ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

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

Μοντέλα Διάχυσης με Εφαρμογές στην Αναπαράσταση Προσώπου και Σύνθεση Ομιλούντος Προσώπου

Diffusion Models with Applications in Face Reenactment and Talking Face Synthesis

Μεταπτυχιακή Διπλωματική Εργασία

Ιωάννης Πίκουλης

Επιβλέπων: Πέτρος Μαραγκός
Καθηγητής, ΣΗΜΜΥ ΕΜΠ

Συνεπιβλέπων: Δρ. Παναγιώτης Π. Φιλντίσης

Αθήνα, Ιούλιος 2023

[This page is left intentionally blank.]

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

Μοντέλα Διάχυσης με Εφαρμογές στην Αναπαράσταση Προσώπου και Σύνθεση Ομιλούντος Προσώπου

Diffusion Models with Applications in Face Reenactment and Talking Face Synthesis

Μεταπτυχιακή Διπλωματική Εργασία

Ιωάννης Πίκουλης

Επιβλέπων: Πέτρος Μαραγκός
Καθηγητής, ΣΗΜΜΥ ΕΜΠ

Συνεπιβλέπων: Δρ. Παναγιώτης Π. Φιλντίσης

Εγκρίθηκε από τη τριμελή εξεταστική επιτροπή. Ημερομηνία εξέτασης: 12/07/2023

(Υπογραφή)	(Υπογραφή)	(Υπογραφή)
.....	.....	.....
Πέτρος Μαραγκός	Αθανάσιος Ροντογιάννης	Γεράσιμος Ποταμιάνος
Καθηγητής	Αναπληρωτής Καθηγητής	Αναπληρωτής Καθηγητής
ΣΗΜΜΥ ΕΜΠ	ΣΗΜΜΥ ΕΜΠ	Τμ. ΗΜΜΥ, Παν/μιο Θεσσαλίας

Αθήνα, Ιούλιος 2023.

(Υπογραφή)

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

Περίληψη

Καθώς τα Παραγωγικά Αντιπαλικά Δίκτυα (ΠΑΔ) έχουν αποδειχθεί ικανά να παράγουν δείγματα υψηλής ποιότητας, κατάφεραν να τραβήξουν μεγάλη προσοχή από την ευρεία επιστημονική κοινότητα τα τελευταία 10 χρόνια. Όμως πρόσφατα, εμφανίστηκαν ακόμη πιο ισχυρά και ικανά παραγωγικά μοντέλα, όπως τα μοντέλα διάχυσης (ΜΔ), που αποτελούν απειλή για την κυριαρχία των ΠΑΔ στην παραγωγή συνθετικών δεδομένων. Τα ΜΔ βρίσκουν γρήγορα χρήση σε εφαρμογές της όρασης υπολογιστών όπως: αποθορυβοποίηση εικόνας, υπερ-ανάλυση εικόνας, σημασιολογική τμηματοποίηση, σημασιολογική σύνθεση εικόνας καθώς και μετάφραση εικόνας-σε-εικόνα.

Σε αυτή τη διατριβή εστιάζουμε ρητά στη μετάφραση εικόνας-σε-εικόνα και πιο συγκεκριμένα στη χειραγώγηση εκφράσεων του προσώπου (γνωστή και ως αναπαράσταση προσώπου). Η φωτορεαλιστική αναπαράσταση προσώπου μπορεί, μεταξύ άλλων, να χρησιμοποιηθεί για ψυχαγωγικούς σκοπούς, αλληλεπιδράσεις ανθρώπου-υπολογιστή και κινούμενα σχέδια προσώπου. Αυτός ο τομέας έχει προσελκύσει σημαντική προσοχή τόσο από τις ακαδημαϊκές όσο και από τις βιομηχανικές-ερευνητικές κοινότητες και έχει παράξει εκπληκτικά αποτελέσματα που διευρύνουν το εύρος της εφευρετικής επεξεργασίας εικόνας και της δημιουργίας περιεχομένου. Επιπλέον, εμβαθύνουμε στη σύνθεση ομιλούντων προσώπων, μια πρόσφατα αναδυόμενη εφαρμογή των ΜΔ που απολαμβάνει επίσης ένα ευρύ φάσμα μεταγενέστερων χρήσεων, όπως τις τηλεδιασκέψεις, μεταγλώττιση ταινιών και εικονικούς βοηθούς. Πιο συγκεκριμένα:

- Πραγματοποιήσαμε πειράματα σχετικά με την αναπαράσταση προσώπου με βάση τις πλήρως ανεξέλεγκτες συνθήκες της βάσης δεδομένων AffectNet. Πειράματα για τη σύνθεση ομιλούντος προσώπου πραγματοποιήθηκαν σε πιο ελεγχόμενες/εργαστηριακές προδιαγραφές, με βάση το σύνολο δεδομένων MEAD, και λαμβάνοντας υπόψη μόνο τις ακολουθίες βίντεο που απεικόνιζαν μετωπικές όψεις προσώπου.

- Από όσο γνωρίζουμε, η εργασία μας στην AffectNet αποτελεί το πρώτο ολοκληρωμένο σύνολο πειραμάτων που διεξήχθη στο προαναφερθέν σύνολο δεδομένων στο πλαίσιο της αναπαράστασης προσώπου βάσει ΜΔ.

- Αξιοποιήσαμε προεκπαιδευμένα μοντέλα CLIP με στόχο την καλύτερη καθοδήγηση των υποκείμενων διαδικασιών συναισθηματικής χειραγώγησης, εμπνεόμενοι από και επεκτείνοντας το μοντέλο DiffusionCLIP.

- Συγκρίναμε τη μέθοδό μας με την τελευταία λέξη της τεχνολογίας όσον αφορά ΠΑΔ, ξεπερνώντας το τελευταία ως προς την ποιότητα εικόνας και τη διατήρηση της ταυτότητας των υποκείμενων εικονιζόμενων προσώπων, επιτυγχάνοντας παράλληλα ανταγωνιστικά αποτελέσματα όσον αφορά την ακρίβεια μετάφρασης των συναισθημάτων.

- Από όσο γνωρίζουμε, προτείνουμε την πρώτη έγκυρη μεθοδολογία προσαρμογής που βασίζεται στην ανάγνωση των χειλιών, στο πλαίσιο της σύνθεσης ομιλούντος προσώπου με Μοντέλα Λανθάνουσας Διάχυσης.

Λέξεις Κλειδιά - αναπαράσταση προσώπου, σύνθεση ομιλούντος προσώπου, μοντέλα διάχυσης, φωτορεαλισμός, χειραγώγηση συναισθημάτων, καθοδήγηση CLIP, ανάγνωση χειλιών

Abstract

As generative adversarial networks (GANs) have proven capable of generating high-quality samples, they managed to draw a lot of attention in the last 10 years. But recently, even more potent generative methods, like diffusion models (DMs) have emerged, posing a threat to the dominance of GANs in the production of synthetic data. DMs are quickly finding use in both low-level and high-level vision tasks because of their incredible generative capabilities, including but not limited to image denoising, image super-resolution, semantic segmentation, semantic image synthesis and image-to-image translation.

In this thesis we explicitly focus on image-to-image translation and more specifically on *facial expression manipulation* (also known as face reenactment) on the basis of "in-the-wild" images. Photo-realistic face reenactment can be used for entertainment purposes, human-computer interactions, and facial animations, among other things. This area has been attracting considerable attention both from academic and industrial research communities and has produced stunning outcomes that broaden the scope of inventive image editing, and content creation. Moreover we delve into *talking face synthesis*, a newly emerging application of DMs that also enjoys a wide range of downstream uses, such as teleconferencing, movie dubbing and virtual assistants. More specifically we:

- Conduct experiments relative to face reenactment on the basis of the fully uncontrolled, "in-the-wild" settings of the AffectNet database. Experiments for talking face synthesis were performed on more controlled/lab settings, on the basis of the MEAD dataset, and only considering video sequences that depicted frontal face views.
- To the best of our knowledge, our work on AffectNet is the first fully-fledged set of experiments conducted on the aforementioned dataset in the context of diffusion-based facial reenactment.
- We leveraged CLIP pre-trained models with the aim of better guiding the underlying emotional manipulation processes. We drew inspiration from and extended the DiffusionCLIP framework.
- We compared our method with SOTA GAN-based models, surpassing the latter in terms of image quality and subject identity preservation, while achieving competitive results regarding emotion translation accuracy.
- To the best of our knowledge, we propose the first proper lip reading-based finetuning methodology, in the context of talking-face synthesis with Latent Diffusion Models.

Keywords - face reenactment, talking face synthesis, diffusion models, photorealism, emotion manipulation, CLIP guidance, lip reading

Contents

List of Figures

List of Tables

Notation

$\mathbb{N}$	The set of natural numbers
$\mathbb{R}$	The set of real numbers
$\{0, 1, ..., K\}$	The set of all integers between 0 and K
$[K]$	Equivalent to $\{0, 1, \dots, K-1\}$ or $\{1, \dots, K\}$, depending on the context
$\|\cdot\|_1$	Absolute-value (ℓ_1) norm
$\|\cdot\|_2$	Euclidean (ℓ_2) norm. Equivalent to $\|\cdot\|$
$\|\cdot\|_L$	Lipschitz constant
$\lfloor\cdot\rfloor$	Floor function
m	A scalar (real or integer)
$\boldsymbol{m}$	A row vector
$\boldsymbol{m}^\top$	Transpose of row vector $\boldsymbol{m}$
$\mathbf{m}$	A multidimensional tensor. Equivalent to $\mathbf{M}$
$M_{a_1,\dots,a_n}$	The element at position $(a_1, \dots, a_n)$ of a n-dimensional tensor $\mathbf{M}$
$M^{a_1,\dots,a_n}$	Equivalent to $M_{a_1,\dots,a_n}$
$\mathbf{M}^\top$	Transpose of 2D matrix $\mathbf{M}$
$\operatorname{diag}(\mathbf{M})$	A square, diagonal matrix with diagonal entries given by $\mathbf{M}$
$\det(\mathbf{M})$	The determinant of matrix $\mathbf{M}$
$\operatorname{tr}(\mathbf{M})$	The trace of matrix $\mathbf{M}$
$\mathbf{Df}$	The Jacobian of $\boldsymbol{f}$
$\mathbf{I}$	Identity matrix with dimensionality implied by context
$\mathbf{I}_n$	An $n \times n$ identity matrix
∇	Gradient vector
$\odot$	Element-wise (Hadamard) product
$\log x$	Natural logarithm of x. Equivalent to $\ln x$
$\sigma(\cdot)$	Element-wise sigmoid operator
$\operatorname{softmax}(\cdot)$	Row-wise softmax operator
$[\cdot]^+$	Equivalent to $\max(0, \cdot)$
$D_{\mathrm{KL}}(p \parallel q)$	Kullback–Leibler divergence of p from q
$\cos\langle \mathbf{x}, \mathbf{y} \rangle$	Equivalent to $\frac{\mathbf{x}^\top \mathbf{y}}{\|\mathbf{x}\| \cdot \|\mathbf{y}\|}$
$f : \mathbb{A} \to \mathbb{B}$	A function f with domain $\mathbb{A}$ and range $\mathbb{B}$
$x \sim D$	Random variable x has distribution D
$\mathcal{N}(x; \mu, \sigma^2)$	Gaussian distribution over x with mean μ and variance σ^2
$p(x)$	Probability distribution over x
$\mathbb{E}(x)$	Expected value of random variable x

Glossary–Acronyms

ΜΔ	Μοντέλο/α Διάχυσης
ΠΑΔ	Παραγωγικό(ά) Αντιπαλικό(ά) Δίκτυο(α)
ΑΠΜΔ	Αποθορυβοποιούμενο(α) Πιθανοτικό(ά) Μοντέλο(α) Διάχυσης
ΑΕΜΔ	Αποθορυβοποιούμενο(α) Έμμεσο(α) Μοντέλο(α) Διάχυσης
ΜΛΔ	Μοντέλο(α) Λανθάνουσας Διάχυσης
ΠΚΜC	Προσαρμογή Καθοδηγούμενη από Μοντέλα CLIP
ΑΑΟ	Αυτόματη Αναγνώριση Ομιλίας
DM	Diffusion Model
DDPM	Denoising Diffusion Probabilistic Model(s)
DDIM	Denoising Diffusion Implicit Model(s)
LDM	Latent Diffusion Model(s)
NCSN	Noise Conditional Score Network
VAE	Variational Autoencoder
HVAE	Hierarchical Variational Autoencoder
GAN	Generative Adversarial Network
EBM	Energy-Based Model
MCMC	Markov Chain Monte Carlo
ELBO	Evidence Lower Bound
FID	Fréchet inception distance
KID	Kernel Inception Distance
LPIPS	Learned Perceptual Image Patch Similarity
PSNR	Peak Signal-to-Noise Ratio
SSIM	Structural Similarity Measure Index
ODE	Ordinary Differential Equation
SDE	Stochastic Differential Equation
VE	Variance Exploding
VP	Variance Preserving
CLIP	Contrastive Language-Image Pre-training
ASR	Automatic Speech Recognition
CGF	CLIP-guided Finetuning
MLP	Multi-layer Perceptron
CNN	Convolutional Neural Network
evidence	The log-likelihood of the observed data
face reenactment	To transfer a target expression and pose to a source face, while preserving the identity/appearance of the latter
talking face/head	A person, shown only from the shoulders up, who speaks without the use of any illustrative material
viseme	A viseme represents the position of the face and mouth when saying a word; it is the visual equivalent of a phoneme

Εκτεταμένη Περίληψη στα Ελληνικά

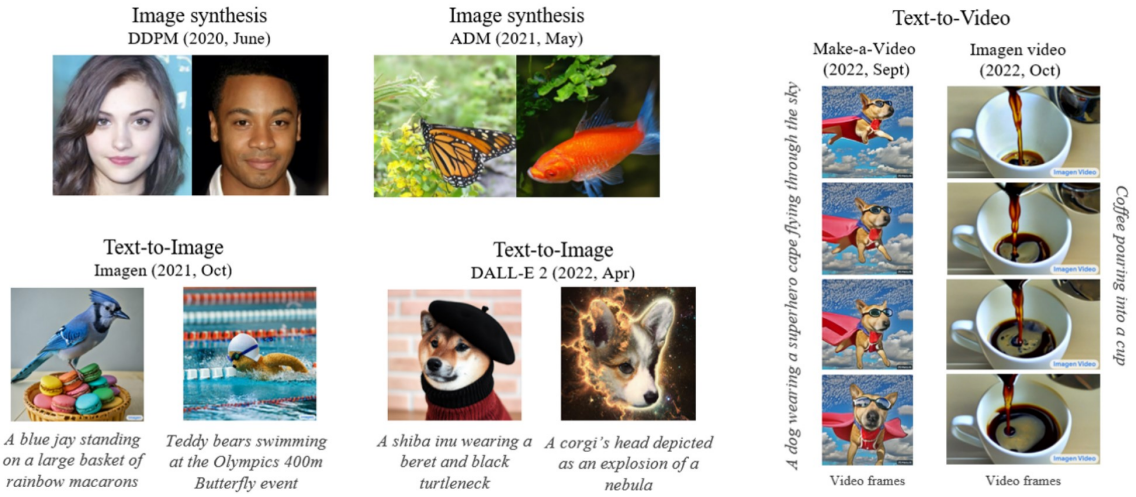
Α Εισαγωγή

Τα βαθιά παραγωγικά μοντέλα αποτελούν ένα από τα πιο ενδιαφέροντα υπολογιστικά εργαλεία σήμερα, δίνοντας ώθηση ακόμη και στην ανθρώπινη εφευρετικότητα. Καθώς τα Παραγωγικά Αντιπαλικά Δίκτυα (ΠΑΔ) έχουν αποδειχθεί ικανά να παράγουν δείγματα υψηλής ποιότητας, κατάφεραν να τραβήξουν μεγάλη προσοχή τα τελευταία 10 χρόνια. Όμως πρόσφατα, εμφανίστηκαν ακόμη πιο ισχυρές παραγωγικές μέθοδοι μάθησης, όπως τα μοντέλα διάχυσης αποτελώντας απειλή για την κυριαρχία των ΠΑΔ στην παραγωγή συνθετικών δεδομένων.

Λόγω της πιο συνεπούς και εύκολης εκπαίδευσής τους συγκριτικά με τα ΠΑΔ και των υψηλότερης ποιότητας δειγμάτων που παράγονται, τα μοντέλα διάχυσης ανέρχονται γρήγορα σε δημοτικότητα. Αυτά τα μοντέλα είναι σε θέση να ξεπεράσουν ορισμένες γνωστές ελλείψεις των ΠΑΔ, όπως η *κατάρρευση λειτουργίας*, το κόστη της ανταγωνιστικής μάθησης και η αποτυχία σύγκλισης. Σε αντίθεση με τα ΠΑΔ, τα οποία μαθαίνουν να ανακτούν τα αρχικά δεδομένα από τα θορυβώδη, μολύνοντας τα δεδομένα εκπαίδευσης με θόρυβο Gauss, τα μοντέλα διάχυσης χρησιμοποιούν μια διαφορετική προσέγγιση όσον αφορά την εκπαίδευση. Αυτά τα μοντέλα βρέθηκε επίσης ότι είναι κατάλληλα από την άποψη της επεκτασιμότητας και του παραλληλισμού, γεγονός που αυξάνει την ελκυστικότητά τους. Επιπλέον, επειδή το θεμέλιο της εκπαιδευτικής τους διαδικασίας είναι η διόρθωση τυχαίων παραποιήσεων που έγιναν στα αρχικά δεδομένα, μαθαίνουν μια κατανομή της οποίας τα δείγματα μοιάζουν πολύ με τα πραγματικά, επομένως τα δείγματα που δημιουργούνται μπορεί να είναι, εν τέλει, αρκετά ρεαλιστικά. Λόγω αυτών των χαρακτηριστικών, τα μοντέλα διάχυσης είχαν σημαντικό αντίκτυπο και αποτελούν τη τελευταία λέξη της τεχνολογίας στην παραγωγή εικόνων, καταφέρνοντας εκπληκτικά αποτελέσματα.

Τα μοντέλα διάχυσης βρίσκουν γρήγορα χρήση σε πολλαπλές εφαρμογές λόγω των απίστευτων δημιουργικών τους δυνατοτήτων, όπως ενδεικτικά η αποθορυβοποίηση εικόνας [65, 109], η ζωγραφική [44], η υπερ-ανάλυση εικόνας [91, 100], η σημασιολογική τμηματοποίηση [9, 56, 175], η σημασιολογική σύνθεση εικόνας [125] και η μετάφραση εικόνας-σε-εικόνα [19, 76, 130, 184]. Δεν αποτελεί έκπληξη, ότι υπήρξε μια συνεχής αύξηση στον αριθμό των ερευνητικών δημοσιεύσεων που γίνονται σε αυτόν τον τομέα μετά την σημαντική ανακάλυψη των πιθανοτικών μοντέλων διάχυσης [65] έναντι της αρχικής ιδέας της μοντελοποίησης διάχυσης [146], με αποτέλεσμα νέα συναρπαστικά μοντέλα αναδύονται κάθε μέρα. Ιδιαίτερα με τα GLIDE [110], DALL-E 2 [122], Imagen [131] και Stable [125] και τα μοντέλα που επέτρεψαν τη δημιουργία κειμένου-σε-εικόνα υψηλής ποιότητας, η μοντελοποίηση με βάση τη διάχυση γνώρισε σημαντικό ενθουσιασμό στα μέσα κοινωνικής δικτύωσης. Οι τεχνικές δημιουργίας κειμένου-σε-βίντεο και κειμένου-σε-3Δ, όπως το Imagen Video [67], Make-A-Video [145], και DreamFusion [118] οι οποίες παράγουν βίντεο και 3Δ αναπαραστάσεις που φαίνονται απίστευτα ρεαλιστικές, αποτελούν πρόσφατο σημείο συζήτησης γύρω από τα μοντέλα διάχυσης. Το Σχ. A.1 επεξηγεί επιμελημένα παραδείγματα που αντιστοιχούν σε ορισμένες από τις προαναφερθείσες διεργασίες και μοντέλα.

Η τρέχουσα διατριβή περιστρέφεται γύρω από τη χειραγώγηση της έκφρασης του προσώπου σε εικόνες και βίντεο. Για το σκοπό αυτό, θα προσπαθήσουμε να αξιοποιήσουμε την εκφραστική δύναμη των μοντέλων διάχυσης, τα οποία έχουν ήδη επιδείξει συναρπαστική ποιότητα στη σύνθετη μοντελοποίηση εικόνων, αλλά εξακολουθούν να παραμένουν ένα κάπως νέο και ανεξερεύνητο μέρος της διαθέσιμης βιβλιογραφίας. Στις επερχόμενες παραγράφους αυτού του εισαγωγικού κεφαλαίου, θα συζητήσουμε εν συντομία τις μεθοδολογίες χειραγώγησης προσώπου, αλλά θα εξετάσουμε επίσης τις τεχνολογίες deepfake για την εναλλαγή προσώπου, καθώς και πιθανά οφέλη και απειλές που προκύπτουν ως επακόλουθα



Σχήμα A.1: Επιμελημένα παραδείγματα από σύνθεση εικόνας, κείμενο-σε-εικόνα και κείμενο-σε-βίντεο βασισμένα σε [33, 65, 67, 122, 145]. Πηγή: [164].

της εφαρμογής αυτής της τεχνολογίας.

A.1 Deepfakes

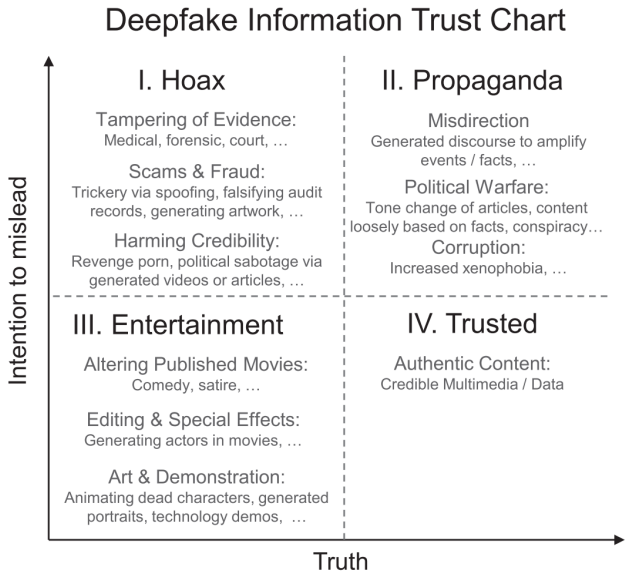
Τα Deepfakes αποτελούν υπερρεαλιστικά βίντεο που έχουν τροποποιηθεί ψηφιακά για να αναπαριστούν άτομα που λένε και κάνουν πράγματα που δεν έχουν συμβεί ποτέ στην πραγματικότητα. Χρησιμοποιούν τεχνικές βαθιάς μάθησης με στόχο την παραγωγή «ψεύτικων» (fake) εικόνων. Πιο συγκεκριμένα, οι μέθοδοι deepfake χρησιμοποιούν νευρωνικά δίκτυα για να μάθουν να αναπαράγουν τις εκφράσεις του προσώπου, τις ιδιαιτερότητες, τον τρόπο ομιλίας και διακυμάνσεις αυτής αναλύοντας τεράστιες ποσότητες δειγμάτων δεδομένων. Προκειμένου να εκπαιδευτεί ένα σύστημα βαθιάς μάθησης για εναλλαγή προσώπων (face swapping), η διαδικασία περιλαμβάνει τη χρήση υλικού βίντεο από δύο άτομα. Με άλλα λόγια, τα Deepfakes, χρησιμοποιούν τεχνολογία χαρτογράφησης προσώπου και τεχνητή νοημοσύνη για να αντικαταστήσουν το πρόσωπο ενός ατόμου σε ένα βίντεο με το πρόσωπο ενός άλλου ατόμου. Μόλις το 2017, όταν ένας χρήστης του Reddit δημοσίευσε ηχογραφήσεις διάσημων ανθρώπων σε μη κολακευτικά σεξουαλικά περιβάλλοντα, τα deepfakes άρχισαν να κερδίζουν την προσοχή του κοινού.

Σε γενικές γραμμές, τα deepfakes στοχεύουν ιστότοπους κοινωνικής δικτύωσης, όπου οι θεωρίες συνωμοσίας, οι φήμες και η παραπληροφόρηση μπορούν να εξαπλωθούν γρήγορα, επειδή οι χρήστες τείνουν να ακολουθούν το «κοπάδι». Εν τω μεταξύ, η τρέχουσα «πληροφοριακή αποκάλυψη» ενθαρρύνει τους ανθρώπους να εμπιστεύονται μόνο πληροφορίες που επιβεβαιώνουν τις προϋπάρχουσες πεποιθήσεις τους και προέρχονται από τα κοινωνικά τους δίκτυα, όπως η οικογένεια, οι στενοί φίλοι ή οι συγγενείς. Φθηνά ψεύτικα ή βίντεο χαμηλής ποιότητας με ελάχιστα τροποποιημένο πραγματικό περιεχόμενο, είναι ήδη διάχυτα λόγω προσβασιμότητας σε φθηνό τεχνολογικό εξοπλισμό, όπως μονάδες επεξεργασίας γραφικών (GPUs). Υπάρχει ένας αυξανόμενος αριθμός λογισμικού ανοιχτού κώδικα που διατίθεται για τη δημιουργία ρεαλιστικού, υψηλής ποιότητας deepfake υλικού, με αποτέλεσμα την συνεχή αύξηση της παραπληροφόρησης μέσω αυτής της τεχνολογίας. Το Σχ. A.2 παρουσιάζει ένα γράφημα εμπιστοσύνης πληροφοριών σχετικά με τα deepfakes.

Εφαρμογές

Ο κινηματογράφος, τα εκπαιδευτικά μέσα, οι ψηφιακές επικοινωνίες, τα βιντεο-παιχνίδια, η ψυχαγωγία, τα μέσα κοινωνικής δικτύωσης, η υγειονομική περίθαλψη, η επιστήμη των υλικών και πολλοί οικονομικοί τομείς, όπως το ηλεκτρονικό εμπόριο και η μόδα, είναι μόνο μερικές από τις βιομηχανίες που δυνητικά επωφελούνται από την τεχνολογία deepfake.

Πρώτον, η τεχνολογία deepfake έχει πολλά πλεονεκτήματα για την κινηματογραφική βιομηχανία. Για παράδειγμα, μπορεί να χρησιμοποιηθεί για την ενημέρωση ή τροποποίηση ορισμένων καρέ ενός φιλμ, πράγμα που ελαχιστοποιεί την ανάγκη επαναληπτικών γυρισμάτων σε σκηνές μιας επερχόμενης ταινίας. Εναλλακτικά, μπορεί να χρησιμοποιηθεί για τη δημιουργία τεχνητών φωνών για καλλιτέχνες που μπορεί

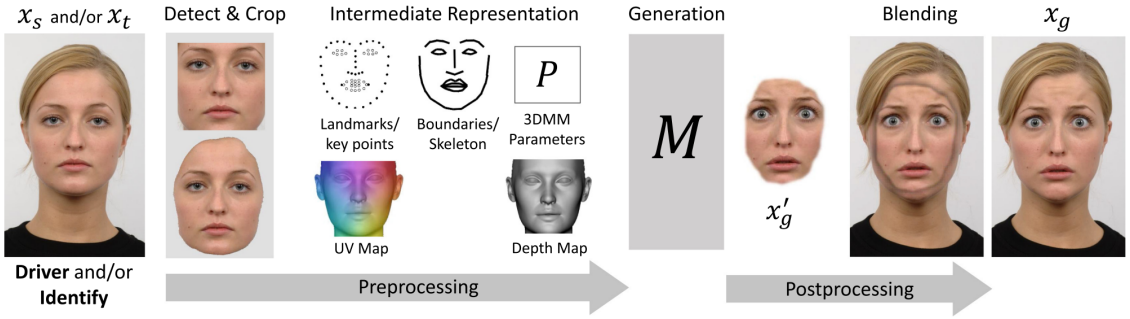


Σχήμα Α.2: Ένα γράφημα εμπιστοσύνης πληροφοριών σχετικά με τα deepfakes. Πηγή: [105].

να έχασαν τις δικές τους λόγω κάποιας ασθένειας. Οι κινηματογραφιστές θα μπορούν να αναπαράγουν εμβληματικές στιγμές ταινιών, να αναπτύσσουν νέες ταινίες με ερμηνευτές που έχουν πεθάνει, να εφαρμόζουν ειδικά εφέ και πρωτοποριακή επεξεργασία προσώπου στο post-production και να μετατρέπουν ερασιτεχνικό υλικό σε κομψά έργα. Αυτή η τεχνολογία επιτρέπει επίσης την αυτόματη και ρεαλιστική μεταγλώττιση φωνής για ταινίες σε οποιαδήποτε γλώσσα, βελτιώνοντας την εμπειρία προβολής για διαφορετικά ακροατήρια ταινιών και εκπαιδευτικών μέσων. Τα γλωσσικά εμπόδια ξεπεράστηκαν από μια παγκόσμια εκστρατεία ευαισθητοποίησης για την ελονοσία το 2019 με τον David Beckham, χάρη σε μια διδακτική διαφήμιση που χρησιμοποιούσε τεχνολογία αλλαγής φωνής/οπτικής για να τον κάνει να φαίνεται δίγλωσσος. Αυτο μπορεί να παρομοιαστεί με το πως η τεχνολογία deepfake μπορεί να βελτιώσει την οπτική επαφή και να καταρρίψει τα γλωσσικά εμπόδια κατά τη διάρκεια μιας τηλεδιάσκεψης, μεταφράζοντας λέξεις ενώ ταυτόχρονα αλλάζει τις κινήσεις των χειλιών και του προσώπου.

Η τεχνολογία deepfake επιτρέπει ψηφιακούς σωσίες, έξυπνους βοηθούς με ρεαλιστικό ήχο, εμφάνιση και αυξημένη τηλεπαρουσία σε διαδικτυακά παιχνίδια και περιβάλλοντα εικονικής συνομιλίας. Αυτό έχει ως αποτέλεσμα καλύτερη επικοινωνία στο διαδίκτυο και διαπροσωπικές αλληλεπιδράσεις. Η τεχνολογία μπορεί επίσης να είναι επωφελής στον κοινωνικό και ιατρικό τομέα. Τα deepfakes μπορεί να βοηθήσουν ένα αγαπημένο πρόσωπο που πενθεί να αποχαιρετήσει έναν αποθανόντα φίλο «φέρνοντάς τον ξανά στη ζωή» με ψηφιακό τρόπο. Αυτό μπορεί να βοηθήσει τους ανθρώπους να αντιμετωπίσουν τον θάνατο ενός αγαπημένου τους προσώπου αποτελεσματικότερα. Επιπλέον, η τεχνολογία μπορεί να χρησιμοποιηθεί για την ψηφιακή ανακατασκευή του άκρου ενός ακρωτηριασμένου ατόμου ή για να βοηθήσει τα διεμφυλικά άτομα να ταυτιστούν καλύτερα με το φύλο που προτιμούν. Ακόμη και οι ενήλικες που πάσχουν από Αλτσχάιμερ μπορούν να επωφεληθούν από την αυτήν τη τεχνολογία αλληλεπιδρώντας με ένα νεότερο πρόσωπο που μπορεί να θυμούνται καλύτερα. Προκειμένου να επιταχυνθεί η ανάπτυξη νέων υλικών και ιατρικών θεραπειών, οι ερευνητές έχουν επίσης διερευνήσει τη χρήση των ΠΑΔ για την ανίχνευση ανωμαλιών στις ακτίνες Χ.

Επιπλέον, οι εταιρείες διανομής τεχνολογίας ενδιαφέρονται για τη δυνατότητα εφαρμογής των deepfake για την επωνυμία, επειδή έχει τη δυνατότητα να αλλάξει σημαντικά το ηλεκτρονικό εμπόριο και τη διαφήμιση. Τα deepfakes επιτρέπουν επίσης τη δημιουργία υπερπροσωπικού περιεχομένου που μπορεί να μετατρέπει τους χρήστες σε μοντέλα. Για παράδειγμα, η τεχνολογία επιτρέπει τις εικονικές εξαρτήσεις, ώστε οι χρήστες να μπορούν να δουν πως θα τους φαίνεται μια στολή/ρούχο πριν κάνουν μια αγορά και να μπορούν να παράγουν προσαρμοσμένες διαφημίσεις μόδας με βάση τον θεατή, την ώρα της ημέρας και τον καιρό.



Σχήμα Α.3: Ένας τυπικός αγωγός επεξεργασίας αναπαράστασης ενός συστήματος deepfake, όπου συνήθως εκτελείται μόνο ένα υποσύνολο των βημάτων που απεικονίζονται. Πηγή: [105].

Πιθανές Απειλές

Τα deepfakes αποτελούν σοβαρή απειλή για την κοινωνία, το πολιτικό σύστημα και την οικονομία επειδή α) ασκούν πίεση στους δημοσιογράφους που προσπαθούν να διακρίνουν μεταξύ πραγματικών και ψεύτικων ειδήσεων, β) θέτουν σε κίνδυνο την εθνική ασφάλεια διαδίδοντας προπαγάνδα και ανακατεύοντας τις εκλογές, γ) υπονομεύουν την εμπιστοσύνη του κοινού σε κυβερνητικές πληροφορίες και δ) εγείρουν ανησυχίες τόσο σε άτομα όσο και σε επιχειρήσεις για την ευρύτερη ασφάλεια στον κυβερνοχώρο.

Αρχικά, είναι πολύ πιθανό ο τομέας των μέσων ενημέρωσης να αντιμετωπίσει ένα σημαντικό πρόβλημα σε σχέση με την εμπιστοσύνη των πελατών/χρηστών. Τα deepfakes είναι πιο επικίνδυνα από τα συμβατικά fake news καθώς είναι πιο δύσκολο να εντοπιστούν, με αποτέλεσμα τη διάδοση παραπληροφόρησης. Για παράδειγμα, η τεχνολογία αυτή επιτρέπει τη δημιουργία βίντεο-ειδήσεων που φαίνονται αληθινά αλλά στην πραγματικότητα δεν είναι, θέτοντας σε κίνδυνο την αξιοπιστία των δημοσιογράφων και των μέσων ενημέρωσης γενικότερα. Επίσης, η απόκτηση πρόσβασης σε βίντεο που τραβήχτηκε από «αυτόπτη» μάρτυρα ενός περιστατικού μπορεί να δώσει σε έναν ειδησεογραφικό οργανισμό ανταγωνιστικό πλεονέκτημα, αλλά ο κίνδυνος αυξάνεται εάν η προσφερόμενη ταινία έχει όντως παραποιηθεί.

Η κοινότητα των μυστικών υπηρεσιών ανησυχεί για τα deepfakes που δυνητικά χρησιμοποιούνται για να υπονομεύσουν προεκλογικές εκστρατείες και να θέσουν σε κίνδυνο την εθνική ασφάλεια διαδίδοντας πολιτική προπαγάνδα. Οι υπηρεσίες πληροφοριών των ΗΠΑ έχουν συχνά εκδώσει προειδοποιήσεις σχετικά με τον κίνδυνο ξένων παρεμβάσεων στην Αμερικανική πολιτική, ιδιαίτερα ενόψει εκλογών. Στους σημερινούς πολέμους παραπληροφόρησης, η εισαγωγή λέξεων σε ένα δημοφιλές βίντεο μπορεί να αποτελέσει ένα ισχυρό όπλο, επειδή μπορεί εύκολα να επηρεάσει τη γνώμη των ψηφοφόρων. Ένα ψεύτικο βίντεο ενός πολιτικού που χρησιμοποιεί ρατσιστικές συκοφαντίες ή δωροδοκία, έναν προεδρικό υποψήφιο που ομολογεί ένα έγκλημα, ειδοποιεί ένα άλλο έθνος για επικείμενο πόλεμο, έναν κυβερνητικό αξιωματούχο που φαίνεται να βρίσκεται σε επισφαλή θέση, να ομολογεί σχέδιο συνωμοσίας, ή στρατιώτες που δολοφονούν αμάχους στο εξωτερικό, θα μπορούσαν να παραχθούν από μια εχθρική υπηρεσία πληροφοριών. Ενώ τέτοιες «φτιαχτές» ταινίες πιθανότατα θα πυροδοτούσαν αναταραχές και διακοπές πιθανών εκλογών, ξένα έθνη μπορεί να καταλήξουν να εκτελούν εξωτερική πολιτική βασισμένη σε μυθοπλασίες, οδηγώντας ακόμη και στο ξέσπασμα πολέμου.

Ένας άλλος κίνδυνος που προκαλείται από την τεχνολογία deepfake είναι ζητήματα που σχετίζονται με την ασφάλεια στον κυβερνοχώρο. Ο εταιρικός κόσμος έχει ήδη εκφράσει ενδιαφέρον να υπερασπιστεί τον εαυτό του από ιογενείς απάτες, επειδή τα deepfakes θα μπορούσαν να χρησιμοποιηθούν για τη χειραγώγηση του χρηματιστηρίου και της αγοράς γενικά, για παράδειγμα, εκθέτοντας έναν διευθύνοντα σύμβουλο χρησιμοποιώντας φυλετικές ή έμφυλες συκοφαντίες, ανακοινώνοντας μια ψεύτικη συγχώνευση, διογκώνοντας οικονομικές απώλειες, κήρυξη πτώχευσης ή ομολογία ενός ένοχου εγκλήματος. «Deepfaked» ανακοινώσεις νέων προϊόντων ή πορνογραφικό υλικό θα μπορούσαν επίσης να χρησιμοποιηθούν για να βλάψουν τη φήμη μιας εταιρείας, να απειλήσουν ή ακόμα και να εκβιάσουν την εκάστοτε διοίκηση. Επιπλέον, η εν λόγω τεχνολογία καθιστά δυνατή την πλαστοπροσωπία ενός στελέχους σε πραγματικό χρόνο μέσω ψηφιακών μέσων, όπως ζητώντας από έναν εργαζόμενο να παράσχει επειγόντως χρήματα ή να αποκαλύψει ευαίσθητες πληροφορίες. Τέλος, δύναται να δημιουργηθεί μια ψεύτικη ταυτότητα και σε ηχογραφήσεις ζωντανής ροής (live-stream), να αλλαχτεί το πρόσωπο ενός ενήλικα σε πρόσωπο ενός νεαρού παιδιού, γεγονός που εγείρει ανησυχίες για το πως θα μπορούσαν οι παιδόφιλοι/παιδεραστές να χρησιμοποιήσουν την εν λόγω τεχνολογία.

Α.2 Αναπαράσταση Προσώπου

Ένας τύπος deepfake που μπορεί να ενισχύσει σημαντικά τον αντίκτυπο αυτής της τεχνολογίας και να διευκολύνει την ενσωμάτωσή της στη βιομηχανία VFX είναι η δυνατότητα απόκτησης σημασιολογικού ελέγχου στα συναισθήματα που εκφράζονται μέσω των εκφράσεων του προσώπου. Η σημασία αυτού του είδους χειραγώγησης αποδεικνύεται ιδιαιτέρως χρήσιμη κατά τη διάρκεια των γυρισμάτων ταινιών, καθώς, παρά την προγραμματισμένη φύση των προφορικών λέξεων, η απόκτηση του απαιτούμενου συναισθήματος του ηθοποιού απαιτεί συχνά πολλές προσπάθειες. Η αλλαγή της απόδοσης του προσώπου θα μπορούσε να γίνει εύκολα σε μεταπαραγωγικό στάδιο, προσφέροντας μια ισχυρή λύση στην επεξεργασία συναισθημάτων.

Η αναπαράσταση προσώπου μετατρέπει το υποκείμενο πρόσωπο σε μαριονέτα, παρέχοντας στους επιτιθέμενους τον μεγαλύτερο βαθμό ελευθερίας ώστε έχουν τον αντίκτυπο που θέλουν. Πρέπει να επισημάνουμε ότι η αναπαράσταση προσώπου υπήρχε πριν γίνουν κοινά τα deepfakes. Το 2003, οι ερευνητές μεταμόρφωσαν μοντέλα τρισδιάστατων σαρωμένων κεφαλών [14]. Το 2005, αποδείχθηκε πως αυτό είναι εφικτό χωρίς ένα τρισδιάστατο μοντέλο [20], μέσω παραμόρφωσης με αντίστοιχες παρόμοιες υφές [53]. Αργότερα, αποδείχθηκε πως μπορούν να χρησιμοποιηθούν τρισδιάστατα παραμετρικά μοντέλα για την επίτευξη υψηλής ποιότητας αποτελεσμάτων και σε πραγματικό χρόνο με ανίχνευση βάθους και συμβατικές κάμερες [157, 159, 160]. Ανεξάρτητα από αυτό, στις μέρες μας, οι προσεγγίσεις που βασίζονται στη βαθιά μάθηση είναι γνωστές ως ο κυρίαρχος τρόπος δημιουργίας αληθοφανούς περιεχομένου. Ένας τυπικός και γενικός αγωγός επεξεργασίας αναπαράστασης προσώπου φαίνεται στο Σχ. Α.3.

Για τους σκοπούς αυτής της εισαγωγικής επισκόπησης, η πρώτη ενότητα θα επικεντρωθεί στις πιο γνωστές μεθόδους: Face2Face [157] και NeuralTextures [158], τα οποία αντικαθιστούν την έκφραση του προσώπου ενός ατόμου σε ένα βίντεο με την έκφραση του προσώπου ενός άλλου ατόμου (επίσης σε ένα βίντεο). Αρχικά, η τεχνική γραφικών υπολογιστή γνωστή ως Face2Face διατηρούσε την ταυτότητα του ατόμου-στόχου, ενώ μετέφερε το συναίσθημα από ένα βίντεο πηγής σε ένα βίντεο-στόχο. Η επιλογή του βασικού καρέ αναφοράς γινόταν χειροκίνητα. Για την παρακολούθηση του συναισθήματος στα υπόλοιπα καρέ, χρησιμοποιήθηκαν τα πρώτα καρέ του εκάστοτε βίντεο για τη δημιουργία μιας προσωρινής ταυτότητας προσώπου (3Δ μοντέλο). Τέλος, δημιουργήθηκαν ψεύτικα βίντεο με την εφαρμογή 76 συντελεστών ανάμειξης σχήματος (blendshape) που αποτελούσαν τις παραμέτρους έκφρασης πηγής κάθε καρέ στο βίντεο-στόχο. Αργότερα, εισήχθη μια νέα στρατηγική εκμάθησης βασισμένη στο NeuralTextures και στη βάση δεδομένων FaceForensics++. Η προαναφερθείσα τεχνική μαθαίνει μια νευρωνική αναπαράσταση υφής του ατόμου-στόχου, συμπεριλαμβανομένου ενός δικτύου απόδοσης, χρησιμοποιώντας τα αρχικά δεδομένα βίντεο.

Γενικά, η ανάπτυξη των ΠΑΔ [55] έχει διεγείρει ένα διευρυνόμενο σώμα έρευνας και μελέτης στον τομέα της χειραγώγησης εικόνων. Όταν η σύνθεση μιας εικόνας απαιτείται να πραγματοποιηθεί βάσει κάποιας άλλης εικόνας, στη συντριπτική πλειονότητα των έργων, χρησιμοποιείται κάποιας μορφής δεσμευμένη γεννήτρια [71]. Μέσω της έννοιας της κυκλικής συνέπειας [187], καθίσταται δυνατή η μετάφραση εικόνων μεταξύ διαφορετικών τομέων διατηρώντας παράλληλα το περιεχόμενο της εκάστοτε αρχικής εικόνας. Η εφαρμογή τέτοιων μεθόδων στις εικόνες προσώπων κατέστησε δυνατή την αλλαγή ορισμένων χαρακτηριστικών του προσώπου, όπως το φύλο και το χρώμα των μαλλιών. Η ικανότητα αλλαγής των συναισθημάτων του προσώπου σε εικόνες μεταφράζοντας τα σύμφωνα με μια δεδομένη σημασιολογική ετικέτα (π.χ. χαρά, θυμός, θλίψη, κ.λπ.) υλοποιήθηκε επιτυχώς στα πλαίσια του μοντέλου StarGAN [25]. Επιπλέον, πολλές γνωστές τεχνικές χειρισμού έκφρασης βάσει εικόνας και με αρχιτεκτονική ΠΑΔ, είναι οι: StarGAN v2 [26], GANimation [120], GANmut [30], ExprGAN [34], FACEGAN [162] και ICface [163]. Μερικές από αυτές εξέτασαν εναλλακτικές κρυφές αναπαραστάσεις για τη μοντελοποίηση του ανθρώπινου συναισθηματικού φάσματος. Για παράδειγμα, το ExprGAN χρησιμοποιεί ετικέτες συνεχών συναισθημάτων που περιγράφουν την *ένταση* των απεικονιζόμενων εκφράσεων του προσώπου, ενώ στο [94], χρησιμοποιήθηκε ο 2Δ χώρος σθένους-διέγερσης (valence-arousal) [103]. Πιο πρόσφατα, το GANmut πρότεινε έναν τρόπο απόκτησης ενός δισδιάστατου δεσμευμένου ερμηνεύσιμου συστήματος ετικετών ακόμη και όταν χρησιμοποιείται ένα σύνολο δεδομένων που χαρακτηρίζεται με αποκλειστικά κατηγορικές ετικέτες βασικών συναισθημάτων.

Β Μοντέλα Διάχυσης

Στο παρόν τμήμα της περίληψης θα προσπαθήσουμε να θέσουμε τις θεωρητικές βάσεις σχετικά με τα Μοντέλα Διάχυσης (ΜΔ) στο πλαίσιο της παραγωγικής μοντελοποίησης, έτσι ώστε να περιγράψουμε

με ακρίβεια τις αρχιτεκτονικές που θα αποτελέσουν στη συνέχεια τους δομικούς λίθους των δικών μας υπολογιστικών μοντέλων. Τα μοντέλα διάχυσης ανήκουν στη κλάση των βαθιών παραγωγικών μοντέλων που παράγουν δείγματα αντιστρέφοντας μια διαδικασία διαφθοράς/θορυβοποίησης. Το επίπεδο θορύβου τυπικά υποδηλώνεται με ένα χρονικό δείκτη t, όπου $t = 0$ αντιστοιχεί σε καθαρές και $t = 1$ σε πλήρως θορυβώδεις εικόνες. Η διαδικασία διάχυσης μπορεί να είναι διακριτή ή συνεχής. Οι δύο γενικές κατηγορίες μοντέλων διάχυσης είναι τα Score-Based [149, 150, 153] και τα Αποθορυβοποιούμενα Πιθανοτικά Μοντέλα Διάχυσης (ΑΠΜΔ) [65, 146]. Στο πλαίσιο αυτής της μελέτης, θα επικεντρωθούμε κυρίως στα τελευταία.

B.1 Αποθορυβοποιούμενα Πιθανοτικά Μοντέλα Διάχυσης

Τα ΑΠΜΔ μπορούν να θεωρηθούν ως παραλλαγές ενός Μαρκοβιανού Ιεραρχικού Αυτοκωδικοποιητή Παραλλαγής που συμμορφώνονται με τους ακόλουθους τρεις περιορισμούς:

- Η διάστατικότητα της λανθάνουσας αναπαράστασης ταιριάζει με αυτή των δεδομένων.
- Η κατανομή του λανθάνοντος κωδικοποιητή είναι προκαθορισμένη ως ένα γραμμικό μοντέλο Gauss που εφαρμόζεται επί της εξόδου του προηγούμενου χρονικού βήματος.
- Οι παράμετροι των λανθανόντων κωδικοποιητών ποικίλλουν με την πάροδο του χρόνου με τέτοιο τρόπο ώστε η κατανομή της λανθάνουσας αναπαράστασης στο τελικό χρονικό βήμα T να είναι τυπική κανονική.

Τα πραγματικά δείγματα δεδομένων καθώς και οι λανθάνουσες/θορυβώδεις εκδοχές τους μπορούν να αναπαρασταθούν από κοινού χρησιμοποιώντας τον συμβολισμό $\mathbf{x}_t$, με το $t = 0$ να αντιστοιχεί σε αληθινά και με το $t \in [1, T]$ να αντιστοιχεί σε λανθάνοντα, με την ιεραρχία λανθανόντων δειγμάτων να υποδηλώνεται με τον χρονικό δείκτη t. Η Μαρκοβιανή ιδιότητα ανάμεσα σε διαδοχικές λανθάνουσες μεταβάσεις υποδηλώνει ότι η εμπρόσθια διαδικασία θορυβοποίησης μπορεί να γραφτεί ως εξής:

$$q(\mathbf{x}_{1:T}|\mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1}) \tag{1}$$

όπου οι μεταβάσεις του κωδικοποιητή είναι της μορφής:

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{a_t}\mathbf{x}_{t-1}, (1 - a_t)\mathbf{I}) \tag{2}$$

όπου το πρόγραμμα της κλίμακας θορύβου a_t μπορεί να είναι είτε προκαθορισμένο είτε εκπαιδεύσιμο. Η Μαρκοβιανή ιδιότητα συνεπάγεται ότι $q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0) = q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{a_t}\mathbf{x}_{t-1}, (1 - a_t)\mathbf{I})$. Αξιοποιώντας το λεγόμενο κόλπο της επαναμετροποίησης και παρατηρώντας ότι το άθροισμα ανεξάρτητων τυχαίων κανονικών μεταβλητών ακολουθεί κανονική κατανομή με μέση τιμή ίση με το άθροισμα των επιμέρους μέσων και με διακύμανση ίση με το άθροισμα των επιμέρους διακυμάνσεων, προκύπτει η ακόλουθη κομψή σχέση:

$$\mathbf{x}_t = \sqrt{\prod_{t=1}^{T} a_t}\mathbf{x}_0 + \sqrt{1 - \prod_{t=1}^{T} a_t}\boldsymbol{\epsilon}_0 = \sqrt{\bar{a}_t}\mathbf{x}_0 + \sqrt{1 - \bar{a}_t}\boldsymbol{\epsilon}_0 \sim \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{a}_t}\mathbf{x}_0, (1 - \bar{a}_t)\mathbf{I}) \tag{3}$$

Κατ' επέκταση, αξιοποιώντας τον νόμο του Bayes, μπορούμε να δείξουμε ότι $\mathbf{x}_{t-1} \sim q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)$, όπου $\mathbf{x}_{t-1}$ είναι κανονικά κατανεμημένες με μέση τιμή $\boldsymbol{\mu}_q(\mathbf{x}_t, \mathbf{x}_0) = \frac{\sqrt{a_t}\mathbf{x}_t(1-\bar{a}_{t-1})+\sqrt{\bar{a}_{t-1}}\mathbf{x}_0(1-a_t)}{1-\bar{a}_t}$ και διακύμανση $\boldsymbol{\Sigma}_q(t) = \sigma_q^2(t)\mathbf{I} = \frac{(1-a_t)(1-\bar{a}_{t-1})}{(1-\bar{a}_t)}\mathbf{I}$. Κατά αυτό το τρόπο, το εκπαιδευόμενο βήμα αποθορυβοποίησης $p_{\boldsymbol{\theta}}(\mathbf{x}_{t-1}|\mathbf{x}_t)$ πρέπει να ακολουθεί όσο πιο πιστά γίνεται το πραγματικό βήμα αποθορυβοποίησης $q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)$, πράγμα που συνοψίζεται στην ελαχιστοποίηση του ακόλουθου όρου *ταιριάσματος αποθορυβοποίησης*:

$$\underset{\boldsymbol{\theta}}{\operatorname{argmin}} \frac{1}{2}\left(\frac{\bar{a}_{t-1}}{1 - \bar{a}_{t-1}} - \frac{\bar{a}_t}{1 - \bar{a}_t}\right)\left[\|\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) - \mathbf{x}_0\|_2^2\right] \tag{4}$$

όπου $\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$ αντιστοιχεί στην εκτίμηση ενός νευρωνικού δικτύου ως προς την αρχική είσοδο $\mathbf{x}_0$ και κατά το επιπέδο θορύβου t. Η Εξ. (3) μπορεί να ξαναγραφτεί στην ακόλουθη μορφή:

$$\mathbf{x}_0 = \frac{\mathbf{x}_t - \sqrt{1 - \bar{a}_t}\boldsymbol{\epsilon}_0}{\sqrt{\bar{a}_t}} \tag{5}$$

Χρησιμοποιώντας την παραπάνω εξίσωση και επικαλούμενοι ξανά το κόλπο της επαναμετροποίησης, μπορούμε να αντικαταστήσουμε το νευρωνικό δίκτυο της Εξ. (4) με ένα νευρωνικό δίκτυο $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$ που μαθαίνει εναλλακτικά να προβλέπει τον πηγαίο θόρυβο $\boldsymbol{\epsilon}_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ (modulo κάποια σταθερά στον αντίστοιχο όρο σφάλματος). Επομένως, η εκμάθηση ενός ΑΠΜΔ μεσω της πρόβλεψης της αρχικής εισόδου $\mathbf{x}_0$ ισοδυναμεί με την εκμάθηση μέσω πρόβλεψης του θορύβου. Εμπειρικά, ωστόσο, έχει βρεθεί ότι η πρόβλεψη του θορύβου οδηγεί συχνά σε καλύτερη απόδοση [65, 79].

Β.2 Αποθορυβοποιούμενα Έμμεσα Μοντέλα Διάχυσης

Τα Αποθορυβοποιούμενα Έμμεσα Μοντέλα Διάχυσης (ΑΕΜΔ) αποτελούν μια κατηγορία παραγωγικών μοντέλων που εκπαιδεύονται με στόχους αυτοκωδικοποιούμενης αποθορυβοποίησης/score matching και χαρακτηρίζονται από μια μη Μαρκοβιανή εμπρόσθια διαδικασία θορυβοποιήσης. Στα ΑΕΜΔ, τα δείγματα μπορούν να προσδιοριστούν μοναδικά από τις λανθάνουσες μεταβλητές, επομένως και παρουσιάζουν παρόμοιες ιδιότητες με τα ΠΑΔ. Τα ΑΕΜΔ βασίζονται στην ακόλουθη οικογένεια κατανομών που χαρακτηρίζονται από ένα πραγματικό διάνυσμα $\boldsymbol{\sigma} \in \mathbb{R}^T_{\geq 0}$:

$$q_{\boldsymbol{\sigma}}(\mathbf{x}_{1:T}|\mathbf{x}_0) := q_{\boldsymbol{\sigma}}(\mathbf{x}_T|\mathbf{x}_0) \prod_{t=2}^{T} q_{\boldsymbol{\sigma}}(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) \tag{6}$$

όπου $q_{\boldsymbol{\sigma}}(\mathbf{x}_T|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_T, \sqrt{a_T}\mathbf{x}_0, (1 - a_T)\mathbf{I})$ και για κάθε $t > 1$ ισχύει:

$$q_{\boldsymbol{\sigma}}(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}\left(\sqrt{a_{t-1}}\mathbf{x}_0 + \left(\sqrt{1 - a_{t-1} - \sigma_t^2}\right)\frac{\mathbf{x}_t - \sqrt{a_t}\mathbf{x}_0}{\sqrt{1 - a_t}}, \sigma_t^2\mathbf{I}\right) \tag{7}$$

Ο Γκαουσιανός μέσος της Εξ. (7) επιλέγεται με τέτοιο τρόπο ώστε να διασφαλίζεται ότι $q_{\boldsymbol{\sigma}}(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t, \sqrt{a_t}\mathbf{x}_0, (1-a_t)\mathbf{I})$. Αποδεικνύεται ότι η διαδικασία δειγματοληψίας στα ΑΕΜΔ μπορεί να περιγραφεί από την ακόλουθη σχέση:

$$\mathbf{x}_{t-1} = \sqrt{a_{t-1}}\left(\frac{\mathbf{x}_t - \sqrt{1 - a_t}\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)}{\sqrt{a_t}}\right) + \left(\sqrt{1 - a_{t-1} - \sigma_t^2}\right)\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) + \sigma_t\boldsymbol{\epsilon} \tag{8}$$

όπου $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ και $a_0 := 1$. Διαφορετικές επιλογές ως προς το διάνυσμα $\boldsymbol{\sigma}$ αντιστοιχούν σε διαφορετικές παραγωγικές διαδικασίες, χρησιμοποιώντας όμως ένα κοινό νευρωνικό δίκτυο $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$ το οποίο, όπως και στα ΑΠΜΔ, εκπαιδεύεται έτσι ώστε να προβλέπει τον πηγαίο θόρυβο. Όταν $\sigma_t = \sqrt{(1 - a_{t-1})/(1 - a_t)}\sqrt{1 - a_t/a_{t-1}}$ για κάθε t, τότε η εμπρόσθια διαδικασία θορυβοποίησης καθίσταται Μαρκοβιανή, και η οπίσθια παραγωγική διαδικασία ταυτίζεται με αυτή ενός ΑΠΜΔ. Όταν εξετάζουμε επιταχυνόμενη σύνθεση με το τ να είναι μια αύξουσα υποακολουθία στο διάστημα $[1, \ldots, T]$, η υπερπαράμετρος της τυπικής απόκλισης συχνά παραμετροποιείται ως $\sigma_{\tau_i}(\eta) = \eta\sqrt{(1 - a_{\tau_{i-1}})/(1 - a_{\tau_i})}\sqrt{1 - a_{\tau_i}/a_{\tau_{i-1}}}$. Επιπλέον, όταν $\sigma_t = 0$, η Εξ. (8) ισοδύναμει με μια μέθοδο Euler προς επίλυση μιας ΣΔΕ, ενώ τόσο η εμπρόσθια διαδικασία θορυβοποίησης όσο και η διαδικασία δειγματοληψίας καθίστανται ντετερμινιστικές.

