robustly optimized bert pretraining approach, 2019. URL https://arxiv.org/abs/1907.11692.

J. Lu, D. Batra, D. Parikh, and S. Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/c74d97b01eae257e44aa9d5bade97baf-Paper.pdf.

A. L. Maas. Rectifier nonlinearities improve neural network acoustic models. In *Proceedings of the 30th International Conference on Machine Learning, Atlanta, Georgia, USA, 2013.*, volume 28, 2013.

D. Mahajan, R. Girshick, V. Ramanathan, K. He, M. Paluri, Y. Li, A. Bharambe, and L. van der Maaten. Exploring the limits of weakly supervised pretraining, 2018.

T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space, 2013a. URL https://arxiv.org/abs/1301.3781.

T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality, 2013b. URL https://arxiv.org/abs/1310.4546.

I. Misra and L. van der Maaten. Self-supervised learning of pretext-invariant representations, 2019. URL https://arxiv.org/abs/1912.01991.

Y. Mori, H. Takahashi, and R. Oka. Image-to-word transformation based on dividing and vector quantizing images with words. In *MISRM'99 First International Workshop on Multimedia Intelligent Storage and Retrieval Management*, 1999. URL citeseer.ist.psu.edu/368129.html.

M. Noroozi and P. Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. *CoRR*, abs/1603.09246, 2016. URL http://arxiv.org/abs/1603.09246.

G. Paraskevopoulos. slp, 2020. URL https://github.com/georgepar/slp.

G. Paraskevopoulos, E. Georgiou, and A. Potamianos. Mmlatch: Bottom-up top-down fusion for multimodal sentiment analysis, 2022. URL https://arxiv.org/abs/2201.09828.

J. Pennington, R. Socher, and C. D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014. URL http://www.aclweb.org/anthology/D14-1162.

H. Pham, P. P. Liang, T. Manzini, L.-P. Morency, and B. Poczos. Found in translation: Learning robust joint representations by cyclic translations between modalities, 2020.

B. A. Plummer, L. Wang, C. M. Cervantes, J. C. Caicedo, J. Hockenmaier, and S. Lazebnik. Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models, 2016.

K. Poulinakis, D. Drikakis, I. Kokkinakis, and M. Spottswood. Deep learning reconstruction of pressure fluctuations in supersonic shock-boundary layer interaction. *Physic of Fluids*.

A. Quattoni, M. Collins, and T. Darrell. Learning visual representations using images with captions. 06 2007. doi: 10.1109/CVPR.2007.383173.

A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever. Learning transferable visual models from natural language supervision, 2021. URL https://arxiv.org/abs/2103.00020.

A. Ramesh, M. Pavlov, G. Goh, S. Gray, C. Voss, A. Radford, M. Chen, and I. Sutskever. Zero-shot text-to-image generation, 2021.

D. E. Rumelhart and J. L. McClelland. *Learning Internal Representations by Error Propagation*, pages 318–362. 1987.

D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, Oct 1986. ISSN 1476-4687. doi: 10.1038/323533a0.

J. Schmidhuber. Learning complex, extended sequences using the principle of history compression. In *Neural Networks, 1992. IJCNN., International Joint Conference on*, pages 219–224. IEEE, 1992.

F. Schroff, D. Kalenichenko, and J. Philbin. FaceNet: A unified embedding for face recognition and clustering. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, jun 2015. doi: 10.1109/cvpr.2015.7298682. URL https://doi.org/10.1109%2Fcvpr.2015.7298682.

B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas. Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE*, 104(1):148–175, 2016. doi: 10.1109/JPROC.2015.2494218.

U. Singer, A. Polyak, T. Hayes, X. Yin, J. An, S. Zhang, Q. Hu, H. Yang, O. Ashual, O. Gafni, D. Parikh, S. Gupta, and Y. Taigman. Make-a-video: Text-to-video generation without text-video data, 2022.

S. Siriwardhana, T. Kaluarachchi, M. Billinghurst, and S. Nanayakkara. Multimodal emotion recognition with transformer-based self supervised feature fusion. *IEEE Access*, 8:176274–176285, 2020. doi: 10.1109/ACCESS.2020.3026823.

N. Srivastava and R. R. Salakhutdinov. Multimodal learning with deep boltzmann machines. In F. Pereira, C. Burges, L. Bottou, and K. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012. URL https://proceedings.neurips.cc/paper_files/paper/2012/file/af21d0c97db2e27e13572cbf59eb343d-Paper.pdf.

R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, second edition, 2018. URL http://incompleteideas.net/book/the-book-2nd.html.