Β.3 Καθοδήγηση

Όλη η συζήτηση μέχρι αυτό το σημείο έχει αφιερωθεί στην εκμάθηση της κατανομής $p(\mathbf{x})$ ενός δεδομένου συνόλου δεδομένων. Ωστόσο, είναι επίσης χρήσιμο να μπορούμε να μαθαίνουμε δεσμευμένες κατανομές δεδομένων, δηλαδή $p(\mathbf{x}|y)$ όπου το y υποδηλώνει οποιοδήποτε είδος επιπρόσθετης πληροφορίας, π.χ., ετικέτες κλάσεων. Σύμφωνα με τη σχετική βιβλιογραφία, υπάρχουν δύο κύριες μέθοδοι για να επιτευχθεί αυτό, συγκεκριμένα η *καθοδήγηση με ταξινομητή* [33] και η *καθοδήγηση χωρίς ταξινομητή* [66].

Καθοδήγηση με Ταξινομητή

Έστω $p_{\boldsymbol{\phi}}(y|\mathbf{x}_t)$ ένας ταξινομητής που στοχεύει στην ταξινόμηση θορυβωδών εικόνων $\mathbf{x}_t$, σε διάφορα επίπεδα θορύβου t. Έστω και πάλι ένα νευρωνικό δίκτυο $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}$ που εκπαιδεύεται έτσι ώστε να προβλέπει τον πηγαίο εισαγόμενο θόρυβο. Τότε, για την αντίστοιχη διαδικασία δειγματοληψίας υπό καθοδήγηση με ταξινομητή, στην περίπτωση των ΑΠΜΔ, ισχύει $\mathbf{x}_{t-1} \sim \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_{\boldsymbol{\theta}} + \boldsymbol{\Sigma}_{\boldsymbol{\theta}}\nabla_{\mathbf{x}_t} \log p_{\boldsymbol{\phi}}(y|\mathbf{x}_t), \boldsymbol{\Sigma}_{\boldsymbol{\theta}})$, όπου $\boldsymbol{\mu}_{\boldsymbol{\theta}}, \boldsymbol{\Sigma}_{\boldsymbol{\theta}}$ είναι ο μέσος και η διακύμανση της εκπαιδευόμενης Gaussian κατανομής που χαρακτηρίζει τις

μεταβάσεις κατά την οπίσθια Μαρκοβιανή αλυσίδα. Κατ' επέκταση, στα ΑΕΜΔ μπορεί να χρησιμοποιηθεί η ίδια διαδικασία δειγματοληψίας που περιγράφηκε στην ενότητα B.2, αντικαθιστώντας το υπάρχον δίκτυο πρόβλεψης θορύβου $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_t)$ από το $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_t|y) = \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_t) - \sqrt{1-\bar{a}_t}\nabla_{\mathbf{x}_t}\log p_{\boldsymbol{\phi}}(y|\mathbf{x}_t)$.

Καθοδήγηση χωρίς Ταξινομητή

Όπως υποδηλώνει το όνομα, η *καθοδήγηση χωρίς ταξινομητή* (*κ.χ.τ.*) [66], ως μεθοδολογία, δεν απαιτεί την εκπαίδευση κάποιου ξεχωριστού ταξινομητή. Αντιθέτως, κάποιος εκπαιδεύει ένα δεσμευμένο μοντέλο διάχυσης $p(\mathbf{x}|y)$ με πιθανότητα απόρριψης της υπό συνθήκη πληροφορίας p_{uncond}, δηλαδή σε τυχαίες περιπτώσεις, η επιπρόσθετη πληροφορία y αφαιρείται (πιθανότητες απόρριψης της τάξης του 10-20% τείνουν να λειτουργούν καλά) και αντικαθίσταται με μια ετικέτα $\varnothing$ (`null`). Αυτή η ετικέτα αποτελεί ειδική τιμή εισόδου που αντιπροσωπεύει την απουσία ρυθμιστικών πληροφοριών. Το τροποποιημένο μοντέλο που προκύπτει είναι πλέον σε θέση να λειτουργεί και υπό συνθήκη $p(\mathbf{x}|y)$ και ως μοντέλο άνευ όρων $p(\mathbf{x})$, ανάλογα με το αν παρέχεται το σήμα ρύθμισης. Στη συνέχεια πραγματοποιείται δειγματοληψία χρησιμοποιώντας τον ακόλουθο γραμμικό συνδυασμό των υπό όρους και άνευ όρων εκτιμήσεων θορύβου:

$$\tilde{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_t|y) = (1-\gamma)\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{x}_t|\varnothing) + \gamma\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{x}_t|y) \tag{9}$$

Για $\gamma = 0$, ανακτούμε το απλό, άνευ όρων μοντέλο και για $\gamma = 1$ παίρνουμε το τυπικό δεσμευμένο μοντέλο. Για $\gamma > 1$, η κατανομή από την οποία δειγματοληπτούμε καθίσταται «υπερ-ρυθμισμένη».

B.4 Μοντέλα Λανθάνουσας Διάχυσης

Τα Μοντέλα Λανθάνουσας Διάχυσης (ΜΛΔ) [125] εισήχθησαν σε μια προσπάθεια να μειωθούν οι υπολογιστικές απαιτήσεις για την εκπαίδευση μοντέλων διάχυσης για σύνθεση εικόνων υψηλής ανάλυσης. Η προτεινόμενη προσέγγιση προτείνει έναν ρητό διαχωρισμό της φάσης συμπίεσης από τη φάση παραγωγικής μάθησης, χρησιμοποιώντας έναν αυτοκωδικοποιητή που μαθαίνει έναν χώρο που είναι αντιληπτικά ισοδύναμος με τον χώρο των εικόνων, αλλά προσφέρει σημαντικά μειωμένη υπολογιστική πολυπλοκότητα τόσο στη διαδικασία εκπαίδευσης όσο και στη διαδικασία δειγματοληψίας.

Δεδομένης μιας RGB εικόνας εισόδου $\mathbf{x} \in \mathbb{R}^{H\times W\times 3}$, ένας κωδικοποιητής $\mathcal{E}$ κωδικοποιεί το $\mathbf{x}$ σε μια λανθάνουσα αναπαράσταση $\mathbf{z} = \mathcal{E}(\mathbf{x}) \in \mathbb{R}^{h\times w\times c}$, ενώ ένας αποκωδικοποιητής $\mathcal{D}$ αποκωδικοποιεί τη λανθάνουσα αναπαράσταση και την επαναφέρει στον αρχικό χώρο των εικόνων, παράγοντας την ανακατασκευή $\tilde{\mathbf{x}} = \mathcal{D}(\mathbf{z}) = \mathcal{D}(\mathcal{E}(\mathbf{x}))$. Ο κωδικοποιητής υποδειγματοληπτεί την εικόνα εισόδου κατά ένα συντελεστή $f = H/h = W/w$. Έχοντας εκπαιδεύσει τα μοντέλα αντιληπτικής συμπίεσης, ο χαμηλών διαστάσεων λανθάνων χώρος μπορεί πλέον να αξιοποιηθεί, στον οποίο υψηλής συχνότητας ανεπαίσθητες λεπτομέρειες αφαιρούνται. Σε σύγκριση με τον υψηλής διαστατικότητας χώρο των pixel, ο λανθάνων χώρος είναι καταλληλότερος για παραγωγικά πιθανοτικά μοντέλα, καθώς μπορούν να επικεντρωθούν στα σημασιολογικά σημαντικά «bit» δεδομένων, εξασφαλίζοντας παράλληλα υπολογιστικά αποδοτική εκπαίδευση.

Εκμεταλλευόμενοι τη δισδιάστατη δομή των παραγόμενων λανθανουσών αναπαραστάσεων, η ραχοκοκαλιά ενός ΜΛΔ βασίζεται στην αρχιτεκτονική U-Net, που αποτελείται κυρίως από 2Δ συνελικτικά στρώματα. Όπως συζητήθηκε στην ενότητα B.1, η εκπαίδευση των ΑΠΜΔ βασίζεται σε μια επαναστάθμισμένη παραλλαγή του μεταβλητού κάτω φράγματος στον λογάριθμο της πιθανοφάνειας της αληθινής κατανομής δεδομένων $\log p(\mathbf{x})$. Τα ΜΛΔ μπορούν να ερμηνευθούν ως εξίσου σταθμισμένη ακολουθία αυτοκωδικοποιητών αποθορυβοποίησης $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{z}_t, t)$ οι οποίοι είναι εκπαιδευμένοι ώστε να προβλέπουν μια αποθορυβοποιημένη εκδοχή της εισόδου $\mathbf{z}_t$, όπου $\mathbf{z}_t$ είναι μια θορυβώδης έκδοση της λανθάνουσας εισόδου $\mathbf{z}$. Ο αντίστοιχος στόχος αποθορυβοποίησης γίνεται:

$$\mathcal{L}_{\text{ΜΛΔ}} = \mathbb{E}_{\mathcal{E}(\mathbf{x}),\boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\mathbf{I}),t\sim\mathcal{U}(1,T)}\left[\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{z}_t, t)\|_2^2\right] \tag{10}$$

Τα ΜΛΔ είναι επίσης ικανά να μοντελοποιούν δεσμευμένες κατανομές της μορφής $p(\mathbf{z}|y)$, όπου το y υποδηλώνει κάθε είδους πρόσθετη πληροφορία όπως ετικέτες κλάσεων, κείμενο, σημασιολογικούς χάρτες κ.λ.π. Προς εξυπηρέτηση αυτού του σκοπού, η ραχοκοκκαλιά U-Net επαυξάνεται με έναν μηχανισμό διασταυρούμενης προσοχής [168], αποτελεσματικό στην εκμάθηση μοντέλων που βασίζονται στην προσοχή πολυτροπικών εισόδων.

Γ Πειραματικά Αποτελέσματα

Γ.1 Σύνθεση Εικόνας & Αναπαράσταση Προσώπου

Σύνολο Δεδομένων και Μετρικές Αξιολόγησης

Εκπαιδεύσαμε τα μοντέλα μας στη βάση δεδομένων AffectNet [107] η οποία περιέχει περίπου 1 εκατ. εικόνες που ανακτήθηκαν από το διαδίκτυο. Συγκεκριμένα, 440 χιλ. από αυτές τις εικόνες σχολιάστηκαν χειροκίνητα από 12 ειδικούς με βασικά διακριτά συναισθήματα (κατηγορικό μοντέλο), την ένταση του σθένους και της διέγερσης (διαστατικό/συνεχές μοντέλο). Η βάση δεδομένων AffectNet είναι μακράν η μεγαλύτερη βάση δεδομένων εκφράσεων προσώπου, με επισημειώσεις σθένους και διέγερσης σε μη ελεγχόμενες ρυθμίσεις, επιτρέποντας την έρευνα για την αυτοματοποιημένη αναγνώριση εκφράσεων προσώπου σύμφωνα με δύο διαφορετικά συναισθηματικά μοντέλα. Από τις 440 χιλ. εικόνες με μη αυτόματο σχολιασμό, 291,651 εικόνες φέρουν ετικέτα με 8 βασικά διακριτά συναισθήματα: ουδέτερο, χαρα, λύπη, έκπληξη, φόβος, αηδία, θυμός και περιφρόνηση. Από αυτές, 287,651 εικόνες ανήκουν στο σετ εκπαίδευσης και 4,000 εικόνες (500 για κάθε ετικέτα συναισθήματος) ανήκουν στο σύνολο επικύρωσης.

Όσον αφορά την αξιολόγηση των μοντέλων μας, χρησιμοποιούμε διαφορετικές μετρικές για να αξιολογήσουμε την ικανότητα σύνθεσης εικόνων και διαφορετικές για την αξιολόγηση της ποιότητας και αποτελεσματικής μεταφοράς συναισθημάτων κατά την αναπαράσταση προσώπου. Η αξιολόγηση ως προς την ικανότητα σύνθεσης εικόνων γίνεται με βάση την *Fréchet Inception Distance* (FID) [64], τη *Kernel Inception Distance* (KID) [13] καθώς και το *Improved Precision & Recall* [88]. Κατά τον χειρισμό εικόνων για αναπαράσταση προσώπου, η ποιότητα των εικόνων αξιολογείται με βάση το *Peak-Signal-to-Noise ratio* (PSNR), το *Structural Similarity Index Measure* (SSIM) [173] και *Learned Perceptual Image Patch Similarity* (LPIPS) [181]. Για την αξιολόγηση της ικανότητας μεταφοράς συναισθήματος αξιοποιούμε ένα προεκπαιδευμένο μοντέλο αναγνώρισης συναισθήματος HSEmotion [135–137] και ως μετρική χρησιμοποιούμε την ακρίβεια ταξινόμησης.

Προεπεξεργασία Δεδομένων

Οι δημιουργοί του συνόλου δεδομένων AffectNet παρέχουν προϋπολογισμένα πλαίσια οριοθέτησης προσώπου καθώς και τις συντεταγμένες ορόσημων σημείων προσώπου ανά εικόνα. Ωστόσο, προτιμήσαμε να εντοπίσουμε εξαρχής τόσο τα οριοθετημένα πλαίσια όσο και τα ορόσημα σημεία του προσώπου, χρησιμοποιώντας το FAN [18]. Προτού τροφοδοτήσουμε τις αρχικές εικόνες στο FAN, αλλάζουμε πρώτα τις διαστάσεις τους σε 256×256 για να αποφύγουμε προβλήματα εξάντλησης μνήμης GPU. Το FAN εντοπίζει τις 2Δ συντεταγμένες για 68 ορόσημα σημεία του προσώπου. Με βάση αυτά τα ορόσημα, τα οριοθετημένα κουτιά μεγέθους 224×224 περικόπτονται κεντρικά και, στη συνέχεια, ευθυγραμμίζονται με εκτίμηση του μετασχηματισμού ομοιότητας που απαιτείται για να παραμορφωθεί το σύνολο των ανιχνευόμενων ορόσημων σημείων προσώπου στις κανονικές τους συντεταγμένες. Τυχόν κενές περιοχές εικόνας που προκύπτουν από την περιστροφή της αρχικής αντιμετωπίζονται ανταναακλώντας την εικόνα κοντά στις άκρες.

Αντιληπτική Συμπίεση Εικόνας

Το πρώτο στάδιο οποιουδήποτε ΜΛΔ αποτελείται από έναν προεκπαιδευμένο αυτοκωδικοποιητή που μεταφέρει οποιαδήποτε δεδομένη είσοδο από το χώρο των εικόνων σε έναν συμπιεσμένο, λανθάνοντα χώρο. Για το σκοπό αυτό, εκπαιδεύουμε ένα VQGAN [45] στη βάση δεδομένων AffectNet.

Το VQGAN υιοθετεί την αρχιτεκτονική U-Net [126] που αποτελείται από μια σειρά υπολειπόμενων (residual) μπλοκ [62] υποδειγματοληψίας και υπερδειγματοληψίας. Ο αριθμός των υπολειπόμενων μπλοκ ανά επίπεδο χωρικής ανάλυσης ορίστηκε ίσος με δύο. Ο αριθμός των βασικών καναλιών ορίστηκε ίσος με 128. Το μέγεθος των εικόνων εισόδου ορίστηκε σε ανάλυση 128×128. Έτσι, με δεδομένες εικόνες εισόδου $\mathbf{x} \in \mathbb{R}^{128\times128\times3}$, ο κωδικοποιητής συμπιέζει τις τελευταίες κατά έναν παράγοντα $f = 4$, παράγοντας λανθάνουσες αναπαραστάσεις $\hat{\mathbf{z}} \in \mathbb{R}^{32\times32\times3}$. Κάθε λανθάνουσα αναπαράσταση υφίσταται κβαντοποίηση στοιχείο-προς-στοιχείο, με αποτέλεσμα μια συλλογή από 32^2 καταχωρήσεις κωδικών $\boldsymbol{z}_{ij} \in \mathbb{R}^3, i, j \in [32]$. Ο πληθάριθμος του βιβλίου κωδικών ορίζεται ίσος με $|\mathcal{Z}| = 16384$. Επιπλέον, οι πολλαπλασιαστές καναλιών για τις 2 ενδιάμεσες χωρικές αναλύσεις ορίζονται ίσοι με 2 και 4 αντίστοιχα. Το τελευταίο επίπεδο χωρικής ανάλυσης περιλαμβάνει επίσης ένα τμήμα χωρικής προσοχής. Το VQGAN εκπαιδεύτηκε για 6 εποχές με τον βελτιστοποιητή Adam [80], με αρχικό ρυθμό εκμάθησης 3.6×10^{-5} και

μέγεθος batch ίσο με 8. Η δομή του αποκωδικοποιητή είναι ακριβώς η ίδια με αυτή του κωδικοποιητή, αλλά με τη σειρά των μπλοκ να αντιστρέφεται.

Σύνθεση Εικόνας

Έχοντας εκπαιδευμένες μονάδες κωδικοποιητή και αποκωδικοποιητή, τις χρησιμοποιούμε ως το πρώτο στάδιο ενός ΜΛΔ. Σύμφωνα με το VQGAN που συζητήθηκε προηγουμένως, προχωράμε περιγράφοντας την επιλεγμένη διαμόρφωση στην περίπτωση του ΜΛΔ.

Η αρχιτεκτονική U-Net που χρησιμοποιεί το ΜΛΔ διαθέτει δύο υπολειπόμενα μπλοκ ανά επίπεδο χωρικής ανάλυσης, με τον αριθμό των βασικών καναλιών να είναι ίσος με 160. Οι λανθάνουσες αναπαραστάσεις εισόδου συμπιέζονται περαιτέρω κατά ένα συντελεστή $f = 4$. Τα μπλοκ προσοχής τοποθετούνται σε κάθε επίπεδο χωρικής ανάλυσης, δηλαδή $32^2, 16^2, 8^2$ και οι πολλαπλασιαστές καναλιών ορίζονται ίσοι με 1, 2 και 4 αντίστοιχα. Ο κωδικοποιητής ετικετών κλάσης υλοποιείται ως ένα ενιαίο επίπεδο ενσωμάτωσης (embedding layer) 512 διαστάσεων, αντιστοιχίζοντας την εκάστοτε ετικέτα συναισθηματικής κλάσης y ως $\boldsymbol{\tau_\theta}(y) \in \mathbb{R}^{1\times 512}$.

Για να ελέγξουμε καλύτερα την ένταση των απεικονιζόμενων συναισθημάτων κατά τη φάση χειραγώγησης, οφείλαμε να εκπαιδεύσουμε το ΜΛΔ χρησιμοποιώντας *καθοδήγηση χωρίς ταξινομητή*. Κατά τη διάρκεια της εκπαίδευσης και με πιθανότητα $p_{\text{uncond}} = 0.2$, οι ετικέτες των κλάσεων εισαγωγής απορρίπτονται και αντικαθίστανται με μια ετικέτα `null`, αυξάνοντας ουσιαστικά τον συνολικό αριθμό των ετικετών συναισθηματικής κλάσης σε εννέα (8 διακριτά συναισθήματα συν την ετικέτα `null`). Το ΜΛΔ εκπαιδεύτηκε για περίπου 30 εποχές με τον βελτιστοποιητή AdamW [98], αρχικό ρυθμό εκμάθησης 2.4×10^{-5} και μέγεθος batch ίσο με 24. Χρησιμοποιήσαμε γραμμικό πρόγραμμα θορύβου και $T_{\text{ΑΠΜΔ}} = 1000$ βήματα διάχυσης.

Για να αξιολογήσουμε την υπό συνθήκη ικανότητα σύνθεσης εικόνων του μοντέλου μας ακολουθήσαμε δύο μεθοδολογίες. Αρχικά, δειγματοληπτήσαμε 5 χιλ. δείγματα ανά συναισθηματική ετικέτα χρησιμοποιώντας ΑΕΜΔ με $T_{\text{ΑΕΜΔ}} = 500$ βήματα και χωρίς άνευ όρων καθοδήγηση ($\gamma = 1, 0$). Για κάθε υποσύνολο δειγμάτων, υπολογίσαμε τις μετρικές αξιολόγησης που συζητήθηκαν στην αρχή της ενότητας Γ.1 χρησιμοποιώντας ολόκληρο το σύνολο δεδομένων εκπαίδευσης που αντιστοιχεί στο δεδομένο συναίσθημα. Στη συνέχεια, δειγματοληπτήσαμε τυχαία 3.75 χιλ. εικόνες από κάθε κατηγορία συναισθημάτων και τις συγκρίνουμε με το προαναφερθέν ολόκληρο σύνολο 40 χιλ. δειγμάτων. Στον Πίν. Γ.1, παρουσιάζονται οι μετρήσεις ως προς τις μετρικές FID, KID, precision και recall[1] και για τα δύο προαναφερθέντα σενάρια αξιολόγησης.

Συναίσθημα Στόχος	FID↓	KID ($\times 10^{-3}$) ↓	Precision↑	Recall↑
Ουδέτερο	6.64	2.1	0.57	0.67
Χαρά	6.90	2.7	0.55	0.66
Λύπη	7.73	2.1	0.58	0.66
Έκπληξη	8.54	2.6	0.55	0.70
Φόβος	10.93	2.4	0.56	0.71
Αηδία	11.72	2.3	0.56	0.73
Θυμός	8.03	2.7	0.57	0.66
Περιφρόνηση	10.16	2.9	0.58	0.71
Μέσος Όρος[2]	8.83	2.48	0.57	0.69
Συνολικά[3]	4.3	2.1	0.56	0.69

Πίνακας Γ.1: Ποσοτικά αποτελέσματα ως προς την υπο συνθήκη σύνθεση εικόνων στη βάση δεδομένων AffectNet χρησιμοποιώντας $T_{\text{ΑΕΜΔ}} = 500$ βήματα, $\eta = 1.0$ και κλίμακα *κ.χ.τ.* $\gamma = 1.0$.

Όπως ήταν αναμενόμενο, η καλύτερη απόδοση παρατηρείται για το ουδέτερο συναίσθημα και για το συναίσθημα της χαράς, καθώς είναι αυτά με τα περισσότερα δείγματα εκπαίδευσης στο σύνολο δεδομένων. Οι υψηλότερες τιμές FID παρατηρούνται για τα συναισθήματα του φόβου και της αηδίας. Από άποψη συναισθημάτων, ο φόβος σχετίζεται στενά με την έκπληξη, ενώ η αηδία συνδέεται στενά με τον θυμό,

[1] Οι μετρικές FID, KID, precision and recall υπολογίστηκαν με βάση το δημόσια διαθέσιμο πακέτο λογισμικού `torch-fidelity`.

[2] Αυτή η γραμμη του πίνακα αντιστοιχεί στον αριθμητικό μέσο της κάθε μετρικής ως προς τις 8 διαφορετικές συναισθηματικές κλάσεις.

[3] Αυτή η αξιολόγηση πραγματοποιήθηκε συγκρίνοντας 40 χιλ. δείγματα (8 χιλ. ανά συναισθηματική κλάση) με 30 χιλ. πραγματικές εικόνες (3.75 χιλ. ανά συναισθηματική κλάση).

όπως μπορεί να φανεί σε μια 2Δ αναπαράσταση σθένους-διέγερσης. Η περιφρόνηση είναι επίσης συχνά οπτικά δύσκολο να διακριθεί από την ευτυχία.

Συναισθηματική Χειραγώγηση Εικόνων & Αναπαράσταση Προσώπου

Βασική Μεθοδολογία Η βασική μεθοδολογία για τη χειραγώγηση εικόνων με ΜΛΔ, ακολουθεί το παράδειγμα του LDEdit [19], προσαρμοσμένο ειδικά για το πρόβλημα της χειραγώγησης συναισθημάτων. Ο τελευταίος χρησιμοποιούσε ντετερμινιστική εμπρόσθια διάχυση σε έναν λανθάνοντα χώρο χαμηλότερης διάστασης, που εξαρτάται από την έξοδο ενός μετασχηματιστή tokenizer που ήταν προεκπαιδευμένος σε ένα σύνολο ζευγών κειμένου-εικόνας (π.χ. LAION-400M [139]), επιτυγχάνοντας χειραγώγηση καθοδηγούμενη από κείμενο σε άγνωστες κατηγορίες με μηδενική υποστήριξη δεδομένων (zero-shot). Στην περίπτωσή μας, ο μετασχηματιστής αντικαθίσταται με ένα στρώμα ενσωμάτωσης που αντιστοιχίζει καθεμία από τις 8 (συν την ετικέτα `null`) πιθανές ετικέτες συναισθημάτων σε ένα εκπαιδευόμενο διάνυσμα $\boldsymbol{\tau_\theta}(y) \in \mathbb{R}^{1\times 512}$.

Η ραχοκοκαλιά της εν λόγω προσέγγισης έγκειται στην σχεδόν κυκλική συνέπεια που εξασφαλίζεται μέσω των ντετερμινιστικών διαδικασιών ΑΕΜΔ, τόσο της εμπρόσθιας διάχυσης όσο και της οπίσθιας διαδικασίας αποθορυβοποίησης. Αυτό αποδεικνύεται περαιτέρω από την ακόλουθη ανάλυση. Η διαδικασία δειγματοληψίας ΑΕΜΔ που περιγράφεται στην ενότητα B.2, μπορεί να ξαναγραφτεί με την ακόλουθη μορφή:

$$\mathbf{z}_{t-1} = \sqrt{\bar{a}_{t-1}}\left(\frac{\mathbf{z}_t - \sqrt{1-\bar{a}_t}\boldsymbol{\epsilon_\theta}(\mathbf{z}_t, t, \boldsymbol{\tau_\theta}(y))}{\sqrt{\bar{a}_t}}\right) + \sqrt{1-\bar{a}_{t-1}}\boldsymbol{\epsilon_\theta}(\mathbf{z}_t, t, \boldsymbol{\tau_\theta}(y)) \tag{11}$$

$$\sqrt{\frac{1}{\bar{a}_{t-1}}}\mathbf{z}_{t-1} - \sqrt{\frac{1}{\bar{a}_t}}\mathbf{z}_t = \left(\sqrt{\frac{1}{\bar{a}_{t-1}} - 1} - \sqrt{\frac{1}{\bar{a}_t} - 1}\right)\boldsymbol{\epsilon_\theta}(\mathbf{z}_t, t, \boldsymbol{\tau_\theta}(y)) \tag{12}$$

Θέτοντας $\mathbf{y}_t := \sqrt{1/\bar{a}_t}\mathbf{z}_t$ και $p_t := \sqrt{1/\bar{a}_t - 1}$, η Εξ. (12) μπορεί να γραφτεί ως εξής:

$$\mathbf{y}_{t-1} - \mathbf{y}_t = (p_{t-1} - p_t)\boldsymbol{\epsilon_\theta}(\mathbf{z}_t, t, \boldsymbol{\tau_\theta}(y)) \Rightarrow \mathrm{d}\mathbf{y}_t = \boldsymbol{\epsilon_\theta}(\mathbf{z}_t, t, \boldsymbol{\tau_\theta}(y))\mathrm{d}p_t \tag{13}$$

Η τελευταία εξίσωση περιγράφει μια ΣΔΕ η οποία κινείται πίσω στο χρόνο ως προς τον χρονικό δείκτη διάχυσης t και περιγράφει την ντετερμινιστική διαδικασία αποθορυβοποίησης ΑΕΜΔ. Αντιστροφή της ίδιας εξίσωσης οδηγεί σε αντίστοιχη ΣΔΕ που περιγράφει την εμπρόσθια ΑΕΜΔ διαδικασία θορυβοποίησης. Θέτοντας $\mathbf{f}_{\boldsymbol{\theta}}(\mathbf{z}_t, t, \boldsymbol{\tau_\theta}(y)) := (\mathbf{z}_t - \sqrt{1-\bar{a}_t}\boldsymbol{\epsilon_\theta}(\mathbf{z}_t, t, \boldsymbol{\tau_\theta}(y)))/\sqrt{\bar{a}_t}$, λαμβάνουμε τις εμπρόσθιες και οπίσθιες διεργασίες ΑΕΜΔ για χειρισμό εικόνων βάσει ΜΛΔ ως προς δύο ετικέτες συναισθημάτων y_{src} (αρχικό) και y_{trg} (στόχος):

$$\mathbf{z}_{t+1} = \sqrt{\bar{a}_{t+1}}\mathbf{f}_{\boldsymbol{\theta}}(\mathbf{z}_t, t, \boldsymbol{\tau_\theta}(y_{\text{src}})) + \sqrt{1-\bar{a}_{t+1}}\boldsymbol{\epsilon_\theta}(\mathbf{z}_t, t, \boldsymbol{\tau_\theta}(y_{\text{src}})) \tag{14}$$

$$\hat{\mathbf{z}}_{t-1} = \sqrt{\bar{a}_{t-1}}\mathbf{f}_{\boldsymbol{\theta}}(\hat{\mathbf{z}}_t, t, \boldsymbol{\tau_\theta}(y_{\text{trg}})) + \sqrt{1-\bar{a}_{t-1}}\boldsymbol{\epsilon_\theta}(\hat{\mathbf{z}}_t, t, \boldsymbol{\tau_\theta}(y_{\text{trg}})) \tag{15}$$

Κατά τη διάρκεια των διαδικασιών συναισθηματικού χειρισμού, τα βάρη του ΜΛΔ $\boldsymbol{\theta}$ παραμένουν σταθερά σύμφωνα με το καλύτερο σωσμένο σημείο ελέγχου από την αρχική φάση εκπαίδευσης. Με δεδομένη μια εικόνα εισόδου $\mathbf{x}_{\text{src}}$, ο κωδικοποιητής πρώτου σταδίου τη μετατρέπει στην αντίστοιχη λανθάνουσα αναπαράστασή $\mathbf{z}_0$. Στη συνέχεια, η ντετερμινιστική ($\eta = 0$) προς τα εμπρός διάχυση εκτελείται μέχρι το χρονικό βήμα $t_0 < T_{\text{ΑΠΜΔ}} = 1000$, με βάση την ετικέτα πηγαίου συναισθήματος y_{src}, που συμβολίζεται ως $\mathbf{z}_{t_0}$. Η αντίστροφη διαδικασία που εξαρτάται από την ετικέτα συναισθήματος στόχου y_{trg} ξεκινά από τον ίδιο θορυβώδη λανθάνοντα κώδικα $\mathbf{z}_{t_0}$ με στόχο την ανακατασκευή του $\hat{\mathbf{z}}_0$. Περνώντας τον λανθάνοντα κώδικα $\hat{\mathbf{z}}_0$ μέσω του αποκωδικοποιητή προκύπτει η ανακατασκευασμένη εικόνα $\mathbf{x}_{\text{gen}}$. Σε αυτό το σημείο θα πρέπει να αναφέρουμε ότι η διαδικασία ΑΕΜΔ μπορεί να επιταχυνθεί σημαντικά χρησιμοποιώντας λιγότερα βήματα διακριτοποίησης $\{\tau_s\}_{s=1}^{T_{\text{AEMΔ}}}$ ομοιόμορφα επιλεγμένα στο εύρος $[1, t_0]$ έτσι ώστε $\tau_1 = 1$ και $\tau_{T_{\text{AEMΔ}}} = t_0$. Προφανώς, όσο μικρότερος είναι ο αριθμός των ενδιάμεσων βημάτων $T_{\text{AEMΔ}}$, τόσο χειρότερη είναι η ποιότητα ανακατασκευής της διαδικασίας ΑΕΜΔ. Ωστόσο, η μείωση του αριθμού των βημάτων δειγματοληψίας είναι απαραίτητη όσον αφορά την επιτάχυνση των πειραμάτων και έχει εμπειρικά προκύψει ότι $T_{\text{AEMΔ}} \geq 20$ παρέχει ικανοποιητικά αποτελέσματα όσον αφορά εφαρμογές μετάφρασης εικόνας-σε-εικόνα.

Το Σχ. Γ.1 απεικονίζει επιμελημένα παραδείγματα χειραγώγησης συναισθημάτων σε εικόνες από το σύνολο επικύρωσης της βάσης δεδομένων AffectNet, χρησιμοποιώντας $T_{\text{AEMΔ}} = 20$, $\gamma = 3.0$ και

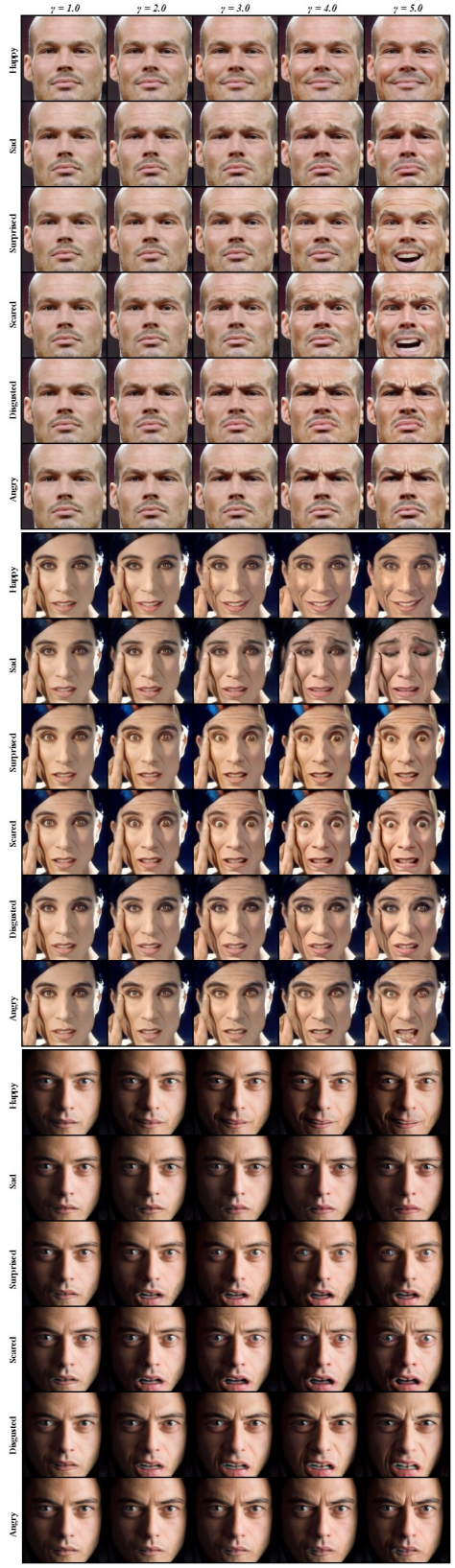
Σχήμα Γ.1: Επιλεγμένα παραδείγματα λανθάνουσας χειραγώγησης συναισθήματος στο σύνολο επικύρωσης AffectNet, χρησιμοποιώντας $T_{\text{AEM}\Delta} = 40$ βήματα, $\eta = 0$, δύναμη επεξεργασίας $t_0 = 500$ και κλίμακα **κ.χ.τ.** $\gamma = 3.0$.

$t_0 = 500$. Η πιο αριστερή εικόνα σε κάθε στήλη είναι ένα αρχικό δείγμα που προέρχεται από το σύνολο δεδομένων, ενώ οι υπόλοιπες έξι στήλες περιλαμβάνουν τα αποτελέσματα χειρισμού για καθεμία από τις 6 βασικές εκφράσεις των ακόλουθων συναισθημάτων: ευτυχία, θλίψη, έκπληξη, φόβος, αηδία και θυμός. Μπορούμε αμέσως να παρατηρήσουμε ότι τα αποτελέσματα στις στήλες που αντιστοιχούν στα συναισθήματα της χαράς, λύπης και θυμού είναι εν γένει ικανοποιητικά, ενώ τα συναισθήματα της έκπληξης, φόβου και αηδίας είναι πιο δύσκολο να μεταδοθούν. Αυτό μπορεί να δικαιολογηθεί εν μέρει από το γεγονός ότι τα προαναφερθέντα συναισθήματα υποεκπροσωπούνται στο σετ εκπαίδευσης της βάσης δεδομένων AffectNet. Επιπλέον, το Σχ. Γ.3 απεικονίζει το αποτέλεσμα της μεταβολής της άνευ όρων κλίμακας καθοδήγησης $\gamma \in \{1.0, 2.0, 3.0, 4.0, 5.0\}$ στην περίπτωση και των έξι διαφορετικών συναισθημάτων-στόχων, ενώ τα $t_0 = 500$ και $T_{\text{AEM}\Delta}$ παραμένουν σταθερά. Είναι προφανές ότι στη συγκεκριμένη ρύθμιση των $t_0 = 500$, οι χαμηλότερες τιμές των γ έχουν ελάχιστη έως καθόλου επίδραση στην αλλαγή του συναισθήματος. Μπορούν να επιτευχθούν καλύτερα αποτελέσματα με $\gamma \in \{4.0, 5.0\}$ με ελάχιστη παραμόρφωση των περιφερειακών χαρακτηριστικών του προσώπου και της ταυτότητας του εικονιζόμενου προσώπου. Τέλος, το Σχ. Γ.2 απεικονίζει το αποτέλεσμα της μεταβολής της ισχύος επεξεργασίας $t_0 \in \{400, 500, 600\}$ στην περίπτωση και των έξι διαφορετικών συναισθημάτων-στόχων,

Σχήμα Γ.2: Επιλεγμένα παραδείγματα χειραγώγησης συναισθημάτων βάσει ΜΛΔ στο σύνολο επικύρωσης της βάσης δεδομένων AffectNet, χρησιμοποιώντας $T_{\text{AEMΔ}} = 40$ βήματα, $\eta = 0$, κλίμακα *κ.χ.τ.* $\gamma = 3.0$ και μεταβλητή ισχύ επεξεργασίας $t_0 \in \{400, 500, 600\}$.

ενώ τα γ και $T_{\text{AEMΔ}}$ παραμένουν σταθερά. Μπορεί κανείς εύκολα να παρατηρήσει ότι το $t_0 = 400$ δεν επαρκεί για την παραγωγή συναισθηματικά διακριτών αποτελεσμάτων. Από την άλλη πλευρά, μια τιμή $t_0 = 600$ έχει ως αποτέλεσμα καλύτερα αποτελέσματα χειραγώγησης, αλλά με κόστος παραμόρφωσης της ταυτότητας του αρχικά εικονιζόμενου προσώπου.

Προσαρμογή Καθοδηγούμενη από Μοντέλα CLIP Για την αποτελεσματική εξαγωγή γνώσεων από τα μοντέλα CLIP, έχουν προταθεί δύο διαφορετικές συναρτήσεις σφάλματος: μια γενικά εφαρμοζόμενη [115] και μια τοπική συνάρτηση σφάλματος κατεύθυνσης [52]. Και οι δύο προαναφερθέντες τύποι σφάλματος προτάθηκαν αρχικά για άμεση βελτιστοποίηση ενδιάμεσων λανθανουσών αναπαραστάσεων στον $\mathcal{W}+$ χώρο του StyleGAN [75]. Ο πρώτος τύπος σφάλματος προσπαθεί να ελαχιστοποιήσει την απόσταση συνημιτόνου εντός ενός προεκπαιδευμένου λανθάνοντος χώρου CLIP μεταξύ της παραγόμενης εικόνας και ενός δεδομένου κειμένου στόχου. Από την άλλη πλευρά, τοπική συνάρτηση σφάλματος κατεύθυνσης επιβάλει την ευθυγράμμιση της κατεύθυνσης, σε επίπεδο ενδιάμεσων αναπαραστάσεων, μεταξύ των αρχικών και των παραγόμενων εικόνων, με την κατεύθυνση μεταξύ των αναπαραστάσεων ενός



Σχήμα Γ.3: Επιλεγμένα παραδείγματα χειραγώγησης συναισθημάτων βάσει ΜΛΔ στο σύνολο επικύρωσης της βάσης δεδομένων AffectNet, χρησιμοποιώντας $T_{\text{AEM}\Delta} = 40$ βήματα, $\eta = 0$, δύναμη επεξεργασίας $t_0 = 500$ και μεταβλητή κλίμακα **κ.χ.τ.** $\gamma \in \{1.0, 2.0, 3.0, 4.0, 5.0\}$.

ζεύγους κειμένων αναφοράς και στόχου εντός του χώρου CLIP ως εξής:

$$\mathcal{L}_{\text{dir}}(\mathbf{x}_{\text{gen}}, y_{\text{trg}}, \mathbf{x}_{\text{src}}, y_{\text{src}}) = 1 - \cos\langle \text{CLIP}_{\text{img}}(\mathbf{x}_{\text{gen}}) - \text{CLIP}_{\text{img}}(\mathbf{x}_{\text{src}}), \text{CLIP}_{\text{text}}(y_{\text{trg}}) - \text{CLIP}_{\text{text}}(y_{\text{src}}) \rangle \quad (16)$$

Η ενσωμάτωση των CLIP μοντέλων στο πλαίσιο χρήσης ΜΔ, εισήχθη μαζί με το μοντέλο DiffusionCLIP [76] το οποίο προσάρμοσε την κατευθυντική απώλεια CLIP στο πλαίσιο των ΑΠΜΔ και πέτυχε μετάφραση εικόνων αγνώστων κατηγοριών με μηδενική υποστήριξη δεδομένων. Κατά τον χειρισμό εικόνων που απεικόνιζαν ανθρώπινα πρόσωπα, ήταν απαραίτητη η χρήση επιπρόσθετων συναρτήσεων σφάλματος διατήρησης ταυτότητας όσο και μια συνάρτηση απωλειών τύπου ℓ_1. Πριν από την εφαρμογή της καθοδήγησης CLIP, απαιτείται εκπαίδευση ενός ΜΔ $\boldsymbol{\epsilon_\theta}$. Στη συνέχεια, οποιαδήποτε δεδομένη εικόνα εισόδου $\mathbf{x}_{\text{src}}$ μετατρέπεται σε μια θορυβώδη λανθάνουσα αναπαράσταση $\mathbf{x}_{\text{src}}^{(t_0)}$. Στη συνέχεια, καθοδηγούμενο από την κατευθυντική απώλεια CLIP, το ΜΔ, στην οπίσθια διαδικασία, ρυθμίζεται έτσι ώστε να δημιουργεί δείγματα ακολουθώντας το εκάστοτε κείμενο-στόχο y_{trg}. Στη δικιά μας περίπτωση, η συνάρτηση σφάλματος διατήρησης ταυτότητας υλοποιείται ως η απόσταση συνημιτόνου στον λανθάνοντα χώρο ενός ArcFace [32] προεκπαιδευμένου δικτύου αναγνώρισης προσώπου IR-SE50[4].

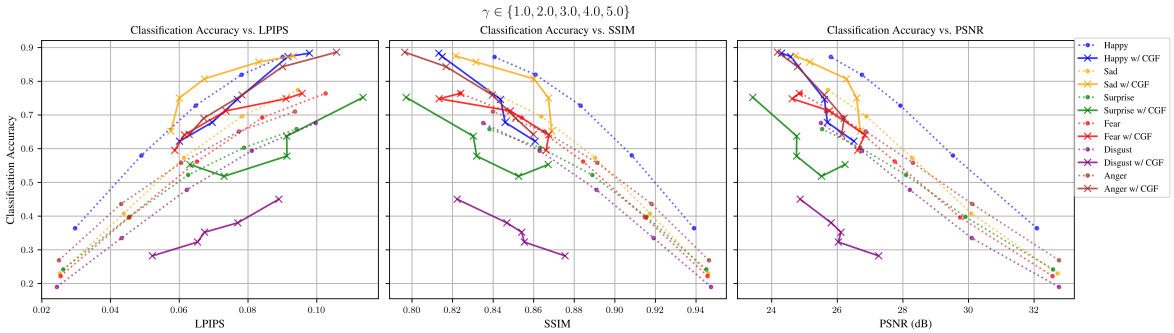
Το πρώτο βήμα προς τη λεπτομερή προσαρμογή ενός ΜΛΔ, περιλαμβάνει τον προϋπολογισμό θορυβωδών λανθανουσών αναπαραστάσεων για έναν προκαθορισμένο αριθμό δειγμάτων εκπαίδευσης καθώς και για ολόκληρο το σύνολο επικύρωσης. Προϋπολογίσαμε θορυβώδεις λανθάνουσες αναπαραστάσεις χρησιμοποιώντας ντετερμινιστικό εμπρόσθιο ΑΕΜΔ ($\eta = 0$) με $T_{\text{AEM}\Delta} = 40$ βήματα και $t_0 = 500$ δύναμη επεξεργασίας, για 4,000 δείγματα εκπαίδευσης (500 ανά συναισθηματική ετικέτα). Στη συνέχεια, δειγματοληπτήσαμε τυχαία 1,000 λανθάνουσες αναπαραστάσεις από το προαναφερθέν σύνολο σε μια προσπάθεια να αντισταθμίσουμε την απόδοση χειραγώγησης με τον μειωμένο χρόνο εκπαίδευσης. Αυτή η διαδικασία επαναλήφθηκε για όλες τις κλίμακες άνευ όρων καθοδήγησης $\gamma \in \{1.0, 2.0, 3.0, 4.0, 5.0\}$. Εκπαιδεύσαμε για 20 εποχές με ρυθμό εκμάθησης 2×10^{-6}, μέγεθος batch ίσο με 4, χρησιμοποιώντας τον βελτιστοποιητή AdamW [98]. Κάθε batch λανθανουσών υποβάλλεται σε δειγματοληψία ΑΕΜΔ για $T_{\text{tune}} = 6$ βήματα και $t_0 = 500$. Προφανώς αυτός ο χαμηλός αριθμός βημάτων στη διαδικασία δειγματοληψίας ΑΕΜΔ παρέχει ατελή ανακατασκευή, αλλά προτείνεται εμπειρικά [76] ως αποδεκτός συμβιβασμός μεταξύ της ποιότητας του δείγματος και των απαιτήσεων για μνήμη GPU. Για απλότητα, κατά τη διάρκεια όλων των πειραμάτων ορίσαμε $\lambda_{\text{id}} = \lambda_{\ell_2} = 1$ και διαφοροποιήσαμε μόνο το λ_{dir}, όπου λ_{id} αποτελεί το συντελεστή σφάλματος διατήρησης ταυτότητας, λ_{ℓ_2} αποτελεί το συντελεστή σφάλματος τύπου ℓ_2 και λ_{dir} αποτελεί το συντελεστή κατευθυντικού σφάλματος CLIP.

Τα γραφήματα Γ.4 και Γ.5 παρουσιάζουν ποσοτικές συγκρίσεις μεταξύ βασικών και προσαρμοσμένων μοντέλων, σε ό,τι αφορά τις μετρικές αξιολόγησης ποιότητας εικόνας (LPIPS/SSIM/PSNR) και την ακρίβεια ταξινόμησης συναισθήματος βάσει των χειραγωγημένων δειγμάτων στο σύνολο επικύρωσης της βάσης δεδομένων AffectNet. Πιο συγκεκριμένα, στο Σχ. Γ.4 τα προσαρμοσμένα μοντέλα παρουσιάζουν υψηλότερες βαθμολογίες ακρίβειας ταξινόμησης για τις ίδιες τιμές της κλίμακας *κ.χ.τ.* γ, με εξαίρεση το συναίσθημα της αηδίας. Η βελτίωση στην ακρίβεια ταξινόμησης έρχεται υπό το κόστος μιας μικρής μείωσης στην ποιότητα της εικόνας σε όλες τις μετρικές, αλλά όπως θα αποδειχθεί αργότερα, το τελικό αποτέλεσμα παραμένει οπτικά αληθοφανές. Επιπλέον, στο Σχ. Γ.5, μπορούμε να δούμε ότι ακόμη και χωρίς άνευ όρων καθοδήγηση ($\gamma = 1$), μια ρύθμιση στην οποία το αποτέλεσμα χειρισμού της βασικής μεθόδου είναι συχνά απαρατήρητο στο ανθρώπινο μάτι, η εισαγωγή της κατευθυντικής απώλειας CLIP επιλύει εν μέρει οποιαδήποτε συναισθηματική ασάφεια σε μεγάλο βαθμό, με εξαίρεση το συναίσθημα της αηδίας, το οποίο είναι οπτικά πανομοιότυπο με το συναίσθημα του θυμού. Πιο συγκεκριμένα, σε μια 2Δ αναπαράσταση σθένους-διέγερσης, και τα δύο συναισθήματα λαμβάνουν παρόμοιες αρνητικές τιμές σθένους, με τον θυμό να κατέχει ελάχιστα μεγαλύτερη τιμή διέγερσης.

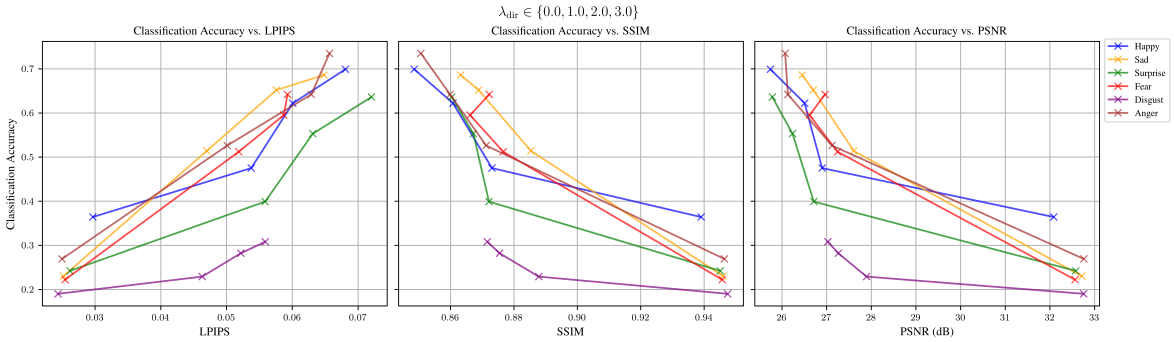
Σύγκριση με ΠΑΔ Καθώς τη στιγμή που γράφουμε την εν λόγω εργασία δεν υπάρχει κανένα συγκρίσιμο ΜΔ με δημόσια διαθέσιμη βάση κώδικα που μπορεί να χρησιμοποιηθεί για αναπαράσταση προσώπου και συναισθηματική χειραγώγηση εικόνων, στρέφουμε την προσοχή μας σε καλά θεμελιωμένες υλοποιήσεις που βασίζονται σε ΠΑΔ, με τις οποίες θα συγκρίνουμε τα ευρήματά μας, τόσο σε ποσοτικό όσο και σε ποιοτικό επίπεδο. Τα μοντέλα αυτά είναι τα ακόλουθα: GANimation [120], StarGAN v2 [26] και GANmut [30] (γραμμικά και Gaussian, που συμβολίζονται ως GANmut και GGANmut, αντίστοιχα).

Στον Πίν. Γ.2 παρέχουμε μια ποσοτική σύγκριση μεταξύ των τριών προαναφερθέντων υλοποιήσεων που βασίζονται σε ΠΑΔ και αυτών που βασίζονται σε διάχυση, με και χωρίς βελτιστοποίηση καθοδηγούμενη από CLIP, όσον αφορά την ακρίβεια ταξινόμησης των συναισθημάτων (χρησιμοποιώντας ένα

[4] Αυτή η αρχιτεκτονική αποτελεί συνδυασμό ενός ResNet [62] με 50 ενδιάμεσα στρώματα και SENet [68] μπλοκς.



Σχήμα Γ.4: Ποσοτική σύγκριση μεταξύ βασικών μοντέλων και προσαρμοσμένων μοντέλων υπό καθοδήγηση CLIP, όσον αφορά τη μέση ακρίβεια ταξινόμησης (στο εύρος [0-1]), και τις μετρικές LPIPS, SSIM και PSNR ως προς το σύνολο των χειραγωγημένων δειγμάτων, χρησιμοποιώντας $T_{\text{AEM}\Delta} = 40$ βήματα, ισχύ επεξεργασίας $t_0 = 500$, $\lambda_{\text{dir}} = 2.0, \lambda_{\text{id}} = \lambda_{\ell_2} = 1.0$, σε όλες τις κλίμακες *κ.χ.τ.* $\gamma \in \{1.0, 2.0, 3.0, 4.0, 5.0\}$.



Σχήμα Γ.5: Ποσοτική σύγκριση μεταξύ βασικών μοντέλων ($\lambda_{\text{dir}} = 0$) και προσαρμοσμένων μοντέλων υπό καθοδήγηση CLIP, όσον αφορά τη μέση ακρίβεια ταξινόμησης (στην περιοχή [0-1]), και τις μετρικές LPIPS, SSIM και PSNR ως προς το σύνολο των χειραγωγημένων δειγμάτων, χρησιμοποιώντας $T_{\text{AEM}\Delta} = 40$, ισχύ επεξεργασίας $t_0 = 500$, $\gamma = 1.0, \lambda_{\text{id}} = \lambda_{\ell_2} = 1.0$, σε μεταβλητές τιμές $\lambda_{\text{dir}} \in \{0.0, 1.0, 2.0, 3.0\}$.

προεκπαιδευμένο HSEmotion [135–137] ως ταξινομητή), PSNR, SSIM, LPIPS καθώς και την ομοιότητα συνημιτόνου στον λανθάνοντα χώρο ενός μοντέλου CosFace [171], προεκπαιδευμένου στο σύνολο δεδομένων Glint360K [2] (CSIM). Η σύγκριση μεταξύ όλων των μοντέλων δεν είναι σε καμία περίπτωση απλή και πρέπει να λάβουμε υπόψη την υποκείμενη αντιστάθμιση μεταξύ της ακρίβειας ταξινόμησης συναισθημάτων και της διατήρησης της ποιότητας σε σχέση με τις πηγαίες εικόνες. Σε όλα τα συναισθήματα, οι υλοποιήσεις που βασίζονται σε ΜΛΔ ξεπερνούν όλες τις αντίστοιχες που βασίζονται σε ΠΑΔ όσον αφορά την ποιότητα των παραγόμενων εικόνων. Επιπλέον, οι υλοποιήσεις ΠΑΔ επιτυγχάνουν γενικά υψηλότερη ακρίβεια ταξινόμησης συναισθημάτων, με τις υψηλότερες μέσες βαθμολογίες να σημειώνονται από το GGANmut. Αυτό μπορεί να εξηγηθεί εν μέρει από το γεγονός ότι το τελευταίο ταιριάζει πολλαπλές Gaussian κατανομές με στόχο την απόκτηση μιας πιο ακριβούς και πλούσιας αναπαράστασης του υπό όρους συναισθηματικού λανθάνοντος χώρου, σε σύγκριση με τις διακριτές ετικέτες συναισθημάτων. Θα μπορούσαμε πιθανώς να αντιμετωπίσουμε αυτό το έλλειμμα πειραματιζόμενοι με διαφορετικές πιθανότητες απόρριψης των υπο συνθήκη συναισθηματικών ετικετών για τη *κ.χ.τ.* κατά τη διάρκεια της εκπαίδευσης και υψηλότερες άνευ όρων κλίμακες καθοδήγησης ($\gamma > 5.0$). Επιπλέον, μπορεί να ακολουθηθεί μια πιο υποκειμενική προσέγγιση για την αξιολόγηση του μοντέλου, π.χ. με τη μορφή ερωτηματολογίων, κάτι που συνηθίζεται στην περίπτωση των πειραμάτων μετάφρασης εικόνας-σε-εικόνα. Αυτό θα συζητηθεί περαιτέρω στις επόμενες παραγράφους.

Πραγματοποιήσαμε επίσης δύο μελέτες χρηστών για να αξιολογήσουμε τον ρεαλισμό και της συναισθηματικής ακρίβειας της προσέγγισής μας έναντι των προαναφερθέντων βασισμένων σε ΠΑΔ, όπως αξιολογήθηκε από ανθρώπινους χρήστες, τα αποτελέσματα των οποίων βρίσκονται στον Πίν. Γ.3. Στην πρώτη μελέτη, συμμετείχαν 24 συμμετέχοντες, στους οποίους παρουσιάστηκαν 28 ζεύγη συγκρίσεων μεταξύ όλων των μεθόδων (συμπεριλαμβανομένων των αρχικών εικόνων) και κλήθηκαν να επιλέξουν την πιο ρεαλιστική. Όπως αποδεικνύεται από τα αποτελέσματα, οι χειραγωγημένες εκ της μεθόδου μας εικόνες έγιναν αντιληπτές ως στατιστικά *σημαντικά* ($p < 0,01$ με διωνυμική δοκιμή) πιο ρεαλιστικές σε σύγκριση με τις άλλες μεθόδους. Ωστόσο, όπως αναμενόταν, δεν ήταν τόσο ρεαλιστικές όσο οι

Συναίσθημα / Μέθοδος	Χαρά					Θλίψη					Έκπληξη				
	Ακρίβεια↑	PSNR↑	SSIM↑	LPIPS↓	CSIM↑	Ακρίβεια↑	PSNR↑	SSIM↑	LPIPS↓	CSIM↑	Ακρίβεια↑	PSNR↑	SSIM↑	LPIPS↓	CSIM↑
Groundtruth [107]	0.758	–	–	–	–	0.638	–	–	–	–	0.606	–	–	–	–
GANimation [120]	0.645	24.07	0.816	0.099	0.547	0.212	24.52	0.830	0.097	0.582	0.360	23.82	0.817	0.101	0.559
StarGAN v2 [26]	**0.958**	17.46	0.659	0.165	0.441	0.569	18.30	0.712	0.149	0.593	0.761	17.99	0.678	0.160	0.503
GANmut [30]	0.879	21.42	0.809	0.106	0.663	0.888	23.85	**0.857**	0.094	0.755	0.829	22.43	0.810	0.112	0.675
GGANmut [30]	0.934	21.91	0.819	0.103	0.717	**0.986**	22.13	0.802	0.115	0.653	**0.970**	22.37	0.777	0.121	0.610
Δικό μας	0.872	**25.80**	**0.841**	**0.090**	0.743	0.774	**25.71**	0.837	0.095	0.778	0.658	**25.54**	**0.838**	**0.094**	0.716
Δικό μας με ΠΚΜΣ[5]	0.883	24.30	0.813	0.098	**0.744**	0.875	24.72	0.822	**0.093**	**0.794**	0.752	23.42	0.797	0.113	**0.721**
	Φόβος					Αηδία					Θυμός				
Groundtruth [107]	0.666	–	–	–	–	0.646	–	–	–	–	0.514	–	–	–	–
GANimation [120]	0.314	23.68	0.814	0.105	0.556	0.271	24.13	0.819	0.103	0.549	0.287	24.80	0.833	0.096	0.580
StarGAN v2 [26]	0.860	17.76	0.676	0.162	0.507	0.879	18.10	0.691	0.154	0.528	0.666	18.14	0.689	0.154	0.509
GANmut [30]	0.932	23.39	**0.841**	0.102	0.721	0.877	22.64	0.815	0.113	0.663	0.856	21.57	0.813	0.106	0.678
GGANmut [30]	**0.967**	20.13	0.763	0.135	0.556	**0.987**	21.68	0.772	0.128	0.562	**0.969**	21.77	0.767	0.133	0.590
Δικό μας	0.764	**24.89**	0.824	0.103	**0.770**	0.676	**25.50**	**0.835**	0.100	0.707	0.710	**25.66**	**0.840**	**0.094**	0.714
Δικό μας με ΠΚΜΣ[5]	0.764	24.83	0.826	**0.096**	0.764	0.450	24.87	0.822	**0.089**	**0.714**	0.886	24.18	0.796	0.106	**0.735**

Πίνακας Γ.2: Ποσοτική σύγκριση μεταξύ του ΜΛΔ μας και των υλοποιήσεων που βασίζονται σε ΠΑΔ για συναισθηματικό χειρισμό εικόνων στο σύνολο επικύρωσης της βάσης δεδομένων AffectNet.

	Ρεαλισμός							
Δικό μας vs.	GANimation		GANmut		StarGAN v2		Groundtruth	
	0.69	0.31	**0.68**	0.32	**0.61**	0.39	0.28	**0.72**
	Αναγνώριση Συναισθήματος							
0.57	0.38		**0.73**		0.57		0.59	

Πίνακας Γ.3: Υποκειμενική μελέτη χρηστών σχετικά με τον ρεαλισμό (πάνω μισό) και την ακρίβεια μετάφρασης των συναισθημάτων (κάτω μισό).

πραγματικές εικόνες.

Στη δεύτερη μελέτη, σε 27 συμμετέχοντες παρουσιάστηκαν 30 εικόνες από κάθε μέθοδο και τους ζητήθηκε να προσδιορίσουν το εμφανιζόμενο συναίσθημα ανάμεσα από μια λίστα έξι πιθανών συναισθημάτων. Το κάτω μισό του Πίν. Γ.3 παρουσιάζει τα αντίστοιχα αποτελέσματα ακρίβειας, τα οποία είναι στενά ευθυγραμμισμένα με τα προηγούμενα αντικειμενικά ποσοτικά αποτελέσματα του Πίν. Γ.2. Η προσέγγισή μας επιτυγχάνει απόδοση ισοδύναμη με το StarGAN v2 και ξεπερνά το GANimation, αν και το GANmut επιτυγχάνει την υψηλότερη ακρίβεια. Συγκεκριμένα, η ακρίβεια στις αρχικές εικόνες της AffectNet είναι παρόμοια με αυτή του StarGAN v2 και της μεθόδου μας. Αυτή η παρατήρηση μπορεί ενδεχομένως να αποδοθεί στο ότι το GANmut δημιουργεί πιο «υπερβολικά» συναισθήματα που αποκλίνουν από την κατανομή συναισθημάτων της AffectNet, θυσιάζοντας τον ρεαλισμό κατά τη πορεία.

Γ.2 Σύνθεση Ομιλούντος Προσώπου

Για την σύνθεση ομιλούντος προσώπου καταφεύγουμε στη βάση δεδομένων MEAD [172]. Η βασική μεθοδολογία για σύνθεση ομιλούντος προσώπου έγκειται σε ένα συνδυασμό της γνωστής έως αυτό το σημείο αρχιτεκτονικής ΜΛΔ, υπό την καθοδήγηση μιας σειράς από μηχανισμούς συνθηκοποίησης:

Συνθηκοποίηση Ήχου Η συνθηκοποίηση ήχου αποτελεί τον πρωταρχικό κινητήριο παράγοντα και τον κοινό παρονομαστή όλων των επιμέρους διαμορφώσεων του μοντέλου σύνθεσης ομιλούντος προσώπου. Πειραματιζόμαστε με την εξαγωγή δύο ειδών χαρακτηριστικών ήχου. Σε μια περίπτωση χρησιμοποιούμε την κρυφή αναπαράσταση $\mathbf{a} \in \mathbb{R}^{T' \times 768}$ όπως προκύπτει από το τελευταίο transformer μπλοκ του wav2vec 2.0 [6]. Στη δεύτερη περίπτωση, λειτουργούμε το μοντέλο σε λειτουργία AAO και εξάγουμε μη κανονικοποιημένα logits $\mathbf{b} \in \mathbb{R}^{T' \times 29}$ που αντιστοιχούν σε ετικέτες 29 χαρακτήρων[6]. Σε κάθε περίπτωση, ο κωδικοποιητής wav2vec 2.0 αρχικοποιείται με τα προεκπαιδευμένα βάρη. Δεδομένου ότι τα δεδομένα βίντεο καταγράφονται με συχνότητα f_m που είναι διαφορετική από τη συχνότητα λειτουργίας του κωδικοποιητή f_a ($f_a = 49$ Hz για το wav2vec 2.0 ενώ $f_m = 30$ fps για MEAD) προσθέτουμε ένα ενδιάμεσο στρώμα γραμμικής παρεμβολής που έχει ως αποτέλεσμα το επιθυμητό μήκος εξόδου $T = kT' = \lfloor \frac{f_m}{f_a} \rfloor T'$.

Συνθηκοποίηση Συναισθήματος

Ως συνήθως, χρησιμοποιούμε ένα μόνο επίπεδο ενσωμάτωσης με δυνατότητα εκμάθησης $\boldsymbol{\tau_\theta}(y) \in \mathbb{R}^{1 \times 256}$ που αντιστοιχίζει καθένα από τις 8 πιθανές ετικέτες συναισθημάτων του συνόλου δεδομένων

[5] ΠΚΜΣ: Προσαρμογή Καθοδηγούμενη από Μοντέλα CLIP

[6] '-' , '|', 'E', 'T', 'A', 'O', 'N', 'I', 'H', 'S', 'R', 'D', 'L', ' U', 'M', 'W', 'C', 'F', 'G', 'Y', 'P', 'B', 'V', 'K', '', 'X' , 'J', 'Q', 'Z'

MEAD (ουδέτερο, χαρά, λύπη, έκπληξη, φόβος, αηδία, θυμός και περιφρόνηση).

Συνθηκοποίηση Ορόσημων Προσώπου Χρησιμοποιούμε μόνο 48 ζεύγη από τα συνολικά διαθέσιμα 68, εξαιρουμένων των 20 ζευγών συντεταγμένων ορόσημων που εντοπίζονται γύρω από την περιοχή του στόματος, έτσι ώστε να μην υπάρχει διαρροή πληροφοριών. Τα 48 ζεύγη συντεταγμένων σχηματίζουν ένα διάνυσμα $\mathbf{l} \in \mathbb{R}^{96}$ και αργότερα κωδικοποιούνται χρησιμοποιώντας ένα εκπαιδεύσιμο ρηχό MLP με ενεργοποιήσεις ReLU $\boldsymbol{\tau_\theta}(\mathbf{l}) \in \mathbb{R}^{1\times 128}$.

Συνθηκοποίηση Ταυτότητας Εκτός από τον θορυβώδη καρέ-στόχο, το U-Net αποθορυβοποίησης παρέχεται με ένα τυχαία επιλεγμένο καρέ-ταυτότητα ως σημείο αναφοράς. Αυτός ο μηχανισμός συνθηκοποίησης αναμένεται να βοηθήσει στη αντιγραφή χαρακτηριστικών που σχετίζονται με την περιφερειακή εμφάνιση του εκάστοτε εικονιζόμενου προσώπου. Δεδομένου ότι η εκάστοτε πραγματική εικόνα-στόχος ενδέχεται να έχει μια εντελώς διαφορετική στάση/πόζα στο χώρο σε σχέση με το φρείμ ταυτότητας, το μοντέλο καλείται να μεταφέρει τη στάση του καρέ ταυτότητας στο καρέ-στόχο χωρίς καμία άλλη ενδιάμεση πληροφορία. Αυτό αποτελεί ένα κακώς ορισμένο πρόβλημα χωρίς μοναδική λύση. Για να μετριαστεί αυτό, απαιτούνται επιπλέον πληροφορίες συνθηκοποίησης.

Συνθηκοποίηση Επικαλυπτόμενου Στόχου Το καρέ επικαλυπτόμενου στόχου παρέχεται ως μια επιπλέον συνθήκη αναφοράς ως καθοδήγηση εκμάθησης της πραγματικής στάσης του εκάστοτε ζητούμενου καρέ. Η περιοχή του στόματος καλύπτεται εντελώς για να διασφαλιστεί ότι οι πραγματικές κινήσεις των χειλιών δεν είναι ορατές στο δίκτυο αποθορυβοποίησης.

Εκμάθηση Οι διάφορες πληροφορίες συνθηκοποίησης που περιγράφονται παραπάνω, μπορούν να χωριστούν σε δύο διακριτές κατηγορίες: α) Αυτές που τροφοδοτούνται στο δίκτυο αποθορυβοποίησης μέσω του μηχανισμού πολλαπλής προσοχής ΜΛΔ, που δηλώνεται ως $\mathbf{c}_{\text{attn}}$. β) Αυτά που τροφοδοτούνται στο δίκτυο αποθορυβοποίησης συνενώνοντας τα ως προς τη διάσταση καναλιού με τον θορυβώδη στόχο, που συμβολίζεται ως $\mathbf{c}_{\text{cnct}}$. Σε κάθε επανάληψη, δειγματοληπτούμε τυχαία ένα βίντεο $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ από το σετ εκπαίδευσης και, στη συνέχεια, ένα τυχαίο καρέ $\mathbf{x}_k$ από το $\mathbf{X}$, όπου n είναι ο συνολικός αριθμός καρέ του βίντεο του δείγματος. Με είσοδο ένα χρονικό βήμα t και τον αντίστοιχο θορυβώδη στόχο, δειγματοληπτούμε ένα καρέ ταυτότητας $\mathbf{x}_{\text{id}}$. Ανάλογα με τη διαμόρφωση, περιλαμβάνεται μια επικαλυπτόμενη έκδοση του καρέ-στόχου, που δηλώνεται ως $\mathbf{x}_{\text{mask}}$. Όλα τα προαναφερθέντα καρέ οδηγούνται πρώτα μέσω ενός προεκπαιδευμένου αυτοκωδικοποιητή, παράγοντας συμπιεσμένες λανθάνουσες αναπαραστάσεις $\mathbf{z}_k, \mathbf{z}_{\text{id}}$ και $\mathbf{z}_{\text{mask}}$, αντιστοίχως. Σε κάθε επανάληψη, το τυχαία δείγμα λανθάνοντος στόχου υφίσταται διάχυση προς τα εμπρός για $t \sim \mathcal{U}(1, T_{\text{ΑΠΜΔ}})$ χρονικά βήματα, που οδηγεί στον θορυβώδη λανθάνοντα στόχο $\mathbf{z}_t$. Ο στόχος της απλής αποθορυβοποίησης ΜΛΔ χρησιμοποιείται για να καθοδηγήσει τη διαδικασία εκπαίδευσης:

$$\mathcal{L}_{\text{ΜΛΔ}} = \mathbb{E}_{\mathbf{z},\mathbf{c}_{\text{cnct}},\mathbf{c}_{\text{attn}},\boldsymbol{\epsilon},t}\left[\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon_\theta}(\mathbf{z}_t, \mathbf{c}_{\text{attn}}, \mathbf{c}_{\text{cnct}}, t)\|_2^2\right] \tag{17}$$

όπου $\boldsymbol{\epsilon_\theta}$ υποδηλώνει το εκπαιδεύσιμο U-Net αποθορυβοποίησης.

Αυτοπαλινδρομική Σύνθεση Βάσει της σχετικής βιβλιογραφίας [140, 154], η σύνθεση ομιλούντος προσώπου με χρήση μοντέλων διάχυσης γίνεται αυτοπαλινδρομικά. Πιο συγκεκριμένα, κατά τη σύνθεση του πρώτου καρέ του εκάστοτε βίντεο, ως καρέ ταυτότητας χρησιμοποιείται αυτούσιο του καρέ-στόχος. Στη συνέχεια, κάθε νέο καρέ που συντίθεται χρησιμοποιείται ως καρέ ταυτότητας στη σύνθεση του επόμενου βήματος. Προφανώς, σε κάθε βήμα παρέχεται το καρέ επικαλυπτόμενου στόχου, η συνθηκοποίηση ήχου καθώς και οποιαδήποτε άλλη μορφή προϋπολογισμένης συνθηκοποίησης που είναι απαραίτητη για την καθοδήγηση της αυτοπαλινδρομικής διαδικασίας σύνθεσης.

Μελέτη Εκτομών Βασικού Μοντέλου Χρησιμοποιώντας το προαναφερθέν βασικό μοντέλο σύνθεσης ομιλούντος προσώπου εκτελούμε μια σειρά πειραμάτων εκτομής για να διαπιστώσουμε τα ακόλουθα: α) Την επίδραση του εκάστοτε μηχανισμού συνθηκοποίησης, β) την επίδραση του μεγέθους ηχητικού παραθύρου. Οι προαναφερθείσες επιδράσεις αξιολογούνται τόσο ως προς την ποιότητα των

Μοντέλο	Συνθηκοποίηση			Ποιότητα Εικόνας			Ανάγνωση Χειλιών			
	Συναίσθημα	Επικαλυπτόμενος Στόχος	Ορόσημα	PSNR↑	SSIM↑	LPIPS↓	WER↓	CER↓	WERV↓	CERV↓
Groundtruth	N/A	N/A	N/A	N/A	N/A	N/A	65.5	41.3	65.1	35.6
Baseline (Transformer)	✓	✗	✗	20.61	0.582	0.114	191.8	137.8	187.7	130.8
	✓	✓	✗	26.59	0.856	0.046	112.9	85.2	109.7	81.1
	✓	✓	✓	26.68	**0.865**	**0.043**	110.1	85.1	108.4	79.7
	✗	✓	✗	**26.69**	0.857	0.047	102.2	**82.1**	99.3	**77.3**
Baseline (AAO)	✗	✓	✗	26.66	0.857	0.045	**101.3**	83.5	**99.1**	78.2

Πίνακας Γ.4: Μελέτη εκτομής όσον αφορά τους διάφορους μηχανισμούς συνθηκοποίησης που χρησιμοποιούνται από το υποκείμενο ΜΛΔ. Χρησιμοποιούμε μήκος ηχητικού παραθύρου $w = 4$ σε όλες τις επιμέρους εξεταζόμενες διαμορφώσεις.

Μοντέλο	Μέγεθος Ηχητικού Παραθύρου w	Ποιότητα Εικόνας			Ανάγνωση Χειλιών			
		PSNR↑	SSIM↑	LPIPS↓	WER↓	CER↓	WERV↓	CERV↓
Groundtruth	N/A	N/A	N/A	N/A	65.5	41.3	65.1	35.6
Baseline (Transformer)	1	26.29	0.851	0.048	121.2	90.5	119.6	88.2
	2	26.53	0.855	0.047	116.3	87.1	111.4	83.4
	4	**26.59**	0.856	**0.046**	112.9	85.2	109.7	81.1
	8	26.43	0.854	**0.046**	108.2	**82.3**	**105.1**	**77.7**
	16	25.74	0.841	0.049	116.2	87.5	111.9	83.1
Baseline (AAO)	8	26.41	**0.859**	**0.046**	**107.1**	83.3	105.3	**77.7**

Πίνακας Γ.5: Συγκριτική ποσοτική μελέτη όσον αφορά τα διάφορα μήκη ηχητικού παραθύρου που χρησιμοποιούνται από την υποκείμενη μονάδα κωδικοποίησης ήχου. Η συνθηκοποίηση ήχου και συναισθημάτων χρησιμοποιείται σε όλες τις επιμέρους εξεταζόμενες διαμορφώσεις.

συντεθέντων εικόνων (μετρικές PSNR, SSIM, LPIPS) όσο και ως προς τον οπτικο-ακουστικό συγχρονισμό χειλιών μεταξύ συντεθέντων και πραγματικών καρέ (μετρικές WER, CER, WERV, CERV[7]). Τα αντίστοιχα αποτελέσματα παρουσιάζονται στους Πιν. Γ.4 και Γ.5.

Άμεσες παρατηρήσεις που πηγάζουν εκ των παραπάνω πινάκων είναι οι ακόλουθες: α) Ο καθοριστικός παράγοντας για την ποιότητα των συντεθέντων καρέ είναι η συνθηκοποίηση επικαλυπτόμενου στόχου. β) Το βέλτιστο μέγεθος ηχητικού παραθύρου φαίνεται να κυμαίνεται στα $w = 8$ καρέ εκατέρωθεν του εκάστοτε καρέ στόχου. γ) Η επίδοση των μοντέλων ως προς τον οπτικο-ακουστικό συγχρονισμό χειλιών είναι κακή, με ποσοστό λάθους λέξης που κυμαίνεται άνω του 100%, πράγμα που υποδεικνύει ότι τα παραγόμενα βίντεο είναι σε «fast-forward». Για αυτό το λόγο, καλούμαστε να βελτιώσουμε την επίδοση του βασικού μοντέλου εφαρμόζοντας κάποιο σφάλμα ανάγνωσης χειλιών.

Βελτιστοποίηση Καθοδηγούμενη Βάσει Ανάγνωσης Χειλιών Για να εκμεταλλευτούμε ενα προεκπαιδευμένο μοντέλο ανάγνωσης χειλιών κατά τη διάρκεια της εκπαίδευσης, πρέπει οι προβλέψεις του αποθορυβοποιητή να γίνονται σε επίπεδο εικόνας και όχι θορύβου. Για το σκοπό αυτό, πρώτα κωδικοποιούμε στοχαστικά τις λανθάνουσες αναπαραστάσεις εισόδου χρησιμοποιώντας ΑΠΜΔ για $t \sim \mathcal{U}(1, T_{\text{ΑΠΜΔ}})$ χρονικά βήματα. Στη συνέχεια, προκειμένου να παρακάμψουμε τους εγγενείς περιορισμούς ενός προεκπαιδευμένου U-Net που παρέχει προβλέψεις σε επίπεδο θορύβου, εφαρμόζουμε διαφοροποιήσιμο αντίστροφο ΑΕΜΔ στις θορυβώδεις λανθάνουσες αναπαράστασεις για $T_{\text{tune}} \ll T_{\text{ΑΠΜΔ}}$ βήματα (8 στην περίπτωσή μας). Στην πράξη όμως, οι ανακατασκευασμένες λανθάνουσες αναπαραστάσεις είναι αρκετά λεπτομερείς, ώστε μετά την αποκωδικοποίησή τους, να μπορούν να τροφοδοτηθούν στο μοντέλο ανάγνωσης χειλιών. Πριν από αυτό, οι προβλεπόμενες εικόνες στόχου υφίστανται περικοπή γύρω από την περιοχή του στόματος (με βάση τα ορόσημα του προσώπου που προβλέπονται με το μοντέλο FAN [18]), μετατροπή σε κλίμακα του γκρι και αλλαγή μεγέθους σε 88×88 εικονοστοιχεία. Υπολογίζουμε τα αντίστοιχα διανύσματα χαρακτηριστικών $f_{lr}(\mathbf{x}_0)$ και $f_{lr}(\hat{\mathbf{x}}_0)$, Αφού υπολογίσουμε τα διανύσματα χαρακτηριστικών, ελαχιστοποιούμε την αντιληπτική απώλεια ανάγνωσης από τα χείλη μεταξύ των πραγματικών και των προβλεπόμενων καρέ. Η απώλεια ορίζεται ως $\mathcal{L}_{lr} = \frac{1}{B}\sum_{i=1}^{B} d(f_{lr}(\mathbf{x}_0), f_{lr}(\hat{\mathbf{x}}_0)$ όπου $d(\cdot,\cdot)$ είναι η απόσταση συνημιτόνου, B είναι το μέγεθος batch και $f_{lr}(\cdot)$ υποδηλώνει τον εξαγωγέα χαρακτηριστικών ανάγνωσης χειλιών. Ο συνολικός στόχος προσαρμογής είναι ο ακόλουθος:

$$\mathcal{L}_{\text{finetune}} = \mathbb{E}_{\mathbf{x}_0,t}\Big[d(f_{lr}(\mathbf{x}_0), f_{lr}(\hat{\mathbf{x}}_0)) + \lambda\|\mathbf{z}_0 - \hat{\mathbf{z}}_0\|_2^2\Big], \qquad \hat{\mathbf{z}}_0 = \text{AEMΔ}(\mathbf{z}_t) \tag{18}$$

[7]WERV: *word error rate* σε επίπεδο *visemes*. Προκύπτει μετατρέποντας τις προβλεπόμενες και πραγματικές μεταγραφές σε visemes χρησιμοποιώντας τη χαρτογράφηση φωνήματος-σε-viseme Amazon Polly. Αντίστοιχα για το CERV, σε επίπεδο χαρακτήρων.

Μοντέλο	Συνθηκοποίηση Συναισθήματος	Μέγεθος Ηχητικού Παραθύρου w	Ποιότητα Εικόνας			Ανάγνωση Χειλιών			
			PSNR↑	SSIM↑	LPIPS↓	WER↓	CER↓	WERV↓	CERV↓
Groundtruth	N/A	N/A	N/A	N/A	N/A	65.5	41.3	65.1	35.6
Baseline	✗	4	**26.69**	0.857	0.047	102.2	82.1	99.3	77.3
	✓	8	26.43	0.854	**0.046**	108.2	82.3	105.1	77.7
Finetuned	✓	4	25.97	**0.858**	0.058	97.3	80.1	96.9	75.0
	✓	8	25.75	0.848	0.054	95.2	79.2	94.0	75.1
	✗	4	25.42	0.845	0.058	91.1	79.1	88.2	74.1
	✗	8	25.87	0.857	0.055	**89.8**	**77.3**	**83.9**	**73.2**

Πίνακας Γ.6: Ποσοτική σύγκριση μεταξύ των μοντέλων σύνθεσης ομιλούντος προσώπου, με βάση το σύνολο δεδομένων MEAD. Όλες οι διαμορφώσεις χρησιμοποιούν ηχητικά χαρακτηριστικά που έχουν εξαχθεί από το τελευταίο transformer μπλοκ κωδικοποιητή του wav2vec 2.0.

όπου η θορυβώδης λανθάνουσα αναπαράσταση $\mathbf{z}_t$ λαμβάνεται χρησιμοποιώντας την κανονική διαδικασία θορυβοποίησης ΑΠΜΔ.

Ο Πίν. Γ.6 παρουσιάζει μια ποσοτική σύγκριση μεταξύ των βασικών και βελτιστοποιημένων μοντέλων μας, με διαφορετικά μήκη ηχητικών παραθύρων και διαμορφώσεις συνθηκοποίησης. Η καλύτερη επίδοση ανάγνωσης χειλιών επιτυγχάνεται ύστερα από εφαρμογή του προτεινόμενου αλγορίθμου βελτιστοποίησης, με μήκος ηχητικού παραθύρου $w = 8$ και μόνο συνθηκοποίηση ήχου. Αν και οι μετρικές αξιολόγησης ανάγνωσης χειλιών παρουσιάζουν σημαντικές βελτιώσεις μετά την βελτιστοποίηση, παρατηρούμε μια ελαφρά υποβάθμιση όσον αφορά την ποιότητα των συντεθέμενων εικόνων. Αυτό δικαιολογείται από το γεγονός ότι εφαρμόζουμε απαθορυβοποίηση ΑΕΜΔ σε θορυβώδεις λανθάνουσες αναπαραστάσεις που έχουν υποστεί θορυβοποίηση με μεταβλητό αριθμό βημάτων t ανά επανάληψη εκπαίδευσης, ενώ χρησιμοποιούμε επίσης έναν εξαιρετικά χαμηλό αριθμό βημάτων ΑΕΜΔ $T_{\text{tune}} = 8$. Για να λειτουργήσει σωστά η απαθορυβοποίηση ΑΕΜΔ, θα πρέπει να είχαμε εφαρμόσει έναν σταθερό αριθμό βημάτων θορυβοποίησης $t = T_{\text{ΑΠΜΔ}}$ ανά επανάληψη. Ωστόσο στην πράξη, μια τέτοια προσέγγιση οδηγεί σε υπερπροσαρμογή του προεκπαιδευμένου ΜΛΔ και πιο σοβαρή υποβάθμιση τόσο στην ποιότητα της εικόνας όσο και σε επιδόσεις σχετικά με ανάγνωση χειλιών.

Chapter 1

Introduction

1.1 Motivation

Deep generative modeling is one of the most intriguing computational technologies today, even pushing human ingenuity. As *Generative Adversarial Networks* (GANs) [55] have proven capable of generating high-quality samples, they managed to draw a lot of attention in the last 10 years. But recently, even more potent generative methods, like *Diffusion Models* [65, 146] have emerged, posing a threat to the dominance of GANs in the production of synthetic data.

Due to more consistent training compared to GANs and higher-quality generated samples, diffusion models are quickly rising in popularity. These models are able to overcome some well-known GAN shortcomings, such as mode collapse, adversarial learning costs, and convergence failure. In contrast to GANs, which learn to retrieve the original data from the noisy ones by contaminating them with Gaussian noise, diffusion models employ a very different approach to training. These models are also found to be appropriate from the perspective of scalability and parallelizability, which increases their appeal. Additionally, because the foundation of their training process is the correction of modest adjustments made to the original data, they develop a data distribution whose samples closely resemble the latter, hence the created samples can be very realistic. Due to these characteristics, diffusion models have had a significant impact on the state-of-the-art in picture production, resulting in astounding outcomes [33, 122, 125].

Diffusion models are quickly finding use in both low-level and high-level vision tasks because of their incredible generative capabilities, including but not limited to image denoising [65, 109], inpainting [44], image super-resolution [91, 100], semantic segmentation [9, 56, 175], semantic image synthesis [125] and image-to-image translation [19, 76, 130, 184]. Unsurprisingly, there has been a constant rise in the number of research publications coming in this area since the key breakthrough of diffusion probabilistic models [65] over the original idea of diffusion modeling [146], and new fascinating models are emerging every day. Particularly with the GLIDE [110], DALL-E 2 [122], Imagen [131], and Stable [125] models that allowed for high quality text-to-image generation, diffusion modeling has experienced significant social media excitement. The text-to-video or text-to-3D creation techniques, like Imagen Video [67], Make-a-Video [145] and DreamFusion [118], which produce videos that look incredibly realistic, have recently added to the overall buzz surrounding diffusion models. Fig. 1.1 illustrates curated examples corresponding to some of the aforementioned tasks and models.

The current thesis revolves around facial expression manipulation of images and talking face videos. To this end, we will try to leverage the expressive power of diffusion models, which have already shown compelling quality in complex image modeling, but still remain a somewhat new and unexplored part of the available literature. In the next upcoming paragraphs of this introductory chapter, we will briefly discuss about facial manipulation methodologies in general but we will also drill down on deepfake technologies for face swapping, as well as discuss potential benefits and threats relative to the latter.

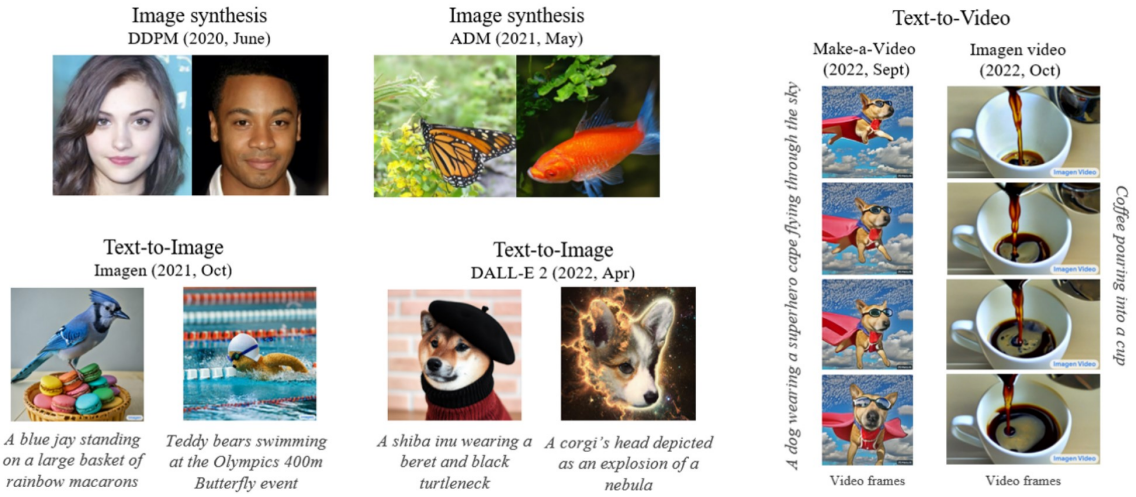


Figure 1.1: Curated examples from image synthesis, text-to-image and text-to-video tasks using [33, 65, 67, 122, 145]. Source: [164].

1.2 Deepfakes

Deepfakes are hyper-realistic videos that have been digitally altered to represent individuals saying and doing things that have never actually occurred. They employ "deep" learning with the aim of producing "fake" imagery. More specifically, deepfake methods use neural networks that learn to replicate a person's facial expressions, mannerisms, speech, and inflections by analyzing enormous quantities of data samples. In order to train a deep learning system to swap faces, two people's video footage is fed into the process. Deepfakes, in other words, utilize facial mapping technology and AI to replace a person's face in a video with the face of another person. It was not until 2017, when a Reddit user posted recordings of famous people in unflattering sexual settings that deepfakes started to gain wide public attention.

In general, deepfakes prey on social media sites, where conspiracy theories, rumors, and disinformation can spread quickly, because users tend to follow the "herd". Meanwhile, an ongoing "infopocalypse" encourages people to only trust information that confirms their preexisting beliefs and originates from their social networks, such as family, close friends, or relatives. In fact, even if they think it might be false, many people are open to anything that supports their preexisting beliefs. Inexpensive fakes, or low-quality videos with minimally altered real content, are already pervasive due to the accessibility of inexpensive hardware, such as effective graphics processing units. There is an increasing amount of open source software available for creating realistic, high-quality deepfakes, thus the current increase in misinformation through deepfake technology. Fig. 1.2 presents an information trust chart for deepfakes.

1.2.1 Benefits

Movies, instructional media, digital communications, games, entertainment, social media, healthcare, material science, and numerous economic sectors including e-commerce and fashion are just a few of the industries that benefit from deepfake technology.

Firstly, deepfake technology has many advantages for the movie industry. For instance, it can be used to update film footage rather than reshoot them or to create artificial voices for performers who lost theirs due to illness. Moviemakers will be able to reproduce iconic movie moments, develop new films with performers who have passed away, apply special effects and cutting-edge face editing in post-production, and turn amateur footage into polished pieces of work. Deepfake technology also enables automatic and lifelike voice dubbing for films in any language, enhancing the viewing experience for different audiences of movies and instructional media. Language barriers were overcome by a 2019 worldwide malaria awareness campaign featuring David Beckham thanks to an instructive commercial that used voice/visual-altering technology to make him appear bilingual. Similar to how deepfake technology can improve eye contact and break down language barriers during video conference calls

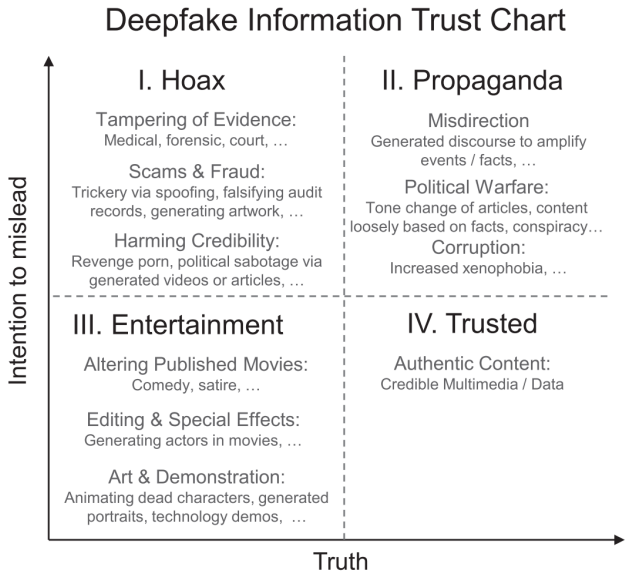


Figure 1.2: A deepfake information trust chart. Source: [105].

by translating words while simultaneously changing lip and facial motions.

Deepfake technology enables digital doubles of humans, realistic-sounding and smart-looking assistants, and increased telepresence in online games and virtual chat environments. Better internet communication and personal interactions can result from this. The technology can also be beneficial in the social and medical sectors. Deepfakes may be able to assist a mourning loved one in saying goodbye to a deceased friend by "bringing them back to life", digitally. This may help people cope with the death of a loved one. Moreover, technology can be used to digitally reconstruct an amputee's limb or help transgender people better identify with their preferred gender. Even adults with Alzheimer's can benefit from deepfake technology by interacting with a younger face they may remember. In order to accelerate the development of new materials and medical treatments, researchers have also investigated the use of GANs to detect anomalies in X-rays.

In addition tech companies are intrigued by the possibility of brand-applicable deepfake technology because it has the potential to significantly alter e-commerce and advertising. Deepfakes also permit the creation of hyperpersonal content that transforms users into models. For example, the technology permits virtual fittings so that users can see how an outfit will look on them before making a purchase and can produce customized fashion ads based on the viewer, the time of day, and the weather.

1.2.2 Possible Threats

Deepfakes pose a serious threat to our society, political system, and economy because they i) put pressure on journalists trying to distinguish between real and fake news, ii) jeopardize national security by spreading propaganda and meddling in elections, iii) undermine public confidence in government information, and iv) raise cybersecurity concerns for individuals and businesses.

Due to deepfakes, it is very likely that the media sector will have to deal with a significant problem relative to customer confidence. Deepfakes are more dangerous than "conventional" fake news since they are more difficult to detect and cause the spread of misinformation. For instance, technology enables the creation of news videos that appear to be real but actually aren't, endangering the credibility of journalists and the media in general. Also, gaining access to video footage taken by a witness of an incident might give a news organization a competitive advantage, but the risk increases if the offered film is actually manipulated.

The intelligence community is concerned about deepfakes being potentially used to undermine election campaigns and endanger national security by spreading political propaganda. U.S. intelligence agencies have frequently issued alerts regarding the danger of foreign interference in American politics, particularly in the run-up to elections. In today's disinformation wars, inserting words into a popular video can be a potent weapon because it can readily sway voter opinion. A deepfaked

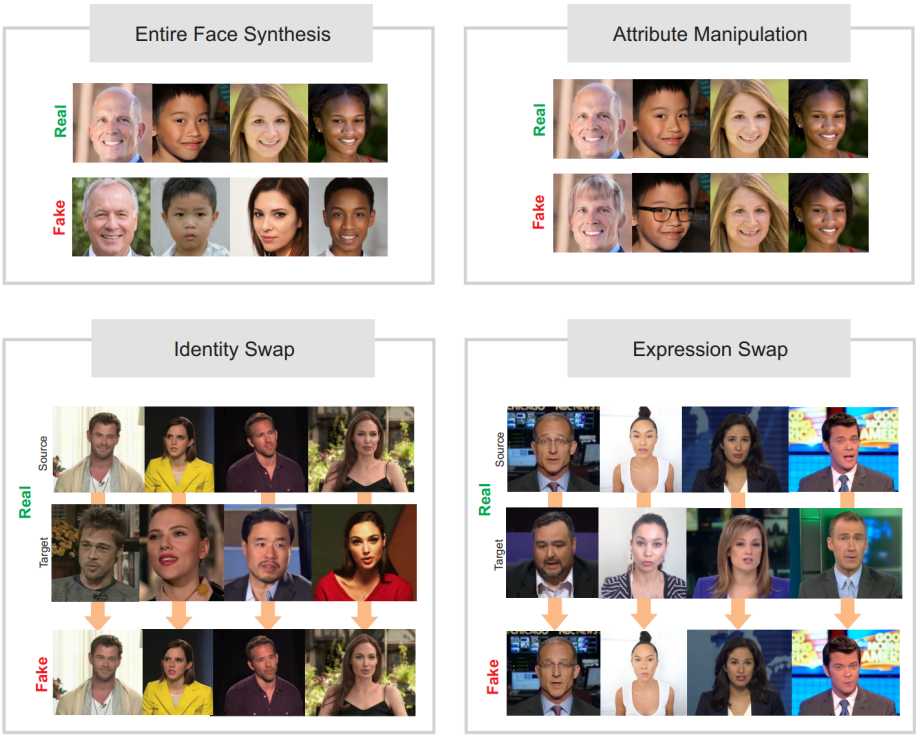


Figure 1.3: Real and fake samples for each one of the four types of facial manipulations, i.e. *entire face synthesis*, *attribute manipulation*, *identity swap* and *expression swap*. Source: [161].

video of a politician using racial slurs or accepting a bribe, a presidential candidate confessing to a crime, alerting another nation to impending war, a government official appearing to be in a precarious position, confessing to a plot to carry out a conspiracy, or soldiers killing civilians abroad, could be produced by a foreign intelligence agency. While such fabricated movies would probably spark riots, turmoil, and disruptions of potential elections, foreign nations might end up executing foreign policies based on fiction, even leading to the outbreak of war.

While fake recordings of government officials saying things that never happened cause people to question authorities, deepfakes are likely to hinder digital literacy and individuals' trust toward information provided by authorities. In fact, AI-generated spam as well as false news that is built on discriminatory text, phony videos, and a slew of conspiracy theories can rapidly spread through society nowadays. The "information" apocalypse or "reality apathy" phenomenon is a result of people feeling that the majority of information, including videos, simply cannot be trusted. This is why the most harmful aspect of deepfakes may not be disinformation per se. Furthermore, due of their ingrained belief that everything they do not want to accept must be fake, people may even reject authentic footage as being fraudulent. In other words, rather than people being tricked, the biggest concern is about people starting to see everything as deception.

Another risk induced by deepfake technology are cybersecurity-related issues. The corporate world has already expressed interest in defending themselves against viral frauds because deepfakes could be used to manipulate the stock and market in general by, for instance, exposing a chief executive using racial or gendered slurs, announcing a fake merger, inflating financial losses, declaring bankruptcy, or appearing to be guilty of a crime. Deeply fabricated announcements of new products or porn could also be used to damage a company's reputation, threaten management, or blackmail them. Moreover, deepfake technology makes it possible to impersonate an executive in real-time via digital means, such as by asking a worker to provide urgent money or divulge sensitive information. Last but not least, deepfake technology may also establish a false identity and, in live-stream recordings, change an adult's face into a child's or younger person's face, which raises concerns about how child predators could utilize this technology.

1.3 Facial Manipulations

In the following paragraphs, we will try to briefly describe the main categories relative to facial manipulations, according to [161]. These categories include, *entire face synthesis*, *face swap*, *attribute manipulation* and *expression swap*. As we will later discuss, deepfakes constitute a *face swapping* methodology. Fig. 1.3 provides an overview of all four of the aforementioned types of facial manipulations.

In *entire face synthesis*, powerful GAN models are employed with the aim of generating full images of faces that do not exist. These methods produce stunning results, producing realistic facial representations of the highest caliber. The video game and 3D modeling businesses stand to gain from this manipulation, but it might also be used for undesirable purposes like the construction of incredibly convincing phony profiles in social networks to spread false information. The primary publicly accessible databases for research on the identification of image manipulation methods reliant on entire face synthesis include 100K-Generated-Images [75], 100K-Faces[1], DFFD [31] and iFakeFaceDB [108]. We ought to mention that all four aforementioned databases only contain fake images generated using the ProGAN [74] and StyleGAN [75] architectures. In order to conduct fake detection experiments, researchers draw real face images from other public databases like FFHQ [75], CelebA [97] and CASIA-Webface [178].

During *face swapping*, the editing involves substituting one person's face with another. Typically, two distinct strategies are often taken into account: FaceSwap[2] which is a traditional computer graphics-based technology and Deepfakes[3]. Numerous videos of this kind of manipulation that are incredibly realistic can be easily found on YouTube. The film industry, in particular, might profit from this kind of manipulation. The manufacture of celebrity pornographic videos, hoaxes, and financial fraud, among many other negative uses, are possible on the other side. Public databases that are commonly used for this type of facial manipulations can be grouped under two separate generations. The first generation includes: i) UADFV [92], comprised of 49 real YouTube videos and 49 fake videos which had been face swapped with Nicholas Cage's face, ii) DeepfakeTIMIT [86], comprised of 620 artificially generated fake videos from 32 participants, iii) FaceForensics++ [127], comprised of 1,000 real YouTube videos and 1,000 fake counterparts, generated with both computer graphics and Deepfake approaches. Second generation datasets, which differ from the latter due to their higher realism include: i) DeepFakeDetection[4], comprised of 363 real videos and 3068 fake ones, generated through the DeepFake FaceSwap GitHub implementation, ii) Celeb-DF [93], comprised of 890 real YouTube videos and 5,639 fake ones, generated through a refined version of a public DeepFake generation algorithm, iii) DFDC Preview [37], comprised of 1,131 real videos from 66 paid actors as well as 4,119 fake ones, generated using two different unknown approaches.

Attribute manipulation, often referred to as face editing or face retouching, entails changing the skin or hair color, the age, the gender, the presence or absence of glasses, and other features of the face. Typically, a GAN is used for this manipulation process. The well-known FaceApp[5] smartphone app is one instance of this kind of manipulation. With the use of this technology, customers may virtually try on a wide variety of things, including eyeglasses, cosmetics, and hairstyles. Few databases are publicly available for research in this area, with the most well known one being DFFD [31]. Notable attribute manipulation techniques include IcGAN [116], Fader Networks [89], StarGAN [25], STGAN [96] and AttGAN [63].

Lastly, *expression swapping* is a form of manipulation, often referred to as face reenactment, that entails changing the subject's expression on their face. The most well-known manipulation techniques suggested in the literature again involve some kind of GAN architecture. Face reenactment poses more challenges compared to static face generation. For example, when the identities of source and driving faces are different, the quality of generated faces can be affected. Texture distortion, identity loss, and shape bias are some additional common problems. Current research aims at enhancing models either with additional forms of input (e.g., AUs [43]) or with better architectures (additional blocks, attention modules, etc.). Expression swapping will constitute the main focus of the current thesis.

[1] https://generated.photos/
[2] https://github.com/MarekKowalski/FaceSwap
[3] https://github.com/deepfakes/faceswap
[4] https://ai.googleblog.com/2019/09/contributing-data-to-deepfake-detection.html
[5] https://www.faceapp.com

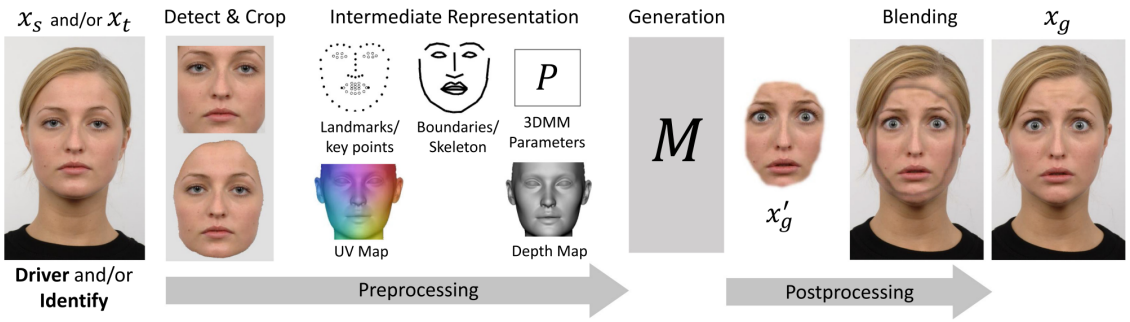


Figure 1.4: A typical reenactment processing pipeline of a deepfake system, where usually only a subset of the depicted steps is executed. Source: [105].

1.4 Expression Swapping (Face Reenactment)

Excluding expression swapping, all of the aforementioned techniques provide stunning results, but they can only be used to change well-defined facial characteristics, like hair color, or to duplicate and composite other people's facial expressions onto their own. An other type of deepfake that might significantly boost the impact of this technology and make it easier for it to be included into the VFX industry is the ability to obtain semantic control over the emotions that are expressed through facial expressions. The significance of this kind of manipulation is amply demonstrated during filming since, despite the planned nature of the spoken words, obtaining the required actor's feeling frequently necessitates numerous attempts. The altering of facial performance would be conveniently done in the post-production stage of a strong emotion editing solution. In static photographs, facial expression manipulation has previously created visually appealing outcomes.

Expression reenactment transforms a person's identity into a puppet, providing assailants the greatest degree of freedom to have the impact they want. We should point out that expression reenactment existed before deepfakes became common. In 2003, researchers morphed models of 3D scanned heads [14]. In 2005, it was shown how this can be done without a 3D model [20], and through warping with matching similar textures [53]. Later, it was demonstrated how 3D parametric models can be used to achieve high-quality and real-time results with depth sensing and ordinary cameras [157, 159, 160]. Regardless, nowadays, deep learning-based approaches are known as the predominant way of generating truthful content. A typical and generic face reenactment processing pipeline can be seen in Fig. 1.4.

1.4.1 Existing Methods

For the purposes of this introductory overview, the first section will concentrate on the most well-known methods; Face2Face [157] and NeuralTextures [158], which swap out one person's facial expression in a video with another person's facial expression (also in a video). FaceForensics++ [127], an expansion of FaceForensics [128], is a well-known database accessible for study in this field. At first, the Face2Face method constituted the primary focus of the FaceForensics database. This computer graphics technique preserved the target person's identity while transferring the emotion from a source video to a target video. Keyframe selection was done manually. To monitor the emotion over the remaining frames of each video, the first frames of each video were used to create a temporary facial identity (3D model). Finally, fake videos were created by applying 76 blendshape coefficients that made up each frame's source expression parameters to the target video. Later, a novel learning strategy was introduced based on NeuralTextures in FaceForensics++. The aforementioned rendering technique learns a neural texture of the target individual, including a rendering network, using the original video data. In particular, a patch-based GAN-loss similar to that seen in Pix2Pix [71] was taken into account. Only the mouth-to-face expression correspondence was changed.

Apart from the latter, several other notable methodologies are worth mentioning, that consider solving the problem at both static (images) and dynamic setting (videos). One such popular approach introduced in [5], automatically animated a still portrait using a driving video depicting a different subject and managed to transfer the expressiveness of the driving video to its source counterpart.

In contrast to Face2Face and NeuralTextures methods, which required videos of both the input and target faces, the latter only required a single image of the target. To that end, [180] provided very good results in both one-shot and few-shot learning.

In general, the development of GANs [55] has stimulated an expanding body of study in the area of image manipulation. In that the synthesized image is dependent on another image, a conditional generator [71] is used in the vast majority of works. Through the concept of cycle consistency [187], this makes it possible to translate images between different domains (i.e., image-to-image translation) while maintaining the source image's content. The application of such methods to face images has made it possible to change certain facial characteristics, such as gender and hair color.

Furthermore, several well-known image-based expression manipulation techniques that are based on GAN architectures include but are not limited to: StarGAN v2 [26], GANimation [120], GANmut [30], ExprGAN [34], FACEGAN [162] and ICface [163]. Some of those have considered alternative latent representations for modeling the human emotional spectrum. For instance, ExprGAN makes use of continuous emotion labels that describe the *intensity* of the depicted facial expressions, while in [94], the 2D Valence-Arousal space was utilized. More recently, GANmut proposed a way of obtaining a 2D interpretable conditional label system even when using a dataset annotated with solely categorical labels of basic emotions.

Even though geometry-based emotion manipulation techniques exceed the scope of the current thesis, we ought to acknowledge their impact in the context of face reenactment, as well their contribution to previously described approaches. In certain works, the target actor is controlled through image-warping [5] or neural rendering [180] using 2D face landmarks to capture the actor's expressions. As they provide a decoupled representation of expressions from identity, 3D morphable models [15] (3DMMs) are a very popular option. Traditional [157, 159] methods render the target subject beneath the source expressions on top of the original target footage while reconstructing the target subject's face in three dimensions on the reference video. Conditional GANs are used by learning-based techniques such as DVP [77] and Head2Head++ [39] to display the target subject while meeting the specified requirements (expressions, stance, and eye-gaze).

1.5 Models of Human Affect

In-depth research and study have been conducted on the interpretation, perception, and recognition of human emotion in a number of scientific fields, including biology, psychology, sociology, neurology, and last but not least, computer science. The fields of computer vision and machine learning aim to automate recognition by developing new techniques and algorithms that are capable of producing efficient and reliable encodings of such information, whereas the aforementioned cognitive sciences concentrate on the extraction of the available affective information. Since it is widely used in situations involving human-robot cooperation, social robotics, medical treatment, mental patient surveillance, driver tiredness surveillance, and many other human-computer interaction scenarios, automatic affect recognition is of utmost practical significance.

We first ought to define the theoretical underpinnings on which emotional states are patterned in order to better grasp the ideas of emotion perception, interpretation, and recognition. For a very long time, there has been discussion over the optimal approach to modeling impact, and many different viewpoints have been put up. Three major categories—categorical, dimensional, and componential—can be used to group together the most pertinent models for affective computing. We will attempt to outline the key characteristics, benefits, and drawbacks of each paradigm in respect to affective computing in the paragraphs that follow.

1.5.1 Categorical Models

Emotions are categorized using categorical models into distinct groups that are simple to identify and articulate in everyday language. The seminal work of Ekman and Friesen [41, 42] and its underlying presumptions regarding the universality of a set of six basic emotions, namely happiness, sadness, fear, anger, disgust, and surprise, are credited for the significant growth in categorical emotion models. A depiction of the aforementioned emotions is illustrated in Fig. 1.5. The universal emotions hypothesis has unquestionably been the main instrument in research related to emotional computing because of the simplicity of the categorical models of emotion and the associated universality claim.

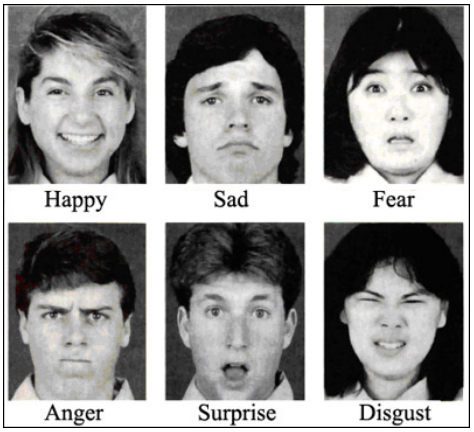


Figure 1.5: The categorical way of describing affect on the basis of the universal set of six emotions, i.e. happiness, sadness, fear, anger, surprise and disgust. Source: [111].

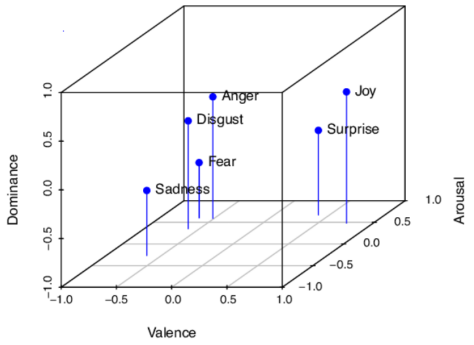


Figure 1.6: The continuous way of describing affect in the form of a point in 3D Valence-Arousal-Dominance (VAD) space. Source: [7].

The categorical models' simplicity, on the other hand, prevents them from accurately capturing and describing more nuanced emotional experiences. In the context of the current thesis, the categorical model will be mostly considered, including minor additions of neutral and contemptuous emotions. Regardless of that, we will briefly mention dimensional/continuous and componential models, for the sake of completeness.

1.5.2 Dimensional Models

Dimensional models [103, 129] are the second most well-known and often employed approach for explaining emotions, just after categorical models. An affective state is portrayed as a point on a continuum spanned by a number of independent dimensions. Over time, psychological study has led to the emergence of a number of different dimensional models. Yet, the Pleasure-Arousal-Dominance (PAD) paradigm is the one that affective computing uses the most frequently.

Many investigations and experiments have shown that the Pleasure-Arousal-Dominance components are independent of one another since any value along one dimension can occur simultaneously with any value along either of the other two. In addition, bipolarity is defined for the three dimensions. Pleasure (or valence) describes the good or negative aspects of an experience, ranging from intense suffering or unhappiness to extreme joy or ecstasy. Arousal is a phrase used to describe the degree of activation, mental attentiveness, and physical activity that can range from sleep to extremes of frenetic enthusiasm. Lastly, dominance represents the amount of control over others and the surrounding environment, ranging from total lack of control and a sense of vulnerability on one end to the opposite extreme of feeling influential and in control. An illustration of the 3D VAD space is shown in Fig. 1.6.

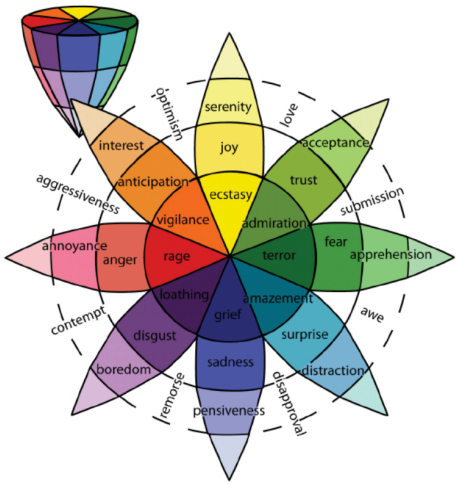


Figure 1.7: The componential way of describing affect on the basis Plutchik's wheel of emotions. Source: [117].

1.5.3 Componential Models

In terms of descriptive capacity, complementary models fall in between categorical and dimensional models. Componential models organize emotions in a hierarchical way, so that emotions belonging to higher or superior layers can be broken into a collection of more primitive and basic emotions belonging to the exact previous layers. The componential emotion model first proposed by Plutchik [117] constitutes the most notable instance of the latter. In addition, he conceptualized primary emotions (anger, fear, sadness, disgust, surprise, anticipation, trust, and joy) in a fashion analogous to a color wheel, placing similar emotions close together and opposites at 180 degrees apart, like complementary colors. Following this pattern of color theory, we can describe emotions which result from the combination of pairs of fundamental emotions, called dyads, in the same way that red and blue make purple. Furthermore, Plutchik extended the initial circumplex model into a third dimension representing the intensity of emotions, resulting in a structured model that was shaped like a cone. The vertical dimension of the cone represents intensity while the horizontal plane represented degrees of similarity among the emotions. An illustration of Plutchik's expanded circumplex model is illustrated in Fig. 1.7.

In contrast to the aforementioned categories and dimensional models, these sorts of models are rarely used in the context of affective computing and automatic emotion detection-related research. Yet, they should be taken into account as a successful compromise between readability and expressiveness.

1.6 Talking Face Generation

The goal of talking face video generation, also known as lip motion sequence generation, is to create lip motion sequences that are consistent with the driving source (a passage of text or speech). Based on synthesizing the lip motion, the talking head's face characteristics, such as facial expressions and head movements, must also be taken into account in the process. Talking face generation can be considered as the dual problem of *visual speech recognition*, as shown in Fig. 1.8.

To achieve the dynamic mapping of driving sources to lip motion data in the early talking face video generating approaches, researchers primarily used cross-modal retrieval [17] or HMM-based algorithms [51, 177]. More recently, talking face video generation has benefited from the quick development of deep learning technology, which has sped up the development of techniques based on convolutional, recurrent, transformer as well as NeRF [104] architectures. We will divide all of the upcoming techniques into four categories, namely coefficient-based, landmark-based, vertex-based and end-to-end. To this end, we consult [141].

Coefficient-based Methods One of the most used face coefficient models, the Active Appearance Model (AAM), represents variations in shape and texture as well as their association. In order to cre-

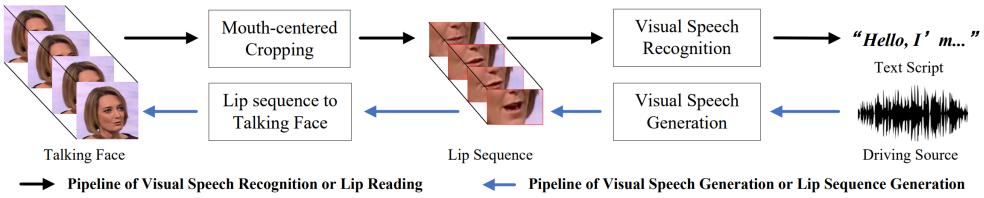


Figure 1.8: The two primal-dual fundamental problems of visual speech analysis; *visual speech recognition* and *talking face generation*. Source: [141].

ate a photo-realistic talking head, in [47] a two-layer BiLSTM network was used to estimate AAM coefficients of the mouth area based on an overlapping triphone input. Apart from 2D coefficient-based, 3DMMs [77, 78] have also been utilized as a 3D face-parametric representation, for video-oriented face2face translation. The 3DMM coefficients contain the rigid head pose parameters, facial identity coefficients, expression coefficients, gaze direction parameters for both eyes and spherical harmonic illumination coefficients. Such frameworks usually include the following steps: i) Train a network to map the driving source to the facial expression coefficients since facial expression coefficients contain implicit visual speech information. ii) To obtain the 3DMM coefficients of the reference identity image, use a deep face reconstruction model that has been previously trained. iii) To create hybrid 3DMM coefficients, combine the 3DMM coefficients from the reference identity image and the projected face expression coefficients. iv) Create talking videos by using a generation network or GPU rendering.

Landmark-based Methods The rigid and non-rigid facial deformations brought on by head motions and facial expressions are captured by facial landmark points around facial components. "Synthesizing Obama" [155] constitutes one of the most pioneering pieces of related literature where the proposed model outputted the synthesized talking face video of former US President Barack Obama, following the pipeline of facial texture synthesis, video re-timing, and target video compositing. Common landmark-based backbones include: LSTM+U-Net [87], LSTM+C-GAN [72] and LSTM+Convolutional-RNN [21]. All the above works, utilize 2D landmarks. Besides 2D landmark based approaches, mapping driving source to 3D landmarks is also widely explored [99, 186].