M. Tagliasacchi, B. Gfeller, F. d. C. Quitry, and D. Roblek. Pre-training audio representations with self-supervision. *IEEE Signal Processing Letters*, 27:600–604, 2020. doi: 10.1109/LSP.2020.2985586.

Y. Taigman, M. Yang, M. Ranzato, and L. Wolf. Deepface: Closing the gap to human-level performance in face verification. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1701–1708, 2014. doi: 10.1109/CVPR.2014.220.

Y.-H. H. Tsai, S. Bai, P. P. Liang, J. Z. Kolter, L.-P. Morency, and R. Salakhutdinov. Multimodal transformer for unaligned multimodal language sequences, 2019. URL https://arxiv.org/abs/1906.00295.

A. van den Oord, Y. Li, and O. Vinyals. Representation learning with contrastive predictive coding. *CoRR*, abs/1807.03748, 2018. URL http://arxiv.org/abs/1807.03748.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need, 2017a. URL https://arxiv.org/abs/1706.03762.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need, 2017b.

P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, 11(110):3371–3408, 2010. URL http://jmlr.org/papers/v11/vincent10a.html.

L. Wang, P. Luc, A. Recasens, J.-B. Alayrac, and A. v. d. Oord. Multimodal self-supervised learning of general audio representations, 2021. URL https://arxiv.org/abs/2104.12807.

W. Wang, D. Tran, and M. Feiszli. What makes training multi-modal classification networks hard?, 2019. URL https://arxiv.org/abs/1905.12681.

Y. Wang, Y. Shen, Z. Liu, P. P. Liang, A. Zadeh, and L.-P. Morency. Words can shift: Dynamically adjusting word representations using nonverbal behaviors, 2018.

K. Weiss, T. M. Khoshgoftaar, and D. Wang. A survey of transfer learning. *Journal of Big Data*, 3 (1):9, May 2016. ISSN 2196-1115. doi: 10.1186/s40537-016-0043-6. URL https://doi.org/10.1186/s40537-016-0043-6.

J. Weston, S. Bengio, and N. Usunier. Wsabie: Scaling up to large vocabulary image annotation. In *Proceedings of the International Joint Conference on Artificial Intelligence, IJCAI*, 2011.

Z. Xie, Z. Zhang, Y. Cao, Y. Lin, J. Bao, Z. Yao, Q. Dai, and H. Hu. Simmim: A simple framework for masked image modeling, 2022.

J. Xu, T. Mei, T. Yao, and Y. Rui. Msr-vtt: A large video description dataset for bridging video and language. IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), June 2016. URL https://www.microsoft.com/en-us/research/publication/msr-vtt-a-large-video-description-dataset-for-bridging-video-and-language/.

X. Yang, X. He, Y. Liang, Y. Yang, S. Zhang, and P. Xie. Transfer learning or self-supervised learning? a tale of two pretraining paradigms, 2020.

J. Yuan and M. Liberman. Speaker identification on the SCOTUS corpus. *The Journal of the Acoustical Society of America*, 123($5_Supplement$) : 3878 − −3878, 052008. *ISSN*0001 − 4966. *doi* : . URL https://doi.org/10.1121/1.2935783.

A. Zadeh, P. P. Liang, N. Mazumder, S. Poria, E. Cambria, and L.-P. Morency. Memory fusion network for multi-view sequential learning, 2018a.

A. Zadeh, P. P. Liang, S. Poria, P. Vij, E. Cambria, and L.-P. Morency. Multi-attention recurrent network for human communication comprehension. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018b.

J. Zbontar, L. Jing, I. Misra, Y. LeCun, and S. Deny. Barlow twins: Self-supervised learning via redundancy reduction, 2021. URL https://arxiv.org/abs/2103.03230.

K. Zhang, Z. Zhang, Z. Li, and Y. Qiao. Joint face detection and alignment using multitask cascaded convolutional networks. *IEEE Signal Processing Letters*, 23(10):1499–1503, oct 2016a. 10.1109/lsp.2016.2603342. URL https://doi.org/10.1109%2Flsp.2016.2603342.

L. Zhang, G.-J. Qi, L. Wang, and J. Luo. Aet vs. aed: Unsupervised representation learning by auto-encoding transformations rather than data, 2019. URL https://arxiv.org/abs/1901.04596.

R. Zhang, P. Isola, and A. A. Efros. Colorful image colorization, 2016b. URL https://arxiv.org/abs/1603.08511.

H. Zhao, C. Gan, A. Rouditchenko, C. Vondrick, J. McDermott, and A. Torralba. URL https://arxiv.org/abs/1804.03160.

H. Zhao, C. Gan, A. Rouditchenko, C. Vondrick, J. McDermott, and A. Torralba. The sound of pixels. In *The European Conference on Computer Vision (ECCV)*, September 2018.