Vertex-based Methods 3D facial vertices are another popularly used 3D face model in talking face generation. Initial approaches were subject-specific [73] and were later generalized to multiple subjects as well, with the well-known VOCA model [29]. The latter combined DeepSpeech [60] features and speaker-specific features with the aim of producing a 3D mesh of 5023 vertex displacements. More recently, the FaceFormer [48] model autoregressively encoded the long-term audio context information and predicted a sequence of 3D face vertices.

End-to-End Methods End-to-end pipelines are designed to synthesize talking face sequences directly from the driving source using an end-to-end learning approach without the use of any intermediary facial parameters. One of the first end-to-end frameworks was Speech2Vid [27] where an image decoder sought to produce synthesized images based on fused speech and identity features, with the latter being extracted from the reference image using an identity encoder, while the former features are extracted from the driving audio using a respective audio encoder. Several GAN-based approaches were proposed in an attempt to overcome the limitations of the aforementioned model. One such approach is DAVS [185] which placed more emphasis on the extraction of disentangled speech and identification features through supervised adversarial training. Apart from GANs, neural radiance fields have also found success in this field. For example, AD-NeRF [59] utilized DeepSpeech audio features as a conditional input so as to learn an implicit neural scene representation function that maps audio features to dynamic neural radiance fields for talking face rendering.

1.7 Thesis Outline

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

Chapter 2 lays the theoretical foundations regarding generative modeling, as a whole. More specifically, we present the structure and features of all major, practical and modern generative frameworks, including GANs, Variational Autoencoders (VAEs), Energy-based models (EBMs), Autoregressive as well as Flow-based models. Furthermore, the necessary mathematical background is going to be established with the aim of obtaining a clearer insight of the functionalities of all the aforementioned modules.

Chapter 3 dives deeper into diffusion models, exploring all of their variants in both continuous and discrete time formulations, with emphasis laid on the latter. There we will also touch upon vector quantization and perceptual image compression, subjects which will prove useful in the context of building lightweight and resource-efficient diffusion models.

Moving on, Chapter 4 constitutes our main case study where we explicitly tackle the problem of image-based face reenactment, on the basis of the largest available, most challenging and emotionally annotated face dataset. We operate diffusion models in a deterministic manner, achieving near cycle consistency between source and target images. Moreover, we adapt a language-image pre-trained model with the sole purpose of directly finetuning diffusion models in latent space. We conduct extensive experiments so that our proposed methodologies are evaluated in terms of both objective metrics as well as subjective studies.

Chapter 5 aims at extending diffusion-based generative techniques in dynamic video settings. More specifically, we tackle the task of talking face synthesis. We conduct extensive ablation studies on the basis of the MEAD dataset. To the best of our knowledge, we propose the first proper finetuning methodology for better lip-synching based on guidance from a lip reading expert network.

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

Chapter 2

An Overview of Generative Models

Given observations of samples $\boldsymbol{x}$, the goal of a generative model is to learn to approximate the true data distribution $p_{\text{data}}(\boldsymbol{x})$, so as to be able to generate new, "synthetic" samples $\tilde{\boldsymbol{x}} \sim \tilde{p}_{\boldsymbol{\theta}}(\tilde{\boldsymbol{x}})$ at will. There are several well-established directions in current literature, such as Generative Adversarial Networks, Variational Autoencoders, Energy-Based, Autoregressive models and Normalizing Flows. We proceed with a brief and high-level introduction of the core concepts behind each of the aforementioned families of generative models. For guidance throughout this process, we also consult [16, 82, 152]. For simplicity, we assume vectorized input and latent representations, unless specified otherwise.

2.1 Generative Adversarial Networks

Generative Adversarial Networks (GANs) [55] consist of two networks, a discriminator $D: \mathbb{R}^n \to [0, 1]$ which estimates the probability that a sample comes from the data distribution $\boldsymbol{x} \sim p_{\text{data}}(\boldsymbol{x})$, and a generator $G: \mathbb{R}^m \to \mathbb{R}^n$ which given a latent variable $\boldsymbol{z} \sim p_{\boldsymbol{z}}(\boldsymbol{z})$ captures p_{data} by tricking the discriminator into thinking its samples are real. This is achieved through adversarial training of the networks: D is trained to correctly label training samples as real and samples from G as fake, while G is trained to minimise the probability that D classifies its samples as fake. This can be formulated as a two-player mini-max game between D and G, optimizing the value function $V(G, D)$:

$$\min_G \max_D V(G, D) = \mathbb{E}_{\boldsymbol{x} \sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z} \sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))] \tag{2.1}$$

For a fixed G, the objective for D can be reformulated as follows:

$$\begin{aligned}
\max_D V(D) &= \mathbb{E}_{\boldsymbol{x} \sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{x} \sim p_g(\boldsymbol{x})}[\log(1 - D(\boldsymbol{x}))] \\
&= \mathbb{E}_{\boldsymbol{x} \sim p_{\text{data}}(\boldsymbol{x})}\left[\log \frac{p_{\text{data}}(\boldsymbol{x})}{p_{\text{data}}(\boldsymbol{x}) + p_g(\boldsymbol{x})}\right] + \mathbb{E}_{\boldsymbol{x} \sim p_g(\boldsymbol{x})}\left[\log \frac{p_g(\boldsymbol{x})}{p_{\text{data}}(\boldsymbol{x}) + p_g(\boldsymbol{x})}\right] \\
&= D_{\text{KL}}\left(p_{\text{data}} \,\middle\|\, \frac{1}{2}(p_{\text{data}} + p_g)\right) + D_{\text{KL}}\left(p_g \,\middle\|\, \frac{1}{2}(p_{\text{data}} + p_g)\right) - \log 4
\end{aligned} \tag{2.2}$$

We recognize in the previous expression the Jensen–Shannon divergence between the model's distribution and the data generating process. Since the JS-divergence between two distributions is always non-negative and zero only when they are equal, we have shown that $V^* = -\log 4$ is the global minimum of $V(G)$ and that the only solution is $p_g = p_{\text{data}}$, i.e., the generative model perfectly replicating the data generating process.

GAN training is known to be notoriously hard, with vanishing gradients and mode collapse constituting the most common problems. Since the cause of these issues can be linked with the use of JS-divergence, other loss functions have been proposed, with the most notable being WGAN [4], LSGAN [102], EBGAN [183] and InfoGAN [23].

2.1.1 Wasserstein GAN

Wasserstein GAN (WGAN) training is guided through the minimization of a reasonable and efficient approximation of the *Earth-Mover* (EM) or *Wasserstein-1* distance. The EM distance between the real data distribution p_{data} and that of generated data p_g, is defined as follows:

$$W(p_{\text{data}}, p_g) = \inf_{\gamma \in \Pi(p_{\text{data}}, p_g)} \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\gamma}[\|\boldsymbol{x} - \boldsymbol{y}\|] \tag{2.3}$$

where $\Pi(p_{\text{data}}, p_g)$ denotes the set of all joint distributions $\gamma(\boldsymbol{x}, \boldsymbol{y})$ whose marginals are p_{data} and p_g, respectively. Intuitively, EM distance indicates how much "mass" must be moved to transform one distribution into another. It is intractable to exhaust all the possible joint distributions in $\Pi(p_{\text{data}}, p_g)$ while calculating the infimum in Eq. (2.3), thus a smart transformation of the objective was proposed, based on the Kantorovich-Rubinstein duality theorem [169]:

$$\min_{\boldsymbol{\theta}} \max_{\substack{\boldsymbol{w}\in\mathcal{W} \\ \|f_{\boldsymbol{w}}\|_L \leq K}} \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[f_{\boldsymbol{w}}(\boldsymbol{x})] - \mathbb{E}_{\boldsymbol{z}\sim p(\boldsymbol{z})}[f_{\boldsymbol{w}}(g_{\boldsymbol{\theta}}(\boldsymbol{z}))] \tag{2.4}$$

where $\{f_{\boldsymbol{w}}\}_{\boldsymbol{w}\in\mathcal{W}}$ is a family of parameterized K-Lipschitz functions corresponding to the discriminator, while $g_{\boldsymbol{\theta}}$ denotes the generator who aims at minimizing the value function w.r.t. its parameters $\boldsymbol{\theta}$. Numerous approaches have been proposed with the aim of enforcing K-Lipschitz continuity to the discriminator function $f_{\boldsymbol{w}}$, with the simplest one being gradient clipping. Most notably, WGAN-GP [58] enforced 1-Lipschitz continuity by penalizing the model if the gradient norm of $f_{\boldsymbol{w}}$ deviated from its target value of 1.

Optimizing the Wasserstein distance offers linear gradients, thus eliminating the vanishing gradient problem. Moreover, it is equivalent to minimizing reverse KL-divergence, providing improved stability while training.

2.1.2 Least Squares GAN

Regular GANs adopt the sigmoid cross entropy loss function for the discriminator. This loss function is susceptible to the problem of vanishing gradients when updating the generator using the fake samples that are on the correct side of the decision boundary, but are still far from the real data. *Least Squares Generative Adversarial Networks* (LSGANs) provide remedy for the aforementioned problem. Let a, b be continuous coded labels for fake data and real data, respectively. The LSGAN objective functions are defined as follows:

$$\begin{aligned} \min_D V_{\text{LSGAN}}(D) &= \frac{1}{2}\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[(D(\boldsymbol{x}) - b)^2] + \frac{1}{2}\mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[D(G(\boldsymbol{z})) - a)^2] \\ \min_G V_{\text{LSGAN}}(G) &= \frac{1}{2}\mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[D(G(\boldsymbol{z})) - c)^2] \end{aligned} \tag{2.5}$$

where c denotes the value that the generator wants the discriminator to believe for fake data. Unlike regular GANs, which cause almost no loss for samples that lie a long way in the correct side of the decision boundary, LSGANs will penalize those samples even though they are correctly classified, which in turn results in more gradients being generated, alleviating the problem the vanishing gradient problem.

2.1.3 Energy-Based GAN

Instead of designing a discriminator similar to a classifier, in the context of *Energy-Based GANs* (EBGAN), the discriminator uses an autoencoder which extracts latent features of the input image using an encoder and reconstructs it again through a decoder. This discriminator D outputs the reconstruction error (MSE) between the input image $\boldsymbol{x}$ and its reconstruction $\hat{\boldsymbol{x}}$. The autoencoder is trained using real images and as a result, for a poorly generated image, the reconstruction error is expected to be high. The EBGAN loss functions are the following:

$$\begin{aligned} \mathcal{L}_D(\boldsymbol{x}, \boldsymbol{z}) &= D(\boldsymbol{x}) + [m - D(G(\boldsymbol{z}))]^+ \\ \mathcal{L}_G(\boldsymbol{z}) &= D(G(\boldsymbol{x})) \end{aligned} \tag{2.6}$$

Therefore, the discriminator is trained with two goals in mind; having low reconstruction error $D(\boldsymbol{x})$ for real images and the reconstruction error for generated images to be higher than a predefined threshold m. The generator is simply trained with the aim of lowering the reconstruction error for generated images.

2.1.4 Information Maximizing GAN

The *Information Maximizing GAN* (InfoGAN) is a type of generative adversarial network that modifies the GAN objective to encourage it to learn interpretable and meaningful representations in a completely unsupervised manner. This is done by maximizing the mutual information between a fixed small subset of the GAN's noise variables and the observations.

Formally, InfoGAN is defined as a minimax game with a variational regularization of mutual information and the hyperparameter λ:

$$\min_{G,Q} \max_{D} V_{\text{InfoGAN}}(D, G, Q) = V(D, G) - \lambda \mathcal{L}_I(G, Q) \tag{2.7}$$

where $V(D, G)$ is the standard GAN objective, $\mathcal{L}_I$ denotes the variational lower bound of the mutual information between the latent code $\boldsymbol{c}$ and generator distribution $G(\boldsymbol{z}, \boldsymbol{c})$, while Q is an auxiliary distribution that approximates the posterior $p(\boldsymbol{c}|\boldsymbol{x})$.

2.2 Energy-Based Models

Energy-Based Models (EBMs) are based on the observation that any probability density function $p(\boldsymbol{x})$ for $\boldsymbol{x} \in \mathbb{R}^n$ can be expressed in terms of an energy function $E_{\boldsymbol{\theta}}(\boldsymbol{x}) : \mathbb{R}^n \to \mathbb{R}$ which associates realistic points with low values and unrealistic points with high values. It is relatively straightforward to extend the models and estimation procedures to the case with multiple dependent variables, or with conditioning variables. The density given by an EBM is:

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}) = \frac{\exp\left(-E_{\boldsymbol{\theta}}(\boldsymbol{x})\right)}{Z_{\boldsymbol{\theta}}} = \frac{\exp\left(-E_{\boldsymbol{\theta}}(\boldsymbol{x})\right)}{\int \exp\left(-E_{\boldsymbol{\theta}}(\boldsymbol{x})\right) \mathrm{d}\boldsymbol{x}} \tag{2.8}$$

where $E_{\boldsymbol{\theta}}(\boldsymbol{x})$ is a nonlinear regression function with parameters $\boldsymbol{\theta}$. Since the denominator in Eq. (2.8) is a function of $\boldsymbol{\theta}$, evaluation and differentiation of $\log p_{\boldsymbol{\theta}}(\boldsymbol{x})$ w.r.t. its parameters is intractable. Let $p_{\text{data}}(\boldsymbol{x})$ be the underlying data distribution of a given dataset. Then, $p_{\boldsymbol{\theta}}(\boldsymbol{x})$ can be fitted to $p_{\text{data}}(\boldsymbol{x})$ by maximizing the expected log-likelihood function over the data distribution, i.e. $\mathbb{E}_{\boldsymbol{x} \sim p_{\text{data}}(\boldsymbol{x})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x})]$. The likelihood of an EBM cannot be directly computed due to the intractable normalizing constant $Z_{\boldsymbol{\theta}}$. However, the gradient of the log-likelihood can be estimated utilizing MCMC approaches and allowing for log-likelihood maximization with gradient ascent. The gradient of the log-probability of an EBM decomposes as a sum of two terms:

$$\nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(\boldsymbol{x}) = -\nabla_{\boldsymbol{\theta}} E_{\boldsymbol{\theta}} - \nabla_{\boldsymbol{\theta}} \log Z_{\boldsymbol{\theta}} \tag{2.9}$$

The challenge in Eq. (2.9) is to approximate the second gradient term. This gradient term can be rewritten as the following expectation:

$$\begin{aligned} \nabla_{\boldsymbol{\theta}} \log Z_{\boldsymbol{\theta}} &= \nabla_{\boldsymbol{\theta}} \log \int \exp\left(-E_{\boldsymbol{\theta}}(\boldsymbol{x})\right) \mathrm{d}\boldsymbol{x} = \frac{\int \nabla_{\boldsymbol{\theta}} \exp\left(-E_{\boldsymbol{\theta}}(\boldsymbol{x})\right) \mathrm{d}\boldsymbol{x}}{\int \exp\left(-E_{\boldsymbol{\theta}}(\boldsymbol{x})\right) \mathrm{d}\boldsymbol{x}} \\ &= \int \frac{\exp\left(-E_{\boldsymbol{\theta}}(\boldsymbol{x})\right)}{Z_{\boldsymbol{\theta}}} \left(-\nabla_{\boldsymbol{\theta}} E_{\boldsymbol{\theta}}(\boldsymbol{x})\right) \mathrm{d}\boldsymbol{x} = \int p_{\boldsymbol{\theta}}(\boldsymbol{x}) \left(-\nabla_{\boldsymbol{\theta}} E_{\boldsymbol{\theta}}(\boldsymbol{x})\right) \mathrm{d}\boldsymbol{x} \\ &= \mathbb{E}_{\boldsymbol{x} \sim p_{\boldsymbol{\theta}}(\boldsymbol{x})}\left[-\nabla_{\boldsymbol{\theta}} E_{\boldsymbol{\theta}}(\boldsymbol{x})\right] \end{aligned} \tag{2.10}$$

Therefore, a one-sample unbiased estimate of the log-likelihood gradient can be obtained using $\nabla_{\boldsymbol{\theta}} \log Z_{\boldsymbol{\theta}} \approx -\nabla_{\boldsymbol{\theta}} E_{\boldsymbol{\theta}}(\tilde{\boldsymbol{x}})$, where $\tilde{\boldsymbol{x}} \sim p_{\boldsymbol{\theta}}(\boldsymbol{x})$, provided that it is possible to draw random samples from the EBM. A method for efficient MCMC sampling from EBMs known as Langevin MCMC [57, 114], makes use of the fact that the gradient of the log-probability w.r.t. $\boldsymbol{x}$, i.e. the score, is equal to the negative gradient of the energy:

$$\nabla_{\boldsymbol{x}} \log p_{\boldsymbol{\theta}}(\boldsymbol{x}) = -\nabla_{\boldsymbol{x}} E_{\boldsymbol{\theta}} - \nabla_{\boldsymbol{x}} \log Z_{\boldsymbol{\theta}} = -\nabla_{\boldsymbol{x}} E_{\boldsymbol{\theta}} \tag{2.11}$$

When using Langevin MCMC to sample from $p_{\boldsymbol{\theta}}$, an initial sample $\boldsymbol{x}$ is first drawn from a simple prior distribution, and then a Langevin diffusion process for K steps with step size $\epsilon > 0$ is simulated as follows:

$$\boldsymbol{x}^{(k+1)} \leftarrow \boldsymbol{x}^{(k)} + \frac{\epsilon^2}{2} \underbrace{\nabla_{\boldsymbol{x}} \log p_{\boldsymbol{\theta}}(\boldsymbol{x}^{(k)})}_{-\nabla_{\boldsymbol{x}} E_{\boldsymbol{\theta}}} + \epsilon \boldsymbol{z}^{(k)}, \quad \boldsymbol{z}^{(k)} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \quad k \in [K] \tag{2.12}$$

When $\epsilon \to 0$ and $K \to \infty$, $\boldsymbol{x}^{(K)}$ is guaranteed to distribute as $p_{\boldsymbol{\theta}}$ under some regularity conditions.

2.2.1 Diffusion Models as Energy Based Models

As we will later discuss in Chapter 3, the sampling procedure of Denoising Diffusion models [65, 146], is functionally similar to that of EBMs. At a timestep t, in diffusion models, images are updated using a learned denoising network $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$, while in EBMs, images are updated using the gradient of the energy function $\nabla_{\mathbf{x}} E_{\boldsymbol{\theta}}(\mathbf{x}_t) \propto \nabla_{\mathbf{x}} \log p_{\boldsymbol{\theta}}(\mathbf{x}_t)$, as indicated by Eq. (2.12). The denoising network $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$ is trained to predict the underlying score of the data distribution when the number of diffusion steps increases to infinity [153]. Similarly, an EBM is trained so that $\nabla_{\mathbf{x}} E_{\boldsymbol{\theta}}(\mathbf{x}_t)$ corresponds to the score of the data distribution as well. In this sense, $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$ and $\nabla_{\mathbf{x}} E_{\boldsymbol{\theta}}(\mathbf{x}_t)$ are functionally the same and a trained diffusion model can be regarded as an implicitly parameterized EBM.

2.2.2 Score Matching

Although Langevin MCMC has allowed EBMs to scale to high dimensional data, training times are still slow due to the need to sample from the model distribution. If two continuously differentiable real-valued functions $f(\boldsymbol{x})$ and $g(\boldsymbol{x})$ have equal first derivatives everywhere, then $f(\boldsymbol{x}) \equiv g(\boldsymbol{x}) + \text{constant}$. When $f(\boldsymbol{x})$ and $g(\boldsymbol{x})$ are log-probability density functions (PDFs), then the normalization requirement implies that $\int \exp(f(\boldsymbol{x}))\, \mathrm{d}\boldsymbol{x} = \int \exp(g(\boldsymbol{x}))\, \mathrm{d}\boldsymbol{x} = 1$, hence $f(\boldsymbol{x}) \equiv g(\boldsymbol{x})$. As a result, one can learn an EBM by matching the first derivatives of its log-PDF (score) to the first derivatives of the log-PDF of the data distribution. For training EBMs, it is easy to transform the equivalence of distributions to the equivalence of scores, because the score of an EBM can be easily obtained by $\nabla_{\boldsymbol{x}} \log p_{\boldsymbol{\theta}}(\boldsymbol{x}) = -\nabla_{\boldsymbol{x}} E_{\boldsymbol{\theta}}$ (Eq. (2.11)).

Assuming $p_{\text{data}}(\boldsymbol{x})$ is the underlying data distribution from which we have a finite number of i.i.d. samples. The score matching objective involves minimizing the Fischer divergence:

$$D_F(p_{\text{data}}(\boldsymbol{x}) \parallel p_{\boldsymbol{\theta}}(\boldsymbol{x})) = \mathbb{E}_{p_{\text{data}}(\boldsymbol{x})}\left[\frac{1}{2} \|\nabla_{\boldsymbol{x}} \log p_{\text{data}}(\boldsymbol{x}) - \nabla_{\boldsymbol{x}} \log p_{\boldsymbol{\theta}}(\boldsymbol{x})\|_2^2\right] \tag{2.13}$$

The expectation w.r.t. $p_{\text{data}}(\boldsymbol{x})$ admits a trivial unbiased Monte Carlo estimator using the empirical mean of samples $\boldsymbol{x} \sim p_{\text{data}}(\boldsymbol{x})$. However, the data score function is usually not available and various methods exist to bypass it. Using integration by parts [70], the intractable term $\nabla_{\boldsymbol{x}} \log p_{\text{data}}(\boldsymbol{x})$ can be replaced with the second derivatives of $E_{\boldsymbol{\theta}}(\boldsymbol{x})$ at the price of increased computational complexity, as the computation of second derivatives is quadratic w.r.t. to the dimensionality of the input.

2.2.3 Denoising Score Matching

The SM objective described in Eq. (2.13) requires that $\nabla_{\boldsymbol{x}} \log p_{\text{data}}(\boldsymbol{x})$ is continuously differentiable and finite everywhere, which is not the case in practice. *Denoising Score Matching* (DSM) [170] aims at alleviating this difficulty by approximating the score using corrupted data samples $\tilde{\boldsymbol{x}} = \boldsymbol{x} + \boldsymbol{\epsilon}$. As long as the noise distribution $p(\boldsymbol{\epsilon})$ is smooth, the resulting noisy data distribution $q(\tilde{\boldsymbol{x}}) = \int q(\tilde{\boldsymbol{x}}|\boldsymbol{x}) p_{\text{data}}(\boldsymbol{x})\, \mathrm{d}\boldsymbol{x}$ is also smooth, and thus $D_F(q(\tilde{\boldsymbol{x}}) \parallel p_{\boldsymbol{\theta}}(\tilde{\boldsymbol{x}}))$ constitutes a proper objective. It holds that:

$$\begin{aligned} D_F(q(\tilde{\boldsymbol{x}}) \parallel p_{\boldsymbol{\theta}}(\tilde{\boldsymbol{x}})) &= \mathbb{E}_{q(\tilde{\boldsymbol{x}})}\left[\frac{1}{2} \|\nabla_{\boldsymbol{x}} \log q(\tilde{\boldsymbol{x}}) - \nabla_{\boldsymbol{x}} \log p_{\boldsymbol{\theta}}(\tilde{\boldsymbol{x}})\|_2^2\right] \\ &= \mathbb{E}_{q(\tilde{\boldsymbol{x}}|\boldsymbol{x})}\left[\frac{1}{2} \|\nabla_{\boldsymbol{x}} \log q(\tilde{\boldsymbol{x}}|\boldsymbol{x}) - \nabla_{\boldsymbol{x}} \log p_{\boldsymbol{\theta}}(\tilde{\boldsymbol{x}})\|_2^2\right] + \text{constant} \end{aligned} \tag{2.14}$$

When $q(\tilde{\boldsymbol{x}}|\boldsymbol{x}) = \mathcal{N}(\tilde{\boldsymbol{x}}; \boldsymbol{x}, \sigma^2\mathbf{I})$ and $\sigma \approx 0$, the DSM objective becomes:

$$\begin{aligned} D_F(q(\tilde{\boldsymbol{x}}) \parallel p_{\boldsymbol{\theta}}(\tilde{\boldsymbol{x}})) &= \mathbb{E}_{p_{\text{data}}(\boldsymbol{x})}\mathbb{E}_{\mathcal{N}(\boldsymbol{z};\mathbf{0},\mathbf{I})}\left[\frac{1}{2}\left\|\frac{\boldsymbol{z}}{\sigma} + \nabla_{\boldsymbol{x}} \log p_{\boldsymbol{\theta}}(\boldsymbol{x} + \sigma\boldsymbol{z})\right\|_2^2\right] \\ &\simeq \frac{1}{2N}\sum_{i=1}^{N}\left\|\frac{\boldsymbol{z}^{(i)}}{\sigma} + \nabla_{\boldsymbol{x}} \log p_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)} + \sigma\boldsymbol{z}^{(i)})\right\|_2^2 \end{aligned} \tag{2.15}$$

where $\{\boldsymbol{x}^{(i)}\}_{i=1}^N \overset{\text{i.i.d.}}{\sim} p_{\text{data}}(\boldsymbol{x})$ and $\{\boldsymbol{z}^{(i)}\}_{i=1}^N \overset{\text{i.i.d.}}{\sim} \mathcal{N}(\mathbf{0}, \mathbf{I})$. The major drawback of adding noise to data arises when $p_{\text{data}}(\boldsymbol{x})$ is already a well-behaved distribution that satisfies the required regularity conditions. In this case, DSM is not a consistent objective because the optimal EBM matches the noisy distribution $q(\tilde{\boldsymbol{x}})$ and not $p_{\text{data}}(\boldsymbol{x})$.

2.2.4 Sliced Score Matching

By adding noise to data, DSM avoids the expensive computation of second-order deriva- tives. However, DSM does not give a consistent estimator of the data distribution, i.e., one cannot directly obtain an EBM that exactly matches the data distribution even with unlimited data. *Sliced Score Matching* (SSM) [151] is an alternative to DSM that is both consistent and computationally efficient. Instead of minimizing the Fisher divergence between two vector-valued scores, SSM randomly samples a projection vector $\boldsymbol{v}$, takes the inner product between $\boldsymbol{v}$ and the two scores, and then compare the resulting two scalars. More specifically, SSM minimizes the following divergence called the sliced Fisher divergence:

$$D_{SF}(p_{\text{data}}(\boldsymbol{x}) \parallel p_{\boldsymbol{\theta}}(\boldsymbol{x})) = \mathbb{E}_{p_{\text{data}}(\boldsymbol{x})}\mathbb{E}_{p(\boldsymbol{v})}\left[\frac{1}{2}\left(\boldsymbol{v}^\top \nabla_{\boldsymbol{x}} \log p_{\text{data}}(\boldsymbol{x}) - \boldsymbol{v}^\top \nabla_{\boldsymbol{x}} \log p_{\boldsymbol{\theta}}(\boldsymbol{x})\right)^2\right] \tag{2.16}$$

where $p(\boldsymbol{v})$ denotes a projection distribution such that $\mathbb{E}_{p(\boldsymbol{v})}[\boldsymbol{v}\boldsymbol{v}^\top]$ is positive definite.

2.3 Variational Autoencoders

One of the key problems associated with EBMs is that sampling is far from straightforward and can require a significant amount of time. To circumvent this issue, it would be beneficial to explicitly sample from the data distribution with a single network pass.

2.3.1 Evidence Lower Bound

In the *Variational Autoencoder* (VAE) setting, we think of the observed data $\boldsymbol{x}$ as represented or generated by an associated unseen latent variable $\boldsymbol{z}$. The latent variables and the observed data are modeled by a joint distribution $p(\boldsymbol{x}, \boldsymbol{z})$. As discussed earlier, likelihood-based approaches aim at maximizing the likelihood $p(\boldsymbol{x})$ of all observed $\boldsymbol{x}$. In this case, this either involves integrating out all latent variables $\boldsymbol{z}$:

$$p(\boldsymbol{x}) = \int p(\boldsymbol{x}, \boldsymbol{z})\, \mathrm{d}\boldsymbol{z} \tag{2.17}$$

or having access to the latent encoder $p(\boldsymbol{z}|\boldsymbol{x})$, as indicated by the chain rule of probability:

$$p(\boldsymbol{x}) = \frac{p(\boldsymbol{x}, \boldsymbol{z})}{p(\boldsymbol{z}|\boldsymbol{x})} \tag{2.18}$$

The above requirements make direct computation and maximization of the likelihood $p(\boldsymbol{x})$ inherently difficult. However, using Eq. (2.17) and (2.18), a term called the Evidence Lower Bound (ELBO) can be derived, which as its name suggests, is a lower bound of the evidence (log-likelihood of the observed data). Given a flexible approximate variational distribution $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$ with parameters $\boldsymbol{\phi}$

which we seek to optimize, the ELBO can be derived as follows:

$$\log p(\boldsymbol{x}) = \log p(\boldsymbol{x}) \int q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}) \, \mathrm{d}\boldsymbol{z} \qquad \left(\int q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}) \, \mathrm{d}\boldsymbol{z} = 1 \right) \tag{2.19}$$

$$= \int \log p(\boldsymbol{x}) q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}) \, \mathrm{d}\boldsymbol{z} \tag{2.20}$$

$$= \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}[\log p(\boldsymbol{x})] \qquad \text{(Def. of expectation)} \tag{2.21}$$

$$= \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{p(\boldsymbol{x}, \boldsymbol{z})}{p(\boldsymbol{z}|\boldsymbol{x})} \right] \qquad \text{(Eq. (2.18))} \tag{2.22}$$

$$= \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{p(\boldsymbol{x}, \boldsymbol{z}) q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}{p(\boldsymbol{z}|\boldsymbol{x}) q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})} \right] \tag{2.23}$$

$$= \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{p(\boldsymbol{x}, \boldsymbol{z})}{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})} \right] + \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}{p(\boldsymbol{z}|\boldsymbol{x})} \right] \tag{2.24}$$

$$= \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{p(\boldsymbol{x}, \boldsymbol{z})}{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})} \right] + D_{\mathrm{KL}}(q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}) \parallel p(\boldsymbol{z}|\boldsymbol{x})) \qquad \text{(Def. of KL divergence)} \tag{2.25}$$

$$\geq \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{p(\boldsymbol{x}, \boldsymbol{z})}{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})} \right] \qquad \text{(KL divergence} \geq 0) \tag{2.26}$$

Maximizing the ELBO becomes a proxy objective with which to optimize a latent variable model; in the best case, when the ELBO is powerfully parameterized and perfectly optimized, it becomes exactly equivalent to the evidence. In the default formulation of the VAE [81], the ELBO is directly maximized. This approach is variational, because we optimize for the best $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$ amongst a family of potential posterior distributions parameterized by $\boldsymbol{\phi}$. The ELBO term can be further dissected:

$$\begin{aligned} \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{p(\boldsymbol{x}, \boldsymbol{z})}{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})} \right] &= \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}) p(\boldsymbol{z})}{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})} \right] \\ &= \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})] - D_{\mathrm{KL}}(q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}) \parallel p(\boldsymbol{z})) = \mathcal{L}_{\boldsymbol{\theta},\boldsymbol{\phi}}(\boldsymbol{x}) \end{aligned} \tag{2.27}$$

where $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$ can be thought as an encoder function and $p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})$ is a deterministic decoder function that converts a given latent vector $\boldsymbol{z}$ into an observation $\boldsymbol{x}$. The encoder of the VAE is commonly chosen to model an isotropic multivariate Gaussian and the prior is often selected to be a standard multivariate Gaussian:

$$q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{z}; \boldsymbol{\mu}_{\boldsymbol{\phi}}(\boldsymbol{x}), \boldsymbol{\sigma}^2_{\boldsymbol{\phi}}\mathbf{I}), \qquad p(\boldsymbol{z}) = \mathcal{N}(\boldsymbol{z}; \mathbf{0}, \mathbf{I}) \tag{2.28}$$

Given a dataset $\mathcal{D}$ with i.i.d. data, the ELBO objective is the sum (or average) of individual-datapoint ELBOs:

$$\mathcal{L}_{\boldsymbol{\theta},\boldsymbol{\phi}}(\mathcal{D}) = \sum_{\boldsymbol{x} \in \mathcal{D}} \mathcal{L}_{\boldsymbol{\theta},\boldsymbol{\phi}}(\boldsymbol{x}) \tag{2.29}$$

Unbiased gradients of the ELBO w.r.t. the generative model parameters $\boldsymbol{\theta}$ are simple to obtain:

$$\begin{aligned} \nabla_{\boldsymbol{\theta}} \mathcal{L}_{\boldsymbol{\theta},\boldsymbol{\phi}}(\boldsymbol{x}) &= \nabla_{\boldsymbol{\theta}} \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z}) - \log q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})] \\ &= \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}[\nabla_{\boldsymbol{\theta}}(\log p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z}) - \log q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}))] \\ &\approx \nabla_{\boldsymbol{\theta}}(\log p_{\boldsymbol{\theta}}(\boldsymbol{x}, \tilde{\boldsymbol{z}}) - \log q_{\boldsymbol{\phi}}(\tilde{\boldsymbol{z}}|\boldsymbol{x})) \\ &= \nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(\boldsymbol{x}, \tilde{\boldsymbol{z}}) \end{aligned} \tag{2.30}$$

where $\tilde{\boldsymbol{z}} \sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$. Unbiased gradients w.r.t. the variational parameters $\boldsymbol{\phi}$ are more difficult to obtain, since the ELBO's expectation is taken w.r.t. the distribution $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$, which is a function of $\boldsymbol{\phi}$. In the case of continuous latent variables, a reparameterization trick can be used for computing unbiased estimates of $\nabla_{\boldsymbol{\theta},\boldsymbol{\phi}} \mathcal{L}_{\boldsymbol{\theta},\boldsymbol{\phi}}(\boldsymbol{x})$.

2.3.2 Reparameterization trick

The reparameterization trick rewrites a random variable as a deterministic function of a noise variable. This allows for the optimization of the non-stochastic terms through gradient descent. The

random variable $\tilde{\boldsymbol{z}} \sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$ can be written as:

$$\tilde{\boldsymbol{z}} = \boldsymbol{g}(\boldsymbol{\epsilon}, \boldsymbol{\phi}, \boldsymbol{x}) \tag{2.31}$$

where the distribution $p(\boldsymbol{\epsilon})$ of random variable is independent of $\boldsymbol{x}$ or $\boldsymbol{\phi}$, e.g., $\mathcal{N}(\mathbf{0}, \mathbf{I})$. The ELBO can be rewritten as:

$$\begin{aligned} \mathcal{L}_{\boldsymbol{\theta},\boldsymbol{\phi}}(\boldsymbol{x}) &= \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z}) - \log q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})] \\ &= \mathbb{E}_{p(\boldsymbol{\epsilon})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z}) - \log q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})], \qquad \boldsymbol{z} = \boldsymbol{g}(\boldsymbol{\epsilon}, \boldsymbol{\phi}, \boldsymbol{x}) \end{aligned} \tag{2.32}$$

As a result we can form a simple Monte Carlo estimator $\tilde{\mathcal{L}}_{\boldsymbol{\theta},\boldsymbol{\phi}}(\boldsymbol{x})$ of the individual-datapoint ELBO where we use a single noise sample from $p(\boldsymbol{\epsilon})$:

$$\begin{aligned} \boldsymbol{\epsilon} &\sim p(\boldsymbol{\epsilon}) \\ \boldsymbol{z} &= \boldsymbol{g}(\boldsymbol{\epsilon}, \boldsymbol{\phi}, \boldsymbol{x}) \\ \tilde{\mathcal{L}}_{\boldsymbol{\theta},\boldsymbol{\phi}}(\boldsymbol{x}) &= \log p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z}) - \log q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}) \end{aligned} \tag{2.33}$$

In addition, the gradient $\nabla_{\boldsymbol{\theta},\boldsymbol{\phi}} \tilde{\mathcal{L}}_{\boldsymbol{\theta},\boldsymbol{\phi}}(\boldsymbol{x}; \boldsymbol{\epsilon})$ is an unbiased estimator of the exact single-datapoint ELBO gradient when averaged over noise $\boldsymbol{\epsilon} \sim p(\boldsymbol{\epsilon})$:

$$\begin{aligned} \mathbb{E}_{p(\boldsymbol{\epsilon})}[\nabla_{\boldsymbol{\theta},\boldsymbol{\phi}} \tilde{\mathcal{L}}_{\boldsymbol{\theta},\boldsymbol{\phi}}(\boldsymbol{x}; \boldsymbol{\epsilon})] &= \mathbb{E}_{p(\boldsymbol{\epsilon})}[\nabla_{\boldsymbol{\theta},\boldsymbol{\phi}}(\log p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z}) - \log q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}))] \\ &= \nabla_{\boldsymbol{\theta},\boldsymbol{\phi}} \mathbb{E}_{p(\boldsymbol{\epsilon})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z}) - \log q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})] \\ &= \nabla_{\boldsymbol{\theta},\boldsymbol{\phi}} \mathcal{L}_{\boldsymbol{\theta},\boldsymbol{\phi}}(\boldsymbol{x}) \end{aligned} \tag{2.34}$$

2.3.3 Hierarchical VAE

A *Hierarchical Variational Autoencoder* (HVAE) [147] is a generalization of a VAE that extends to multiple hierarchies over latent variables. In the general HVAE formulation, each latent can condition on all latents from previous hierarchical levels. In the context of this study, all latent transitions down the hierarchy will be Markovian, i.e., decoding each latent $\boldsymbol{z}_t$ only conditions on previous latent $\boldsymbol{z}_{t+1}$. The joint distribution and posterior of an Markovian HVAE, with T hierarchical levels, are:

$$\begin{aligned} p(\boldsymbol{x}, \boldsymbol{z}_{1:T}) &= p(\boldsymbol{z}_T) p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}_1) \prod_{i=2}^{T} p_{\boldsymbol{\theta}}(\boldsymbol{z}_{t-1}|\boldsymbol{z}_t) \\ q_{\boldsymbol{\phi}}(\boldsymbol{z}_{1:T}|\boldsymbol{x}) &= q_{\boldsymbol{\phi}}(\boldsymbol{z}_1|\boldsymbol{x}) \prod_{t=2}^{T} q_{\boldsymbol{\phi}}(\boldsymbol{z}_t|\boldsymbol{z}_{t-1}) \end{aligned} \tag{2.35}$$

In this case, ELBO derivation becomes:

$$\begin{aligned} \log p(\boldsymbol{x}) &= \log \int p(\boldsymbol{x}, \boldsymbol{z}_{1:T}) \, \mathrm{d}\boldsymbol{z}_{1:T} \\ &= \log \int \frac{p(\boldsymbol{x}, \boldsymbol{z}_{1:T}) q_{\boldsymbol{\phi}}(\boldsymbol{z}_{1:T}|\boldsymbol{x})}{q_{\boldsymbol{\phi}}(\boldsymbol{z}_{1:T}|\boldsymbol{x})} \, \mathrm{d}\boldsymbol{z}_{1:T} \\ &= \log \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}_{1:T}|\boldsymbol{x})} \left[\frac{p(\boldsymbol{x}, \boldsymbol{z}_{1:T})}{q_{\boldsymbol{\phi}}(\boldsymbol{z}_{1:T}|\boldsymbol{x})} \right] \\ &\geq \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}_{1:T}|\boldsymbol{x})} \left[\log \frac{p(\boldsymbol{x}, \boldsymbol{z}_{1:T})}{q_{\boldsymbol{\phi}}(\boldsymbol{z}_{1:T}|\boldsymbol{x})} \right] \qquad \text{(Jensen's inequality)} \end{aligned} \tag{2.36}$$

2.4 Autoregressive Generative Models

In the setting of autoregressive generative models [11], we again assume that we are given access to a dataset with n-dimensional datapoints $\boldsymbol{x}$ that can be decomposed in terms of their components as $\boldsymbol{x} = x_1, x_2, \ldots, x_n$. For simplicity, we assume the datapoints are binary, i.e. $\boldsymbol{x} \in \{0, 1\}^n$. By the chain rule of probability, we can factorize the joint distribution over the n dimensions as follows:

$$p(\boldsymbol{x}) = p(x_1, x_2, \ldots, x_n) = \prod_{i=1}^{n} p(x_i | x_1, x_2, \ldots, x_{i-1}) = \prod_{i=1}^{n} p(x_i | \boldsymbol{x}_{<i}) \tag{2.37}$$

Unlike GANs and EBMs, it is possible to directly maximize the likelihood of the data by training a recurrent neural network to model $p(x_i|\boldsymbol{x}_{<i})$ and by minimizing the negative log-likelihood:

$$-\log p(\boldsymbol{x}) = -\sum_{i=1}^{n} \log p(x_i|x_1, x_2, \ldots, x_{i-1}) \tag{2.38}$$

The chain rule factorization can be expressed graphically as a Bayesian network. Such a Bayesian network that makes no conditional independence assumptions is said to obey the autoregressive property. The term autoregressive originates from the literature on time-series models where observations from the previous time-steps are used to predict the value at the current time step. If we allow for every conditional $p(x_i|\boldsymbol{x}_{<i})$ to be specified in a tabular form, then such a representation is fully general and can represent any possible distribution over n random variables. However, the space complexity for such a representation grows exponentially with n. This is because in order to fully specify this conditional, we need to specify a probability for 2^{n-1} configurations of the variables $x_1, x_2, \ldots, x_{n-1}$. Since the probabilities should sum to 1, the total number of parameters for specifying this conditional is given by $2^{n-1} - 1$. Hence, a tabular representation for the conditionals is impractical for learning the joint distribution factorized via chain rule.

In an autoregressive generative model, the conditionals are specified as parameterized functions with a fixed number of parameters. That is, we assume the conditional distributions $p(x_i|\boldsymbol{x}_{<i})$ to correspond to a Bernoulli random variable and learn a function that maps the preceding random variables $x_1, x_2, \ldots, x_{i-1}$ to the mean of this distribution. In the simplest case, we can specify the function as a linear combination of the input elements followed by a sigmoid non-linearity (to restrict the output to lie between 0 and 1). This gives us the formulation of a fully-visible sigmoid belief network (FVSBN) [50]:

$$p(x_i = 1|\boldsymbol{x}_{<i}) = \sigma(a_0^{(i)} + a_1^{(i)} x_1 + \cdots + a_{i-1}^{(i)} x_{i-1}) \tag{2.39}$$

where $\{a_k^{(i)}\}_{k=0}^{i-1}$ denote the parameters of the mean function. One approach to build more expressive autoregressive models is to mask the weights of simple multilayer perceptron (MLP) autoencoders so as to satisfy the autoregressive property. The neural autoregressive density estimator (NADE) [90] which can be viewed as a mean-field approximation of a restricted Boltzmann machine, achieves this for binary data by placing time-dependent masks on an MLP with one hidden layer. Specifically, at time step i, weights are masked so that the entire hidden state h_i and output $p(x_i|\boldsymbol{x}_{<i})$ are dependent only on $\boldsymbol{x}_{<i}$; formally this can be defined as follows:

$$\boldsymbol{h}_i = \sigma(\mathbf{W}_{\cdot,<i}\boldsymbol{x}_{<i} + \boldsymbol{c}), \quad p(x_i = 1|\boldsymbol{x}_{<i}) = \sigma(\mathbf{W}_{i,\cdot}^{\top}\boldsymbol{h}_i + b_i) \tag{2.40}$$

where $\mathbf{W}_{\cdot,<i} \in \mathbb{R}^{d\times(i-1)}$ denotes the first $i-1$ columns of the shared weight matrix $\mathbf{W}$, $\boldsymbol{c} \in \mathbb{R}^d$, $\boldsymbol{b} \in \mathbb{R}^n$ and $\boldsymbol{h}_i \in \mathbb{R}^d$, $1 \leq i \leq n$. RNADE [165] generalises NADE to real valued data by instead modeling $p(x_i|\boldsymbol{x}_{<i})$ with mixture distributions parameterised by the network. An alternative masking procedure known as MADE [54] allows for parallel density estimation by placing a mask fixed over time on an MLP so that no connections exist between $p(x_i|\boldsymbol{x}_{<i})$ and $\boldsymbol{x}_{\geq i}$.

Apart from RNNs and their variants such as LSTMs and GRUs which constitute a natural architecture for implementing autoregressive models, CNNs also provide an alternative approach in the form of causal convolutions. PixelCNN [166], PixelCNN++ [134] and PixelSNAIL [24] are notable realizations of convolutional generative autoregressive models.

While autoregressive models are extremely powerful density estimators, sampling is inherently a sequential process and can be exceedingly slow on high dimensional data. Additionally, data must be decomposed into a fixed ordering; while the choice of ordering can be clear for some modalities (e.g. text and audio), it is not obvious for others such as images and can affect performance depending on the network architecture used.

2.5 Normalizing Flows

While training autoregressive models through maximum likelihood offers plenty of benefits including stable training, density estimation, and a useful validation metric, they still suffer from slow

sampling speed and poor scaling properties. *Normalizing Flows* [85, 113] are a technique that also allows exact likelihood calculation while being efficiently parallelizable as well as offering a useful latent space for downstream tasks.

Let $\boldsymbol{z} \in \mathbb{R}^D$ be a random variable with a known and tractable probability density function $p_z(\boldsymbol{z})$. Let $\boldsymbol{g}$ be an invertible function and $\boldsymbol{x} = \boldsymbol{g}(\boldsymbol{z})$. Then using the change of variables formula, the probability density function of the random variable $\boldsymbol{x}$ can be computed:

$$\begin{aligned} p_x(\boldsymbol{x}) &= p_z(\boldsymbol{f}(\boldsymbol{x}))|\det(\mathbf{Df}(\boldsymbol{x}))| \\ &= p_z(\boldsymbol{f}(\boldsymbol{x}))|\det(\mathbf{Dg}(\boldsymbol{f}(\boldsymbol{x})))|^{-1} \end{aligned} \tag{2.41}$$

where $\boldsymbol{f}$ is the inverse of $\boldsymbol{g}$, $\mathbf{Df}(\boldsymbol{x}) = \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}$ denotes the Jacobian of $\boldsymbol{f}$, while $\mathbf{Dg}(\boldsymbol{z}) = \frac{\partial \boldsymbol{g}}{\partial \boldsymbol{z}}$ is the Jacobian of $\boldsymbol{g}$. The new density function $p_x(\boldsymbol{x})$ is called a *pushforward* of the density p_z by the function $\boldsymbol{g}$ and denoted by $\boldsymbol{g}_\star p_z$. In order to generate a data point $\boldsymbol{x}$, one can sample $\boldsymbol{z}$ from the base distribution and then apply the generator $\boldsymbol{x} = \boldsymbol{g}(\boldsymbol{z})$. The inverse function $\boldsymbol{f}$ moves in the so called *normalizing direction*, transforming a complicated and irregular data distribution into a simpler, more regular form, denoted by p_z. In practice, p_z is chosen as a standard normal distribution.

Intuitively, if the transformation $\boldsymbol{g}$ can be arbitrarily complex, one can generate any distribution p_x from any base distribution p_z under reasonable assumptions on the two distributions. Constructing arbitrarily complicated non-linear invertible functions (bijections) can be difficult. By the term *Normalizing Flows* people mean bijections which are convenient to compute, invert, and calculate the determinant of their Jacobian. One approach to this is to note that the composition of invertible functions is itself invertible and the determinant of its Jacobian has a specific form. Let, $\boldsymbol{g}_1, \boldsymbol{g}_2, \ldots, \boldsymbol{g}_N$ be a set of N bijective function and define $\boldsymbol{g} = \boldsymbol{g}_N \circ \boldsymbol{g}_{N-1} \circ \cdots \circ \boldsymbol{g}_1$ to be the composition of those functions. It can be shown that $\boldsymbol{g}$ is bijective with the corresponding inverse being $\boldsymbol{f} = \boldsymbol{f} \circ \cdots \circ \boldsymbol{f}_{N-1} \circ \boldsymbol{f}_N$. The determinant of the Jacobian of $\boldsymbol{f}$ becomes:

$$\det(\mathbf{Df}) = \det\left(\prod_{i=1}^{N} \mathbf{Df}_i\right) = \prod_{i=1}^{N} \det(\mathbf{Df}_i) \tag{2.42}$$

Thus, a set of nonlinear bijective functions can be composed to construct successively more complicated functions.

2.5.1 Density Estimation and Sampling

The natural and most obvious use of normalizing flows is to perform density estimation. Without loss of generality, let $\boldsymbol{g}_{\boldsymbol{\theta}}$ be a single flow parameterized by $\boldsymbol{\theta}$, while the base density p_z is parameterized by $\boldsymbol{\phi}$. Given a set of data observed from a data distribution $\mathcal{D} = \{\boldsymbol{x}_i\}_{i=1}^M$, then maximum likelihood estimation of the parameters $(\boldsymbol{\theta}, \boldsymbol{\phi})$ can be performed:

$$\max_{\boldsymbol{\theta}, \boldsymbol{\phi}} \sum_{i=1}^{M} \log p_z(\boldsymbol{f}(\boldsymbol{x}_i; \boldsymbol{\theta}); \boldsymbol{\phi}) + \log|\det(\mathbf{Df})(\boldsymbol{x}_i; \boldsymbol{\theta})| \tag{2.43}$$

where the first term is the log-likelihood of the samples under the base density and the second term, sometimes called the log-determinant or volume correction, accounts for the change of volume induced by the transformation of the normalizing flows. Evaluating the likelihood of a distribution modelled by a normalizing flow requires computing $\boldsymbol{f}$ and its log-determinant. Therefore, the efficiency of these operations is particularly important during training.

Sampling from the data distribution, requires sampling from the base density p_z and then evaluating the inverse $\boldsymbol{g}$, i.e., the generative direction. In that way, sampling performance is determined by the cost of the generative direction. Even though a flow must be theoretically invertible, computation of the inverse may be difficult in practice.

2.5.2 Normalizing Flow Layers

Normalizing Flows should satisfy several conditions in order to be practical: (i) They should be invertible; $\boldsymbol{g}$ is needed for sampling, while $\boldsymbol{f}$ is needed for density estimation. (ii) They need to be

sufficiently expressive to model the distribution of interest. (iii) Lastly, they need to be computationally efficient, both in terms of computing $\boldsymbol{f}$ and $\boldsymbol{g}$ but also in terms of the calculation of the Jacobian. In the remainder of current section, we will briefly present different types of flows and comment on their properties.

Planar Flows

Planar flows [124] expand and contract the distribution along certain specific directions and take the form:

$$\boldsymbol{g}(\boldsymbol{z}) = \boldsymbol{z} + \boldsymbol{u}h(\boldsymbol{w}^\top \boldsymbol{z} + b) \tag{2.44}$$

where $\boldsymbol{u}, \boldsymbol{w} \in \mathbb{R}^D$ and $b \in \mathbb{R}$, while $h : \mathbb{R} \to \mathbb{R}$ is a smooth non-linearity. The determinant of the Jacobian for this transformation is:

$$\det\left(\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{z}}\right) = \det\left(\mathbf{I}_D + \boldsymbol{u}h'(\boldsymbol{w}^\top \boldsymbol{z} + b)\boldsymbol{w}^\top\right) = 1 + h'(\boldsymbol{w}^\top \boldsymbol{z} + b)\boldsymbol{u}^\top \boldsymbol{w} \tag{2.45}$$

which can be computed in $\mathcal{O}(D)$. There is no closed form for the inverse of this flow, which may not even exist for certain choices of $h(\cdot)$ and parameter settings. Planar flows can be interpreted as a multilayer perceptron with a bottleneck hidden layer with a single unit. Higher expressivity can be achieved by stacking multiple planar flow layers. *Sylvester flows* remove the well-known single-unit bottleneck from planar flows, making the latter much more flexible. They are defined as follows:

$$\boldsymbol{g}(\boldsymbol{z}) = \boldsymbol{z} + \mathbf{U}\boldsymbol{h}(\mathbf{W}^\top \boldsymbol{z} + \boldsymbol{b}) \tag{2.46}$$

where $\mathbf{U}, \mathbf{W} \in \mathbb{R}^{D \times M}$, $\boldsymbol{b} \in \mathbb{R}^M$ and $\boldsymbol{h} : \mathbb{R}^M \to \mathbb{R}^M$, with $M \leq D$. In this case, the determinant of the Jacobian becomes:

$$\begin{aligned} \det\left(\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{z}}\right) &= \det\left(\mathbf{I}_D + \mathbf{U}\mathrm{diag}(\boldsymbol{h}(\mathbf{W}^\top \boldsymbol{z} + \boldsymbol{b}))\mathbf{W}^\top\right) \\ &= \det\left(\mathbf{I}_M + \mathrm{diag}(\boldsymbol{h}(\mathbf{W}^\top \boldsymbol{z} + \boldsymbol{b}))\mathbf{W}^\top \mathbf{U}\right) \end{aligned} \tag{2.47}$$

where the last equality is due to *Sylvester's determinant identity*, hence the name.

Coupling Flows

Let $(\boldsymbol{z}_A, \boldsymbol{z}_B) \in \mathbb{R}^d \times \mathbb{R}^{D-d}$ be a dijoint partition of the input $\boldsymbol{z} \in \mathbb{R}^D$ and $\boldsymbol{h}(\cdot, \boldsymbol{\theta}) : \mathbb{R}^d \to \mathbb{R}^d$ be a bijection parameterized by $\boldsymbol{\theta}$. Then one can define a function $\boldsymbol{g} : \mathbb{R}^D \times \mathbb{R}^D$ by the formula:

$$\begin{aligned} \boldsymbol{x}_A &= \boldsymbol{h}(\boldsymbol{z}_A; \Theta(\boldsymbol{z}_B)) \\ \boldsymbol{x}_B &= \boldsymbol{z}_B \end{aligned} \tag{2.48}$$

where the parameters $\boldsymbol{\theta}$ are defined by any arbitrary function $\Theta(\boldsymbol{x}_B)$ which only uses $\boldsymbol{x}_B$ as input. This function is called a *conditioner*. The bijection $\boldsymbol{h}$ is called a *coupling function* and the resulting function $\boldsymbol{g}$ is called a *coupling flow*. A coupling flow is invertible if and only if $\boldsymbol{h}$ is invertible and has inverse:

$$\begin{aligned} \boldsymbol{z}_A &= \boldsymbol{h}^{-1}(\boldsymbol{x}_A; \Theta(\boldsymbol{x}_B)) \\ \boldsymbol{z}_B &= \boldsymbol{x}_B \end{aligned} \tag{2.49}$$

The Jacobian of $\boldsymbol{g}$ is a block triangular matrix where the diagonal blocks are $\mathbf{Dh}$ and the identity matrix, respectively. Hence the determinant of the Jacobian of the coupling flow is simply the determinant of $\mathbf{Dh}$. The power of a coupling flow resides in the ability of a conditioner $\Theta(\boldsymbol{x}_B)$ to be arbitrarily complex. The initial partition can be done by splitting the dimensions in half, potentially after a random permutation [35]. More sophisticated methods have also been used, such as "masked" flows [36] and 1×1 convolutions [83].

Autoregressive Flows Autoregressive models can also be used as a form of normalizing flow. Let $h(\cdot, \boldsymbol{\theta}) : \mathbb{R} \to \mathbb{R}$ be a bijection parameterized by $\boldsymbol{\theta}$. Then an autoregressive model is a function $\boldsymbol{g} : \mathbb{R}^D \to \mathbb{R}^D$, which outputs each entry of $\boldsymbol{x} = \boldsymbol{g}(\boldsymbol{z})$ conditioned on the previous entries of the input:

$$x_t = h(z_t; \Theta_t(\boldsymbol{z}_{1:t-1})) \tag{2.50}$$

The Jacobian matrix of the autoregressive transformation $\boldsymbol{g}$ is triangular. Each output x_t only depends on $\boldsymbol{z}_{1:t}$, and so the determinant is just a product of its diagonal entries, i.e., $\det(\mathbf{Dg}) = \prod_{t=1}^{D} \frac{\partial x_t}{\partial z_t}$. Given the inverse of h, the inverse of $\boldsymbol{g}$ can be found with recursion as $z_t = h^{-1}(x_t; \Theta_t(\boldsymbol{z}_{1:t-1}))$. Contrary to the above formulation, *Inverse Autoregressive Flow* (IAF) [84] outputs each entry of $\boldsymbol{x}$ conditioned on its own previous entries:

$$x_t = h(z_t; \Theta_t(\boldsymbol{x}_{1:t-1})) \tag{2.51}$$

One can see that the functional form of the IAF is the same as the form of the inverse of the regular autoregressive flow. Therefore, computation of the IAF is sequential and expensive, but the inverse of IAF can be computed relatively efficiently.

Residual Flows

Residual networks [62] are compositions of the form:

$$\boldsymbol{g}(\boldsymbol{z}) = \boldsymbol{z} + \boldsymbol{h}(\boldsymbol{z}) \tag{2.52}$$

where the *residual block* $\boldsymbol{h}(\cdot)$ can be any feed-forward neural network. According to [10], the residual connection described in Eq. (2.52) is invertible if $\|\boldsymbol{h}\|_L < 1$. There is no analytically closed form for the inverse, but it can be found numerically using fixed-point iterations. The specific architecture proposed in [10], called iResNet, models the residual block as a CNN and constrains the spectral radius of each convolutional layer to be less than one. Given that $\|\boldsymbol{h}\|_L < 1$, the following hold:

$$\mathbf{Dg} = \mathbf{I} + \mathbf{Dh}, \qquad |\det(\mathbf{I} + \mathbf{Dh})| = \det(\mathbf{I} + \mathbf{Dh}) \tag{2.53}$$

Using the identity $\log\det(\mathbf{A}) = \mathrm{tr}(\log \mathbf{A})$, we get:

$$\log\det(\mathbf{Dg}) = \log\det(\mathbf{I} + \mathbf{Dh}) = \mathrm{tr}(\log(\mathbf{I} + \mathbf{Dh})) = \sum_{k=1}^{\infty} (-1)^{k+1} \frac{\mathrm{tr}(\mathbf{Dh})^{\mathrm{k}}}{k} \tag{2.54}$$

To efficiently compute each member of the truncated series, the Hutchinson trace estimator [69] was used. Instead of the latter, in [22], in a model called *Residual Flow*, the authors used the *Russian roulette* estimator.

2.6 Unified View

Putting aside EBMs and autoregressive generative models, the four main contenders in the field of generative modeling are: GANs, VAEs, normalizing flows and the most recent one, Denoising Diffusion models. In general, GANs are known to generate high-quality samples rapidly, but they have poor mode coverage. On the contrary, VAEs and normalizing flows cover data modes faithfully, but they often suffer from low sample quality. Recently, diffusion models have emerged as powerful generative models. They demonstrate surprisingly good results in sample quality, beating GANs [33] in image generation. They also obtain good mode coverage, indicated by high likelihood [79, 150] but sampling from them often requires thousands of network evaluations, making their application expensive in practice. The latter will constitute the main subject of the current thesis. Fig. 2.1 illustrates the trilemma discussed above. Moreover, Fig. 2.2 provides an illustrative overview of GANs, VAEs, normalizing flows, as well as diffusion models, which will be covered in more detail in the upcoming chapter.

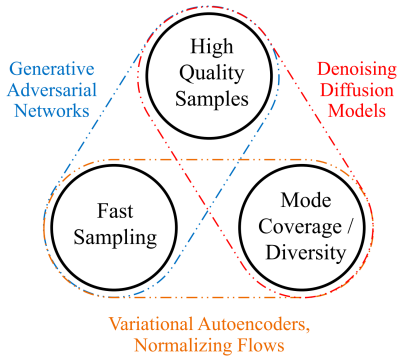


Figure 2.1: Generative learning trilemma. Source: [176]

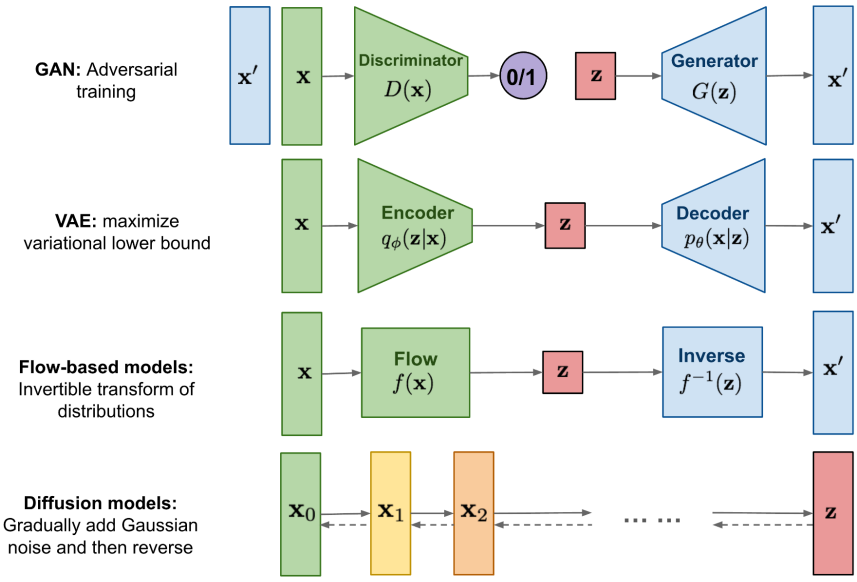


Figure 2.2: Overview of GANs, VAEs, Flow-based and diffusion-based models. Source: https://lilianweng.github.io/posts/2021-07-11-diffusion-models/

Chapter 3

Diffusion Models

In the current chapter we will try to establish the theoretical foundations regarding Diffusion Models (DMs) in the context of generative modeling, as well as accurately describe the basic architectures which will later constitute the building blocks of our own computational models. Diffusion models are generative models that produce samples by inverting a corruption process. The corruption level is typically indexed by a time t, with $t = 0$ corresponding to clean and $t = 1$ to fully corrupted images (continuous case). The diffusion process can be discrete or continuous. The two general classes of diffusion models are *Score-Based Models* [149, 150, 153] and *Denoising Diffusion Probabilistic Models* (DDPM) [65, 146]. In context of this study, we will mostly focus on the latter. For guidance throughout this process, we also consult [28, 101].

3.1 Denoising Diffusion Probabilistic Models

DDPM can be thought as variants of a Markovian HVAE that abide by the following three restrictions:

- The dimensionality of the latents matches that of the data.
- The distribution of the latent encoder is predefined as a linear Gaussian model centered around the output of the previous timestep.
- The parameters of the latent encoders vary over time in such a way that the distribution of the latent at the final timestep T is a standard Gaussian.

True data samples and latents can be jointly represented using the $\mathbf{x}_t$ notation, where $t = 0$ corresponds to true data samples and $t \in [1, T]$ corresponds to latents with hierarchy indexed by t. The Markov property between subsequent latent transitions implies that the DDPM posterior (*forward/noising* process) can be written as follows:

$$q(\mathbf{x}_{1:T}|\mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1}) \tag{3.1}$$

Encoder transitions are of the form:

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{a_t}\mathbf{x}_{t-1}, (1 - a_t)\mathbf{I}) \tag{3.2}$$

The noise schedule a_t is either fixed [65] or learnable [109] and evolves over time so that the distribution of the final latent is standard normal, i.e., $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. The joint distribution $p_{\boldsymbol{\theta}}(\mathbf{x}_{0:T})$ is called the *reverse* process and is defined as a Markov chain with learnable mean and variance parameters:

$$p_{\boldsymbol{\theta}}(\mathbf{x}_{0:T}) = p_{\boldsymbol{\theta}}(\mathbf{x}_T) \prod_{t=1}^{T} p_{\boldsymbol{\theta}}(\mathbf{x}_{t-1}|\mathbf{x}_t), \quad p_{\boldsymbol{\theta}}(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{x}_t, t), \boldsymbol{\Sigma}(\mathbf{x}_t, t)) \tag{3.3}$$

As the linear Gaussian models in the *forward* process are fixed, the conditionals $p_{\boldsymbol{\theta}}(\mathbf{x}_{t-1}|\mathbf{x}_t)$ constitute the sole learnable parameters in the DDPM setting.

Encoder transitions can be rewritten as $q(\mathbf{x}_t|\mathbf{x}_{t-1}) = q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0)$, where the extra conditioning term $\mathbf{x}_0$ is rendered redundant due to the Markov property. Using the Bayes rule, each transition can also be rewritten as follows:

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0) = \frac{q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)q(\mathbf{x}_t|\mathbf{x}_0)}{q(\mathbf{x}_{t-1}|\mathbf{x}_0)} \tag{3.4}$$

Similarly to what we've discussed in section 2.3, DDPM can be optimized by maximizing the following ELBO:

$$\begin{aligned}
\log p(\mathbf{x}) &= \log \int p(\mathbf{x}_{0:T})\, \mathrm{d}\mathbf{x}_{1:T} && (3.5)\\
&= \log \int \frac{p(\mathbf{x}_{0:T})q(\mathbf{x}_{1:T}|\mathbf{x}_0)}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\, \mathrm{d}\mathbf{x}_{1:T} && (3.6)\\
&= \log \mathbb{E}_{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\left[\frac{p(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\right] && (3.7)\\
&\geq \mathbb{E}_{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\left[\log \frac{p(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\right] && (3.8)\\
&= \mathbb{E}_{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\left[\log \frac{p(\mathbf{x}_T)\prod_{t=1}^{T} p_{\boldsymbol{\theta}}(\mathbf{x}_{t-1}|\mathbf{x}_t)}{\prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1})}\right] && (3.9)\\
&= \mathbb{E}_{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\left[\log \frac{p(\mathbf{x}_T)p_{\boldsymbol{\theta}}(\mathbf{x}_0|\mathbf{x}_1)\prod_{t=2}^{T} p_{\boldsymbol{\theta}}(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_1|\mathbf{x}_0)\prod_{t=2}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1})}\right] && (3.10)\\
&= \mathbb{E}_{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\left[\log \frac{p(\mathbf{x}_T)p_{\boldsymbol{\theta}}(\mathbf{x}_0|\mathbf{x}_1)\prod_{t=2}^{T} p_{\boldsymbol{\theta}}(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_1|\mathbf{x}_0)\prod_{t=2}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0)}\right] && (3.11)\\
&= \mathbb{E}_{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\left[\log \frac{p(\mathbf{x}_T)p_{\boldsymbol{\theta}}(\mathbf{x}_0|\mathbf{x}_1)}{q(\mathbf{x}_1|\mathbf{x}_0)} + \log \prod_{t=2}^{T} \frac{p_{\boldsymbol{\theta}}(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0)}\right] && (3.12)\\
&= \mathbb{E}_{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\left[\log \frac{p(\mathbf{x}_T)p_{\boldsymbol{\theta}}(\mathbf{x}_0|\mathbf{x}_1)}{q(\mathbf{x}_1|\mathbf{x}_0)} + \log \prod_{t=2}^{T} \frac{p_{\boldsymbol{\theta}}(\mathbf{x}_{t-1}|\mathbf{x}_t)}{\frac{q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)q(\mathbf{x}_t|\mathbf{x}_0)}{q(\mathbf{x}_{t-1}|\mathbf{x}_0)}}\right] && (3.13)\\
&= \mathbb{E}_{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\left[\log \frac{p(\mathbf{x}_T)p_{\boldsymbol{\theta}}(\mathbf{x}_0|\mathbf{x}_1)}{q(\mathbf{x}_1|\mathbf{x}_0)} + \log \frac{q(\mathbf{x}_1|\mathbf{x}_0)}{q(\mathbf{x}_T|\mathbf{x}_0)} + \log \prod_{t=2}^{T} \frac{p_{\boldsymbol{\theta}}(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)}\right] && (3.14)\\
&= \mathbb{E}_{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\left[\log \frac{p(\mathbf{x}_T)p_{\boldsymbol{\theta}}(\mathbf{x}_0|\mathbf{x}_1)}{q(\mathbf{x}_T|\mathbf{x}_0)} + \sum_{t=2}^{T} \frac{p_{\boldsymbol{\theta}}(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)}\right] && (3.15)\\
&= \mathbb{E}_{q(\mathbf{x}_1|\mathbf{x}_0)}[\log p_{\boldsymbol{\theta}}(\mathbf{x}_0|\mathbf{x}_1)] + \mathbb{E}_{q(\mathbf{x}_T|\mathbf{x}_0)}\left[\log \frac{p(\mathbf{x}_T)}{q(\mathbf{x}_T|\mathbf{x}_0)}\right] && (3.16)\\
&\qquad + \sum_{t=2}^{T} \mathbb{E}_{q(\mathbf{x}_{t-1}, \mathbf{x}_t|\mathbf{x}_0)}\left[\log \frac{p_{\boldsymbol{\theta}}(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)}\right] &&\\
&= \mathbb{E}_{q(\mathbf{x}_1|\mathbf{x}_0)}[\log p_{\boldsymbol{\theta}}(\mathbf{x}_0|\mathbf{x}_1)] - D_{\mathrm{KL}}(q(\mathbf{x}_T|\mathbf{x}_0) \parallel p(\mathbf{x}_T)) && (3.17)\\
&\qquad - \sum_{t=2}^{T} \mathbb{E}_{q(\mathbf{x}_T|\mathbf{x}_0)}[D_{\mathrm{KL}}(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) \parallel p_{\boldsymbol{\theta}}(\mathbf{x}_{t-1}|\mathbf{x}_t))] &&
\end{aligned}$$

This formulation also has an elegant interpretation, which is revealed when inspecting each individual term:

- $\mathbb{E}_{q(\mathbf{x}_1|\mathbf{x}_0)}[\log p_{\boldsymbol{\theta}}(\mathbf{x}_0|\mathbf{x}_1)]$ can be interpreted as a *reconstruction* loss that can be approximated and optimized using a Monte Carlo sampling.
- $D_{\mathrm{KL}}(q(\mathbf{x}_T|\mathbf{x}_0) \parallel p(\mathbf{x}_T))$ measures the distance between the final noisy version of the initial input and the standard normal prior. It has no trainable parameters and is ideally equal to zero.

- $\mathbb{E}_{q(\mathbf{x}_T|\mathbf{x}_0)}[D_{\mathrm{KL}}(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) \parallel p_{\boldsymbol{\theta}}(\mathbf{x}_{t-1}|\mathbf{x}_t))]$ can be interpreted as a *denoising matching* term as it measures the distance between the learnable denoising transition step $p_{\boldsymbol{\theta}}(\mathbf{x}_{t-1}|\mathbf{x})$ and the ground-truth denoising transition step $q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)$. The term $q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)$ acts as a ground-truth signal, since it defines how to denoise a noisy image $\mathbf{x}_t$, given its initial, noise free version $\mathbf{x}_0$.

By making further use of the Bayes rule, we get:

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = \frac{q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0)q(\mathbf{x}_{t-1}|\mathbf{x}_0)}{q(\mathbf{x}_t|\mathbf{x}_0)} \tag{3.18}$$

The Markov property implies that $q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0) = q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{a_t}\mathbf{x}_{t-1}, (1-a_t)\mathbf{I})$. Next, we need to derive a formula for calculating the $q(\mathbf{x}_{t-1}|\mathbf{x}_0), q(\mathbf{x}_t|\mathbf{x}_0)$ terms, based on the fact that DDPM transitions are linear Gaussian models. Using the reparameterization trick described in section 2.3.2, samples $\mathbf{x}_t \sim q(\mathbf{x}_t|\mathbf{x}_{t-1})$ can be rewritten as:

$$\mathbf{x}_t = \sqrt{a_t}\mathbf{x}_{t-1} + \sqrt{1-a_t}\boldsymbol{\epsilon}, \qquad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \tag{3.19}$$

We repeatedly apply the reparameterization trick for all transitions $q(\mathbf{x}_{t-1}|\mathbf{x}_{t-2}), \ldots, q(\mathbf{x}_1|\mathbf{x}_0)$, given access to noise variables $\{\boldsymbol{\epsilon}_t, \boldsymbol{\epsilon}_t^*\}_{t=0}^T \overset{\text{i.i.d.}}{\sim} \mathcal{N}(\mathbf{0}, \mathbf{I})$, and derive the following recursive formula:

$$\begin{aligned}
\mathbf{x}_t &= \sqrt{a_t}\mathbf{x}_{t-1} + \sqrt{1-a_t}\boldsymbol{\epsilon}_t^* && (3.20)\\
&= \sqrt{a_t}(\sqrt{a_{t-1}}\mathbf{x}_{t-2} + \sqrt{1-a_{t-1}}\boldsymbol{\epsilon}_{t-1}) + \sqrt{1-a_t}\boldsymbol{\epsilon}_t^* && (3.21)\\
&= \sqrt{a_t a_{t-1}}\mathbf{x}_{t-2} + \sqrt{a_t(1-a_{t-1})}\boldsymbol{\epsilon}_{t-1} + \sqrt{1-a_t}\boldsymbol{\epsilon}_t^* && (3.22)\\
&= \sqrt{a_t a_{t-1}}\mathbf{x}_{t-2} + \sqrt{\left(\sqrt{a_t(1-a_{t-1})}\right)^2 + \left(\sqrt{1-a_t}\right)^2}\boldsymbol{\epsilon}_{t-2}^* && (3.23)\\
&= \sqrt{a_t a_{t-1}}\mathbf{x}_{t-2} + \sqrt{1-a_t a_{t-1}}\boldsymbol{\epsilon}_{t-2}^* && (3.24)\\
&= \sqrt{a_t a_{t-1}}(\sqrt{a_{t-2}}\mathbf{x}_{t-3} + \sqrt{1-a_{t-2}}\boldsymbol{\epsilon}_{t-2}) + \sqrt{1-a_t a_{t-1}}\boldsymbol{\epsilon}_{t-2}^* && (3.25)\\
&= \sqrt{a_t a_{t-1} a_{t-2}}\mathbf{x}_{t-3} + \sqrt{a_t a_{t-1}(1-a_{t-2})}\boldsymbol{\epsilon}_{t-2} + \sqrt{1-a_t a_{t-1}}\boldsymbol{\epsilon}_{t-2}^* && (3.26)\\
&= \sqrt{a_t a_{t-1} a_{t-2}}\mathbf{x}_{t-3} + \sqrt{\left(\sqrt{a_t a_{t-1}(1-a_{t-2})}\right)^2 + \left(\sqrt{1-a_t a_{t-1}}\right)^2}\boldsymbol{\epsilon}_{t-3}^* && (3.27)\\
&= \sqrt{a_t a_{t-1} a_{t-2}}\mathbf{x}_{t-3} + \sqrt{1-a_t a_{t-1} a_{t-2}}\boldsymbol{\epsilon}_{t-3}^* && (3.28)\\
&= \ldots && (3.29)\\
&= \sqrt{\prod_{t=1}^{T} a_t}\mathbf{x}_0 + \sqrt{1-\prod_{t=1}^{T} a_t}\boldsymbol{\epsilon}_0^* && (3.30)\\
&= \sqrt{\bar{a}_t}\mathbf{x}_0 + \sqrt{1-\bar{a}_t}\boldsymbol{\epsilon}_0^* \sim \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{a}_t}\mathbf{x}_0, (1-\bar{a}_t)\mathbf{I}) && (3.31)
\end{aligned}$$

Eqs. (3.23) and (3.27) are derived based in the fact that the sum of independent normal variables is normally distributed with a mean equal to the sum of their means and variance equal to the sum of their variances.

Having specified all the distributions in the right hand side of Eq. (3.18), we can now proceed

with calculating the form of $q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)$:

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = \frac{q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0)q(\mathbf{x}_{t-1}|\mathbf{x}_0)}{q(\mathbf{x}_t|\mathbf{x}_0)} \tag{3.32}$$

$$= \frac{\mathcal{N}(\mathbf{x}_t; \sqrt{a_t}\mathbf{x}_{t-1}, \sqrt{1-a_t}\mathbf{I})\mathcal{N}(\mathbf{x}_{t-1}; \sqrt{\bar{a}_{t-1}}\mathbf{x}_0, \sqrt{1-\bar{a}_{t-1}}\mathbf{I})}{\mathcal{N}(\mathbf{x}_t; \sqrt{\bar{a}_t}\mathbf{x}_0, \sqrt{1-\bar{a}_t}\mathbf{I})} \tag{3.33}$$

$$\propto \exp\left\{ -\frac{1}{2}\left[\frac{(\mathbf{x}_t - \sqrt{a_t}\mathbf{x}_{t-1})^2}{1-a_t} + \frac{(\mathbf{x}_{t-1} - \sqrt{\bar{a}_{t-1}}\mathbf{x}_0)^2}{1-\bar{a}_{t-1}} - \frac{(\mathbf{x}_t - \sqrt{\bar{a}_t}\mathbf{x}_0)^2}{1-\bar{a}_t} \right] \right\} \tag{3.34}$$

$$= \exp\left\{ -\frac{1}{2}\left[\frac{(-2\sqrt{a_t}\mathbf{x}_{t-1}\mathbf{x}_t + a_t\mathbf{x}_{t-1}^2)}{1-a_t} + \frac{(\mathbf{x}_{t-1}^2 - 2\sqrt{\bar{a}_{t-1}}\mathbf{x}_0\mathbf{x}_{t-1})}{1-\bar{a}_{t-1}} + C(\mathbf{x}_t, \mathbf{x}_0) \right] \right\} \tag{3.35}$$

$$\propto \exp\left\{ -\frac{1}{2}\left[-\frac{2\sqrt{a_t}\mathbf{x}_{t-1}\mathbf{x}_t}{1-a_t} + \frac{a_t\mathbf{x}_{t-1}^2}{1-a_t} + \frac{\mathbf{x}_{t-1}^2}{1-\bar{a}_{t-1}} - \frac{2\sqrt{\bar{a}_{t-1}}\mathbf{x}_0\mathbf{x}_{t-1}}{1-\bar{a}_{t-1}} \right] \right\} \tag{3.36}$$

$$= \exp\left\{ -\frac{1}{2}\left[\left(\frac{a_t}{1-a_t} + \frac{1}{1-\bar{a}_{t-1}} \right)\mathbf{x}_{t-1}^2 - 2\left(\frac{\sqrt{a_t}\mathbf{x}_t}{1-a_t} + \frac{\sqrt{\bar{a}_{t-1}}\mathbf{x}_0}{1-\bar{a}_{t-1}} \right)\mathbf{x}_{t-1} \right] \right\} \tag{3.37}$$

$$= \exp\left\{ -\frac{1}{2}\left[\frac{1-\bar{a}_t}{(1-a_t)(1-\bar{a}_{t-1})}\mathbf{x}_{t-1}^2 - 2\left(\frac{\sqrt{a_t}\mathbf{x}_t}{1-a_t} + \frac{\sqrt{\bar{a}_{t-1}}\mathbf{x}_0}{1-\bar{a}_{t-1}} \right)\mathbf{x}_{t-1} \right] \right\} \tag{3.38}$$

$$= \exp\left\{ -\frac{1}{2}\left(\frac{1-\bar{a}_t}{(1-a_t)(1-\bar{a}_{t-1})} \right)\left[\mathbf{x}_{t-1}^2 - 2\frac{\left(\frac{\sqrt{a_t}\mathbf{x}_t}{1-a_t} + \frac{\sqrt{\bar{a}_{t-1}}\mathbf{x}_0}{1-\bar{a}_{t-1}} \right)(1-a_t)(1-\bar{a}_{t-1})}{1-\bar{a}_t}\mathbf{x}_{t-1} \right] \right\} \tag{3.39}$$

$$= \exp\left\{ -\frac{1}{2}\left(\frac{1}{\frac{(1-a_t)(1-\bar{a}_{t-1})}{(1-\bar{a}_t)}} \right)\left[\mathbf{x}_{t-1}^2 - 2\frac{\sqrt{a_t}\mathbf{x}_t(1-\bar{a}_{t-1}) + \sqrt{\bar{a}_{t-1}}\mathbf{x}_0(1-a_t)}{1-\bar{a}_t}\mathbf{x}_{t-1} \right] \right\} \tag{3.40}$$

$$\propto \mathcal{N}\left(\mathbf{x}_{t-1}; \frac{\sqrt{a_t}\mathbf{x}_t(1-\bar{a}_{t-1}) + \sqrt{\bar{a}_{t-1}}\mathbf{x}_0(1-a_t)}{1-\bar{a}_t}, \frac{(1-a_t)(1-\bar{a}_{t-1})}{(1-\bar{a}_t)}\mathbf{I} \right) \tag{3.41}$$

The term $C(\mathbf{x}_t, \mathbf{x}_0)$ depends only on $\mathbf{x}_t, \mathbf{x}_0$ and therefore is considered to be constant. At each timestep, $\mathbf{x}_{t-1}$ is normally distributed with mean $\boldsymbol{\mu}_q(\mathbf{x}_t, \mathbf{x}_0) = \frac{\sqrt{a_t}\mathbf{x}_t(1-\bar{a}_{t-1}) + \sqrt{\bar{a}_{t-1}}\mathbf{x}_0(1-a_t)}{1-\bar{a}_t}$ and variance $\boldsymbol{\Sigma}_q(t) = \sigma_q^2(t)\mathbf{I} = \frac{(1-a_t)(1-\bar{a}_{t-1})}{(1-\bar{a}_t)}\mathbf{I}$. Therefore, in order to match the ground-truth transition step $q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)$, $p_{\boldsymbol{\theta}}(\mathbf{x}_{t-1}|\mathbf{x}_t)$ should also be modeled as a Gaussian. In general, the KL divergence between two Normal distributions $\mathcal{N}_0, \mathcal{N}_1$ in d dimensions is as follows:

$$D_{\mathrm{KL}}(\mathcal{N}_0 \parallel \mathcal{N}_1) = \frac{1}{2}\left[\log\frac{\det(\boldsymbol{\Sigma}_1)}{\det(\boldsymbol{\Sigma}_0)} + \mathrm{tr}(\boldsymbol{\Sigma}_1^{-1}\boldsymbol{\Sigma}_0) + (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)^\top\boldsymbol{\Sigma}_1^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0) - d \right] \tag{3.42}$$

Let $\boldsymbol{\mu}_{\boldsymbol{\theta}}$ and $\boldsymbol{\Sigma}_{\boldsymbol{\theta}}$ be the learnable mean and variance of $p_{\boldsymbol{\theta}}(\mathbf{x}_{t-1}|\mathbf{x}_t)$ respectively. We can directly set $\boldsymbol{\Sigma}_{\boldsymbol{\theta}}(t) = \boldsymbol{\Sigma}_q(t)$ for all t. Maximizing the ELBO requires minimizing the *denoising matching term* in Eq. (3.17):

$$\operatorname*{argmin}_{\boldsymbol{\theta}} D_{\mathrm{KL}}(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) \parallel p_{\boldsymbol{\theta}}(\mathbf{x}_{t-1}|\mathbf{x}_t)) \tag{3.43}$$

$$= \operatorname*{argmin}_{\boldsymbol{\theta}} D_{\mathrm{KL}}(\mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_q, \boldsymbol{\Sigma}_q) \parallel \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_{\boldsymbol{\theta}}, \boldsymbol{\Sigma}_{\mathbf{q}})) \tag{3.44}$$

$$= \operatorname*{argmin}_{\boldsymbol{\theta}} \left[\mathrm{tr}(\mathbf{I}) - d + (\boldsymbol{\mu}_{\boldsymbol{\theta}} - \boldsymbol{\mu}_q)^\top\boldsymbol{\Sigma}_q^{-1}(\boldsymbol{\mu}_{\boldsymbol{\theta}} - \boldsymbol{\mu}_q) \right] \tag{3.45}$$

$$= \operatorname*{argmin}_{\boldsymbol{\theta}} \left[\mathrm{tr}(\mathbf{I}) - d + (\boldsymbol{\mu}_{\boldsymbol{\theta}} - \boldsymbol{\mu}_q)^\top\left(\sigma_q^2\mathbf{I}\right)^{-1}(\boldsymbol{\mu}_{\boldsymbol{\theta}} - \boldsymbol{\mu}_q) \right] \tag{3.46}$$

$$= \operatorname*{argmin}_{\boldsymbol{\theta}} \frac{1}{2\sigma_q^2}\left[\|\boldsymbol{\mu}_{\boldsymbol{\theta}} - \boldsymbol{\mu}_q\|_2^2 \right] \tag{3.47}$$

Eq. (3.47) implies that $\boldsymbol{\mu}_{\boldsymbol{\theta}}$ must exactly match the ground-truth mean, as expected. As $\boldsymbol{\mu}_{\boldsymbol{\theta}}$ only conditions on $\mathbf{x}_t$, it can closely match $\boldsymbol{\mu}_q$ by being set equal to:

$$\boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) = \frac{\sqrt{a_t}(1-\bar{a}_{t-1})\mathbf{x}_t + \sqrt{\bar{a}_{t-1}}(1-a_t)\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)}{1-\bar{a}_t} \tag{3.48}$$

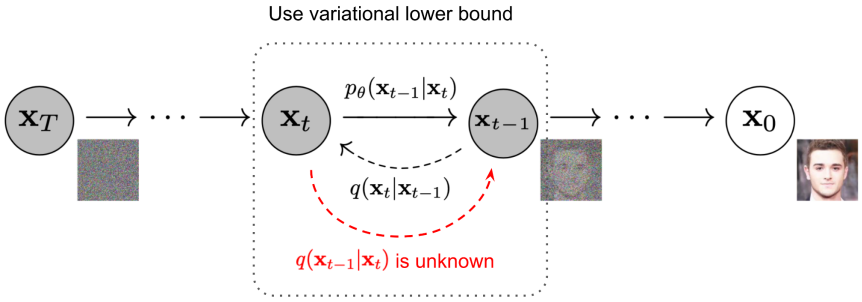


Figure 3.1: The DDPM Markov chain of forward (reverse) noising process of generating a sample by slowly adding (removing) noise. Source: https://lilianweng.github.io/posts/2021-07-11-diffusion-models/. Inspired by [65].

where $\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$ denotes a neural network's prediction of $\mathbf{x}_0$ at noise level t. By substituting Eq. (3.48) and $\boldsymbol{\mu}_q$ into Eq. (3.47), the latter simplifies to:

$$\begin{aligned} &\underset{\boldsymbol{\theta}}{\operatorname{argmin}} \frac{1}{2\sigma_q^2} \frac{\bar{a}_{t-1}(1-a_t)^2}{(1-\bar{a}_t)^2} \left[\|\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) - \mathbf{x}_0\|_2^2\right] \\ =&\underset{\boldsymbol{\theta}}{\operatorname{argmin}} \frac{1}{2} \left(\frac{\bar{a}_{t-1}}{1-\bar{a}_{t-1}} - \frac{\bar{a}_t}{1-\bar{a}_t}\right) \left[\|\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) - \mathbf{x}_0\|_2^2\right] \end{aligned} \tag{3.49}$$

By solving Eq. (3.31) w.r.t. the ground-truth $\mathbf{x}_0$, we get the following expression:

$$\mathbf{x}_0 = \frac{\mathbf{x}_t - \sqrt{1-\bar{a}_t}\boldsymbol{\epsilon}_0}{\sqrt{\bar{a}_t}} \tag{3.50}$$

Plugging Eq. (3.50) into the ground-truth mean $\boldsymbol{\mu}_q$, we get an equivalent parameterization of the objective (3.47) where a neural network $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$ learns to predict the source noise $\boldsymbol{\epsilon}_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. Hence, learning a DDPM by predicting the original input $\mathbf{x}_0$ is equivalent to learning to predict the noise; empirically, however, it has been found that predicting the noise results in better performance [65, 79]. Fig. 3.1 illustrates the noising and denoising Markov chains within the DDPM mechanism.

Lastly, we mention a third equivalent interpretation of the DDPM training objective, based on the score function $\nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t)$ and *Tweedie's Formula* [40]. The latter states that the true mean of an exponential family distribution, given samples drawn from it, can be estimated by the maximum likelihood estimate of the samples, plus some correction term involving the score of the estimate. In the case of just one observed sample, the empirical mean is just the sample itself. So, given a Gaussian random variable $\mathbf{z} \sim \mathcal{N}(\boldsymbol{\mu}_\mathbf{z}, \boldsymbol{\Sigma}_\mathbf{z})$ the following holds true:

$$\mathbb{E}[\boldsymbol{\mu}_\mathbf{z}|\mathbf{z}] = \mathbf{z} + \boldsymbol{\Sigma}_\mathbf{z} \nabla_\mathbf{z} \log p(\mathbf{z}) \tag{3.51}$$

Using Eq. (3.31), the true posterior mean of $\mathbf{x}_t$ given its samples can be approximated as follows:

$$\mathbb{E}[\boldsymbol{\mu}_{\mathbf{x}_t}|\mathbf{x}_t] = \mathbf{x}_t + (1-\bar{a}_t)\nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t) \tag{3.52}$$

Substituting the true mean $\boldsymbol{\mu}_{\mathbf{x}_t} = \sqrt{\bar{a}_t}\mathbf{x}_0$ back into Eq. (3.52), we get:

$$\begin{aligned} \sqrt{\bar{a}_t}\mathbf{x}_0 &= \mathbf{x}_t + (1-\bar{a}_t)\nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t) \\ \Rightarrow \mathbf{x}_0 &= \frac{\mathbf{x}_t + (1-\bar{a}_t)\nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t)}{\sqrt{\bar{a}_t}} \end{aligned} \tag{3.53}$$

Plugging Eq. (3.53) into the ground-truth mean $\boldsymbol{\mu}_q$, we get an equivalent parameterization of the objective (3.47) where a neural network $\hat{\boldsymbol{s}}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$ learns to predict the score function $\nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t)$. Combining Eqs. (3.50) and (3.53), it can be shown that the source noise $\boldsymbol{\epsilon}_0$ and the score function $\nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t)$ off by a constant factor that scales with time:

$$\begin{aligned} \frac{\mathbf{x}_t + (1-\bar{a}_t)\nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t)}{\sqrt{\bar{a}_t}} &= \frac{\mathbf{x}_t - \sqrt{1-\bar{a}_t}\boldsymbol{\epsilon}_0}{\sqrt{\bar{a}_t}} \\ \Rightarrow \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t) &= -\frac{1}{\sqrt{1-\bar{a}_t}}\boldsymbol{\epsilon}_0 \end{aligned} \tag{3.54}$$

3.2 Denoising Diffusion Implicit Models

Denoising Diffusion Implicit Models (DDIM) [148] constitute a class of generative models trained with denoising auto-encoding/score-matching objectives and are characterized by a non-Markovian *forward* process. In the DDIM setting, samples can be uniquely determined from the latent variables, hence DDIM have certain properties that resemble GANs. In the upcoming analysis, in order to be consistent with the derivation in [148], we use the notation a_t to represent the variable $\bar{a}_t$. This is equivalent with choosing a decreasing sequence $a_{1:T} \in (0, 1]$ with the following latent transition step:

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}\left(\mathbf{x}_t; \sqrt{\frac{a_t}{a_{t-1}}}\mathbf{x}_{t-1}, \left(1 - \frac{a_t}{a_{t-1}}\right)\mathbf{I}\right) \tag{3.55}$$

DDIM are based on the following family of inference distributions indexed by a real vector $\boldsymbol{\sigma} \in \mathbb{R}^T_{\geq 0}$:

$$q_{\boldsymbol{\sigma}}(\mathbf{x}_{1:T}|\mathbf{x}_0) := q_{\boldsymbol{\sigma}}(\mathbf{x}_T|\mathbf{x}_0) \prod_{t=2}^{T} q_{\boldsymbol{\sigma}}(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) \tag{3.56}$$

where $q_{\boldsymbol{\sigma}}(\mathbf{x}_T|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_T, \sqrt{a_T}\mathbf{x}_0, (1 - a_T)\mathbf{I})$ and for all $t > 1$:

$$q_{\boldsymbol{\sigma}}(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}\left(\sqrt{a_{t-1}}\mathbf{x}_0 + \left(\sqrt{1 - a_{t-1} - \sigma_t^2}\right)\frac{\mathbf{x}_t - \sqrt{a_t}\mathbf{x}_0}{\sqrt{1 - a_t}}, \sigma_t^2\mathbf{I}\right) \tag{3.57}$$

The Gaussian mean function in Eq. (3.57) is chosen in such way so as to ensure that $q_{\boldsymbol{\sigma}}(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t, \sqrt{a_t}\mathbf{x}_0, (1 - a_t)\mathbf{I})$, leading to a joint inference distribution that matches the desired marginals. The corresponding *forward* process can be derived from Bayes' rule:

$$q_{\boldsymbol{\sigma}}(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0) = \frac{q_{\boldsymbol{\sigma}}(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)q_{\boldsymbol{\sigma}}(\mathbf{x}_t|\mathbf{x}_0)}{q_{\boldsymbol{\sigma}}(\mathbf{x}_{t-1}|\mathbf{x}_0)} \tag{3.58}$$

The aforementioned *forward* process, although it is Gaussian, it is no longer Markovian because each $\mathbf{x}_t$ depends on both $\mathbf{x}_{t-1}$ and $\mathbf{x}_0$. The magnitude of $\boldsymbol{\sigma}$ controls how stochastic the forward process is.

The trainable generative process is modeled so that given a noisy observation $\mathbf{x}_t$, a prediction $\mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$ corresponding to $\mathbf{x}_0$ is made, which is then used to obtain a sample $\mathbf{x}_{t-1}$ through the ground-truth conditional distribution $q_{\boldsymbol{\sigma}}(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)$. The model $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$ attempts to predict the noise from $\mathbf{x}_t$ without knowledge of $\mathbf{x}_0$. Based on Eq. (3.50), the prediction of the *denoised observation* is defined as follows:

$$\mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) := \frac{\mathbf{x}_t - \sqrt{1 - a_t}\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)}{\sqrt{a_t}} \tag{3.59}$$

The generative process is defined with a fixed prior $p_{\boldsymbol{\theta}}(\mathbf{x}_T) = \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I})$ and transition steps:

$$p_{\boldsymbol{\theta}}(\mathbf{x}_{t-1}|\mathbf{x}_t) = \begin{cases} \mathcal{N}(\mathbf{x}_0; \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}_1, 1)), \sigma_1^2\mathbf{I}) & \text{if } t = 1 \\ q_{\boldsymbol{\sigma}}(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}_t, t))) & \text{otherwise} \end{cases} \tag{3.60}$$

Combining Eqs. (3.60) and (3.57), the DDIM sampling process becomes:

$$\mathbf{x}_{t-1} = \sqrt{a_{t-1}}\left(\frac{\mathbf{x}_t - \sqrt{1 - a_t}\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)}{\sqrt{a_t}}\right) + \left(\sqrt{1 - a_{t-1} - \sigma_t^2}\right)\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) + \sigma_t\boldsymbol{\epsilon} \tag{3.61}$$

where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and $a_0 := 1$. Different choices for $\boldsymbol{\sigma}$ result in different generative processes, all while using the same model $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$. When $\sigma_t = \sqrt{(1 - a_{t-1})/(1 - a_t)}\sqrt{1 - a_t/a_{t-1}}$ for all t, then the forward process becomes Markovian, and the generative process becomes a DDPM, i.e., the mean and variance in Eq. (3.57) become equal to the $\boldsymbol{\mu}_q(\mathbf{x}_t, \mathbf{x}_0)$ and $\sigma_q^2(t)$, respectively. Moreover, when $\sigma_t = 0$ for all t, both the *forward* and *reverse* processes become deterministic. The resulting model becomes an implicit probabilistic model [106]. When considering accelerated generation with τ being an increasing subsequence of $\{1, \ldots, T\}$, the variance hyperparameter is often parameterized as $\sigma_{\tau_i}(\eta) = \eta\sqrt{(1 - a_{\tau_{i-1}})/(1 - a_{\tau_i})}\sqrt{1 - a_{\tau_i}/a_{\tau_{i-1}}}$. The latter notation will prove useful in simplifying comparisons during experimentation with different variance hyperparameters $\boldsymbol{\sigma}$.

When $\sigma_t = 0$, Eq. (3.61) can be considered as an Euler method to solve ODEs:

$$\frac{\mathbf{x}_{t-\Delta t}}{\sqrt{a_{t-\Delta t}}} - \frac{\mathbf{x}_t}{\sqrt{a_t}} = \left(\sqrt{\frac{1-a_{t-\Delta t}}{a_{t-\Delta t}}} - \sqrt{\frac{1-a_t}{a_t}} \right) \hat{\epsilon}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) \tag{3.62}$$

To derive the corresponding ODE, we can re-parameterize $w := \sqrt{1-a}/\sqrt{a}$ and $\bar{\mathbf{x}} := \mathbf{x}/\sqrt{a}$. In the continuous case, w and $\mathbf{x}$ are functions of t, where $w : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ is continuous, increasing with $w(0) = 0$. Then, Eq. (3.62) can be treated as a Euler method over the following ODE:

$$\mathrm{d}\bar{\mathbf{x}}(t) = \hat{\epsilon}_{\boldsymbol{\theta}} \left(\frac{\bar{\mathbf{x}}(t)}{\sqrt{\sigma^2 + 1}}, t \right) \mathrm{d}w(t) \tag{3.63}$$

This suggests that with enough discretization steps, the above generation process can be reversed (going from $t = 0$ to T), which encodes $\mathbf{x}_0$ to $\mathbf{x}_T$.

3.3 Guidance

All discussion up to this point has been dedicated to learning the distribution $p(\mathbf{x})$ of a given dataset. However, it also useful to be able to learn conditional data distributions $p(\mathbf{x}|y)$ where y denotes any kind of conditioning information, e.g., class labels. According to related literature, there are two main methods for achieving this, namely *Classifier Guidance* [33] and *Classifier-Free Guidance* [66].

3.3.1 Classifier Guidance

Let $p_{\boldsymbol{\phi}}(y|\mathbf{x}_t)$ be a classifier that aims at classifying noisy images $\mathbf{x}_t$, at various noise levels t. According to [33], the corresponding conditional denoising process becomes:

$$p_{\boldsymbol{\theta},\boldsymbol{\phi}}(\mathbf{x}_{t-1}|\mathbf{x}_t, y) \propto p_{\boldsymbol{\theta}}(\mathbf{x}_{t-1}|\mathbf{x}_t) p_{\boldsymbol{\phi}}(y|\mathbf{x}_{t-1}) \tag{3.64}$$

From the previous section, we know that $p_{\boldsymbol{\theta}}(\mathbf{x}_{t-1}|\mathbf{x}_t)$ is normally distributed with mean $\boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$ and covariance matrix $\boldsymbol{\Sigma}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$. Assuming that $\log p_{\boldsymbol{\phi}}(y|\mathbf{x}_{t-1})$ has low curvature compared to $\boldsymbol{\Sigma}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$ (which is reasonable in the limit $T \to \infty$), the latter can be approximated using a Taylor expansion around $\mathbf{x}_{t-1} = \boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$ (we drop dependence on $\mathbf{x}_t$ and t for simplicity):

$$\begin{aligned} \log p_{\boldsymbol{\phi}}(y|\mathbf{x}_{t-1}) &\approx \log p_{\boldsymbol{\phi}}(y|\mathbf{x}_{t-1})\big|_{\mathbf{x}_{t-1}=\boldsymbol{\mu}_{\boldsymbol{\theta}}} + (\mathbf{x}_{t-1} - \boldsymbol{\mu}_{\boldsymbol{\theta}})^\top \nabla \log p_{\boldsymbol{\phi}}(y|\mathbf{x}_{t-1})\big|_{\mathbf{x}_{t-1}=\boldsymbol{\mu}_{\boldsymbol{\theta}}} \\ &= (\mathbf{x}_{t-1} - \boldsymbol{\mu}_{\boldsymbol{\theta}})^\top \mathbf{g} + \text{constant} \end{aligned} \tag{3.65}$$

where $\mathbf{g} = \nabla \log p_{\boldsymbol{\phi}}(y|\mathbf{x}_{t-1})\big|_{\mathbf{x}_{t-1}=\boldsymbol{\mu}_{\boldsymbol{\theta}}}$. Substituting back into Eq. (3.64), gives:

$$\begin{aligned} \log p_{\boldsymbol{\theta},\boldsymbol{\phi}}(\mathbf{x}_{t-1}|\mathbf{x}_t, y) &\propto -\frac{1}{2}(\mathbf{x}_{t-1} - \boldsymbol{\mu}_{\boldsymbol{\theta}})^\top \boldsymbol{\Sigma}_{\boldsymbol{\theta}}^{-1}(\mathbf{x}_{t-1} - \boldsymbol{\mu}_{\boldsymbol{\theta}}) + (\mathbf{x}_{t-1} - \boldsymbol{\mu}_{\boldsymbol{\theta}})^\top \mathbf{g} \\ &\propto -\frac{1}{2}(\mathbf{x}_{t-1} - \boldsymbol{\mu}_{\boldsymbol{\theta}} - \boldsymbol{\Sigma}_{\boldsymbol{\theta}}\mathbf{g})^\top \boldsymbol{\Sigma}_{\boldsymbol{\theta}}^{-1}(\mathbf{x}_{t-1} - \boldsymbol{\mu}_{\boldsymbol{\theta}} - \boldsymbol{\Sigma}_{\boldsymbol{\theta}}\mathbf{g}) + \frac{1}{2}\mathbf{g}^\top \boldsymbol{\Sigma}_{\boldsymbol{\theta}} \mathbf{g} \\ &\propto \log p(\mathbf{z}), \qquad \mathbf{z} \sim \mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{\theta}} + \boldsymbol{\Sigma}_{\boldsymbol{\theta}}\mathbf{g}, \boldsymbol{\Sigma}_{\boldsymbol{\theta}}) \end{aligned} \tag{3.66}$$

Therefore, the conditional transition operator can be approximated by a Gaussian similar to the unconditional transition operator, but with its mean shifted by $\boldsymbol{\Sigma}_{\boldsymbol{\theta}}\mathbf{g}$. A gradient scaling factor γ can also be incorporated, i.e., $\gamma \nabla_{\mathbf{x}} \log p(y|\mathbf{x}) \propto \nabla \log p(y|\mathbf{x})^\gamma$. As a result, the conditioning process is still theoretically grounded in a classifier distribution proportional to $p(y|\mathbf{x})^\gamma$. When $\gamma > 1$, the latter becomes sharper than $p(y|\mathbf{x})$, since larger values are amplified by the exponent. Consequently, using larger gradient scale focuses more on the modes of the classifier, producing higher fidelity, but potentially, less diverse samples.

Classifier guidance can also be applied in the DDIM setting. To this end, Eq. (3.54) will be put in use, establishing the necessary connection between the conditional score function $\nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t|y)$

and the corresponding network's noise predictions $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_t|y)$ (we drop the time index t for simplicity). Using Bayes' rule, we first notice that:

$$\begin{aligned} \log p(\mathbf{x}_t|y) = \log \frac{p(y|\mathbf{x}_t)p(\mathbf{x}_t)}{p(y)} &= \log p(y|\mathbf{x}_t) + \log p(\mathbf{x}_t) - \log p(y) \\ \Rightarrow \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t|y) &= \nabla_{\mathbf{x}_t} \log p(y|\mathbf{x}_t) + \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t) \end{aligned} \tag{3.67}$$

Applying Eq. (3.54) in this case, results in the following formula for $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_t|y)$:

$$\begin{aligned} -\frac{1}{\sqrt{1-\bar{a}_t}}\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_t|y) &= -\frac{1}{\sqrt{1-\bar{a}_t}}\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_t) + \nabla_{\mathbf{x}_t} \log p_{\boldsymbol{\phi}}(y|\mathbf{x}_t) \\ \Rightarrow \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_t|y) &= \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_t) - \sqrt{1-\bar{a}_t}\nabla_{\mathbf{x}_t} \log p_{\boldsymbol{\phi}}(y|\mathbf{x}_t) \end{aligned} \tag{3.68}$$

We can then use the exact same sampling procedure as used in regular DDIM, but with the modified noise predictions $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_t|y)$ instead of $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_t)$.

3.3.2 Classifier-Free Guidance

As the name implies, *Classifier-Free Guidance* [66] does not require training a separate classifier. Instead, one trains a conditional diffusion model $p(\mathbf{x}|y)$ with conditioning dropout probability p_{uncond}; some percentage of the time, the conditioning information y is removed (10-20% tends to work well) and is replaced with a $\varnothing$ (`null`) label. In practice, it is often replaced with a special input value representing the absence of conditioning information. The resulting model is now able to function both as a conditional model $p(\mathbf{x}|y)$, and as an unconditional model $p(\mathbf{x})$, depending on whether the conditioning signal is provided.

As discussed in the previous section, classifier guidance results in sampling from a distribution $p_\gamma(\mathbf{x}_t|y) \propto p(\mathbf{x}_t)p(y|\mathbf{x}_t)^\gamma$, where $\gamma \in [0, +\infty)$ plays the role of an inverse temperature parameter. This time, we apply Bayes' rule in a different direction relative to Eq. (3.67) in order to eliminate terms containing the classifier $p(y|\mathbf{x}_t)$:

$$\begin{aligned} \nabla_{\mathbf{x}_t} \log p_\gamma(\mathbf{x}_t|y) &= \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t) + \gamma\nabla_{\mathbf{x}_t} \log p(y|\mathbf{x}_t) \\ &= \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t) + \gamma(\nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t|y) - \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t)) \\ &= (1-\gamma)\nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t) + \gamma\nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t|y) \end{aligned} \tag{3.69}$$

Then sampling is performed using the following linear combination of the conditional and unconditional score estimates:

$$\tilde{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_t|y) = (1-\gamma)\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{x}_t|\varnothing) + \gamma\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{x}_t|y) \tag{3.70}$$

This is a barycentric combination of the conditional and the unconditional score function. For $\gamma = 0$, we recover the unconditional model, and for $\gamma = 1$ we get the standard conditional model. For $\gamma > 1$ the resulting distribution becomes "super-conditioned".

3.4 Latent Diffusion Models

Latent Diffusion Models (LDM) [125] were introduced in an attempt to lower the computational demands of training diffusion models towards high-resolution image synthesis. The proposed approach suggests an explicit separation of the compressive from the generative learning phase, utilizing an autoencoding model which learns a space that is perceptually equivalent to the image space, but offers significantly reduced computational complexity in both training and sampling procedures.

3.4.1 Perceptual Image Compression

The image compression models used in [125] consisted of an autoencoder trained with a combination of a perceptual loss and patch-based adversarial objective. The usage of a GAN-based loss enforces local realism upon the reconstructed images, avoiding bluriness introduced by relying solely on pixel-space losses such as ℓ_1 or ℓ_2 objectives.

Given an RGB input image $\mathbf{x} \in \mathbb{R}^{H\times W\times 3}$, an encoder $\mathcal{E}$ encodes $\mathbf{x}$ into a latent representation $\mathbf{z} = \mathcal{E}(\mathbf{x}) \in \mathbb{R}^{h\times w\times c}$, while a decoder $\mathcal{D}$ decodes the latent and restores it back in the original image space, producing the reconstruction $\tilde{\mathbf{x}} = \mathcal{D}(\mathbf{z}) = \mathcal{D}(\mathcal{E}(\mathbf{x}))$. The encoder downsamples the input image by a factor $f = H/h = W/w$. In [125], the authors investigated downsampling factors of the form $f = 2^m, m \in \mathbb{N}$.

Furthermore, the first stage compression models undergo different regularization schemes with the aim of avoiding learned latent spaces of arbitrarily high variance. The first of the two proposed autoencoder variants, namely *KL-reg*, features a slight KL-penalty towards a standard normal on the learned latent, similar to a regular VAE. The second variant, namely *VQ-reg* is built upon the VQGAN [45] architecture and uses a vector quantization layer in order to learn codebook consisting of discrete latent representations.

Learning Discrete Latent Spaces

In general, a big portion of the data encountered in real-world, everyday scenarios, often favor discrete representations over continuous ones. For example, language is inherently discrete, similarly speech is typically represented as a sequence of symbols. Images can often be described concisely by language. Furthermore, discrete representations are a natural fit for complex reasoning, planning and predictive learning (e.g., if it rains, I will use an umbrella). A new family of generative models [45, 123, 167, 179] has recently emerged by successfully combining the VAE framework with discrete latent representations based on *Vector Quantisation* (VQ).

The first notable realization of a VAE that made effective use of discrete latents came in the form of the VQ-VAE [167]. The latter is a type of variational autoencoder that uses VQ to obtain discrete latent representations. It differs from VAEs in two key ways: the encoder network outputs discrete, rather than continuous codes; and the prior is learned rather than being static. In order to learn a discrete latent representation, ideas from VQ are incorporated, as the latter allows the model to circumvent issues of posterior collapse - where the latents are ignored when they are paired with a powerful autoregressive decoder - typically observed in the VAE framework. Pairing these representations with an autoregressive prior, the model can generate high quality images, videos, and speech as well as doing high quality speaker conversion and unsupervised learning of phonemes.

More specifically, let $\{\boldsymbol{e}_i\}_{i=1}^K \subset \mathbb{R}^d$ be a latent embedding space where K is the size of the discrete latent space and d is the dimensionality of each latent embedding vector $\boldsymbol{e}_i \in \mathbb{R}^d, i \in [K]$. The model takes an input $\mathbf{x}$, that is passed through an encoder producing output $\boldsymbol{z}_e(\mathbf{x})$. Subsequently, the discrete latent variables z are calculated by a nearest neighbour look-up using the shared embedding space. The posterior categorical distribution $q(z|\mathbf{x})$ probabilities are defined as one-hot as follows:

$$q(z = k|\mathbf{x}) = \begin{cases} 1 & \text{for } k = \operatorname{argmin}_j \|\boldsymbol{z}_e(\mathbf{x}) - \boldsymbol{e}_j\|_2 \\ 0 & \text{otherwise} \end{cases} \tag{3.71}$$

The representation $\boldsymbol{z}_e(\mathbf{x})$ is passed through the discretization bottleneck followed by a mapping onto the nearest embedding element:

$$\boldsymbol{z}_q(\mathbf{x}) = \boldsymbol{e}_k, \text{ where } k = \operatorname*{argmin}_j \|\boldsymbol{z}_e(\mathbf{x}) - \boldsymbol{e}_j\|_2 \tag{3.72}$$

No real gradient can be defined for Eq. (3.72), however it can be approximated similarly to the *Straight-Through Estimator* (STE) [12] and just copy gradients from decoder input $\boldsymbol{z}_q(\mathbf{x})$ to encoder output $\mathbf{z}_e(\mathbf{x})$. VQ-VAE training is guided through the minimization of the following loss function:

$$\mathcal{L}_{\text{VQ}} = \log(p(\mathbf{x}|\boldsymbol{z}_q(\mathbf{x}))) + \|\text{sg}[\boldsymbol{z}_e(\mathbf{x})] - \boldsymbol{e}\|_2^2 + \beta\|\boldsymbol{z}_e(\mathbf{x}) - \text{sg}[\boldsymbol{e}]\|_2^2 \tag{3.73}$$

where β is a scaling factor and $\text{sg}[\cdot]$ stands for the stopgradient operator which is defined as identity at forward computation time and has zero partial derivatives, thus effectively constraining its operand to be a non-updated constant. The first term constitutes the reconstruction loss which optimizes the decoder and the encoder through the STE. Furthermore, the embedding space is learned using the simplest dictionary learning algorithm, that is, VQ. The VQ objective uses an ℓ_2 error to guide the embedding vectors towards the encoder outputs, as indicated by the second term of Eq. (3.73). Lastly, since the volume of the embedding space is dimensionless, it can grow arbitrarily in case the

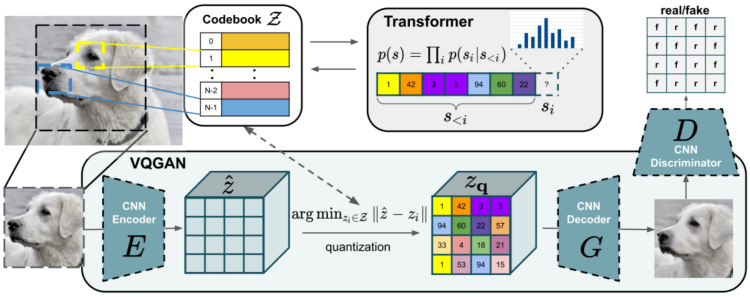


Figure 3.2: Approach for using a convolutional VQGAN to learn a codebook of context-rich visual parts, whose composition is subsequently modeled with an autoregressive transformer architecture. Source: [45].

embeddings do not train as fast as the encoder parameters. Therefore, a commitment loss is added in the form of the third term in Eq. (3.73), so as to ensure that the encoder commits to an embedding and its output does not grow.

The prior distribution over the discrete latents $p(z)$ is a categorical distribution, and can be made autoregressive by depending on other z in the feature map. Whilst training the VQ-VAE, the prior is kept constant and uniform. After training, an autoregressive distribution is fitted over z, $p(z)$, so that $\mathbf{x}$ can be generated via ancestral sampling. The authors in [167] used a PixelCNN over the discrete latents for images.

As opposed to vanilla VQ-VAE, the authors in [123], inspired by the ideas of coarse-to-fine generation, proposed a hierarchy of vector quantized codes to tackle large scale, high-fidelity image generation. The corresponding model, i.e., VQ-VAE-2, has an architecture that resembles a 3-level U-Net [126] with concatenating skip connections, except that each feature map undergoes quantization before being passed to the decoder. In order to sample from the model, a separate autoregressive prior (PixelCNN) is learned over the sequences of latent codes at each resolution level. The authors use self-attention layers in the top level prior since it has lower resolution, and large conditional stacks coming from the top prior to the bottom prior with higher resolution due to memory constraints. Each prior is trained separately. Sampling from the model requires passing a class label to the trained top level PixelCNN to obtain the top level codes, then passing the class label along with the generated codes to the bottom level to generate the higher resolution code, and then use the decoder to generate an image from the top and bottom level codes.

VQGAN [45] differs from the aforementioned implementations in the following two ways: (i) The reconstruction loss in the original VQ-VAE objective function is replaced with a perceptual loss, while an adversarial training procedure with a patch-based discriminator, is also introduced. (ii) Image generation is formulated as autoregressive next-index prediction using a transformer that operates on sequences of quantized tokens, drawn from a learned discrete codebook, and learns to predict the distribution for next possible tokens given a sequence of previous tokens.

Any RGB image $\mathbf{x} \in \mathbb{R}^{H \times W \times 3}$ can be represented by a spatial collection of codebook entries $\mathbf{z}_q \in \mathbb{R}^{h \times w \times n_z}$ where n_z denotes the dimensionality of codes. An equivalent representation is a sequence of $h \cdot w$ indices which specify the respective entries in the learned codebook. The discrete spatial codebook $\mathcal{Z} = \{\boldsymbol{z}_k\}_{k=1}^{K} \subset \mathbb{R}^{n_z}$ is learned using a CNN-based encoder-decoder architecture. Encodings $\hat{\mathbf{z}} = \mathcal{E}(\mathbf{x}) \in \mathbb{R}^{h \times w \times n_z}$ undergo element-wise quantization $\mathbf{q}(\cdot)$ on each one of their spatial codes $\hat{\boldsymbol{z}}_{ij} \in \mathbb{R}^{n_z}$ so as to get mapped to their closest codebook entries:

$$\mathbf{z}_q = \mathbf{q}(\hat{\mathbf{z}}) := \underset{\mathbf{z}_k \in \mathcal{Z}}{\operatorname{argmin}} \|\hat{\boldsymbol{z}}_{ij} - \boldsymbol{z}_k\| \in \mathbb{R}^{h \times w \times n_z} \tag{3.74}$$

The reconstruction $\hat{\mathbf{x}}$ generated by the decoder $\mathcal{G}$ is given by:

$$\hat{\mathbf{x}} = \mathcal{G}(\mathbf{z}_q) = \mathcal{G}(\mathbf{q}(\mathcal{E}(\mathbf{x}))) \tag{3.75}$$

Apart from the vector quantization objective of Eq. (3.73), where the reconstruction loss is replaced with the *Learned Perceptual Image Patch Similarity* (LPIPS) [181] metric, an adversarial loss term is added with a patch-based discriminator D aiming to differentiate between real and reconstructed images:

$$\mathcal{L}_{\text{GAN}} = \log D(\mathbf{x}) + \log\left(1 - D(\hat{\mathbf{x}})\right) \tag{3.76}$$

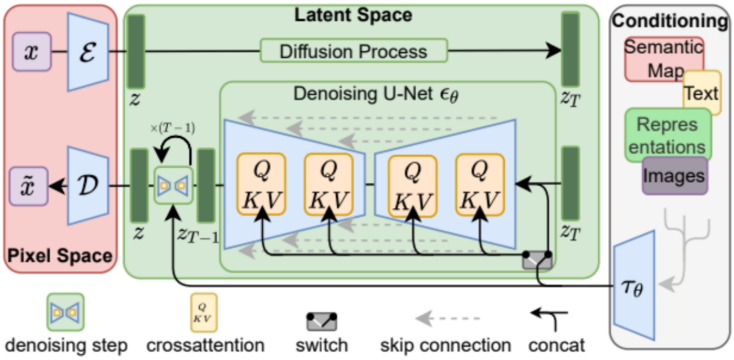


Figure 3.3: Overview of the LDM framework. Source: [125].

The complete objective for finding the optimal compression model becomes:

$$\underset{\mathcal{E},\mathcal{G},\mathcal{Z}}{\operatorname{argmin}} \max_{D} \mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}[\mathcal{L}_{\text{VQ}}(\mathcal{E},\mathcal{G},\mathcal{Z}) + \lambda \mathcal{L}_{\text{GAN}}(\{\mathcal{E},\mathcal{G},\mathcal{Z}\},D)] \tag{3.77}$$

where λ is an adaptive weight parameter. With $\mathcal{E}$ and $\mathcal{G}$ available, images can be represented as sequences of codebook-indices $\mathbf{s} \in [|\mathcal{Z}|]^{h\times w}$, where:

$$s_{ij} = k \text{ such that } (\mathbf{z}_q)_{ij} = \boldsymbol{z}_k \tag{3.78}$$

Thus, after choosing some ordering of the indices in $\mathbf{s}$, image-generation is formulated as autoregressive next-index prediction: Given indices $\mathbf{s}_{<i}$, a transformer learns to predict the distribution of possible next indices, i.e., $p(s_i|\mathbf{s}_{<i})$. The transformer is trained by maximizing the log-likelihood of the data representations:

$$\mathcal{L}_{\text{transformer}} = \mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}\left[-\prod_i p(s_i|\mathbf{s}_{<i})\right] \tag{3.79}$$

A high level view of the VQGAN coupled with a transformer-based autoregressive prior can be seen in Fig. 3.2.

In [179], the aforementioned approach is taken one step further by replacing both the CNN encoder and decoder with a ViT [38]. With the proposed ViT-VQGAN, images are encoded into discrete tokens represented by integers, each of which encompasses an 8×8 patch of the input image. Using these tokens, a decoder-only transformer is trained to predict a sequence of image tokens autoregressively.

3.4.2 Generative Modeling of Latent Representations

Having trained perceptual compression models consisting of an encoder $\mathcal{E}$ and a decoder $\mathcal{D}$, an efficient, learned, low-dimensional latent space can now be used, in which high-frequency, imperceptible details are abstracted away. Compared to the high-dimensional pixel space, the latent space is more suitable for likelihood-based generative models, as they can focus on the semantically important bits of data as well as train in a lower dimensional, computationally efficient space.

Taking advantage of the 2D structure of the produced latents, the backbone of the diffusion model is built upon the U-Net architecture, primarily consisting of 2D convolutional layers. As discussed in section 3.1, DDPM training relies on a reweighted variant of the variational lower bound on the evidence of the true data distribution $\log p(\mathbf{x})$, which resembles denoising score matching over multiple noise scales indexed by t. LDMs can be interpreted as a equally weighted sequence of denoising autoencoders $\boldsymbol{\epsilon_\theta}(\mathbf{z}_t, t)$ which are trained to predict a denoised variant of their input $\mathbf{z}_t$, where $\mathbf{z}_t$ is a noisy version of the input latent $\mathbf{z}$. The corresponding denoising objective becomes:

$$\mathcal{L}_{\text{LDM}} = \mathbb{E}_{\mathcal{E}(\mathbf{x}),\boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\mathbf{I}),t\sim\mathcal{U}(1,T)}\left[\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon_\theta}(\mathbf{z}_t,t)\|_2^2\right] \tag{3.80}$$

The neural backbone $\boldsymbol{\epsilon_\theta}(\cdot,t)$ of the LDM is realized as a time-conditional U-Net.

LDMs are also be capable of modeling conditional distributions of the form $p(\mathbf{z}|y)$, where y denotes any kind of additional information such as class labels, text, semantic maps, etc. To that end, the

U-Net backbone is augmented with a cross-attention [168] mechanism which is effective for learning attention-based models of various input modalities. In order to preprocess y, a domain specific encoder $\boldsymbol{\tau_\theta}$ is introduced that projects the latter to an intermediate representation $\boldsymbol{\tau_\theta}(y) \in \mathbb{R}^{M \times d_\tau}$, which is then mapped to the intermediate layers of the U-Net via a cross-attention layer:

$$\text{Attention}(\mathbf{Q}, \mathbf{V}, \mathbf{K}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d}}\right)\mathbf{V} \tag{3.81}$$

$$\mathbf{Q} = \boldsymbol{\phi}_i(\mathbf{z}_t)\mathbf{W}_Q^{(i)}, \qquad \mathbf{K} = \boldsymbol{\tau_\theta}(y)\mathbf{W}_K^{(i)}, \qquad \mathbf{V} = \boldsymbol{\tau_\theta}(y)\mathbf{W}_V^{(i)} \tag{3.82}$$

where i is the attention head index, $\boldsymbol{\phi}_i(\mathbf{z}_t) \in \mathbb{R}^{N \times d_\epsilon^i}$ denotes a flattened intermediate representation of the U-Net implementing $\boldsymbol{\epsilon_\theta}$, while $\mathbf{W}_Q^{(i)} \in \mathbb{R}^{d_\epsilon^i \times d}, \mathbf{W}_K^{(i)} \in \mathbb{R}^{d_\tau \times d}, \mathbf{W}_V^{(i)} \in \mathbb{R}^{d_\tau \times d}$ are learnable projection matrices. Then, the conditional LDM denoising objective becomes:

$$\mathcal{L}_{\text{LDM}} = \mathbb{E}_{\mathcal{E}(\mathbf{x}), \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), t \sim \mathcal{U}(1,T)}\left[\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon_\theta}(\mathbf{z}_t, t, \boldsymbol{\tau_\theta}(y))\|_2^2\right] \tag{3.83}$$

An overview of the LDM framework is illustrated in Fig. 3.3.

3.5 Score-Based Diffusion Models

Score-based DMs were initially proposed as a means of generative modeling through estimating the gradients of data distributions. More formally, let $\{\mathbf{x}_i \in \mathbb{R}^D\}_{i=1}^N$ be a dataset of i.i.d. samples from an unknown data distribution $p_{\text{data}}(\mathbf{x})$. The *score network* $\mathbf{s_\theta} : \mathbb{R}^D \to \mathbb{R}^D$ is a neural network parameterized by $\boldsymbol{\theta}$, which is trained to approximate the score of $p_{\text{data}}(\mathbf{x})$, i.e., $\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x})$. The two main ingredients of score-based DMs are score matching and Langevin dynamics, as discussed in section 2.2.

Although the aforementioned approach can be used for data generation, several issues were emphasized in [149] regarding the application of this method on real data. Most of the problems are linked with the manifold hypothesis. For example, the score estimation $\mathbf{s_\theta}$ is inconsistent when the data resides on a low-dimensional manifold and, among other implications, this could cause the Langevin dynamics to never converge to the high-density regions. In the same work the authors further demonstrated that these problems can be addressed by perturbing the data with Gaussian noise at different scales.

3.5.1 Noise Conditional Score Networks

The proposed method learns score estimations for the resulting noisy distributions via a single *Noise Conditioned Score Network* (NCSN) [149]. Given a sequence of Gaussian noise scales $\sigma_1 < \sigma_2 < \cdots < \sigma_T$ such that $p_{\sigma_1}(\mathbf{x}) \approx p(\mathbf{x}_0)$ and $p_{\sigma_T}(\mathbf{x}) \approx \mathcal{N}(\mathbf{x}; \mathbf{0}, \mathbf{I})$, an NCSN $\mathbf{s_\theta}(\mathbf{x}, \sigma_t)$ can be trained with denoising score matching so that $\mathbf{s_\theta}(\mathbf{x}, \sigma_t) \approx \nabla_{\mathbf{x}} \log p_{\sigma_t}(\mathbf{x}), \forall t \in [T]$. Given that $p_{\sigma_t}(\mathbf{x}_t|\mathbf{x}) = \mathcal{N}(\mathbf{x}_t; \mathbf{x}, \sigma_t^2\mathbf{I})$, the score of the perturbation kernel is:

$$\nabla_{\mathbf{x}_t} \log p_{\sigma_t}(\mathbf{x}_t|\mathbf{x}) = -\frac{\mathbf{x}_t - \mathbf{x}}{\sigma_t} \tag{3.84}$$

where $\mathbf{x}_t$ is a noised version of $\mathbf{x}$. Generalizing Eqs. (2.13) and (2.14) for all noise scales $\{\sigma_t\}_{t=1}^T$, the NCSN objective becomes:

$$\mathcal{L}_{\text{DSM}} = \frac{1}{T}\sum_{t=1}^{T} \lambda(\sigma_t)\mathbb{E}_{p_{\text{data}}(\mathbf{x})}\mathbb{E}_{p_{\sigma_t}(\mathbf{x}_t|\mathbf{x})}\left[\left\|\mathbf{s_\theta}(\mathbf{x}_t, \sigma_t) + \frac{\mathbf{x}_t - \mathbf{x}}{\sigma_t}\right\|_2^2\right] \tag{3.85}$$

where $\lambda(\sigma_t)$ is a weighting function. After training, the neural network $\mathbf{s_\theta}(\mathbf{x}_t, \sigma_t)$ returns estimates of the scores of $p_{\sigma_t}(\mathbf{x}_t|\mathbf{x})$, having as input noisy images $\mathbf{x}_t$ and corresponding time step t. Sampling during inference is performed using annealed Langevin dynamics.

In their subsequent work [150], the authors proposed a set of techniques to scale score-based generative models to high resolution images. Based on the analysis of a simplified mixture model, they provided a method to analytically compute an effective set of Gaussian noise scales from training

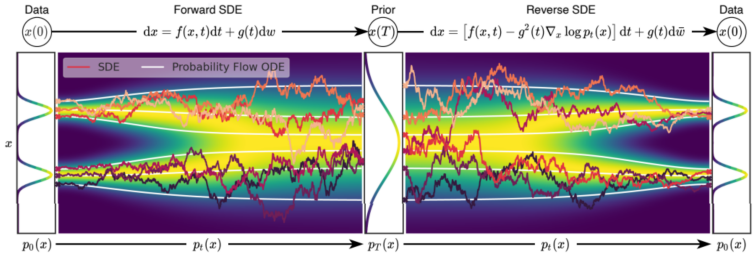


Figure 3.4: Overview of score-based generative modeling through SDEs. Source: [153].

data. In addition, they reparameterized the original NCSN so as to efficiently handle a large (possibly infinite) number of noise scales and further optimized the underlying Langevin dynamics sampling procedure w.r.t. the noise scales. Lastly, they enhanced stability by maintaining an exponential moving average (EMA) of model weights. With all these improvements, score-based generative models were successfully scaled to various image datasets, with diverse resolutions ranging from 64×64 to 256×256.

3.6 Connection to SDEs

The diffusions of both score-based models and DDPM can be expressed as solutions of Stochastic Differential Equations (SDEs). That was the observation made in [153], where the authors generalized the aforementioned two methodologies to continuous time and an infinite number of noise scales. A high-level view of this framework is illustrated in Fig. 3.4.

The goal is to construct a diffusion process $\{\mathbf{x}(t)\}_{t=0}^{T}$ indexed by a continuous variable $t \in [0, T]$, s.t. $\mathbf{x}(0) \sim p_0$ for which a dataset of i.i.d. samples is available and $\mathbf{x}(T) \sim p_T$, for which there is a tractable form to generate samples efficiently. This diffusion process can be modeled as the solution to an Itô SDE:

$$\mathrm{d}\mathbf{x} = \mathbf{f}(\mathbf{x}, t)\mathrm{d}t + g(t)\mathrm{d}\mathbf{w} \tag{3.86}$$

where $\boldsymbol{w}$ is the standard Wiener process, $\mathbf{f}(\cdot, t) : \mathbb{R}^D \to \mathbb{R}^D$ is a vector-valued function called the drift coefficient of $\mathbf{x}(t)$ and $g(\cdot) : \mathbb{R} \to \mathbb{R}$ is a scalar function known as the diffusion coefficient of $\mathbf{x}(t)$.

By starting from samples of $\mathbf{x}(T) \sim p_T$ and reversing the process, samples $\mathbf{x}(0) \sim p_0$ can be obtained. The reverse of the diffusion process in Eq. (3.86) is also a diffusion process, running backwards in time and given by the reverse-time SDE:

$$\mathrm{d}\mathbf{x} = [\mathbf{f}(\mathbf{x}, t) - g^2(t)\nabla_{\mathbf{x}} \log p_t(\mathbf{x})]\mathrm{d}t + g(t)\mathrm{d}\bar{\mathbf{w}} \tag{3.87}$$

where $\bar{\mathbf{w}}$ is a standard Wiener process when time flows backwards from T to 0, and $\mathrm{d}t$ is an infinitesimal negative timestep. Once the score $\nabla_{\mathbf{x}} \log p_t(\mathbf{x})$ is known for all t, we can derive the reverse diffusion process and simulate it to sample from p_0. The score of a distribution can be estimated by training a score-based model on samples with score matching. A time conditional score network $\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}(t), t)$ by optimizing a continuous variant of Eq. (3.85):

$$\mathcal{L}^*_{\mathrm{DSM}} = \mathbb{E}_{t\sim\mathcal{U}(0,T)}\left\{\lambda(t)\mathbb{E}_{p_0(\mathbf{x}(0))}\mathbb{E}_{p_{0t}(\mathbf{x}(t)|\mathbf{x}(0))}\left[\left\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}(t), t) - \nabla_{\mathbf{x}(t)} \log p_{0t}(\mathbf{x}(t)|\mathbf{x}(0))\right\|_2^2\right]\right\} \tag{3.88}$$

where $\lambda(t)$ is a weighting function. Moreover, when $\mathbf{f}$ is affine, $p_{0t}(\mathbf{x}(t)|\mathbf{x}(0))$ is a Gaussian distribution. When $\mathbf{f}$ does not conform to this property, denoising score matching cannot be applied, but sliced score matching can be used instead. In the following sections, we show how both DDPM and NCSN can be formulated as special cases of the general Itô SDE.

3.6.1 VE SDE

In the NCSN setting, when using a total of T noise scales, each perturbation kernel $p_{\sigma_t}(\mathbf{x}_t|\mathbf{x}_0)$ can be derived from the following Markov chain:

$$\mathbf{x}_i = \mathbf{x}_{i-1} + \sqrt{\sigma_t^2 - \sigma_{i-1}^2}\mathbf{z}_{i-1}, \quad i \in [N] \tag{3.89}$$

where $\mathbf{z}_{i-1} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, $\mathbf{x}_0 \sim p_{\text{data}}(\mathbf{x})$ and $\sigma_0 = 0$. In the limit of $N \to \infty$ the Markov chain $\{\mathbf{x}_i\}_{i=1}^N$ becomes a continuous stochastic process $\{\mathbf{x}(t)\}_{t=0}^1$, $\{\sigma_i\}_{i=1}^N$ becomes a function $\sigma(t)$ and $\mathbf{z}_i$ becomes $\mathbf{z}(t)$, where a continuous time variable $t \in [0, 1]$ is used instead of discrete timesteps $i \in [N]$. By letting $\mathbf{x}\left(\frac{i}{N}\right) = \mathbf{x}_i$, $\sigma\left(\frac{i}{N}\right) = \sigma_i$ and $\mathbf{z}\left(\frac{i}{N}\right) = \mathbf{z}_i$, Eq. (3.89) can be rewritten as follows:

$$\mathbf{x}(t + \Delta t) = \mathbf{x}(t) + \sqrt{\sigma^2(t + \Delta t) - \sigma^2(t)}\mathbf{z}(t) \approx \mathbf{x}(t) + \mathbf{z}(t)\sqrt{\frac{\mathrm{d}[\sigma^2(t)]}{\mathrm{d}t}\Delta t} \tag{3.90}$$

where $\Delta t = \frac{1}{N}$, while the approximate equality holds when $\Delta t \ll 1$. In the limit of $\Delta t \to 0$, this converges to:

$$\mathrm{d}\mathbf{x} = \sqrt{\frac{\mathrm{d}[\sigma^2(t)]}{\mathrm{d}t}}\mathrm{d}\mathbf{w} \tag{3.91}$$

which is the so called *Variance Exploding* (VE) SDE, as it always gives a process with exploding variance when $t \to \infty$.

3.6.2 VP SDE

In the DDPM setting, the corresponding discrete Markov chain relative to the perturbation kernels $\{q_{a_i}(\mathbf{x}_t|\mathbf{x}_0)\}_{i=1}^N$ of section 3.1, is:

$$\mathbf{x}_i = \sqrt{1 - \beta_i}\mathbf{x}_{i-1} + \sqrt{\beta_i}\mathbf{z}_{i-1}, \quad i \in [N] \tag{3.92}$$

where $a_i = 1 - \beta_i$ and $\mathbf{z}_{i-1} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. To obtain the limit of this Markov chain when $N \to \infty$, an auxiliary set of noise scales $\{\bar{\beta}_i = N\beta_i\}_{i=1}^N$ is defined. In the limit of $N \to \infty$, $\{\bar{\beta}_i\}_{i=1}^N$ becomes a function $\beta(t)$ indexed by $t \in [0, 1]$. Similarly to the case of the VE SDE, the Markov chain of Eq. (3.92) can be rewritten as:

$$\mathbf{x}(t + \Delta t) = \mathbf{x}(t)\sqrt{1 - \beta(t + \Delta t)\Delta t} + \mathbf{z}(t)\sqrt{\beta(t + \Delta t)\Delta t} \tag{3.93}$$

$$\approx \mathbf{x}(t) - \frac{1}{2}\beta(t + \Delta t)\Delta t\mathbf{x}(t) + \mathbf{z}(t)\sqrt{\beta(t + \Delta t)\Delta t} \tag{3.94}$$

$$\approx \mathbf{x}(t) - \frac{1}{2}\beta(t)\Delta t\mathbf{x}(t) + \mathbf{z}(t)\sqrt{\beta(t)\Delta t} \tag{3.95}$$

where $\Delta t = \frac{1}{N}$, while the approximate equality holds when $\Delta t \ll 1$. In the limit of $\Delta t \to 0$, this converges to:

$$\mathrm{d}\mathbf{x} = -\frac{1}{2}\beta(t)\mathbf{x}\mathrm{d}t + \sqrt{\beta(t)}\mathrm{d}\mathbf{w} \tag{3.96}$$

which is the so called *Variance Preserving* (VP) SDE, as it yields a process with a fixed variance of one when the initial distribution has unit variance.

Chapter 4

Experiments–Face Reenactment

The current chapter is dedicated to presenting our experimental results regarding emotion-conditional and image-based emotional manipulation using diffusion models. The first step of this process includes the establishment of all the relevant metrics used for model evaluation during the aforementioned tasks.

4.1 Evaluation Metrics

Below, we present and establish a list of metrics used to evaluate our diffusion models in image generation, but more importantly, in emotional image manipulation.

4.1.1 Image Generation

Fréchet Inception Distance

The *Fréchet Inception Distance* (FID) [64] is a metric that was initially proposed as a means of evaluating the quality of synthetic images and by extension the performance of GANs. The score was proposed as an improvement over the existing Inception Score (IS) [133] which estimates the quality of a collection of synthetic images based on how well the top-performing image classification model Inception-v3 [156] classifies them as one of the 1,000 object classes supported by the ILSVRC2012 dataset. IS combines both the confidence of the conditional class predictions for each synthetic image (quality) and the integral of the marginal probability of the predicted classes (diversity).

Like in the case of IS, FID utilizes a pre-trained Inception-v3 model. Specifically, the coding layer of the model (the last pooling layer prior to the output classification of images) is used to capture 2048-dim, computer-vision-specific features of an input image. These activations are calculated for a collection of real and generated images. The activations are summarized as a multivariate Gaussian by calculating the mean and covariance of the images. These statistics are then calculated for the activations across the collection of real and generated images. The distance between these two distributions is then calculated using the Fréchet distance [49], also called the Wasserstein-2 [174] distance. Let $\mathcal{N}(\boldsymbol{\mu}_r, \boldsymbol{\Sigma}_r)$ and $\mathcal{N}(\boldsymbol{\mu}_g, \boldsymbol{\Sigma}_g)$ be the fitted multidimensional Gaussian distributions for the real and generated images, respectively. Then, FID is calculated as follows:

$$d^2_{\text{FID}}\big(\mathcal{N}(\boldsymbol{\mu}_r, \boldsymbol{\Sigma}_r), \mathcal{N}(\boldsymbol{\mu}_g, \boldsymbol{\Sigma}_g)\big) = \|\boldsymbol{\mu}_r - \boldsymbol{\mu}_g\|_2^2 + \text{tr}\big(\boldsymbol{\Sigma}_r + \boldsymbol{\Sigma}_g - 2(\boldsymbol{\Sigma}_r\boldsymbol{\Sigma}_g)^{\frac{1}{2}}\big) \tag{4.1}$$

A lower FID indicates better-quality images; conversely, a higher score indicates a lower-quality image and the relationship may be linear.

Kernel Inception Distance

The *Kernel Inception Distance* (KID) [13] aims to improve on FID by relaxing the Gaussian assumption. KID measures the squared *Maximum Mean Discrepancy* (MMD) between the Inception representations of the real and generated samples using a polynomial kernel $k(\mathbf{x}, \mathbf{y}) = (\frac{1}{d}\mathbf{x}^\top\mathbf{y} + 1)^n$, where d is the representation dimension. This is a non-parametric test so it does not have the strict

Gaussian assumption, only assuming that the kernel is a good similarity measure. It also requires fewer samples as fitting the quadratic covariance matrix is not required.

Improved Precision and Recall

In [132], *precision* and *recall* for distributions (PRD) was explicitly introduced to the study of generative models and was formulated through the relative probability densities of two distributions. It was motivated by the observation that FID and related density metrics cannot be used for making conclusions about precision and recall, i.e., a low FID may indicate high precision (realistic images), high recall (large amount of variation), or anything in between. According to the classic viewpoint, *precision* denotes the fraction of generated images that are realistic, and *recall* measures the fraction of the training data manifold covered by the generator.

An improved precision and recall metric was introduced in [88] based on k-nearest neighbors. Real and generated samples are drawn from $\mathbf{x}_r \sim p_r$ and $\mathbf{x}_g \sim p_g$, respectively, and embedded into a high-dimensional feature space using a pre-trained classifier network. Feature vectors for real and generated images are denoted as $\boldsymbol{\phi}_r$ and $\boldsymbol{\phi}_g$, with the corresponding sets being $\boldsymbol{\Phi}_r$ and $\boldsymbol{\Phi}_g$. For each set of feature vectors $\boldsymbol{\Phi} \in \{\boldsymbol{\Phi}_r, \boldsymbol{\Phi}_g\}$, the corresponding manifold in the feature space is estimated by calculating pairwise Euclidean distances between all feature vectors in the set and, for each feature vector, forming a hypersphere with radius equal to the distance to its kth nearest neighbor. The following binary membership function $f(\boldsymbol{\phi}, \boldsymbol{\Phi})$ determines whether a sample $\boldsymbol{\phi}$ is located within the volume induced by $\boldsymbol{\Phi}$:

$$f(\boldsymbol{\phi}, \boldsymbol{\Phi}) = \begin{cases} 1, & \text{if } \|\boldsymbol{\phi} - \boldsymbol{\phi}'\|_2 \leq \|\boldsymbol{\phi}' - \mathrm{NN}_k(\boldsymbol{\phi}', \boldsymbol{\Phi})\|_2 \text{ for at least one } \boldsymbol{\phi}' \in \boldsymbol{\Phi} \\ 0, & \text{otherwise} \end{cases} \tag{4.2}$$

where $\mathrm{NN}_k(\boldsymbol{\phi}', \boldsymbol{\Phi})$ denotes the kth nearest feature vector of $\boldsymbol{\phi}'$ from set $\boldsymbol{\Phi}$. Precision and recall metrics are defined as follows:

$$\mathrm{precision}(\boldsymbol{\Phi}_r, \boldsymbol{\Phi}_g) = \frac{1}{|\boldsymbol{\Phi}_g|} \sum_{\boldsymbol{\phi}_g \in \boldsymbol{\Phi}_g} f(\boldsymbol{\phi}_g, \boldsymbol{\Phi}_r), \qquad \mathrm{recall}(\boldsymbol{\Phi}_r, \boldsymbol{\Phi}_g) = \frac{1}{|\boldsymbol{\Phi}_r|} \sum_{\boldsymbol{\phi}_r \in \boldsymbol{\Phi}_r} f(\boldsymbol{\phi}_r, \boldsymbol{\Phi}_g) \tag{4.3}$$

According to Eq. (4.3), precision is quantified by querying for each generated image whether the image is within the estimated manifold of real images. Symmetrically, recall is calculated by querying for each real image whether the image is within the estimated manifold of generated images.

4.1.2 Image Manipulation

Peak-Signal-to-Noise Ratio

Any processing applied to an image may cause an important loss of information or quality. The most famous and widely used image quality metric is the *Peak-Signal-to-Noise ratio* (PSNR). The simplest definition of this starts out from the *mean-squared-error* (MSE). Let there be two images: $\mathbf{I}_1, \mathbf{I}_2 \in \mathbb{R}^{H \times W \times C}$, with corresponding spatial dimension indices i, j and channel index c. The MSE between the two images is defined as:

$$\mathrm{MSE} = \frac{1}{HWC} \sum_{i,j,c} (I_1^{ijc} - I_2^{ijc})^2 \tag{4.4}$$

Then the PSNR is expressed as:

$$\mathrm{PSNR} = 10 \log_{10} \left(\frac{\mathrm{MAX}_I^2}{\mathrm{MSE}} \right) \tag{4.5}$$

where MAX_I denotes the maximum valid image value, i.e. 255 for `uint8` RGB images. When two images are the same, the MSE will give zero, resulting in an invalid division by zero operation in Eq. (4.5). In this case the PSNR is undefined and this case needs to be handled separately. The transition to a logarithmic scale is made because the pixel values have a very wide dynamic range. Higher PSNR indicates higher image quality. This similarity check is easy and fast to calculate, however in practice it may turn out somewhat inconsistent with human eye perception. The *Structural Similarity* algorithm aims to correct this.

Structural Similarity

The *Structural Similarity Index Measure* (SSIM) [173] extracts three key features from an image: luminance, contrast and structure. Let's assume $\mathbf{x}$ and $\mathbf{y}$ are two spatially aligned, non-negative image signals. First, the luminance of each signal is compared. Assuming discrete signals, this is estimated as the mean intensity:

$$\mu_x = \frac{1}{N}\sum_{i=1}^{N} x_i \tag{4.6}$$

Moreover, the standard deviation of intensity is used as an estimate of the signal contrast. An unbiased estimate in discrete form is given by:

$$\sigma_x = \left(\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \mu_x)^2\right)^{\frac{1}{2}} \tag{4.7}$$

Third, each signal is normalized by its own standard deviation, so that the two signals being compared have unit standard deviation. The cosine angle between the normalized image vectors corresponds to the correlation coefficient:

$$\sigma_{xy} = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \mu_x)(y_i - \mu_y) \tag{4.8}$$

The luminance, contrast and structure comparison functions are defined as follows:

$$l(\mathbf{x},\mathbf{y}) = \frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1}, \qquad c(\mathbf{x},\mathbf{y}) = \frac{2\sigma_x\sigma_y + C_2}{\sigma_x^2 + \sigma_y^2 + C_2}, \qquad s(\mathbf{x},\mathbf{y}) = \frac{\sigma_{xy} + C_3}{\sigma_x\sigma_y + C_3} \tag{4.9}$$

where the constants C_1, C_2, C_3 are included to avoid instability when the respective denominators are very close to zero. Finally, by combining the above expressions, the SSIM index is calculated as follows:

$$\text{SSIM}(\mathbf{x},\mathbf{y}) = [l(\mathbf{x},\mathbf{y})]^{\alpha}[c(\mathbf{x},\mathbf{y})]^{\beta}[s(\mathbf{x},\mathbf{y})]^{\gamma} \tag{4.10}$$

where $\alpha, \beta, \gamma > 0$ are parameters used to adjust the relative importance of the three components. In order to evaluate image quality, this formula is usually applied only on luma, although it may also be applied on color (e.g., RGB) values or chromatic (e.g. YCbCr) values. The resultant SSIM is a decimal value between -1 and 1, where 1 indicates perfect similarity, 0 indicates no similarity, and -1 indicates perfect anti-correlation. In the original paper, the authors set $\alpha = \beta = \gamma = 1$ and $C_3 = C_2/2$.

Learned Perceptual Image Patch Similarity

The *Learned Perceptual Image Patch Similarity* (LPIPS) [181] is used to judge the perceptual similarity between two images. It is argued that widely used image quality metrics like SSIM and PSNR are simple and shallow functions that may fail to account for many nuances of human perception. The original paper introduces a new dataset of human perceptual similarity judgments to systematically evaluate deep features across different architectures and tasks and compare them with classic metrics. LPIPS essentially computes the similarity between the activations of two image patches $\mathbf{x}, \mathbf{x}_0$ for some predefined network $\mathcal{F}$. Feature stacks are extracted from L layers and are unit-normalized in the channel dimension, denoted as $\hat{\mathbf{y}}_l, \hat{\mathbf{y}}_{l,0} \in \mathbb{R}^{H_l \times W_l \times C_l}, \forall l$. The activations are scaled channel-wise by a vector $\boldsymbol{w}_l \in \mathbb{R}^{C_l}$ and then their ℓ_2 distance is computed. The exact distance formula is the following:

$$d(\mathbf{x},\mathbf{x}_0) = \sum_l \frac{1}{H_l W_l}\sum_{i,j}\|\boldsymbol{w}_l \odot (\boldsymbol{y}_l^{ij} - \boldsymbol{y}_{l,0}^{ij})\|_2^2 \tag{4.11}$$

This measure has been shown to match human perception well. A low LPIPS score means that image patches are perceptually similar.

4.2 Dataset

We trained our models and baselines from scratch on AffectNet [107]. It contains approximately 1M images retrieved from the Internet. Queries were performed on major search engines (Google, Bing, Yahoo), using 1,250 emotion-related keywords in six different languages. Notably, 440K of these images were annotated manually by 12 experts with basic discrete emotions (categorical model) and the intensity of valence and arousal (dimensional model). AffectNet is by far the largest database of facial expressions, valence, and arousal in the wild enabling research in automated facial expression recognition in two different emotion models. Out of the 440K manually annotated images, 291,651 images are labeled with 8 basic discrete emotions, i.e. neutral, happy, sad, surprised, scared, disgusted, angry and contemptuous. Out of those, 287,651 images belong to the training set and 4K images (500 for emotion label) belong to the validation set.

4.3 Emotion-Conditional Image Generation

Before moving onto the main subject of the current thesis, that is image manipulation, we are first required to actually train a diffusion model so as to learn the actual underlying distribution of the given dataset. Due to both time and resource constraints, we immediately shifted our attention towards LDMs due to the fact that the latter operate in latent, compressed space rather than the original high-dimensional image space. In this way, GPU memory requirements become tractable but at the cost of computational overhead for latent encoding/decoding.

4.3.1 Landmark Detection, Cropping & Face Alignment

The creators of the AffectNet dataset have provided precomputed face bounding boxes as well as facial landmark keypoint coordinates per image. However, we rather opted for manually locating both the bounding boxes as well as the facial landmarks from scratch, using FAN [18]. Before feeding the original images to FAN, we first resize them to 256×256 pixels in order to avoid running out of GPU memory. FAN locates the 2D coordinates for 68 facial landmarks. Based on those landmarks, bounding boxes of size 224×224 are center-cropped and then aligned by estimating the affine transform required to warp the set of detected facial keypoints to their canonical coordinates. Any empty image regions resulting from rotating the original image are dealt with by reflecting the image near the edges.

4.3.2 Image Compression

The first stage of any LDM consists of a pre-trained autoencoder that transfers any given input from image space to a compressed, latent space. To this end, we train a VQGAN on AffectNet. Following the notation that was introduced in section 3.4.1, we proceed by describing the chosen configuration.

The VQGAN follows the U-Net architecture consisting of a series of downsampling and upsampling residual blocks. The number of residual blocks per spatial resolution level was set equal to two. The number of basic channels was set equal to 128. The given input images were first resized to a resolution of 128×128 pixels. Thus, given input images $\mathbf{x} \in \mathbb{R}^{128 \times 128 \times 3}$, the encoder compresses the latter by a factor $f = 4$, resulting in latents $\hat{\mathbf{z}} \in \mathbb{R}^{32 \times 32 \times 3}$. Each latent undergoes element-wise quantization resulting in a collection of 32^2 codebook entries $\boldsymbol{z}_{ij} \in \mathbb{R}^3, i, j \in [32]$. The cardinality of the entire codebook is set equal to $|\mathcal{Z}| = 16384$. Moreover, the channel multipliers for the 2 intermediate spatial resolutions is set equal to 2 and 4 respectively. The last spatial resolution level also includes a spatial attention block. The VQGAN was trained for 6 epochs with the Adam [80] optimizer, an initial learning rate of 3.6×10^{-5} and a batch size of 8, reaching a reconstruction loss equal to 0.138 (sum of ℓ_1 and LPIPS distance) on the AffectNet validation set. The decoder's structure is exactly the same as the encoder's, but with the order of the blocks being reversed, resulting in approximately 61M trainable parameters.

4.3.3 Image Generation

Having trained encoder and decoder modules, we utilize them as the first stage of an LDM. In accordance with the previously discussed VQGAN, we proceed by describing the chosen configuration in the case of the LDM.

The underlying U-Net architecture features two residual blocks per spatial resolution level, with the number of basic channels being equal to 160. Input latents are further downsampled by a factor of $f = 4$. Attention blocks are placed at every spatial resolution level, i.e. $32^2, 16^2, 8^2$ and the channel multipliers are set equal to 1, 2 and 4 respectively. Each attention block is implemented according to the cross-attention mechanism described in section 3.4.2. The class label encoder is implemented as a single learnable embedding layer with a dimensionality of 512, mapping class labels y to $\boldsymbol{\tau_\theta}(y) \in \mathbb{R}^{1\times 512}$. All the above led to a total number of 156M trainable parameters.

In order to better control the intensity of the depicted emotions during the manipulation phase, we were obliged to train the LDM using *classifier-free guidance*. During training and with a probability $p_{\text{uncond}} = 0.2$, the input class labels are randomly dropped and replaced with a `null` label, effectively raising the total number of class labels to nine. The LDM was trained for around 30 epochs with the AdamW [98] optimizer, an initial learning rate of 2.4×10^{-5} and a batch size of 24. We used a linear noise schedule and $T_{\text{DDPM}} = 1,000$ diffusion steps.

Now is the time to actually evaluate the class conditional generative capabilities of the model. We came up with two evaluation settings. We first generated 5K samples per emotion class using DDIM sampling with $T_{\text{DDIM}} = 500$ steps and no unconditional guidance ($\gamma = 1.0$). For each set of generated samples, we calculated the evaluation metrics discussed in section 4.1.1 using the entire training dataset corresponding to the given emotion. Then, we randomly sampled 3.75K images from each emotion class and compared it to the previously mentioned combined set of 40K samples. In Tab. 4.1, we report FID, KID, precision and recall[6] metrics for the aforementioned evaluation scenarios.

Target Emotion	FID↓	KID ($\times 10^{-3}$) ↓	Precision↑	Recall↑
Neutral	6.64	2.1	0.57	0.67
Happy	6.90	2.7	0.55	0.66
Sad	7.73	2.1	0.58	0.66
Surprise	8.54	2.6	0.55	0.70
Fear	10.93	2.4	0.56	0.71
Disgust	11.72	2.3	0.56	0.73
Anger	8.03	2.7	0.57	0.66
Contempt	10.16	2.9	0.58	0.71
Mean[7]	8.83	2.48	0.57	0.69
Overall[8]	4.3	2.1	0.56	0.69

Table 4.1: Quantitative results for emotion conditional image generation on AffectNet using $T_{\text{DDIM}} = 500$ DDIM steps, $\eta = 1.0$ and *c.f.g.* scale $\gamma = 1.0$.

As expected, the best generation performance is observed for 'Neutral' and 'Happy' emotions, as they are the ones with the most training samples in the dataset. The highest FID values are observed for 'Fear' and 'Disgust'. Emotion-wise, fear is closely related to surprise while disgust is closely related to anger, as it can be seen in a 2D valence-arousal representation. Contempt is also often visually difficult to distinguish from happiness. Fig. 4.1 presents 1K uncurated samples, with 100 generated images per emotion class.

4.3.4 Nearest Neighbor Search

Next, before moving onto the emotion manipulation part, we took some time to further study the generation capabilities of our trained model. One important aspect of generative models is their ability to learn the underlying data distribution without actually memorizing individual training samples. Therefore, we randomly selected generated samples and found their 10 closest neighbors in

[6] FID, KID, precision and recall are calculated using the publicly available `torch-fidelity` package.
[7] This table row corresponds to the mean evaluation metrics across the 8 emotion classes.
[8] This evaluation was performed by comparing 40K samples (8K per emotion label) with 30K real samples (3.75K per emotion label).

(a) Neutral **(b)** Happy

(c) Sad **(d)** Surprise

(e) Fear **(f)** Disgust

(g) Anger **(h)** Contempt

Figure 4.1: Uncurated samples drawn with emotional conditional sampling using $T_{\text{DDIM}} = 500$ DDIM steps, $\eta = 1.0$ and *c.f.g.* scale $\gamma = 1.0$, from an LDM trained on AffectNet.

Figure 4.2: Nearest neighbors of our best AffectNet model, computed in Euclidean space in terms of pixel-wise *squared error*. The leftmost sample is from our model. The remaining samples in each row are its 10 nearest neighbors from the training set.

terms of pixel-wise squared error. The corresponding qualitative results are illustrated in Fig. 4.2. The leftmost column corresponds to the generated samples from our model, while the rest of the columns corresponds to their closest neighbors from the training set. In [125], they considered finding the nearest neighbors in VGG [144] feature space rather than pixel space.

4.3.5 Latent Interpolation

We can interpolate source images $\mathbf{x}_1, \mathbf{x}_2$ in latent space using the trained LDM. First we have to produce the corresponding two latent codes using the pre-trained encoder, resulting in $\mathbf{z}_1, \mathbf{z}_2$, respectively. Then we can use deterministic DDIM noising process ($\eta = 0$) on the input latents for T_{DDIM} steps, up to $t_0 < T_{\text{DDPM}}$, leading to $\mathbf{z}_1^{(t_0)}, \mathbf{z}_2^{(t_0)}$. Then we can apply *spherical linear interpolation* (Slerp) between the two noisy latent codes as follows:

$$\text{Slerp}(\mathbf{z}_1^{(t_0)}, \mathbf{z}_2^{(t_0)}, \lambda) = \frac{\sin(1-\lambda)\Omega}{\sin\Omega}\mathbf{z}_1^{(t_0)} + \frac{\sin\lambda\Omega}{\sin\Omega}\mathbf{z}_2^{(t_0)}, \qquad \cos\Omega = \mathbf{z}_1^{(t_0)} \cdot \mathbf{z}_2^{(t_0)}[9] \tag{4.12}$$

By applying either deterministic or stochastic ($\eta \neq 0$) reverse DDIM process on the interpolated noisy latent and decoding it with the first stage LDM decoder, we can visualize the intermediate results. Figs. 4.3 and 4.4 illustrate interpolation examples using $\eta \in \{0, 1\}$ in reverse DDIM process, having fixed $T_{\text{DDIM}} = 20$ and $t_0 = 500$. In both cases we used 10 uniform intepolation steps for each pair of original images. In the case of $\eta = 0$ we achieve nearly perfect reconstructions of the original images ($\lambda \in \{0, 1\}$), as expected, while the intermediate results are more consistent in between consecutive steps. Obviously, this is not the case when $\eta = 1$ were the intermediate results are more diverse but of equally high quality.

4.4 Image-Based Emotional Manipulation

4.4.1 Baseline Method

The baseline methodology towards image-based emotion manipulation follows the paradigm of LDEdit [19], adapted specifically for the task of emotion translation. The latter employed determin-

[9]The above formulation assumes unit-length elements of any arbitrary-dimensional inner product space, i.e. a vector space that also has an inner product).

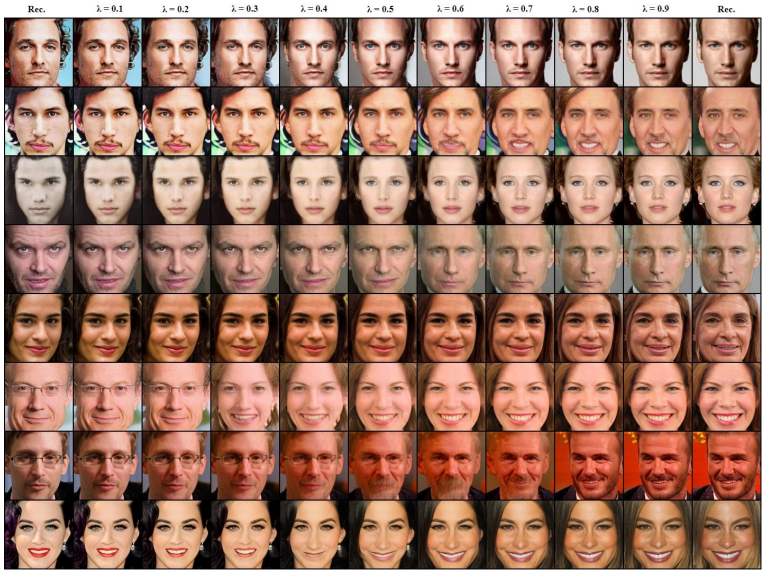


Figure 4.3: Curated examples of latent-space spherical linear interpolation on AffectNet, using $T_{\text{DDIM}} = 20$ steps, $\eta = 0$ and $t_0 = 500$.

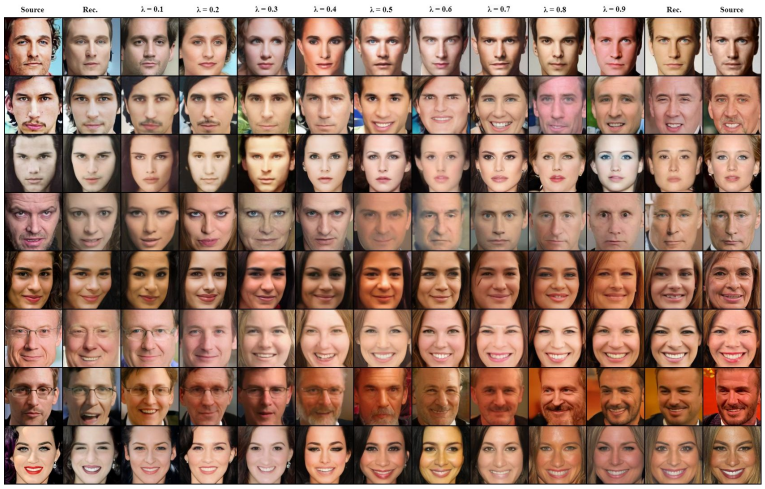


Figure 4.4: Curated examples of latent-space spherical linear interpolation on AffectNet, using $T_{\text{DDIM}} = 20$ steps, $\eta = 1$ and $t_0 = 500$.

istic forward diffusion in a lower dimensional latent space, conditioned on the output of a transformer tokenizer that was pre-trained on a large-scale text-image dataset (e.g. LAION-400M [139]), achieving zero-shot text-guided manipulation. In our case, the transformer is replaced with a class-embedder that maps each of the 8 possible emotion labels to a learnable embedding vector $\boldsymbol{\tau}_{\boldsymbol{\theta}}(y) \in \mathbb{R}^{1\times 512}$.

The backbone of this approach resides in the near cycle-consistency induced by the deterministic forward and backward DDIM diffusion processes. This is further demonstrated by the following analysis. The DDIM sampling process described in section 3.2, can be rewritten in the following form:

$$\mathbf{z}_{t-1} = \sqrt{\bar{a}_{t-1}}\left(\frac{\mathbf{z}_t - \sqrt{1-\bar{a}_t}\epsilon_{\boldsymbol{\theta}}(\mathbf{z}_t, t, \boldsymbol{\tau}_{\boldsymbol{\theta}}(y))}{\sqrt{\bar{a}_t}}\right) + \sqrt{1-\bar{a}_{t-1}}\epsilon_{\boldsymbol{\theta}}(\mathbf{z}_t, t, \boldsymbol{\tau}_{\boldsymbol{\theta}}(y)) \tag{4.13}$$

$$\sqrt{\frac{1}{\bar{a}_{t-1}}}\mathbf{z}_{t-1} - \sqrt{\frac{1}{\bar{a}_t}}\mathbf{z}_t = \left(\sqrt{\frac{1}{\bar{a}_{t-1}} - 1} - \sqrt{\frac{1}{\bar{a}_t} - 1}\right)\epsilon_{\boldsymbol{\theta}}(\mathbf{z}_t, t, \boldsymbol{\tau}_{\boldsymbol{\theta}}(y)) \tag{4.14}$$

Figure 4.5: Curated examples of latent emotion manipulation on the AffectNet validation set, using $T_{\text{DDIM}} = 40$ DDIM steps, $\eta = 0$, editing strength $t_0 = 500$ and *c.f.g* scale $\gamma = 3.0$.

If we set $\mathbf{y}_t := \sqrt{1/\bar{a}_t}\mathbf{z}_t$ and $p_t := \sqrt{1/\bar{a}_t - 1}$, Eq. (4.14) can be written as:

$$\mathbf{y}_{t-1} - \mathbf{y}_t = (p_{t-1} - p_t)\epsilon_{\boldsymbol{\theta}}(\mathbf{z}_t, t, \boldsymbol{\tau}_{\boldsymbol{\theta}}(y)) \Rightarrow \mathrm{d}\mathbf{y}_t = \epsilon_{\boldsymbol{\theta}}(\mathbf{z}_t, t, \boldsymbol{\tau}_{\boldsymbol{\theta}}(y))\mathrm{d}p_t \tag{4.15}$$

The last equation describes an ODE that moves backward in terms of the diffusion time index t. Reversal of this ODE leads to the forward DDIM process:

$$\mathbf{y}_{t+1} - \mathbf{y}_t = (p_{t+1} - p_t)\epsilon_{\boldsymbol{\theta}}(\mathbf{z}_t, t, \boldsymbol{\tau}_{\boldsymbol{\theta}}(y)) \tag{4.16}$$

By setting the following:

$$\mathbf{f}_{\boldsymbol{\theta}}(\mathbf{z}_t, t, \boldsymbol{\tau}_{\boldsymbol{\theta}}(y)) := \frac{\mathbf{z}_t - \sqrt{1 - \bar{a}_t}\epsilon_{\boldsymbol{\theta}}(\mathbf{z}_t, t, \boldsymbol{\tau}_{\boldsymbol{\theta}}(y))}{\sqrt{\bar{a}_t}} \tag{4.17}$$

we get the forward/backward DDIM processes for LDM-based manipulation w.r.t. emotion labels

Figure 4.6: Curated examples of LDM-based emotion manipulation on the AffectNet validation set, using $T_{\text{DDIM}} = 40$ steps, $\eta = 0$, *c.f.g* scale $\gamma = 3.0$ and variable editing strength $t_0 \in \{400, 500, 600\}$.

y_{src} and y_{trg}:

$$\mathbf{z}_{t+1} = \sqrt{\bar{a}_{t+1}}\mathbf{f}_{\boldsymbol{\theta}}(\mathbf{z}_t, t, \boldsymbol{\tau}_{\boldsymbol{\theta}}(y_{\text{src}})) + \sqrt{1-\bar{a}_{t+1}}\epsilon_{\boldsymbol{\theta}}(\mathbf{z}_t, t, \boldsymbol{\tau}_{\boldsymbol{\theta}}(y_{\text{src}})) \tag{4.18}$$

$$\hat{\mathbf{z}}_{t-1} = \sqrt{\bar{a}_{t-1}}\mathbf{f}_{\boldsymbol{\theta}}(\hat{\mathbf{z}}_t, t, \boldsymbol{\tau}_{\boldsymbol{\theta}}(y_{\text{trg}})) + \sqrt{1-\bar{a}_{t-1}}\epsilon_{\boldsymbol{\theta}}(\hat{\mathbf{z}}_t, t, \boldsymbol{\tau}_{\boldsymbol{\theta}}(y_{\text{trg}})) \tag{4.19}$$

During the manipulation processes, the LDM weights $\boldsymbol{\theta}$ remain frozen in accordance with our best model checkpoint from the initial training phase. Thus, the baseline method does not introduce any new trainable parameters. Given an input image $\mathbf{x}_{\text{src}}$, the first stage encoder transforms it into its corresponding latent representation $\mathbf{z}_0$. Then deterministic ($\eta = 0$) forward diffusion is performed up to timestep $t_0 < T_{\text{DDPM}} = 1000$, conditioned on the source emotion label y_{src}, denoted as $\mathbf{z}_{t_0}$. The reverse process conditioned on the target emotion label y_{trg} initiates from the same noised latent code $\mathbf{z}_{t_0}$ with the aim of reconstructing $\hat{\mathbf{z}}_0$. Passing the manipulated latent code $\hat{\mathbf{z}}_0$ through the decoder results in the reconstructed target image $\mathbf{x}_{\text{gen}}$. At this point we ought to mention that the DDIM process can be significantly sped up by using fewer discretization steps $\{\tau_s\}_{s=1}^{T_{\text{DDIM}}}$ evenly selected in the range $[1, t_0]$ such that $\tau_1 = 1$ and $\tau_{T_{\text{DDIM}}} = t_0$. Obviously, the lower the number of intermediate steps T_{DDIM}, the worse the reconstruction quality of the DDIM process. However, lowering the number of sampling steps is essential in terms of speeding up experiments and it has been found that

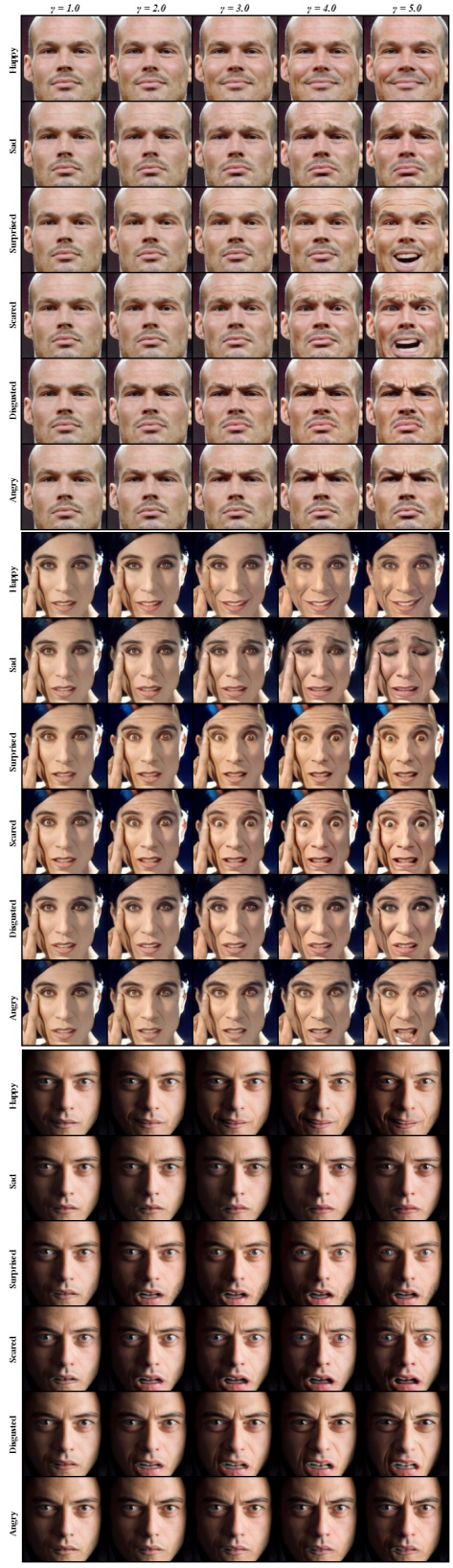


Figure 4.7: Curated examples of LDM-based emotion manipulation on the AffectNet validation set, using $T_{\text{DDIM}} = 40$ steps, $\eta = 0$, editing strength $t_0 = 500$ and variable *c.f.g.* scale $\gamma \in \{1.0, 2.0, 3.0, 4.0, 5.0\}$.

$T_{\text{DDIM}} \geq 20$ provides satisfactory results in terms of image-to-image translation.

Important hyperparameters of the aforementioned manipulation process are the stochasticity parameter η, the editing strength t_0, the number of steps T_{DDIM} and the unconditional guidance scale γ. We begin with by setting $\eta = 0$ for all of the upcoming experiments. In general, values $\eta > 0$ can produce diverse outputs, reducing the consistency with the original image. In the context of facial expression manipulation, identity and facial attribute preservation of the depicted subjects is of primary importance and as the current baseline methodology does not include any other form of supervision upon the latter, we find ourselves constrained to use $\eta = 0$. Moreover, we chose to name t_0 as the editing strength hyperparameter due to the observation that the higher its value, the more prominent the manipulation effect becomes, compared to the original image. Lastly, as we chose to train the LDM using *classifier-free guidance*, we have one extra degree of freedom regarding the intensity of the produced manipulation in the form of the unconditional guidance scale γ.

Fig. 4.5 illustrates curated examples of emotion manipulation on images from the AffectNet validation set, using $T_{\text{DDIM}} = 20$, $\gamma = 3.0$ and $t_0 = 500$. The leftmost image in each column is an original sample drawn from the dataset, while the rest six columns include the manipulation results for each one of the 6 basic expressions of *happiness*, *sadness*, *surprise*, *fear*, *disgust* and *anger*. We can immediately notice that the results in the 'happy', 'sad' and 'angry' columns are overall satisfactory, the emotions of surprise, fear and disgust are more difficult to convey. This can be partly justified by the fact that the aforementioned emotions are underrepresented in the AffectNet training set. Furthermore, Fig. 4.6 illustrates the effect of varying the editing strength $t_0 \in \{400, 500, 600\}$ in the case of all six different target emotions, while γ and T_{DDIM} are fixed. One can easily notice that $t_0 = 400$ is not sufficient for producing emotionally distinguishable results. On the other end, a value of $t_0 = 600$ results in better manipulation results but at the cost of subject identity distortion. Lastly, Fig. 4.7 illustrates the effect of varying the unconditional guidance scale $\gamma \in \{1.0, 2.0, 3.0, 4.0, 5.0\}$ in the case of all six different target emotions, while $t_0 = 500$ and T_{DDIM} are fixed. It is evident that in the specific setting of $t_0 = 500$, lower values of γ have little to no effect in changing the emotion of a given subject. Better results can be obtained with $\gamma \in \{4.0, 5.0\}$ with minimal distortion of peripheral facial features and subject identity.

Detailed quantitative results are presented in Tab. 4.2 and 4.3. The quality of the produced images and consistency with the original ones is measured in terms of PSNR, SSIM and LPIPS[10]. We also employ HSEmotion[11] [135–137], a state-of-the-art lightweight emotion recognition framework for evaluating the emotion transfer capabilities of our baseline methodology. In Tab. 4.2, we notice that the most difficult expressions, as far as emotion transfer is concerned, are (according to HSEmotion) surprise and disgust, while the easiest one is happiness. More specifically, a maximum emotion accuracy of 0.961 is achieved in terms of happiness, while 0.731 and 0.734 for surprise and disgust, respectively. As expected, the highest recognition accuracy for all six emotions is achieved using $\gamma = 5.0$ and $t_0 = 600$. This setting also results in the lowest image quality, that regardless of the target emotion, revolves around 22.00 dB PSNR, 0.730 SSIM and 0.150 LPIPS. Tab. 4.3, quantifies the effect of varying the number of steps T_{DDIM} in terms of the three aforementioned image quality metrics and recognition accuracy. For $t_0 = 400$ and regardless of the target emotion, increasing T_{DDIM} leads to either no change or a small drop in recognition performance. This is not the case for higher $t_0 \in \{500, 600\}$. PSNR always increases in accordance with the increase in sampling steps. SSIM behaves differently, as it increases along with T_{DDIM} when $t_0 \in \{400, 500\}$ and decreases when $t_0 = 600$. A similar counter-intuitive behavior is noticed in the case of LPIPS which gradually increases, as the number of DDIM steps increases, when $t_0 \in \{500, 600\}$.

Why does this work? An intuitive explanation of why this works is the following: DMs are trained by asking them to denoise corrupted images by predicting the added noise, along with the corresponding noisy pixel locations, across all image channels. In that way, DMs end up learning to synthesize images out of pure noise. Therefore, the actual learnt embedding space of a DM lies within the noise itself. Moreover, given that the forward DDIM noising process is deterministic, there is a 1-1 correspondence between each (image, y_{src}) pair and its noisy latent representation, at some arbitrary noise level t_0, given a fixed pre-trained denoising network $\boldsymbol{\epsilon_\theta}$. The deterministic forward process is also

[10] PSNR, SSIM and LPIPS are calculated using the publicly available `piq` package.

[11] Ranked #1 in 8-emotion classification accuracy on AffectNet, according to https://paperswithcode.com/sota/facial-expression-recognition-on-affectnet

y_{target}	*c.f.g.* Scale (γ)	Str. (t_0)	Image Quality			Emotion Recognition Accuracy w/ HSEmotion [135–137]						
			PSNR↑	SSIM↑	LPIPS↓	Neutral	Happy	Sad	Surprised	Disgusted	Disgusted	Angry
Happiness	1.0	400	35.25	0.966	0.013	0.24	0.248	0.076	0.146	0.093	0.075	0.122
	2.0		33.15	0.95	0.021	0.254	0.344	0.047	0.154	0.058	0.052	0.092
	3.0		31.71	0.937	0.029	0.244	0.447	0.028	0.148	0.035	0.032	0.066
	4.0		30.59	0.924	0.037	0.216	0.537	0.018	0.138	0.023	0.021	0.047
	5.0		29.66	0.911	0.045	0.179	0.626	0.01	0.121	0.016	0.018	0.031
	1.0	500	32.09	0.939	0.03	0.223	0.364	0.057	0.138	0.068	0.06	0.09
	2.0		29.53	0.908	0.049	0.191	0.58	0.02	0.115	0.025	0.025	0.043
	3.0		27.92	0.883	0.065	0.13	0.728	0.01	0.082	0.012	0.018	0.019
	4.0		26.75	0.861	0.078	0.084	0.819	0.006	0.066	0.007	0.009	0.01
	5.0		25.80	0.841	0.090	0.056	0.872	0.003	0.05	0.004	0.007	0.007
	1.0	600	28.39	0.889	0.062	0.163	0.566	0.035	0.103	0.036	0.044	0.052
	2.0		25.68	0.839	0.093	0.079	0.821	0.007	0.052	0.011	0.018	0.012
	3.0		24.13	0.803	0.113	0.038	0.916	0.003	0.027	0.004	0.008	0.004
	4.0		23.00	0.773	0.129	0.022	0.945	0.002	0.02	0.002	0.005	0.003
	5.0		22.04	0.747	0.143	0.014	0.961	0.002	0.015	0.002	0.003	0.003
Sadness	1.0	400	35.39	0.967	0.012	0.202	0.144	0.174	0.107	0.13	0.099	0.146
	2.0		33.24	0.952	0.020	0.188	0.092	0.27	0.079	0.144	0.094	0.135
	3.0		31.68	0.937	0.029	0.159	0.052	0.393	0.052	0.138	0.083	0.123
	4.0		30.41	0.921	0.039	0.129	0.03	0.507	0.036	0.13	0.064	0.104
	5.0		29.33	0.906	0.049	0.098	0.014	0.609	0.022	0.121	0.05	0.086
	1.0	500	32.72	0.946	0.025	0.212	0.104	0.23	0.091	0.131	0.092	0.14
	2.0		30.08	0.917	0.044	0.179	0.038	0.408	0.048	0.126	0.077	0.124
	3.0		28.28	0.890	0.061	0.128	0.014	0.573	0.027	0.106	0.052	0.1
	4.0		26.89	0.864	0.078	0.084	0.009	0.696	0.014	0.086	0.034	0.077
	5.0		25.71	0.837	0.095	0.06	0.006	0.774	0.008	0.067	0.025	0.06
	1.0	600	29.85	0.913	0.047	0.213	0.064	0.296	0.075	0.126	0.082	0.144
	2.0		26.95	0.867	0.078	0.142	0.014	0.549	0.032	0.099	0.055	0.109
	3.0		25.06	0.827	0.103	0.09	0.008	0.712	0.017	0.069	0.029	0.076
	4.0		23.59	0.789	0.126	0.052	0.005	0.806	0.008	0.051	0.02	0.058
	5.0		22.35	0.752	0.146	0.036	0.005	0.849	0.005	0.04	0.016	0.049
Surprise	1.0	400	35.45	0.967	0.012	0.205	0.183	0.089	0.184	0.126	0.079	0.134
	2.0		33.38	0.953	0.020	0.204	0.19	0.063	0.252	0.126	0.053	0.112
	3.0		31.87	0.940	0.028	0.191	0.18	0.048	0.337	0.11	0.04	0.094
	4.0		30.64	0.927	0.037	0.168	0.173	0.035	0.418	0.105	0.028	0.074
	5.0		29.58	0.913	0.046	0.135	0.167	0.027	0.482	0.106	0.024	0.059
	1.0	500	32.58	0.945	0.026	0.209	0.165	0.072	0.242	0.137	0.063	0.112
	2.0		29.90	0.916	0.045	0.174	0.148	0.044	0.398	0.12	0.034	0.082
	3.0		28.10	0.889	0.063	0.127	0.136	0.027	0.522	0.115	0.024	0.048
	4.0		26.70	0.863	0.079	0.091	0.125	0.017	0.603	0.12	0.014	0.03
	5.0		25.54	0.838	0.094	0.066	0.114	0.014	0.658	0.12	0.008	0.022
	1.0	600	29.43	0.907	0.051	0.191	0.131	0.064	0.335	0.143	0.049	0.088
	2.0		26.44	0.856	0.084	0.191	0.131	0.064	0.335	0.143	0.049	0.088
	3.0		24.45	0.813	0.110	0.078	0.085	0.016	0.662	0.133	0.008	0.018
	4.0		22.96	0.774	0.133	0.048	0.092	0.008	0.714	0.122	0.006	0.012
	5.0		21.73	0.738	0.152	0.036	0.1	0.006	0.731	0.115	0.004	0.009
Fear	1.0	400	35.33	0.967	0.012	0.164	0.15	0.122	0.148	0.174	0.093	0.148
	2.0		33.12	0.952	0.020	0.124	0.108	0.13	0.149	0.272	0.088	0.13
	3.0		31.47	0.937	0.030	0.087	0.071	0.124	0.14	0.388	0.081	0.109
	4.0		30.11	0.921	0.040	0.053	0.048	0.115	0.13	0.498	0.066	0.09
	5.0		28.94	0.904	0.051	0.033	0.032	0.098	0.113	0.598	0.057	0.068
	1.0	500	32.57	0.946	0.025	0.151	0.113	0.118	0.166	0.222	0.093	0.136
	2.0		29.75	0.915	0.045	0.085	0.058	0.117	0.17	0.396	0.066	0.109
	3.0		27.75	0.884	0.065	0.043	0.033	0.092	0.148	0.562	0.05	0.072
	4.0		26.19	0.854	0.084	0.02	0.022	0.072	0.112	0.692	0.035	0.048
	5.0		24.89	0.824	0.103	0.012	0.017	0.054	0.093	0.764	0.028	0.033
	1.0	600	29.54	0.910	0.049	0.133	0.073	0.114	0.195	0.291	0.073	0.122
	2.0		26.26	0.855	0.085	0.052	0.024	0.089	0.18	0.541	0.044	0.071
	3.0		24.12	0.806	0.115	0.016	0.016	0.062	0.136	0.708	0.023	0.039
	4.0		22.49	0.760	0.141	0.009	0.014	0.045	0.096	0.794	0.017	0.024
	5.0		21.16	0.717	0.164	0.008	0.012	0.038	0.081	0.832	0.013	0.016
Disgust	1.0	400	35.38	0.967	0.012	0.177	0.151	0.111	0.122	0.122	0.144	0.173
	2.0		33.18	0.952	0.020	0.149	0.106	0.108	0.1	0.116	0.224	0.197
	3.0		31.55	0.937	0.030	0.115	0.073	0.098	0.074	0.111	0.317	0.212
	4.0		30.21	0.921	0.040	0.094	0.05	0.082	0.056	0.094	0.413	0.211
	5.0		29.07	0.904	0.051	0.065	0.032	0.076	0.041	0.08	0.509	0.196
	1.0	500	32.76	0.947	0.024	0.17	0.122	0.112	0.104	0.11	0.19	0.192
	2.0		30.10	0.919	0.043	0.119	0.064	0.088	0.069	0.099	0.335	0.226
	3.0		28.21	0.891	0.062	0.073	0.04	0.072	0.04	0.073	0.478	0.224
	4.0		26.75	0.863	0.081	0.044	0.029	0.057	0.027	0.055	0.594	0.194
	5.0		25.50	0.835	0.100	0.029	0.025	0.042	0.017	0.043	0.676	0.168
	1.0	600	29.91	0.915	0.047	0.15	0.095	0.097	0.083	0.099	0.247	0.229
	2.0		26.95	0.869	0.078	0.082	0.042	0.069	0.043	0.069	0.452	0.243
	3.0		24.98	0.826	0.106	0.038	0.033	0.047	0.02	0.052	0.593	0.216
	4.0		23.49	0.787	0.131	0.021	0.034	0.036	0.012	0.039	0.682	0.176
	5.0		22.24	0.750	0.154	0.015	0.036	0.028	0.009	0.031	0.734	0.147
Anger	1.0	400	35.57	0.968	0.011	0.203	0.146	0.102	0.119	0.115	0.105	0.21
	2.0		33.48	0.954	0.019	0.188	0.102	0.091	0.104	0.1	0.124	0.291
	3.0		31.94	0.940	0.027	0.168	0.071	0.085	0.084	0.088	0.132	0.372
	4.0		30.70	0.926	0.036	0.146	0.044	0.076	0.072	0.073	0.141	0.448
	5.0		29.63	0.912	0.045	0.127	0.03	0.068	0.057	0.064	0.142	0.512
	1.0	500	32.76	0.946	0.025	0.201	0.102	0.098	0.104	0.101	0.125	0.269
	2.0		30.12	0.918	0.043	0.168	0.041	0.077	0.076	0.071	0.13	0.436
	3.0		28.30	0.891	0.061	0.127	0.021	0.063	0.05	0.054	0.128	0.558
	4.0		26.86	0.865	0.077	0.091	0.012	0.048	0.04	0.042	0.117	0.651
	5.0		25.66	0.840	0.094	0.072	0.008	0.039	0.025	0.037	0.108	0.71
	1.0	600	29.63	0.910	0.049	0.191	0.058	0.088	0.085	0.084	0.129	0.364
	2.0		26.60	0.861	0.083	0.123	0.01	0.059	0.042	0.052	0.119	0.595
	3.0		24.67	0.819	0.109	0.078	0.006	0.043	0.025	0.032	0.097	0.72
	4.0		23.17	0.781	0.132	0.046	0.007	0.028	0.017	0.023	0.086	0.794
	5.0		21.88	0.744	0.155	0.033	0.008	0.019	0.013	0.017	0.072	0.838

Table 4.2: LDM-based emotion manipulation on the AffectNet validation set, using *forward*/*backward* DDIM processes with $T_{\text{DDIM}} = 40$ steps, variable *c.f.g.* scale $\gamma \in \{1.0, 2.0, 3.0, 4.0, 5.0\}$ and editing strength $t_0 \in \{400, 500, 600\}$.

y_{target}	DDIM Steps (T_{DDIM})	Str. (t_0)	Image Quality			Emotion Recognition Accuracy w/ HSEmotion [135–137]						
			PSNR↑	SSIM↑	LPIPS↓	Neutral	Happy	Sad	Surprised	Scared	Disgusted	Angry
Happiness	20	400	31.25	0.934	0.031	0.238	0.451	0.028	0.15	0.035	0.033	0.065
	40		31.71	0.937	0.029	0.244	0.447	0.028	0.148	0.035	0.032	0.066
	80		32.04	0.938	0.029	0.245	0.446	0.03	0.15	0.034	0.031	0.065
	20	500	27.67	0.883	0.064	0.133	0.712	0.011	0.094	0.013	0.015	0.022
	40		27.92	0.883	0.065	0.13	0.728	0.01	0.082	0.012	0.018	0.019
	80		28.18	0.883	0.065	0.127	0.734	0.011	0.081	0.012	0.016	0.019
	20	600	24.11	0.811	0.107	0.045	0.905	0.002	0.031	0.004	0.007	0.005
	40		24.13	0.803	0.113	0.038	0.916	0.003	0.027	0.004	0.008	0.004
	80		24.28	0.800	0.115	0.037	0.919	0.003	0.026	0.004	0.007	0.005
Sadness	20	400	31.18	0.933	0.031	0.157	0.052	0.398	0.051	0.141	0.081	0.119
	40		31.68	0.937	0.029	0.159	0.052	0.393	0.052	0.138	0.083	0.123
	80		32.02	0.938	0.028	0.159	0.054	0.393	0.053	0.139	0.08	0.122
	20	500	27.97	0.889	0.061	0.122	0.016	0.562	0.029	0.111	0.055	0.104
	40		28.28	0.890	0.061	0.128	0.014	0.573	0.027	0.106	0.052	0.1
	80		28.56	0.891	0.061	0.128	0.015	0.572	0.027	0.105	0.052	0.102
	20	600	25.00	0.833	0.097	0.089	0.006	0.694	0.015	0.082	0.032	0.082
	40		25.06	0.827	0.103	0.09	0.008	0.712	0.017	0.069	0.029	0.076
	80		25.23	0.824	0.106	0.089	0.008	0.716	0.016	0.066	0.029	0.076
Surprise	20	400	31.37	0.937	0.030	0.186	0.18	0.046	0.34	0.112	0.042	0.096
	40		31.87	0.940	0.028	0.191	0.18	0.048	0.337	0.11	0.04	0.094
	80		32.22	0.942	0.027	0.188	0.182	0.048	0.339	0.11	0.041	0.092
	20	500	27.82	0.888	0.062	0.129	0.137	0.029	0.512	0.117	0.021	0.054
	40		28.10	0.889	0.063	0.127	0.136	0.027	0.522	0.115	0.024	0.048
	80		28.36	0.890	0.063	0.127	0.138	0.026	0.525	0.114	0.021	0.048
	20	600	24.47	0.821	0.104	0.078	0.086	0.017	0.661	0.128	0.01	0.021
	40		24.45	0.813	0.110	0.078	0.085	0.016	0.662	0.133	0.008	0.018
	80		24.60	0.810	0.113	0.075	0.083	0.014	0.67	0.131	0.008	0.018
Fear	20	400	30.99	0.933	0.031	0.084	0.068	0.126	0.144	0.389	0.08	0.108
	40		31.47	0.937	0.030	0.087	0.071	0.124	0.14	0.388	0.081	0.109
	80		31.82	0.938	0.029	0.087	0.072	0.121	0.142	0.388	0.08	0.11
	20	500	27.46	0.883	0.065	0.043	0.034	0.098	0.14	0.562	0.053	0.072
	40		27.75	0.884	0.065	0.043	0.033	0.092	0.148	0.562	0.05	0.072
	80		28.03	0.885	0.065	0.042	0.032	0.094	0.146	0.564	0.05	0.072
	20	600	24.09	0.813	0.109	0.018	0.016	0.068	0.133	0.696	0.027	0.041
	40		24.12	0.806	0.115	0.016	0.016	0.062	0.136	0.708	0.023	0.039
	80		24.29	0.804	0.117	0.016	0.016	0.061	0.134	0.715	0.021	0.038
Disgust	20	400	31.09	0.934	0.031	0.116	0.072	0.099	0.074	0.11	0.316	0.214
	40		31.55	0.937	0.030	0.115	0.073	0.098	0.074	0.111	0.317	0.212
	80		31.88	0.938	0.029	0.115	0.073	0.097	0.077	0.108	0.317	0.213
	20	500	27.92	0.890	0.061	0.079	0.041	0.071	0.04	0.079	0.466	0.224
	40		28.21	0.891	0.062	0.073	0.04	0.072	0.04	0.073	0.478	0.224
	80		28.48	0.891	0.062	0.074	0.041	0.072	0.04	0.072	0.481	0.219
	20	600	24.94	0.833	0.100	0.041	0.032	0.05	0.025	0.051	0.579	0.222
	40		24.98	0.826	0.106	0.038	0.033	0.047	0.02	0.052	0.593	0.216
	80		25.15	0.824	0.108	0.038	0.036	0.05	0.019	0.048	0.596	0.212
Anger	20	400	31.42	0.937	0.029	0.169	0.069	0.083	0.082	0.088	0.131	0.377
	40		31.94	0.940	0.027	0.168	0.071	0.085	0.084	0.088	0.132	0.372
	80		32.30	0.942	0.026	0.17	0.07	0.083	0.083	0.088	0.133	0.374
	20	500	27.98	0.890	0.060	0.125	0.02	0.064	0.053	0.055	0.128	0.554
	40		28.30	0.891	0.061	0.127	0.021	0.063	0.05	0.054	0.128	0.558
	80		28.59	0.892	0.061	0.13	0.02	0.061	0.051	0.052	0.125	0.562
	20	600	24.60	0.825	0.103	0.078	0.006	0.043	0.026	0.033	0.105	0.709
	40		24.67	0.819	0.109	0.078	0.006	0.043	0.025	0.032	0.097	0.72
	80		24.83	0.817	0.112	0.075	0.005	0.041	0.024	0.029	0.1	0.725

Table 4.3: LDM-based emotion manipulation on the AffectNet validation set, using *forward*/*backward* DDIM processes with *c.f.g.* scale $\gamma = 3.0$, variable number of steps $T_{\text{DDIM}} \in \{20, 40, 80\}$ and editing strength $t_0 \in \{400, 500, 600\}$.

responsible for encoding general identity specific characteristics of each depicted subject that ought to be kept intact throughout the manipulation process. Deterministic decoding conditioned on y_{trg} is expected to maintain all peripheral/appearance-related features and only manipulate facial expression characteristics so as to match the target conditioning. In this case, learnt emotion embeddings encode all the necessary semantic information relative to each emotion, within the deterministic DDIM setting (similar to $\mathbf{z}_{\text{sem}}$ of the DiffAE framework [119]).

4.5 CLIP-Guided Finetuning

4.5.1 CLIP

Contrastive Language-Image Pre-training (CLIP) [121], constitutes an efficient method of image representation learning with natural language supervision and can be considered as a simplified version of the ConVIRT [182] model, trained from scratch. CLIP jointly trains an image encoder and a text encoder to predict the correct pairings of a batch of (image, text) training examples. At test time the learned text encoder synthesizes a zero-shot linear classifier by embedding the names or descriptions of the target dataset's classes.

For pre-training, given a batch of N (image, text) pairs, CLIP is trained to predict which of the $N \times N$ possible (image, text) pairings across a batch actually occurred. CLIP learns a multi-modal embedding space by jointly training an image encoder and text encoder to maximize the cosine similarity of the image and text embeddings of the N real pairs in the batch while minimizing the

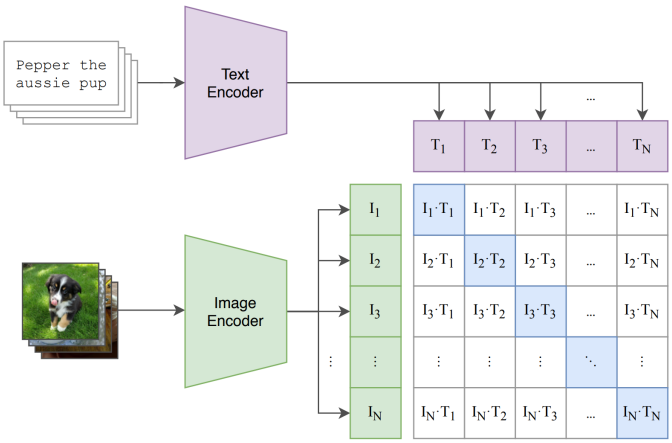


Figure 4.8: Overview of the CLIP model. Source: [121]

cosine similarity of the embeddings of the $N^2 - N$ incorrect pairings. A symmetric cross entropy loss is optimized over these similarity scores. An high-level view of the CLIP framework is illustrated in Fig. 4.8.

4.5.2 CLIP Guidance for Emotional Manipulation

To effectively extract knowledge from CLIP, two different losses have been proposed: a global target loss [115], and local directional loss [52]. Both of the aforementioned loss types were initially proposed for direct optimization of intermediate latent representations in StyleGAN's [75] $\mathcal{W}+$ space. The global CLIP loss tries to minimize the cosine distance in the CLIP space between the generated image and a given target text as follows:

$$\mathcal{L}_{\text{global}}(\mathbf{x}_{\text{gen}}, y_{\text{trg}}) = 1 - \cos\langle \text{CLIP}_{\text{img}}(\mathbf{x}_{\text{gen}}), \text{CLIP}_{\text{text}}(y_{\text{trg}}) \rangle \tag{4.20}$$

where y_{trg} is a text description of a target and $\mathbf{x}_{\text{gen}}$ denotes the generated image. On the other hand, the local directional loss is designed to alleviate the issues of global CLIP loss such as low diversity and susceptibility to adversarial attacks. The local directional CLIP loss induces the direction between the embeddings of the reference and generated images to be aligned with the direction between the embeddings of a pair of reference and target texts in the CLIP space as follows:

$$\mathcal{L}_{\text{dir}}(\mathbf{x}_{\text{gen}}, y_{\text{trg}}, \mathbf{x}_{\text{src}}, y_{\text{src}}) = 1 - \cos\langle \text{CLIP}_{\text{img}}(\mathbf{x}_{\text{gen}}) - \text{CLIP}_{\text{img}}(\mathbf{x}_{\text{src}}), \text{CLIP}_{\text{text}}(y_{\text{trg}}) - \text{CLIP}_{\text{text}}(y_{\text{src}}) \rangle \tag{4.21}$$

The manipulated images guided by the directional CLIP loss are known robust to mode-collapse issues because by aligning the direction between the image representations with the direction between the reference text and the target text, distinct images should be generated.

DiffusionCLIP [76] adapted the directional CLIP loss to the DDPM framework and achieved zero-shot image translation between unseen domains as well as image editing. Aside from the directional CLIP loss, when operating on human face images, they additionally employed both an identity-preserving and ℓ_1 loss. Before the application of CLIP guidance, training of a DM $\epsilon_{\boldsymbol{\theta}}$ is required. Then, any given input image $\mathbf{x}_{\text{src}}$ was converted to a noisy latent $\mathbf{x}_{\text{src}}^{(t_0)}$. Subsequently, guided by the CLIP loss, the DM, at the reverse path, was finetuned to generate samples driven by the target text y_{trg}. The deterministic forward and reverse processes ($\eta = 0$) were based on DDIM.

The identity preservation loss is implemented as the cosine distance in the embedding space of a pre-trained face recognition network. More specifically, an IR-SE50[12] model coupled with ArcFace [32] is employed. The authors of [76] used a ℓ_1 loss term but we opted for using its ℓ_2 counterpart, similar to the original implementations [52, 115]. The final combined loss objective takes the following form:

$$\mathcal{L}(\mathbf{x}_{\text{gen}}, y_{\text{trg}}, \mathbf{x}_{\text{src}}, y_{\text{src}}) = \lambda_{\text{dir}} \mathcal{L}^*_{\text{dir}}(\mathbf{x}_{\text{gen}}, y_{\text{trg}}, \mathbf{x}_{\text{src}}, y_{\text{src}}) + \lambda_{\text{id}} \mathcal{L}_{\text{id}}(\mathbf{x}_{\text{gen}}, \mathbf{x}_{\text{src}}) + \lambda_{\ell_2} \|\mathbf{x}_{\text{gen}} - \mathbf{x}_{\text{src}}\|_2^2 \tag{4.22}$$

[12] This model is basically a combination of a 50-layer ResNet [62] with SENet [68] blocks.

where:

$$\mathcal{L}^{*}_{\text{dir}} = -\log\left(\frac{2-\mathcal{L}_{\text{dir}}}{2}\right) \in [0, +\infty) \tag{4.23}$$

The main two differentiating factors between the original DiffusionCLIP and our adaptation are the following: i) The original one used a DDPM as its backbone while we use a LDM. ii) The original one did not take advantage of source class-label information when applying the directional CLIP loss, as the majority of experiments were performed in unconditional settings. It is worth mentioning that in the original paper, a requirement of 24 GB GPU memory is reported for the full optimization configuration of DiffusionCLIP. This could be reduced to 12 GB when using a memory-efficient implementation of the main optimization loop. On the other hand, due to hardware constraints, we were obliged to adapt this framework in the LDM setting. Doing so, we managed to perform the full optimization routine on a single GPU and 11 GB of available memory. Moreover, the text editing direction in the original implementation was static, i.e. it was the same for all input images, regardless of their context. This meant that all pairs of generated-original images was forced to be aligned with the same direction in the CLIP embedding space, resulting in lack of diversity in the manipulated images. Taking advantage of the class of each source image, we allowed images to be manipulated in effectively 36 different directions, that is 'neutral' $\rightarrow$ {'happy', 'sad', 'surprised', 'scared', 'disgusted', 'angry'} and $y_{\text{src}} \rightarrow y_{\text{trg}}$, with $y_{\text{trg}} \in$ {'happy', 'sad', 'surprised', 'scared', 'disgusted', 'angry'} and $y_{\text{src}} \neq y_{\text{trg}}$ (contempt was mapped to neutral). Lastly, as our LDM was trained using *c.f.g.*, we are able to couple CLIP guidance with different values of unconditional guidance scale γ.

The first step in finetuning an LDM, involves the precomputation of noisy latents for a predefined number of training instances and the entirety of the testing instances. We precomputed noisy latents using deterministic forward DDIM ($\eta = 0$) with $T_{\text{DDIM}} = 40$ steps and $t_0 = 500$ editing strength for 4,000 training instances (500 per emotion class). Subsequently, we randomly sampled 1,000 latents from the latter pool in an attempt to trade off model performance against reduced tuning time. This

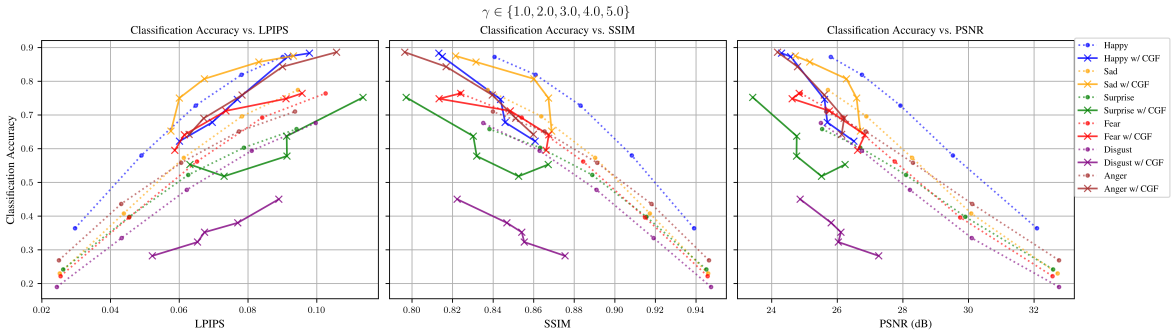


Figure 4.9: Quantitative comparison between baseline and CLIP-guided finetuned (CGF) models, in terms of mean classification accuracy (in the range [0-1]), LPIPS, SSIM and PSNR of manipulated samples, using $T_{\text{DDIM}} = 40$ steps, editing strength $t_0 = 500$, $\lambda_{\text{dir}} = 2.0, \lambda_{\text{id}} = \lambda_{\ell_2} = 1.0$, across all *c.f.g.* scales $\gamma \in \{1.0, 2.0, 3.0, 4.0, 5.0\}$.

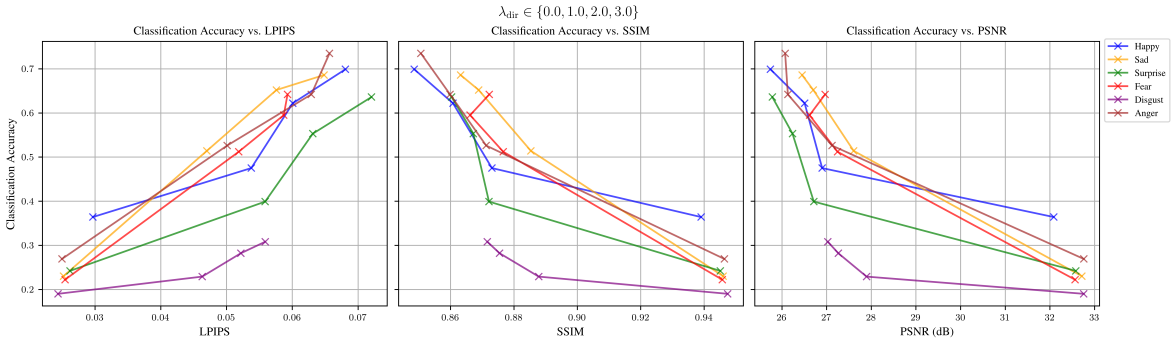
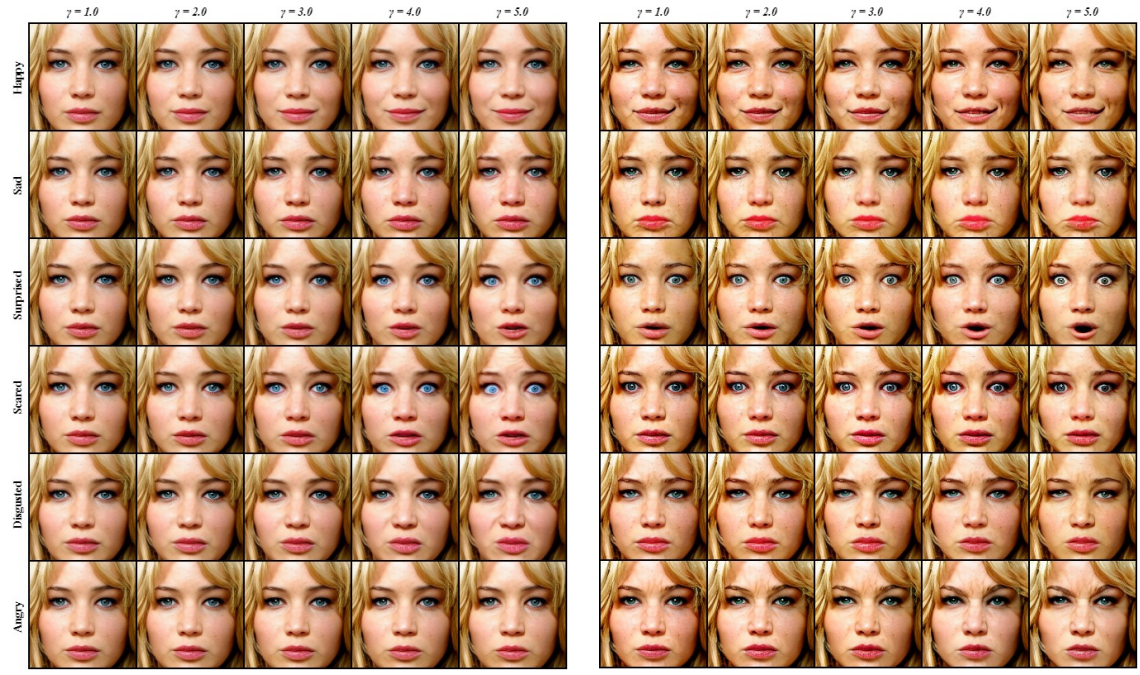


Figure 4.10: Quantitative comparison between baseline ($\lambda_{\text{dir}} = 0$) and CLIP-guided finetuned (CGF) models ($\lambda_{\text{dir}} > 0$), in terms of mean classification accuracy (in the range [0-1]), LPIPS, SSIM and PSNR of manipulated samples, using $T_{\text{DDIM}} = 40$ steps, editing strength $t_0 = 500$, $\gamma = 1.0, \lambda_{\text{id}} = \lambda_{\ell_2} = 1.0$, across different values of $\lambda_{\text{dir}} \in \{0.0, 1.0, 2.0, 3.0\}$.

procedure was replicated for all guidance scales $\gamma \in \{1.0, 2.0, 3.0, 4.0, 5.0\}$. We ran finetuning for 20 epochs with a learning rate of 2×10^{-6}, batch size of 4, using the AdamW [98] optimizer. Each batch of latents undergoes DDIM sampling for $T_{\text{tune}} = 6$ steps and $t_0 = 500$. Obviously this low number of steps in the DDIM sampling process gives imperfect reconstruction but is empirically chosen [76] as an acceptable compromise between sample quality and GPU memory requirements. The tuning procedure is repeated for each one of the six target emotions, leading to six different models per value of γ and combination of $(\lambda_{\text{dir}}, \lambda_{\text{id}}, \lambda_{\ell_2})$ coefficients. For simplicity, during all experiments we set $\lambda_{\text{id}} = \lambda_{\ell_2} = 1$ and only varied λ_{dir}.

Figs. 4.9 and 4.10 present quantitative comparisons between baseline and finetuned models, in terms of image quality metrics (LPIPS/SSIM/PSNR) and classification accuracy measured on top of the manipulated samples from the AffectNet validation set. More specifically, in Fig. 4.9 the finetuned models demonstrate higher classification accuracy scores for the same values of *c.f.g* scale γ, with the exception of the emotion of 'disgust'. Images that undergo CLIP guidance with the



(a) Baseline **(b)** Finetuned

Figure 4.11: Emotion manipulation comparison between baseline and finetuned models on curated examples from the AffectNet validation set, using $T_{\text{DDIM}} = 40$ steps, $\eta = 0$, editing strength $t_0 = 500$, variable *c.f.g.* scale $\gamma \in \{1.0, 2.0, 3.0, 4.0, 5.0\}$, $\lambda_{\text{dir}} = 2.0, \lambda_{\text{id}} = \lambda_{\ell_2} = 1.0$.

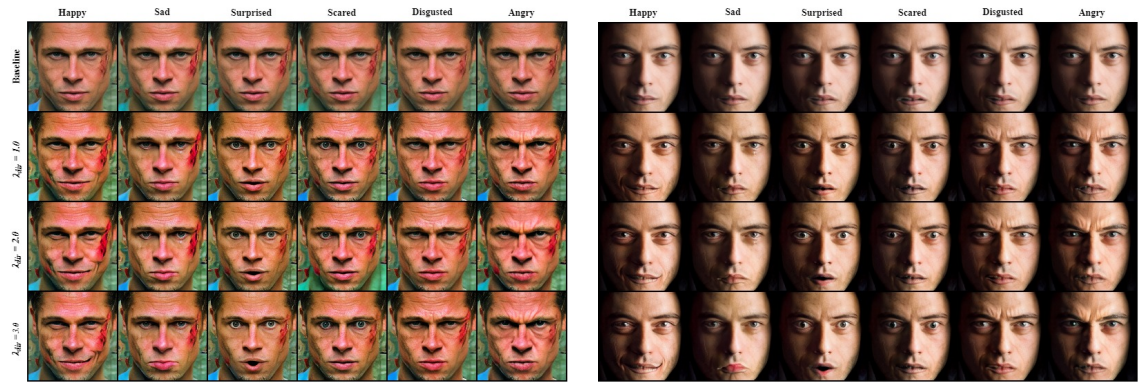


Figure 4.12: Qualitative comparison between baseline and finetuned models on curated examples from the AffectNet validation set, using $T_{\text{DDIM}} = 40$ steps, $\eta = 0$, editing strength $t_0 = 500$, $\gamma = 1$ $\lambda_{\text{id}} = \lambda_{\ell_2} = 1.0$ and variable $\lambda_{\text{dir}} \in \{1.0, 2.0, 3.0\}$.

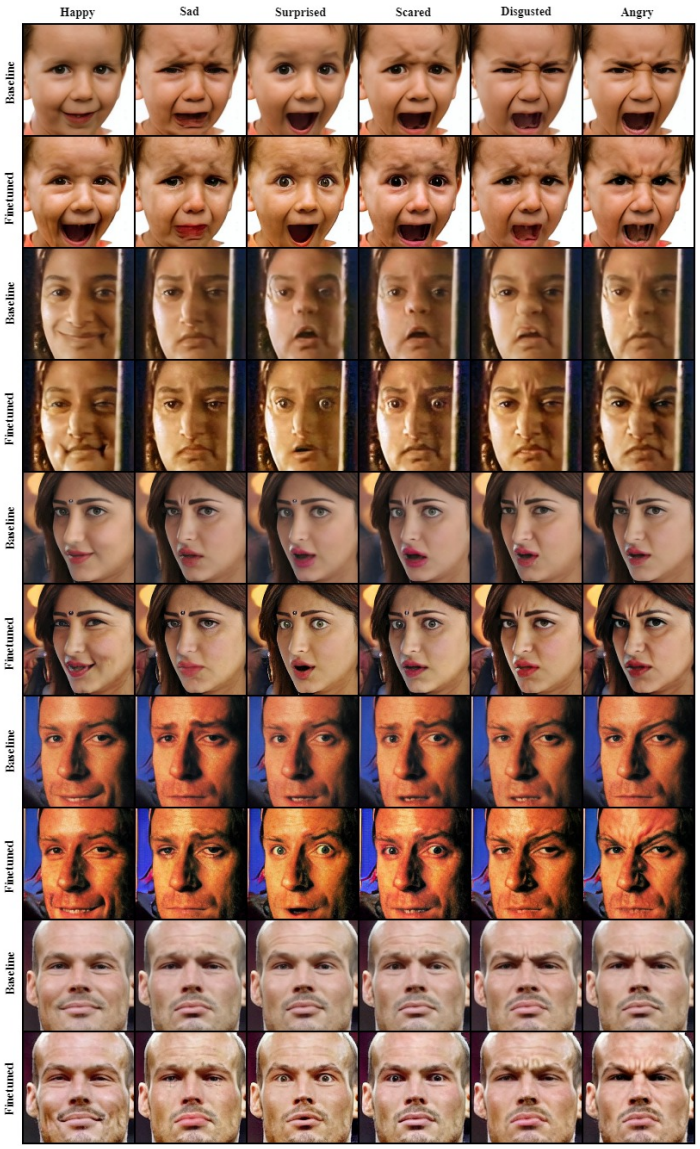


Figure 4.13: Qualitative comparison between baseline and finetuned models on curated examples from the AffectNet validation set, using $T_{\text{DDIM}} = 40$ steps, $\eta = 0$, editing strength $t_0 = 500$, *c.f.g* scale $\gamma = 3.0$, $\lambda_{\text{dir}} = 2.0, \lambda_{\text{id}} = \lambda_{\ell_2} = 1.0$.

latter being the target emotion, end up looking very similar to their 'angry' counterparts, hence the drop in accuracy. The boost in classification accuracy comes at the cost of a slight dicrease in image quality across all quality metrics, but as it will be demonstrated later, the end result remains visually compelling. Furthermore, in Fig. 4.10, we can see that even with no unconditional guidance ($\gamma = 1$), a setting in which the manipulation effect of the baseline method is often unnoticeable to the human eye, the introduction of directional CLIP loss resolves emotional ambiguity to a large extend, with the exception of 'disgust', as discussed above.

A qualitative comparison between the baseline and tuned models is presented in Figs. 4.11, 4.12 and 4.13, where the values of *c.f.g* scale γ and weight λ_{dir} are varied, respectively. Increasing γ in the finetuned models, does not particularly affect the cases of happiness and disgust while a slight shift in colors can be noticed in some cases. This could either be a result of the small number of sampling steps used during the tuning process ($T_{\text{tune}} = 6$) higher required coefficients for λ_{ℓ_2}.

4.6 Comparison with GAN-based Approaches

There is a scarcity in both diffusion-based and GAN-based implementations for image-based emotion manipulation that are publicly available. We search the recent related literature with the aim of finding methodologies that fall under either of the two aforementioned classes of networks. As at the time of writing, no applicable diffusion-based model with a publicly available codebase exists, we shift our attention towards well-grounded GAN-based implementations, with which we will compare our findings, in both a quantitative and qualitative level. These models are: GANimation [120], StarGAN v2 [26] and GANmut [30]. The following paragraphs present the core concepts behind each of the three aforementioned implementations, before moving onto the extensive quantitative and qualitative comparisons that we have conducted relative to our own implementations.

4.6.1 GANmut

GANmut [30] is a framework that utilizes the categorical labels of basic emotions to jointly learn the underlying emotion-conditional space as well as the ability to emotionally manipulate images. Motivated by the fact that basic emotion labels cannot faithfully cover the full gamut of human emotions (e.g., a happily surprised face might be only labelled as happy) GANmut introduces the problem of making the GAN conditional space learnable in an attempt to refine the inherent imperfections of emotion labeling.

To this end, two different parameterizations of the emotion-conditional space are proposed. The first approach, denotes as *Linear*, parameterizes a 2D conditional space Z with polar coordinates (θ, ρ). The conditional latent code is interpreted as a random variable $\boldsymbol{z} = (\theta, \rho)$, with its coordinates coming from a uniform distribution: $\theta \sim \mathcal{U}(0, 2\pi), \rho \sim \mathcal{U}(0, 1)$. Thus, each basic emotion $c \in \{1, \ldots, C\}$ ($C = 7$ in AffectNet, excluding the emotion of contempt) is parameterized with $\boldsymbol{z}_c = (\theta_c, \cdot)$, where θ_c is the learned direction for emotion c.

The second approach, denoted as *Gaussian*, aims to describe facial expressions using a label distribution rather than a single categorical emotion. In this case, the conditional latent code is a 2D random variable that is uniformly distributed, i.e., $\boldsymbol{z} \sim \mathcal{U}(Z), Z = [-1, 1]^2$. Then, a basic emotion $c \in \{1, \ldots, C\}$ is represented by a mode $(\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c)$. This is characterized by a mean $\boldsymbol{\mu}_c = \tanh(\boldsymbol{w}_c), \boldsymbol{w}_c \in \mathbb{R}^2$, with $\boldsymbol{w}_c$ being learnable parameters. Moreover, the corresponding covariance matrix $\boldsymbol{\Sigma}_c \in \mathbb{R}^{2\times 2}$ is parameterized by its eigenvalues $\sigma^2_{1,c}, \sigma^2_{2,c}$ and eigenvector orientation θ_c, which define the alignment and the length of the covariance ellipses axes, respectively.

For testing the aforementioned implementation against our own, we used the official codebase[13] along with publicly available models that have been pre-trained on AffectNet. We evaluated both linear and Gaussian models, denoted as GANmut and GGANmut, respectively. In the case of the linear model, we manipulated images using maximum intensity ($\rho = 1$). In order to achieve that, we use the learned unit vectors $\boldsymbol{v}_c = (x_c, y_c)$ corresponding to each emotion $c \in \{1, \ldots, C\}$, and calculate the principal direction $\theta_c = \arctan2(y_c, x_c)$.

4.6.2 GANimation

GANimation [120] is a conditional GAN framework that conditions the generation process on normalized AU intensities rather than discrete emotion labels. In that way, it is able to animate a continuous manifold of anatomical facial movements without being constrained by the limited expressiveness of discrete basic emotions.

Let G be a generator block. Given an RGB image $\mathbf{I}_{\boldsymbol{y}_o} \in \mathbb{R}^{H\times W\times 3}$ captured under an arbitrary facial expression and the N-dimensional vector $\boldsymbol{y}_f$ encoding the desired expression ($N = 17$), the input of G is formulated as a concatenation $(\mathbf{I}_{\boldsymbol{y}_o}, \boldsymbol{y}_f) \in \mathbb{R}^{H\times W\times (N+3)}$, where each entry in $\boldsymbol{y}_f$ has been replicated across the other two spatial dimensions. One key ingredient of the current system is to make G focus only on those regions of the image that are responsible of synthesizing the novel expression and keep the surrounding context untouched. To that end, instead of regressing a full image, the generator outputs two masks, a color mask $\mathbf{C}$ and attention mask $\mathbf{A}$. The final image is obtained as:

$$\mathbf{I}_{\boldsymbol{y}_f} = (1 - \mathbf{A}) \cdot \mathbf{C} + \mathbf{A} \cdot \mathbf{I}_{\boldsymbol{y}_o} \tag{4.24}$$

[13] https://github.com/stefanodapolito/GANmut

where $\mathbf{A} = G_A(\mathbf{I}_{\boldsymbol{y}_o}|\boldsymbol{y}_f) \in \{0,1\}^{H\times W}$ and $\mathbf{C} = G_C(\mathbf{I}_{\boldsymbol{y}_o}|\boldsymbol{y}_f) \in \mathbb{R}^{H\times W\times 3}$. The mask $\mathbf{A}$ indicates to which extent each pixel of $\mathbf{C}$ contributes to the output image $\mathbf{I}_{\boldsymbol{y}_f}$. The generator is applied twice, first to map the input image $\mathbf{I}_{\boldsymbol{y}_o} \rightarrow \mathbf{I}_{\boldsymbol{y}_f}$ and then to reconstruct it $\mathbf{I}_{\boldsymbol{y}_f} \rightarrow \hat{\mathbf{I}}_{\boldsymbol{y}_o}$. Moreover, a conditional patch-based discriminator D is used to evaluate the generated images in terms of their photo-realism and desired expression fulfillment. Conditioning is evaluated using an auxiliary regression head that estimates the AU activations $\hat{\boldsymbol{y}} = (\hat{y}_1, \ldots, \hat{y}_N)^\top$ of the generated image. The full model is trained using a weighted sum of adversarial, attention, conditional expression and identity losses. Training is performed in an unsupervised manner, using triplets of the form $\{\mathbf{I}_{\boldsymbol{y}_o}, \boldsymbol{y}_o, \boldsymbol{y}_f\}$, where the target AU vector $\boldsymbol{y}_f$ is randomly picked for every generated triplet.

For the application of GANimation in the emotion manipulation scenario with discrete emotions, we used the following scheme: we randomly sampled 500 images corresponding to each emotion label from the training set and computed the required target AU vectors for each one of them, using the publicly available OpenFace[14] [8] toolkit. Each vector contains intensities in the range $[0, 5]$ for each on of 17 detected AUs which are later normalized to $[0, 1]$. As GAN training is notoriously unstable, we opted for using a publicly available model checkpoint[15], pre-trained on more than 400K images with annotated AUs belonging to the EmotionNet database [46].

4.6.3 StarGAN v2

StarGAN v2 [26] differentiated itself among existing (at that time) implementations by ensuring both diverse image generation as well as scalability across multiple domains. Assuming $\mathcal{X}$ and $\mathcal{Y}$ denote a set of images and their respective domains, a single generator G is trained so that it can generate diverse images from each domain $y \in \mathcal{Y}$.

To begin with, a generator G translates an input image $\mathbf{x}$ into an output image $G(\mathbf{x}, \boldsymbol{s})$ reflecting a domain-specific style code $\boldsymbol{s}$, which is provided either by a mapping network F or by a style encoder E. The mapping network F generates style codes $\boldsymbol{s} = F_y(\boldsymbol{z}) \in \mathbb{R}^{d_s}$ where $\boldsymbol{z} \in \mathbb{R}^{d_z}$ is a latent code ($\boldsymbol{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ in practice). F consists of an MLP with multiple output branches to provide appropriate style codes for all available domains. Moreover, the style encoder receives an image $\mathbf{x}$ and its corresponding domain y as inputs and extracts its corresponding style code $\boldsymbol{s} = E_y(\mathbf{x})$. Similarly to F, the style encoder has the same number of output branches as the number of available domains. A multitask discriminator D is also employed, consisting of multiple output branches where each branch D_y solves the binary classification problem of whether $\mathbf{x}$ is a real image that belongs to domain y or a fake one produced by G.

The aforementioned framework is trained using a composition of various loss objectives. Apart from the regular adversarial objective, the generator is trained using a style reconstruction loss. More specifically, given a target style code $\tilde{\boldsymbol{s}}$ corresponding to target domain $\tilde{y}$, the generator is forced to utilize the latter when generating the image $G(\mathbf{x}, \tilde{\boldsymbol{s}})$ through the following loss term:

$$\mathcal{L}_{\text{sty}} = \mathbb{E}_{\mathbf{x},\tilde{y},\boldsymbol{z}}[\|\tilde{\boldsymbol{s}} - E_{\tilde{y}}(G(\mathbf{x}, \tilde{\boldsymbol{s}}))\|_1] \tag{4.25}$$

so that the learned encoder E allows G to transform an input image, reflecting the style of a reference image. Additionally, the generator is encouraged to generate diverse images through a style diversification loss:

$$\mathcal{L}_{\text{div}} = -\mathbb{E}_{\mathbf{x},\tilde{y},\boldsymbol{z}_1,\boldsymbol{z}_2}[\|G(\mathbf{x}, \tilde{\boldsymbol{s}}_1) - G(\mathbf{x}, \tilde{\boldsymbol{s}}_2)\|_1] \tag{4.26}$$

where the target style codes $\tilde{\boldsymbol{s}}_1, \tilde{\boldsymbol{s}}_2$ are produced through F conditioned on two random latent codes $\boldsymbol{z}_1, \boldsymbol{z}_2$, respectively. Lastly, a cycle consistency loss is employed to ensure that the generated image $G(\mathbf{x}, \tilde{\boldsymbol{s}})$ preserves any domain-invariant characteristics:

$$\mathcal{L}_{\text{cyc}} = \mathbb{E}_{\mathbf{x},y,\tilde{y},\boldsymbol{z}}[\|\mathbf{x} - G(G(\mathbf{x}, \tilde{\boldsymbol{s}}), \hat{\boldsymbol{s}})\|_1] \tag{4.27}$$

where $\hat{\boldsymbol{s}} = E_y(\mathbf{x})$ is the estimated style code of the input image $\mathbf{x}$ and y its respective original domain. All above training objectives (plus the adversarial objective) are combined through weight coefficients $\lambda_{\text{sty}}, \lambda_{\text{div}}, \lambda_{\text{cyc}}, \lambda_{\text{adv}}$, respectively.

We trained a StarGAN v2 model, using the publicly available official codebase[16], on AffectNet for 100K iterations using a batch size of 8. The learning rate for D, E and G is set equal to 10^{-4},

[14] https://github.com/TadasBaltrusaitis/OpenFace
[15] https://github.com/donydchen/ganimation_replicate
[16] https://github.com/clovaai/stargan-v2

y_{target} / Method	Happy					Sad					Surprised				
	Accuracy↑	PSNR↑	SSIM↑	LPIPS↓	CSIM↑	Accuracy↑	PSNR↑	SSIM↑	LPIPS↓	CSIM↑	Accuracy↑	PSNR↑	SSIM↑	LPIPS↓	CSIM↑
Groundtruth [107]	0.758	–	–	–	–	0.638	–	–	–	–	0.606	–	–	–	–
GANimation [120]	0.645	24.07	0.816	0.099	0.547	0.212	24.52	0.830	0.097	0.582	0.360	23.82	0.817	0.101	0.559
StarGAN v2 [26]	**0.958**	17.46	0.659	0.165	0.441	0.569	18.30	0.712	0.149	0.593	0.761	17.99	0.678	0.160	0.503
GANmut [30]	0.879	21.42	0.809	0.106	0.663	0.888	23.85	**0.857**	0.094	0.755	0.829	22.43	0.810	0.112	0.675
GGANmut [30]	0.934	21.91	0.819	0.103	0.717	**0.986**	22.13	0.802	0.115	0.653	**0.970**	22.37	0.777	0.121	0.610
Ours	0.872	**25.80**	**0.841**	**0.090**	0.743	0.774	**25.71**	0.837	0.095	0.778	0.658	**25.54**	**0.838**	**0.094**	0.716
Ours w/ CGF	0.883	24.30	0.813	0.098	**0.744**	0.875	24.72	0.822	**0.093**	**0.794**	0.752	23.42	0.797	0.113	**0.721**
	Scared					Disgusted					Angry				
Groundtruth [107]	0.666	–	–	–	–	0.646	–	–	–	–	0.514	–	–	–	–
GANimation [120]	0.314	23.68	0.814	0.105	0.556	0.271	24.13	0.819	0.103	0.549	0.287	24.80	0.833	0.096	0.580
StarGAN v2 [26]	0.860	17.76	0.676	0.162	0.507	0.879	18.10	0.691	0.154	0.528	0.666	18.14	0.689	0.154	0.509
GANmut [30]	0.932	23.39	**0.841**	0.102	0.721	0.877	22.64	0.815	0.113	0.663	0.856	21.57	0.813	0.106	0.678
GGANmut [30]	**0.967**	20.13	0.763	0.135	0.556	**0.987**	21.68	0.772	0.128	0.562	**0.969**	21.77	0.767	0.133	0.590
Ours	0.764	**24.89**	0.824	0.103	**0.770**	0.676	**25.50**	**0.835**	0.100	0.707	0.710	**25.66**	**0.840**	**0.094**	0.714
Ours w/ CGF	0.764	24.83	0.826	**0.096**	0.764	0.450	24.87	0.822	**0.089**	**0.714**	0.886	24.18	0.796	0.106	**0.735**

Table 4.4: Quantitative comparison among our LDM-based model and GAN-based implementations for image-based emotional manipulation on the AffectNet validation set.

Method	Mean				
	Accuracy↑	PSNR↑	SSIM↑	LPIPS↓	CSIM↑
Groundtruth [107]	0.638	–	–	–	–
GANimation [120]	0.348	24.17	0.822	0.100	0.562
StarGAN v2 [26]	0.782	17.95	0.684	0.157	0.514
GANmut [30]	0.877	22.55	0.824	0.106	0.693
GGANmut [30]	0.969	21.67	0.783	0.123	0.614
Ours[17]	0.742	**25.51**	**0.836**	**0.096**	0.738
Ours w/ CGF[18]	0.768	24.38	0.813	0.099	**0.745**

Table 4.5: Quantitative comparison among our LDM-based model and GAN-based implementations for image-based emotional manipulation on the AffectNet validation set, using mean aggregated metrics across all six target emotions.

while for F is set equal to 10^{-6}, using the Adam optimizer. The loss weighting coefficients were set equal to $\lambda_{\text{sty}} = \lambda_{\text{div}} = \lambda_{\text{cyc}} = \lambda_{\text{adv}} = 1$, for simplicity. Input image size is 128^2 pixels, with $|\mathcal{Y}| = 8$ domains (in accordance with the basic 7 emotions of the AffectNet dataset, plus neutral). We used latent code dimension $d_{\boldsymbol{z}} = 16$, style code dimension $d_{\boldsymbol{s}} = 64$ and 512 units for the hidden layers of the mapping network. For image-manipulation, we used latent-based translation, generating 10K latent codes per image, per target emotion. Subsequently, these latent codes were mapped to their corresponding style codes using the trained mapping network, while the generator was fed with the average of the computed style codes, along with the images to be translated.

Fig. 4.14 provides a qualitative comparison among the three aforementioned implementations, considering both *linear* and *Gaussian* variants of GANmut, on curated examples from the AffectNet validation set. We immediately notice that samples manipulated with GANmut, feature accurate depictions of the target emotions but often suffer from noise and subject identity deformation. Moreover, as it was expected, it is far more difficult to capture the desired emotions with GANimation and randomly selected driving AUs, rather that curated ones, but such a comparison would have been unfair with regard to other implementations. On the other hand, StarGAN v2 seems to strike a better balance between image quality and emotion transfer, but mostly fails to preserve the subjects' identity, skin tone and illumination. Our diffusion model successfully compromises between style preservation and emotion transfer, while also maintaining high image quality.

In Tab. 4.4 we provide a quantitative comparison among the three aforementioned GAN-based implementations and our diffusion-based ones, with and without CLIP-guided finetuning, in terms of emotion classification accuracy (using a pre-trained HSEmotion [135–137] as the classifier), PSNR, SSIM, LPIPS as well as feature cosine similarity in the embedding space of a CosFace [171] model, pre-trained on the Glint360K [2] dataset (CSIM). Boldface indicates the best score relative to each table column. The comparison all models is far from straightforward and we need to take into consideration the underlying trade-off between emotion classification accuracy and quality preservation with regard to source images. Across all emotions, our LDM-based implementations surpass all GAN-based counterparts in terms of image quality, something which was expected given the previous qualitative comparison on curated examples. Furthermore, GAN-based manipulators achieve generally higher emotion classification accuracies, with the highest mean scores being noted by GGANmut. This can be partly explained by the fact the the latter fits multiple Gaussians with the aim of acquiring

[17] *c.f.g* scale $\gamma = 5.0$, editing strength $t_0 = 500$ and $T_{\text{DDIM}} = 40$ steps.
[18] *c.f.g* scale $\gamma = 5.0$, editing strength $t_0 = 500$ and $T_{\text{DDIM}} = 40$ steps, $\lambda_{\text{dir}} = 2.0$, $\lambda_{\text{id}} = \lambda_{\ell_2} = 1.0$

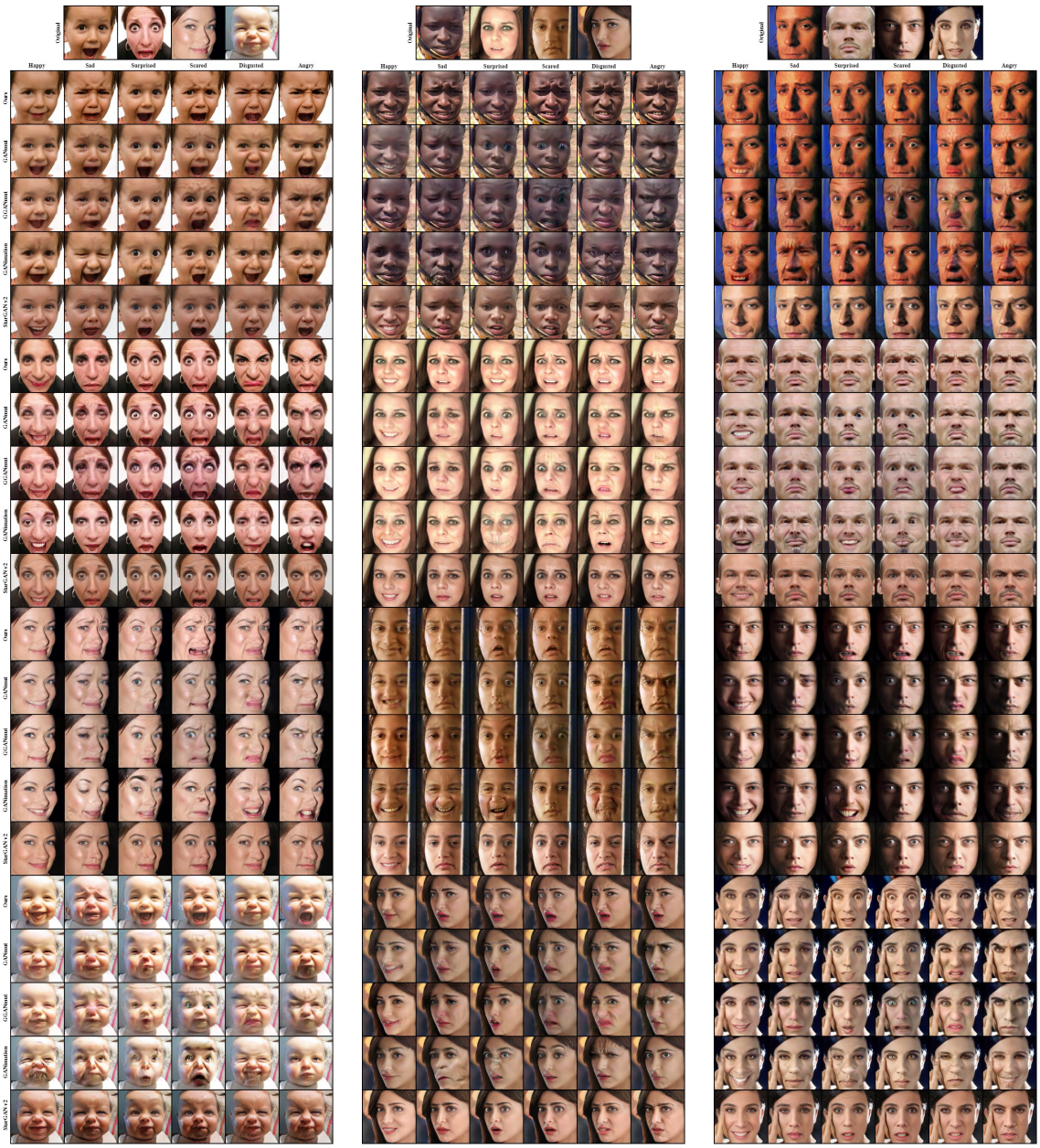


Figure 4.14: Qualitative comparison among our LDM-based model and GAN-based implementations for image-based emotional manipulation on curated examples from the AffectNet validation set.

a more accurate and rich representation of the conditional emotional latent space, compared to discrete emotion labels. We could probably counter this deficit by experimenting with different dropout probabilities for *c.f.g* during training and higher unconditional guidance scales ($\gamma > 5.0$). Moreover, a more subjective approach towards model evaluation can be followed, e.g., in the form of questionnaires, which is common is the case of image-to-image translation experiments. This will be further discussed in the upcoming paragraphs. Tab. 4.5 presents mean aggregated statistics across all six target emotions in terms of all five evaluation metrics.

We also conducted two user studies to evaluate the realism and emotion accuracy of our approach, as assessed by human users, the results of which can be found in Tab. 4.6. In the first study, we involved 24 participants, who were presented with 28 pairwise comparisons between all methods (including original images) and were asked to choose the most realistic one. As demonstrated by the results, our method's manipulated images were perceived *significantly* ($p < 0.01$ with binomial test) more realistic compared to the other methods. However, as anticipated, they were not as realistic as

	Realism							
Ours vs.	GANimation		GANmut		StarGAN v2		Groundtruth	
	0.69	0.31	**0.68**	0.32	**0.61**	0.39	0.28	**0.72**
	Emotion Recognition							
0.57	0.38		**0.73**		0.57		0.59	

Table 4.6: Head-to-head subjective study regarding realism (top half) and emotion translation accuracy (bottom half).

the actual images.

In the second study, 27 participants were shown 30 images from each method and asked to identify the displayed emotion from a list of six emotions. The bottom half of Tab. 4.6 presents the corresponding accuracy results, which are closely aligned with the previous objective results. Our approach achieves performance on-par with StarGAN v2 and surpasses GANimation, although GANmut achieves the highest accuracy. Notably, the accuracy in the original images of AffectNet is similar to that of StarGAN v2 and our method. This observation can potentially be attributed to GANmut generating more "exaggerated" emotions that deviate from the emotion distribution in AffectNet, sacrificing realism in the process. This is further strengthened through the initial qualitative comparison of Fig. 4.14, where the superiority of our models in terms of image quality and realism becomes more evident.

Chapter 5

Experiments–Talking Face Generation

The current short chapter is dedicated to presenting our experimental results regarding diffusion-based talking face generation. The basis of the presented models has already been established in the previous chapters and is centered around LDMs. The methodologies presented are heavily influenced from [154] and [140]. At the time of writing, no publicly available and relative code implementation exists, so part of the current experimentation is the actual reproduction of the models themselves. The first of the upcoming paragraphs focuses on introducing all necessary computational components of the talking face model, starting from the audio encoder.

5.1 Dataset and Preprocessing

Dataset Experiments are based on the Multi-view Emotional Audio-visual Dataset (MEAD) [172], a talking-face video corpus featuring 60 actors and actresses. The data acquisition process for MEAD focuses on two aspects. Firstly, capturing natural and stable emotions during speech, and secondly, ensuring distinguishable intensities of emotions. A guidance team, led by a professional actor, supervised the process. Fluent English speakers aged 20 to 35 with acting experience were recruited, and their skills were evaluated through imitation tasks based on video samples performed by the professional actor.

The MEAD dataset offers several distinctive features compared to other audio-visual datasets. It provides a more fine-grained manipulation of emotions and intensity levels by including neutral and 7 basic emotions with 3 intensity levels each. This richness in emotion information sets it apart from recent datasets that typically offer fewer intensity levels. MEAD also stands out for its multi-view data, with recordings captured from 7 cameras at different viewpoints, making it the dataset with the largest number of viewpoints among recent audio-visual datasets. Other datasets like Ouluvs2 [3] and TCD-TIMIT [61] offer a limited number of viewpoints or focus primarily on front and side views. MEAD videos have a resolution of 1920×1080, allowing for high-fidelity portrait video generation. The audio sample rate is 48 kHz, and the video frame rate is 30 fps, which is widely used and sufficient for various video tasks such as emotion recognition and portrait video generation.

No predefined training/test splits are provided by the distributors of the MEAD dataset. We create custom splits that do not share any common human subject. The training set ends up containing 21,739 videos while the test set contains 3,692 videos. The subjects contained in the test set are: M003, M023, M033, M040, W009, W021 and W040.

Preprocessing Input frames are first resized to 216×384. Next, they undergo facial landmark detection using FAN [18]. Using the pixel coordinates of the detected facial landmarks, we extract frame-level center crops, after further adding a 10 pixel margin along all directions (up, down, left, right). In order to avoid inter-frame inconsistencies in terms of the scale and size of the extracted face crops, we fix the expanded facial bounding box based on the detected facial landmark coordinates of the first frame of each video sequence. Our choice is partly justified by the fact that MEAD was assembled under heavily controlled lab settings, that is the possibility of a subject's head to fall outside of the initially detected bounding box is relatively low. The extracted bounding boxes are

later resized to 128×128. Based of the latter, we also recalculate and save facial landmark coordinates for each frame.

5.2 Baseline Model

The backbone of the upcoming models is again based on LDMs. What is worth mentioning is the additional feature extractor that produces the necessary audio conditioning that is eventually fed into the LDM cross-attention mechanism.

5.2.1 Audio Feature Extractor

Our design follows the SOTA self-supervised pre-trained speech model, wav2vec 2.0 [6], which can be seen in Fig. 5.1. The latter builds upon its predecessor, i.e. wav2vec [138], and introduces several key improvements and differentiating factors.

Contrastive Predictive Coding It employs a contrastive self-supervised learning objective called Contrastive Predictive Coding (CPC) [112] in its initial training stage. This allows the model to learn useful representations from large amounts of unlabeled speech data. By predicting masked portions of the audio, the model captures contextual information and learns to understand the relationships between different parts of the audio.

Transformer Architecture Its first stage encoder consists of several temporal convolutions layers (TCN) and transforms the raw waveform input into feature vectors. The outputs of the temporal convolutions are discretized to a finite set of speech units via a quantization module. While wav2vec used a CNN as its primary architecture, wav2vec 2.0 further incorporates a transformer encoder [168] composed of a stack of multi-head self-attention and feed-forward layers, converting the audio feature vectors into contextualized speech representations. Transformers have shown remarkable success in various natural language processing tasks, and their utilization enables the model to capture long-range dependencies in the audio data, leading to improved performance.

Large-Scale Training It benefits from being trained on massive amounts of unlabeled data. By leveraging large-scale training, the model can learn robust representations that generalize well to different downstream tasks. This allows the model to perform effectively even when finetuned on relatively smaller labeled datasets.

Supervised Finetuning It utilizes a supervised finetuning stage, where it is trained with labeled speech data using a connectionist temporal classification (CTC) loss. This finetuning process aligns the learned representations with the actual speech labels, further improving the model's accuracy in recognizing spoken words and other speech-related tasks.

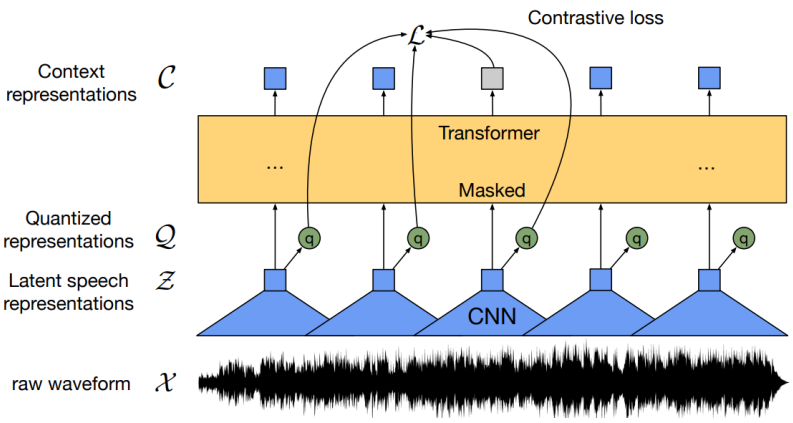


Figure 5.1: Illustration of the wav2vec 2.0 framework. Source: [6].

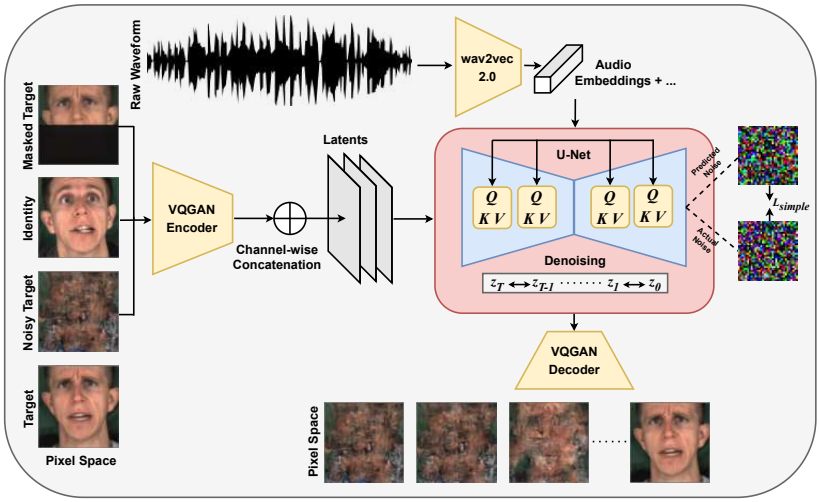


Figure 5.2: Illustration of the diffusion-based talking face generation framework. Inspired by: [125, 140, 154].

5.2.2 Methodology Overview

Due to the 2D inherent nature of diffusion models, we are constrained to frame-level generation during both training and inference. The rest of the current section describes the different conditioning mechanisms used for driving the talking face synthesis process. A high level overview of the approach is illustrated in Fig. 5.2.

Audio Conditioning Audio conditioning constitutes the primary driving factor and common denominator of all talking face synthesis configurations which we will consider. We experiment with extracting two kinds of audio features. In one case we use the hidden state $\mathbf{a} \in \mathbb{R}^{T' \times 768}$ as obtained from the last transformer encoder block of wav2vec 2.0. In the second case, we operate the model in ASR mode and extract unnormalized logits $\mathbf{b} \in \mathbb{R}^{T' \times 29}$ for 29 character labels[19]. We initialize our encoder with the pre-trained wav2vec 2.0 weights. Since the facial motion data might be captured with a frequency f_m that is different to the operating frequency of the feature extractor f_a ($f_a = 49$ Hz for wav2vec 2.0 while $f_m = 30$ fps for MEAD) we add a linear interpolation layer which results in the output length $T = kT' = \lfloor \frac{f_m}{f_a} \rfloor T'$. In this way, we achieve temporal alignment of the visual and audio input modalities. In order to add expressive power to the cross-attention mechanism, we add learnable audio meta-encoders $\boldsymbol{\tau_\theta}(\mathbf{a}) \in \mathbb{R}^{1 \times 768}$ (Conv1D temporal attention) and $\boldsymbol{\tau_{\theta'}}(\mathbf{b}) \in \mathbb{R}^{1 \times 64}$ (Conv1D temporal attention paired with a feature expander).

Emotion Conditioning As usual, we use a single learnable embedding layer $\boldsymbol{\tau_\theta}(y) \in \mathbb{R}^{1 \times 256}$ that maps each one of the 8 possible emotion labels of the MEAD dataset, i.e. neutral, happy, sad, surprised, scared, disgusted, angry and contempt. The emotion conditioning is not necessarily applied in all configuration but is essential in case of emotion conditional talking face synthesis.

Landmark Conditioning Landmark conditioning is directly adapted from [140] and evaluated as an additional conditioning mechanism. More specifically, we use a per-frame masked version of the landmarks detected using FAN [18] during the preprocessing stage. We use only 48 pairs out of the total 68 keypoints, excluding 20 pairs of pixel coordinates relative to the mouth area, so as to avoid shortcuts. The 48 coordinate pairs are flattened, forming a vector $\mathbf{l} \in \mathbb{R}^{96}$ and are later encoded using a learnable shallow MLP with ReLU activations $\boldsymbol{\tau_\theta}(\mathbf{l}) \in \mathbb{R}^{1 \times 128}$. This auxiliary facial landmark condition is also included for better control of the face outline.

Identity Frame Apart from the noisy target, the denoising U-Net is provided with a randomly chosen identity frame as reference. This conditioning mechanism is expected to help the denoising

[19] '-', '|', 'E', 'T', 'A', 'O', 'N', 'I', 'H', 'S', 'R', 'D', 'L', 'U', 'M', 'W', 'C', 'F', 'G', 'Y', 'P', 'B', 'V', 'K', "'", 'X', 'J', 'Q', 'Z'

component in transferring peripheral-appearance related features to the target synthesized image without information leakage relative to the mouth region. Since the ground-truth/target face image has a completely different pose from model is expected to transfer the pose of the identity frame to the target face without any prior information. This constitutes an ill-posed problem with no unique solution. In order to alleviate this issue, extra conditioning information is needed.

Masked Target The masked ground-truth image is provided as another reference condition for target head pose guidance. The mouth region of is completely masked to ensure that the ground truth lip movements are not visible to the network.

5.2.3 Training

The various conditioning information described above, can be separated into two distinct categories: i) Those that are fed to the denoising network through the LDM cross-attention mechanism, denoted as $\mathbf{c}_{\text{attn}}$. ii) Those that are fed to the denoising network by concatenating them channel-wise with the noisy target, denoted as $\mathbf{c}_{\text{cnct}}$. We train the LDM to learn the distribution of frames extracted from videos. At each iteration, we randomly sample a video $\mathbf{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}$ from the training set, and then a frame $\mathbf{x}_k$ from $\mathbf{X}$, where n is the total number of frames of the sampled video. In addition to the standard diffusion model's inputs, i.e. a time step t and the noisy target, we also sample an identity frame $\mathbf{x}_{\text{id}}$. To prevent training shortcuts, the selection of $\mathbf{x}_{\text{id}}$ is limited to 60 frames beyond the target image. Depending on the configuration, a masked version of the target frame is included, denoted as $\mathbf{x}_{\text{mask}}$. All of the aforementioned frames are first passed through a pre-trained perceptual encoder, leading to compressed latent representations $\mathbf{z}_k, \mathbf{z}_{\text{id}}$ and $\mathbf{z}_{\text{mask}}$, respectively. At each iteration, the randomly sampled target latent undergoes forward diffusion for $t \sim \mathcal{U}(1, T_{\text{DDPM}})$ timesteps, leading to the noisy target latent $\mathbf{z}_t$.

In terms of conditioning, $\mathbf{c}_{\text{cnct}}$ is formed through channel-wise concatenation of $\mathbf{z}_t$ with $\mathbf{z}_{\text{id}}$ and $\mathbf{z}_{\text{mask}}$, whenever applicable. That is $\mathbf{c}_{\text{cnct}} = \mathbf{z}_t \oplus_c \mathbf{z}_{\text{id}} \oplus_c \mathbf{z}_{\text{mask}}$. Similarly, $\mathbf{c}_{\text{attn}}$ is formed through vector concatenation of the respective audio, emotion or landmark embeddings. That is $\mathbf{c}_{\text{attn}} = [\boldsymbol{\tau}_{\boldsymbol{\theta}_1}(\mathbf{a}); \boldsymbol{\tau}_{\boldsymbol{\theta}_2}(y) \dots]$, where $\boldsymbol{\tau}_{\boldsymbol{\theta}_1}, \boldsymbol{\tau}_{\boldsymbol{\theta}_2}, \dots$ are the respective audio, emotion or landmarks learnable encoders. The simple denoising LDM objective is used to guide the training process:

$$\mathcal{L}_{\text{simple}} = \mathbb{E}_{\mathbf{z}, \mathbf{c}_{\text{cnct}}, \mathbf{c}_{\text{attn}}, \boldsymbol{\epsilon}, t}\left[\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{z}_t, \mathbf{c}_{\text{attn}}, \mathbf{c}_{\text{cnct}}, t)\|_2^2\right] \tag{5.1}$$

where $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}$ denotes the learnable denoising U-Net.

5.2.4 Autoregressive Synthesis

Following the same practices as [140, 154], we apply an autoregressive sampling scheme for talking face synthesis. In [154], the authors proposed the use of motion frames with the aim of producing smooth videos. During every sampling step, the current motion frame is replaced with the latest synthesized frame. During the early stages of development, we neglected this type of approach due to the fact that the latter methodology was based on pixel-space DDPM rather than LDM. This led to the problem of gradual error accumulation, especially in the case of longer videos. One extra error source in the case of LDMs is the inherent quantization/reconstruction error during latent encoding/decoding (decoding is necessary as the authors of [154] suggest motion frames to be in grayscale).

So we ended up using the proposed methodology of [140]. When rendering a talking video, for the first frame, the first identity frame matches the groundtruth target frame. Subsequently, every newly synthesized frame is utilized as the identity frame during the next generation step.

5.2.5 Ablation Study

Evaluation Metrics

We evaluate all configurations coupled with quantitative indicators. PSNR, SSIM and LPIPS are the three selected metrics for assessing image quality. As far as lip reading metrics are concerned, a straightforward way would to evaluate the compared methods by applying a pre-trained lipreading

Model	Conditioning			Image Quality			Lip Reading			
	Emotion	Masked Target	Landmarks	PSNR↑	SSIM↑	LPIPS↓	WER↓	CER↓	WERV↓	CERV↓
Groundtruth	N/A	N/A	N/A	N/A	N/A	N/A	65.5	41.3	65.1	35.6
Baseline (Transformer)	✓	✗	✗	20.61	0.582	0.114	191.8	137.8	187.7	130.8
	✓	✓	✗	26.59	0.856	0.046	112.9	85.2	109.7	81.1
	✓	✓	✓	26.68	**0.865**	**0.043**	110.1	85.1	108.4	79.7
	✗	✓	✗	**26.69**	0.857	0.047	102.2	**82.1**	99.3	**77.3**
Baseline (ASR)	✗	✓	✗	26.66	0.857	0.045	**101.3**	83.5	**99.1**	78.2

Table 5.1: Ablation study in terms of various conditioning mechanisms used in the underlying LDM backbone, on the basis of the MEAD dataset. Audio window length $w = 4$ in all configurations.

Model	Audio Window Length w	Image Quality			Lip Reading			
		PSNR↑	SSIM↑	LPIPS↓	WER↓	CER↓	WERV↓	CERV↓
Groundtruth	N/A	N/A	N/A	N/A	65.5	41.3	65.1	35.6
Baseline (Transformer)	1	26.29	0.851	0.048	121.2	90.5	119.6	88.2
	2	26.53	0.855	0.047	116.3	87.1	111.4	83.4
	4	**26.59**	0.856	**0.046**	112.9	85.2	109.7	81.1
	8	26.43	0.854	**0.046**	108.2	**82.3**	**105.1**	**77.7**
	16	25.74	0.841	0.049	116.2	87.5	111.9	83.1
Baseline (ASR)	8	26.41	**0.859**	**0.046**	**107.1**	83.3	105.3	**77.7**

Table 5.2: Quantitative comparative study in terms of various audio window lengths used by the underlying audio encoding module, on the basis of the MEAD dataset. Audio and emotion conditioning used in all configurations.

network on the output rendered images. We choose AV-HuBERT [142, 143] as the evaluation model. The following lipreading metrics are considered: Character Error Rate (CER), and Word Error Rate (WER) as well as their viseme variants (obtained by converting the predicted and ground truth transcriptions to visemes using the Amazon Polly phoneme-to-viseme mapping). CER calculation is based on the concept of Levenshtein distance, where we count the minimum number of character-level operations required to transform the groundtruth text into the OCR output. On the other hand, the formula for WER is the same as that of CER, but WER operates at the word level instead. It represents the number of word substitutions, deletions, or insertions needed to transform one sentence into another. The formulas for both error rates are the following:

$$\text{WER} = \frac{S + D + I}{N}, \qquad \text{CER} = \frac{S_w + D_w + I_w}{N_w} \tag{5.2}$$

where S denotes the number of character substitutions, D denotes the number of character deletions, I denotes the number of character insertion, N denotes the number of characters in the reference text while subscript w indicates their word-level counterparts.

Conditioning First we investigate the effect of the various conditioning mechanisms in both image quality and lip synching of the generated talking face sequences. Audio conditioning and identity frames are supplied as base conditioning in all cases. In terms of audio, we consider using either intermediate features from the last transformer block of wav2vec 2.0 or the unnormalized logits for 29 character labels. Evaluation metrics are calculated on the basis of 150 videos for a random subset of the MEAD test set. We ought to note that generation of 150 talking face sequences on an NVIDIA RTX 3090 GPU took approximately 24 hours. Due to both resource and time constraints, the number of generated sequences per compared model configuration was kept to a bare minimum.

Results are presented in Tab. 5.1. Providing only an identity frame and audio as conditioning results in the worst performance in terms of both image quality and lip reading. This is expected as the corresponding sequence generation task constitutes an ill-posed problem and calls for further guidance. Word/character error rates are above 100%, i.e. the generated sequences include more words/characters than the reference transcript. As we will later demonstrate, this happens because the generated sequences suffer from severe temporal inconsistencies. Moreover, we notice that the addition of landmark conditioning does not add anything to the lip reading metrics as landmark coordinates of the 20 mouth-related keypoints is excluded from the conditioning mechanism so as to avoid shortcuts. However, landmark conditioning provides a slight boost in image quality as they essentially help in capturing a more accurate outline of inter-frame facial deformations. Concatenating

Figure 5.3: Qualitative comparison in terms of LDM conditioning mechanisms on uncurated talking face frame sequences of the MEAD dataset.

emotion with audio conditioning results in a slight degradation in lip reading metrics. This is partly justified by the fact the resulting concatenated embedding is of a mush higher dimensionality (1024-dim in our case), making it more difficult for the underlying cross-attention mechanism to properly

distinguish the effect of each of aforementioned conditionings.

Fig. 5.3 illustrates uncurated sample sequences, as obtained through progressive inference on our MEAD test set, under different conditioning configurations. 'A' stands for audio, 'I' stands for identity, 'M' denotes the masked target, 'L' denotes the landmark, while 'E' denotes the emotion conditioning. We immediately notice that the identity conditioning does not suffice for capturing identity specific facial details and textures. Emotion conditioning does not seem to help in lip synching, a conclusion which is also reinforced by the quantitative results of Tab. 4.2. Masked target conditioning definitely plays an important role in grounding a rather ill-posed problem, providing strong guidance in terms of both head pose and subject identity preservation.

Audio Window Length Apart from the identity and masked target frames, another way to smoothen inter-frame transitions is by basing the audio conditioning on a series of adjacent frames rather than a single one. This means that when model tries to infer frame t, the audio encoder module is fed with wav2vec 2.0 features from $[t-w,\ldots,t,\ldots,t+w]$, where w denotes the audio window length. For example, all results presented in Tab. 5.1 correspond to an audio window length $w=4$. The corresponding ablation study is presented in Tab. 5.2, where we have used audio, coupled with masked target and emotion conditionings. In both ASR and transformer modes, the windowed audio features are weighted using a temporal attention module consisting of 5 Conv1D layers, followed by a linear and softmax layer. Small window lengths ($w=1,2$) are not sufficient for achieving smooth inter-frame transitions. Best lip reading performance is achieved for $w=8$. On the other hand, using a big audio window of length $w=16$ results in excessive smoothing and consequently, degradation in terms of lip reading metrics.

5.3 Lip Reading-Based Finetuning

A big portion of existing talking face synthesis methodologies incorporate some kind of lip reading expert that explicitly guides the model for better audio-lip synchronization. To the best of our knowledge, in the diffusion-based setting, no such approach has ever been applied. In [154], the authors proposed the use of a local denoising loss term on the basis of noisy and groundtruth mouth crops. However, this approach is not applicable in the LDM setting, due to the fact that training is performed in latent space rather that in pixel-space, which is the case for regular DDPM.

5.3.1 Caveats

Another fundamental blocker regarding the addition of a lip synching loss term in the LDM training objective is the fact that (in order to be able to sample using DDIM) LDMs are constrained to the so called *epsilon-prediction* mode, i.e. they learn to predict the added noise rather than the target image. The latter implies that a target image prediction needs to be obtained through the following transformation:

$$\hat{\mathbf{z}}_0 = \frac{\mathbf{z}_t - \sqrt{1-\bar{a}_t}\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{z}_t, \mathbf{c}_{\text{attn}}, \mathbf{c}_{\text{cnct}}, t)}{\sqrt{\bar{a}_t}}, \qquad \hat{\mathbf{x}}_0 = \mathcal{D}(\hat{\mathbf{z}}_0) \tag{5.3}$$

where $\mathcal{D}$ denotes the perceptual decoder that maps latents to their respective pixel-space counterparts. Getting back from predicted noise to $\hat{\mathbf{x}}_0$, which is required to apply any type of lip reading loss, is not accurate enough in a single step, and computationally inefficient in more steps. In order to demonstrate the inaccuracies introduced using this approach, we gathered $(\mathbf{x}_0, \hat{\mathbf{x}}_0)$ pairs using a pre-trained LDM. Results are illustrated in Fig. 5.4. The first couple of rows correspond to cases of relatively accurate prediction w.r.t. the target image, while the second couple of rows corresponds to cases of imperfect reconstruction. Considering the fact that the provided samples are obtained using the best available model checkpoint, it becomes evident that during earlier training epochs, artifacts in predicted targets $\hat{\mathbf{x}}_0$ will definitely be even more pronounced, rendering the addition of the lip-synching term pointless and subsequently, the training process unstable.

Therefore, we propose a lip reading finetuning training stage, guided with the help of a lip reading expert. For this purpose we use a network that has been trained on LRS3 [1]. The lipreading network receives sequences of grayscale images cropped around the mouth as input and outputs the

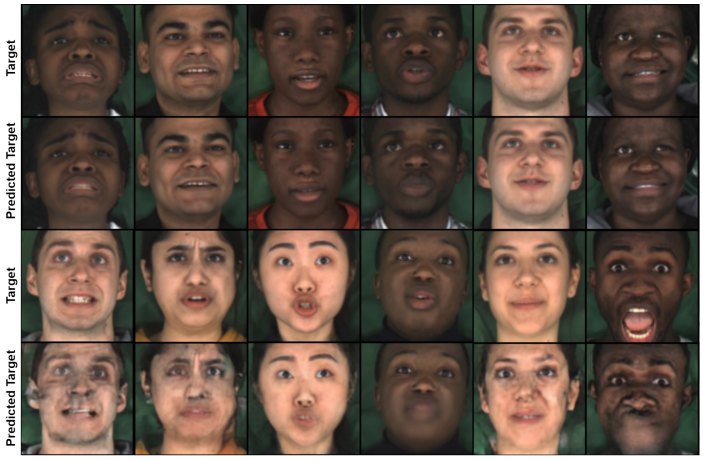


Figure 5.4: Groundtruth versus predicted targets using the ϵ-predicting network as a proxy.

predicted character sequence. The network has been trained with a combination of Connectionist Temporal Classification (CTC) loss with attention. The model architecture consists of a Conv3D kernel, followed by a ResNet-18, a 12-layer conformer, and finally a transformer decoder layer which outputs the predicted sequence.

5.3.2 Proposed Method

In order to take advantage of the lip reading expert during training, we need to get access to predicted target images. To this end, we first stochastically encode input latents using regular DDPM for $t \sim \mathcal{U}(1, T_{\text{DDPM}})$ timesteps. Next, in order to circumvent the inherent limitations of the *epsilon-predicting* pre-trained U-Net, we apply differentiable reverse DDIM on the noisy latents for $T_{\text{tune}} \ll T_{\text{DDPM}}$ steps (8 in our case). Obviously, the variable number of noising steps as well as the extremely low number of denoising steps will yield imperfect reconstructions. In practice though, the reconstructed latents detailed enough so that after decoding them, they can be fed to the lip reading expert. Before that, the predicted target images undergo cropping around the mouth region (based on facial landmarks predicted with FAN [18]), conversion to grayscale and resizing to 88×88 pixels. The batch dimension is collapsed in place of the sequence dimension. We calculate the corresponding feature vectors $f_{lr}(\mathbf{x}_0)$ and $f_{lr}(\hat{\mathbf{x}}_0)$. After calculating the feature vectors, we minimize the perceptual lip reading loss between the groundtruth mouth crops and the output rendered crops. The loss is defined as $\mathcal{L}_{lr} = \frac{1}{B}\sum_{i=1}^{B} d(f_{lr}(\mathbf{x}_0), f_{lr}(\hat{\mathbf{x}}_0))$ where $d(\cdot,\cdot)$ is the cosine distance, B is the batch size and $f_{lr}(\cdot)$ denotes the lip reading feature extractor.

However, the lip reading loss by itself proved capable of disrupting the pre-trained backbone, leading inferred frames to saturation just after few epochs of finetuning, as illustrated in Fig. 5.5. In order to alleviate this issue, we propose adding an ℓ_2-based consistency term so as to ensure accurate denoising of input latents. The overall finetuning objective is as follows:

$$\mathcal{L}_{\text{finetune}} = \mathbb{E}_{\mathbf{x}_0,t}\Big[d(f_{lr}(\mathbf{x}_0), f_{lr}(\hat{\mathbf{x}}_0)) + \lambda\|\mathbf{z}_0 - \hat{\mathbf{z}}_0\|_2^2\Big], \qquad \hat{\mathbf{z}}_0 = \text{DDIM}(\mathbf{z}_t) \tag{5.4}$$

where $\mathbf{z}_t$ is obtained using the regular DDPM noising process. For simplicity, we set $\lambda = 1$ across all of our finetuning experiments.

Tab. 5.3 presents a quantitative comparison between baseline and finetuned models, across different audio window length and conditioning configurations. Best lip reading performance is achieved after applying finetuning on pre-trained LDM with $w = 8$ audio window length and only audio conditioning. Although lip reading metrics showcase significant improvements after the application of our proposed finetuning scheme, we notice a slight degradation in terms of image quality. This is justified by the fact that we apply DDIM denoising on noisy latents which have been corrupted with a variable number of noising steps t per instance, while also using an extremely low number of DDIM steps $T_{\text{tune}} = 8$. In order for DDIM denoising to function properly, we should have applied a constant number of noising steps $t = T_{\text{DDPM}}$ per iteration. However, we found in practice that such an

Figure 5.5: Denoising rows across different finetuning epochs, with columns corresponding to intermediate denoising steps, guided solely by a lip reading loss. Saturation occurs within the four first finetuning epochs.

Model	Emotion Conditioning	Audio Window Length w	Image Quality			Lip Reading			
			PSNR↑	SSIM↑	LPIPS↓	WER↓	CER↓	WERV↓	CERV↓
Groundtruth	N/A	N/A	N/A	N/A	N/A	65.5	41.3	65.1	35.6
Baseline	✗	4	**26.69**	0.857	0.047	102.2	82.1	99.3	77.3
	✓	8	26.43	0.854	**0.046**	108.2	82.3	105.1	77.7
Finetuned	✓	4	25.97	**0.858**	0.058	97.3	80.1	96.9	75.0
	✓	8	25.75	0.848	0.054	95.2	79.2	94.0	75.1
	✗	4	25.42	0.845	0.058	91.1	79.1	88.2	74.1
	✗	8	25.87	0.857	0.055	**89.8**	**77.3**	**83.9**	**73.2**

Table 5.3: Quantitative comparison between our baseline and finetuned talking face generation models, on the basis of the MEAD dataset. All configurations utilize audio features as extracted from the last layer of the wav2vec 2.0 transformer encoder block.

approach leads to overfitting and a more severe degradation in both image quality and lip reading performance. Furthermore, Fig. 5.6 provides a qualitative comparison on the basis of uncurated talking face sequences, among groundtruth, baseline and finetuned models. Supplementary red boxes are used in order to highlight frames where considerable differences in terms of lip synching can be noticed. For example in the first row triplet, in the second frame, the finetuned model correctly follows the groundtruth with a slightly opened mouth, while the baseline model fails, showcasing a closed mouth. Also, in the second to last frame, the groundtruth and finetuned model frames almost exactly match while the baseline frame showcases a widely open mouth. In the second row triplet, in the last four frames, the finetuned model better follows the groundtruth, where the baseline falsely maintains an open mouth that likely corresponds to a vowel, while the groundtruth visually suits a fricative consonant sound. In the last row triplet, in the last five frames, the finetuned model follows the groundtruth mouth opening, while the baseline model showcases a mostly closed mouth sub-sequence that goes visually near an open vowel sound in the last two frames.

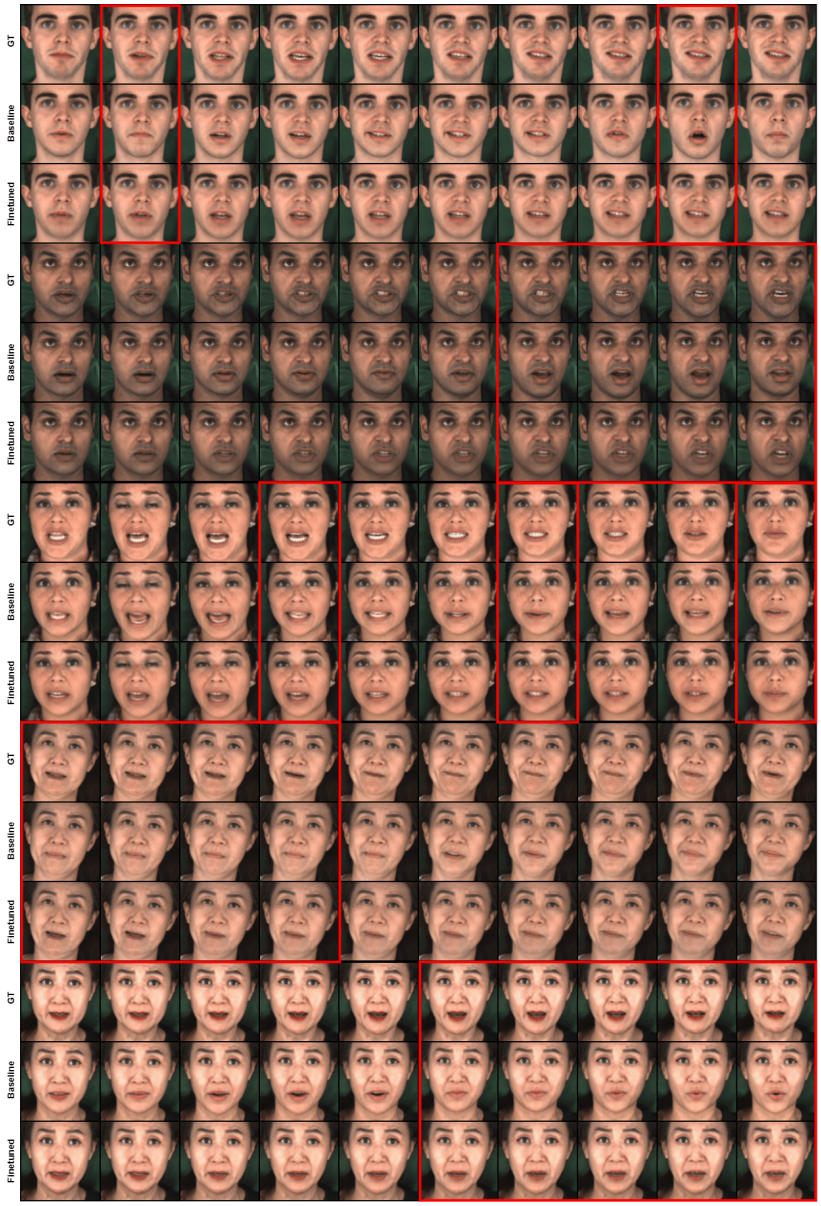


Figure 5.6: Qualitative comparison between groundtruth talking face sequences, and synthesized sequences with our baseline and finetuned models, on the basis of the MEAD dataset.

5.3.3 Future Directions

We decide to keep the current chapter short but at the same time concise enough to raise the main pain-points regarding talking face synthesis with diffusion models. One possible future research direction involves facilitating the need for masked target conditioning through an alternative, such as motion frames/latents. Provided that the upper half of the target image is not provided as conditioning, then emotional talking face manipulation will come into play. At the moment, the latter is not possible due to the fact that only the bottom half of the synthesized image is inferred, and subsequently can be manipulated using techniques similar to those described in Chapter 4. Additionally, more effective audio conditioning mechanisms need to be proposed with the aim of improving lip reading performance of the end-model. Lip reading performance at the moment is rather poor. Lastly, progressive inference is a rather slow process, raising questions as to whether other samplers apart from DDIM need to be considered.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

In this thesis we addressed the problems of both facial reenactment and talking face generation with the use of diffusion models. Experiments for the former were conducted in the fully uncontrolled, "in-the-wild" settings of the AffectNet database. Experiments for the latter were performed on more controlled-lab settings on the basis of the MEAD dataset, and only considering video sequences that depicted frontal face views.

To the best of our knowledge, our work on AffectNet is the first fully-fledged set of experiments conducted on the aforementioned dataset in the context of diffusion-based facial reenactment. Our framework enables emotion conditioning based on multiple degrees of freedom, namely editing strength t_0, classifier-free guidance scale γ, allowing for both emotion intensity and variety control on top of the manipulated images. Our baseline method is based on deterministic ($\eta = 0$) DDIM noising paired with deterministic DDIM sampling, conditioned on pairs of source and target emotion labels.

Furthermore, we leveraged CLIP pre-trained models with the aim of better guiding the underlying emotional manipulation processes. We drew inspiration from and extended the DiffusionCLIP framework. The following are the two primary distinctions between the original DiffusionCLIP and our adaptation: i) The original one's backbone was a DDPM, whereas ours is a LDM. ii) Because the majority of the original evaluations were conducted in unconditional settings, the former did not utilize source class-label information when applying the directional CLIP loss. Through our adaptation, we managed to run the full optimization routine of DiffusionCLIP using half of the GPU resource specifications of the latter. Moreover, we allowed for greater variety in the end result, as depicted in manipulated images, by taking advantage of the emotional source labels. More specifically, images could be finetuned in 36 different directions, that is 'neutral' $\rightarrow$ {'happy', 'sad', 'surprised', 'scared', 'disgusted', 'angry'} and $y_{\text{src}} \rightarrow y_{\text{trg}}$, with $y_{\text{trg}} \in$ {'happy', 'sad', 'surprised', 'scared', 'disgusted', and $y_{\text{src}} \neq y_{\text{trg}}$ (contempt was mapped to neutral). Lastly, we coupled CLIP guidance loss coefficients with different values of unconditional guidance scale γ, enabling more fine-grained control over the emotional intensity of the depicted end result.

We compared our method with SOTA GAN-based models. To that end, we conducted both extensive quantitative and qualitative experiments. Qualitative experiments revolved around user-case studies in terms of both realism and emotion translation. Both objective and subjective evaluations indicated that our method is superior in terms of image quality and subject identity preservation, while achieving competitive results regarding emotion translation accuracy against both conventional and dedicated GAN-based methodologies.

Subsequently, we shifted our attention towards talking face generation with diffusion models. We conducted extensive ablation studies, mainly regarding conditioning mechanisms, with emphasis laid on audio. Results obtained with our baseline model indicated poor lip synching performance. This pushed us into finding a way to use a lip reading expert as additional source of guidance during training. After failing to integrate a lip reading loss directly in the main LDM training loop, we proposed a LDM finetuning stage. To the best of our knowledge, our proposed finetuning stage constitutes the first proper lip reading-based algorithm for finetuning an LDM with the help of a lip reading expert. Post-tuning evaluations indicate that finetuned models perform significantly better

in terms of lip reading metrics, at the cost of a slight degradation in image quality.

6.2 Future Work

Many different adaptations, tests, and experiments have been left for the future due to lack of time (i.e. the experiments with real data are usually very time consuming, requiring even days to finish a single run). Future work concerns deeper analysis of particular mechanisms, new proposals to try different methods, or simply curiosity.

Improvements upon our facial reenactment pipeline include:

- Condition the LDM backbone on 3DMM [15] coefficients, allowing for head pose conditioning as well as fine-grained control over the depicted facial expressions.
- Substitute the discrete emotion labels with a more sophisticated and robust parameterization of the emotion-conditional space, such as the one proposed by GANmut [30].
- Integrate AU-based conditioning as a substitute for discrete emotion labels, similar to GANimation [120] or ICface [163].

As far as talking face generation is concerned, future research directions might revolve around the following key-points:

- Facilitating the need for masked target conditioning through alternative conditioning mechanisms, potentially involving motion frames/latents.
- Emotional talking face manipulation, provided that the upper half of the target image is not offered as conditioning. The latter is currently not feasible because only the bottom half of the synthesized image is inferred.
- More effective audio conditioning mechanisms, as current lip reading performance is rather poor.
- Progressive inference speed-up and potential replacement of the DDIM sampler (e.g. PLMS [95]).

Lastly, code for the reproduction of all experiments is made publicly available in the form of the following GitHub repository: https://github.com/GiannisPikoulis/dsml-thesis

Bibliography

[1] T. Afouras, J. S. Chung, A. Senior, O. Vinyals, and A. Zisserman, "Deep audio-visual speech recognition", *IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 44, pp. 8717–8727, 2018.

[2] X. An *et al.*, "Partial FC: Training 10 million identities on a single machine", in *Proc. IEEE/CVF Int. Conf. on Computer Vision (ICCV)*, 2021.

[3] I. Anina, Z. Zhou, G. Zhao, and M. Pietikäinen, "Ouluvs2: A multi-view audiovisual database for non-rigid mouth motion analysis", in *IEEE Int. Conf. on Automatic Face & Gesture Recognition (FG)*, 2015.

[4] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks", in *Int. Conf. on Machine Learning (ICML)*, 2017.

[5] H. Averbuch-Elor, D. Cohen-Or, J. Kopf, and M. F. Cohen, "Bringing portraits to life", *ACM Trans. on Graphics (ToG)*, vol. 36, pp. 1–13, 2017.

[6] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations", *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[7] O. Bălan, G. Moise, L. Petrescu, A. Moldoveanu, M. Leordeanu, and F. Moldoveanu, "Emotion classification based on biophysical signals and machine learning techniques", *Symmetry*, vol. 12, pp. 1–21, 2020.

[8] T. Baltrusaitis, A. Zadeh, Y. C. Lim, and L.-P. Morency, "OpenFace 2.0: Facial behavior analysis toolkit", in *IEEE Int. Conf. on Automatic Face & Gesture Recognition (FG)*, 2018.

[9] D. Baranchuk, A. Voynov, I. Rubachev, V. Khrulkov, and A. Babenko, "Label-efficient semantic segmentation with diffusion models", in *Int. Conf. on Learning Representations (ICLR)*, 2022.

[10] J. Behrmann, W. Grathwohl, R. T. Chen, D. Duvenaud, and J.-H. Jacobsen, "Invertible residual networks", in *Int. Conf. on Machine Learning (ICML)*, 2019.

[11] Y. Bengio, R. Ducharme, and P. Vincent, "A neural probabilistic language model", in *Advances in Neural Information Processing Systems (NIPS)*, 2000.

[12] Y. Bengio, N. Léonard, and A. Courville, *Estimating or propagating gradients through stochastic neurons for conditional computation*, 2013. arXiv: 1308.3432 [cs.LG].

[13] M. Binkowski, D. J. Sutherland, M. Arbel, and A. Gretton, "Demystifying MMD GANs", in *Int. Conf. on Learning Representations (ICLR)*, 2018.

[14] V. Blanz, C. Basso, T. Poggio, and T. Vetter, "Reanimating faces in images and video", *Computer Graphics Forum*, vol. 22, pp. 641–650, 2003.

[15] V. Blanz and T. Vetter, "A morphable model for the synthesis of 3D faces", in *Proc. 26th Annual Conf. on Computer Graphics and Interactive Techniques (SIGGRAPH)*, 1999.

[16] S. Bond-Taylor, A. Leach, Y. Long, and C. G. Willcocks, "Deep generative modelling: A comparative review of VAEs, GANs, normalizing flows, energy-based and autoregressive models", *IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 44, pp. 7327–7347, 2022.

[17] C. Bregler, M. Covell, and M. Slaney, "Video rewrite: Driving visual speech with audio", in *Proc. 24th Annual Conf. on Computer Graphics and Interactive Techniques (SIGGRAPH)*, 1997.

[18] A. Bulat and G. Tzimiropoulos, "How far are we from solving the 2D & 3D face alignment problem? (And a dataset of 230,000 3D facial landmarks)", in *Proc. IEEE/CVF Int. Conf. on Computer Vision (ICCV)*, 2017.

[19] P. Chandramouli and K. V. Gandikota, "LDEdit: Towards generalized text guided image manipulation via latent diffusion models", in *British Machine Vision Conf. (BMVC)*, 2022.

[20] Y.-J. Chang and T. Ezzat, "Transferable videorealistic speech animation", in *Proc. ACM SIGGRAPH/Eurographics Symp. on Computer Animation*, 2005.

[21] L. Chen, R. K. Maddox, Z. Duan, and C. Xu, "Hierarchical cross-modal talking face generation with dynamic pixel-wise loss", in *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[22] R. T. Chen, J. Behrmann, D. K. Duvenaud, and J.-H. Jacobsen, "Residual flows for invertible generative modeling", in *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[23] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel, "InfoGAN: Interpretable representation learning by information maximizing generative adversarial nets", in *Advances in Neural Information Processing Systems (NIPS)*, 2016.

[24] X. Chen, N. Mishra, M. Rohaninejad, and P. Abbeel, "PixelSNAIL: An improved autoregressive generative model", in *Int. Conf. on Machine Learning (ICML)*, 2018.

[25] Y. Choi, M. Choi, M. Kim, J.-W. Ha, S. Kim, and J. Choo, "StarGAN: Unified generative adversarial networks for multi-domain image-to-image translation", in *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[26] Y. Choi, Y. Uh, J. Yoo, and J.-W. Ha, "StarGAN v2: Diverse image synthesis for multiple domains", in *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[27] J. S. Chung, A. Jamaludin, and A. Zisserman, "You said that?", in *British Machine Vision Conf. (BMVC)*, 2017.

[28] F.-A. Croitoru, V. Hondru, R. T. Ionescu, and M. Shah, *Diffusion models in vision: A survey*, 2022. arXiv: 2209.04747 [cs.CV].

[29] D. Cudeiro, T. Bolkart, C. Laidlaw, A. Ranjan, and M. J. Black, "Capture, learning, and synthesis of 3D speaking styles", in *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[30] S. d'Apolito, D. P. Paudel, Z. Huang, A. Romero, and L. Van Gool, "GANmut: Learning interpretable conditional space for gamut of emotions", in *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[31] H. Dang, F. Liu, J. Stehouwer, X. Liu, and A. K. Jain, "On the detection of digital face manipulation", in *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[32] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, "ArcFace: Additive angular margin loss for deep face recognition", in *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[33] P. Dhariwal and A. Nichol, "Diffusion models beat GANs on image synthesis", in *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[34] H. Ding, K. Sricharan, and R. Chellappa, "ExprGAN: Facial expression editing with controllable expression intensity", in *Proc. AAAI Conf. on Artificial Intelligence*, 2018.

[35] L. Dinh, D. Krueger, and Y. Bengio, *Nice: Non-linear independent components estimation*, 2015. arXiv: 1410.8516 [cs.LG].

[36] L. Dinh, J. Sohl-Dickstein, and S. Bengio, "Density estimation using Real NVP", in *Int. Conf. on Learning Representations (ICLR)*, 2017.

[37] B. Dolhansky, R. Howes, B. Pflaum, N. Baram, and C. C. Ferrer, *The Deepfake detection challenge (DFDC) preview dataset*, 2019. arXiv: 1910.08854 [cs.CV].

[38] A. Dosovitskiy *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale", in *Int. Conf. on Learning Representations (ICLR)*, 2021.

[39] M. C. Doukas, M. R. Koujan, V. Sharmanska, A. Roussos, and S. Zafeiriou, "Head2head++: Deep facial attributes re-targeting", *IEEE Trans. on Biometrics, Behavior, and Identity Science*, vol. 3, pp. 31–43, 2021.

[40] B. Efron, "Tweedie's formula and selection bias", *Journal of the American Statistical Association*, vol. 106, pp. 1602–1614, 2011.

[41] P. Ekman, "Strong evidence for universals in facial expressions: A reply to Russell's mistaken critique", *Psychological Bulletin*, vol. 115, pp. 268–287, 1994.

[42] P. Ekman and W. V. Friesen, "Constants across cultures in the face and emotion", *Journal of Personality and Social Psychology*, vol. 17, pp. 124–129, 1971.

[43] P. Ekman and W. V. Friesen, "Measuring facial movement", *Environmental Psychology and Nonverbal Behavior*, vol. 1, pp. 56–75, 1976.

[44] P. Esser, R. Rombach, A. Blattmann, and B. Ommer, "ImageBART: Bidirectional context with multinomial diffusion for autoregressive image synthesis", in *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[45] P. Esser, R. Rombach, and B. Ommer, "Taming transformers for high-resolution image synthesis", in *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[46] C. Fabian Benitez-Quiroz, R. Srinivasan, and A. M. Martinez, "EmotioNet: An accurate, real-time algorithm for the automatic annotation of a million facial expressions in the wild", in *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[47] B. Fan, L. Wang, F. K. Soong, and L. Xie, "Photo-real talking head with deep bidirectional LSTM", in *IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, 2015.

[48] Y. Fan, Z. Lin, J. Saito, W. Wang, and T. Komura, "FaceFormer: Speech-driven 3D facial animation with transformers", in *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[49] M. Fréchet, "Sur la distance de deux lois de probabilité", *Comptes Rendus Hebdomadaires des Seances de L Academie des Sciences*, vol. 244, pp. 689–692, 1957.

[50] B. J. Frey, G. E. Hinton, and P. Dayan, "Does the wake-sleep algorithm produce good density estimators?", in *Advances in Neural Information Processing Systems (NIPS)*, 1995.

[51] S. Fu, R. Gutierrez-Osuna, A. Esposito, P. K. Kakumanu, and O. N. Garcia, "Audio/visual mapping with cross-modal hidden Markov models", *IEEE Trans. on Multimedia (TMM)*, vol. 7, pp. 243–252, 2005.

[52] R. Gal, O. Patashnik, H. Maron, A. H. Bermano, G. Chechik, and D. Cohen-Or, "StyleGAN-NADA: CLIP-guided domain adaptation of image generators", *ACM Trans. on Graphics (ToG)*, vol. 41, pp. 1–13, 2022.

[53] P. Garrido, L. Valgaerts, O. Rehmsen, T. Thormahlen, P. Perez, and C. Theobalt, "Automatic face reenactment", in *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[54] M. Germain, K. Gregor, I. Murray, and H. Larochelle, "MADE: Masked autoencoder for distribution estimation", in *Int. Conf. on Machine Learning (ICML)*, 2015.

[55] I. Goodfellow *et al.*, "Generative adversarial nets", in *Advances in Neural Information Processing Systems (NIPS)*, 2014.

[56] A. Graikos, N. Malkin, N. Jojic, and D. Samaras, "Diffusion models as plug-and-play priors", in *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[57] U. Grenander and M. I. Miller, "Representations of knowledge in complex systems", *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 56, pp. 549–581, 1994.

[58] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of Wasserstein GANs", in *Advances in Neural Information Processing Systems (NIPS)*, 2017.

[59] Y. Guo, K. Chen, S. Liang, Y.-J. Liu, H. Bao, and J. Zhang, "AD-NeRF: Audio driven neural radiance fields for talking head synthesis", in *Proc. IEEE/CVF Int. Conf. on Computer Vision (ICCV)*, 2021.

[60] A. Hannun *et al.*, *Deep Speech: Scaling up end-to-end speech recognition*, 2014. arXiv: 1412.5567 [cs.CL].

[61] N. Harte and E. Gillen, "TCD-TIMIT: An audio-visual corpus of continuous speech", *IEEE Trans. on Multimedia*, vol. 17, pp. 603–615, 2015.

[62] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition", in *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[63] Z. He, W. Zuo, M. Kan, S. Shan, and X. Chen, "AttGAN: Facial attribute editing by only changing what you want", *IEEE Trans. on Image Processing*, vol. 28, pp. 5464–5478, 2019.

[64] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "GANs trained by a two time-scale update rule converge to a local nash equilibrium", in *Advances in Neural Information Processing Systems (NIPS)*, 2017.

[65] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models", in *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[66] J. Ho and T. Salimans, "Classifier-free diffusion guidance", in *NeurIPS Workshop on Deep Generative Models and Downstream Appl.*, 2021.

[67] J. Ho *et al.*, *Imagen Video: High definition video generation with diffusion models*, 2022. arXiv: 2210.02303 [cs.CV].

[68] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks", in *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[69] M. F. Hutchinson, "A stochastic estimator of the trace of the influence matrix for Laplacian smoothing splines", *Communications in Statistics-Simulation and Computation*, pp. 1059–1076, 1989.

[70] A. Hyvärinen and P. Dayan, "Estimation of non-normalized statistical models by score matching", *Journal of Machine Learning Research (JMLR)*, vol. 6, pp. 695–709, 2005.

[71] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks", in *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[72] S. A. Jalalifar, H. Hasani, and H. Aghajan, *Speech-driven facial reenactment using conditional generative adversarial networks*, 2018. arXiv: 1803.07461 [cs.CV].

[73] T. Karras, T. Aila, S. Laine, A. Herva, and J. Lehtinen, "Audio-driven facial animation by joint end-to-end learning of pose and emotion", *ACM Trans. on Graphics (ToG)*, vol. 36, pp. 1–12, 2017.

[74] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of GANs for improved quality, stability, and variation", in *Int. Conf. on Learning Representations (ICLR)*, 2018.

[75] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks", in *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[76] G. Kim, T. Kwon, and J. C. Ye, "DiffusionCLIP: Text-guided diffusion models for robust image manipulation", in *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[77] H. Kim *et al.*, "Deep video portraits", *ACM Trans. on Graphics (ToG)*, vol. 37, pp. 1–14, 2018.

[78] H. Kim *et al.*, "Neural style-preserving visual dubbing", *ACM Trans. on Graphics (ToG)*, vol. 38, pp. 1–13, 2019.

[79] D. Kingma, T. Salimans, B. Poole, and J. Ho, "Variational diffusion models", in *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[80] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization", in *Int Conf. on Learning Representations (ICLR)*, 2015.

[81] D. P. Kingma and M. Welling, *Auto-encoding variational Bayes*, 2013. arXiv: 1312.6114 [stat.ML].

[82] D. P. Kingma and M. Welling, "An introduction to variational autoencoders", *Foundations and Trends in Machine Learning*, vol. 12, pp. 307–392, 2019.

[83] D. P. Kingma and P. Dhariwal, "Glow: Generative flow with invertible 1x1 convolutions", in *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

[84] D. P. Kingma, T. Salimans, R. Jozefowicz, X. Chen, I. Sutskever, and M. Welling, "Improved variational inference with inverse autoregressive flow", in *Advances in Neural Information Processing Systems (NIPS)*, 2016.

[85] I. Kobyzev, S. J. Prince, and M. A. Brubaker, "Normalizing flows: An introduction and review of current methods", *IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 43, pp. 3964–3979, 2020.

[86] P. Korshunov and S. Marcel, *DeepFakes: A new threat to face recognition? Assessment and detection*, 2018. arXiv: 1812.08685 [cs.CV].

[87] R. Kumar, J. Sotelo, K. Kumar, A. de Brebisson, and Y. Bengio, *ObamaNet: Photo-realistic lip-sync from text*, 2017. arXiv: 1801.01442 [cs.CV].

[88] T. Kynkäänniemi, T. Karras, S. Laine, J. Lehtinen, and T. Aila, "Improved precision and recall metric for assessing generative models", in *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[89] G. Lample, N. Zeghidour, N. Usunier, A. Bordes, L. Denoyer, and M. Ranzato, "Fader networks: Manipulating images by sliding attributes", in *Advances in Neural Information Processing Systems (NIPS)*, 2017.

[90] H. Larochelle and I. Murray, "The neural autoregressive distribution estimator", in *Proc. Int. Conf. on Artificial Intelligence and Statistics (AISTATS)*, 2011.

[91] H. Li *et al.*, "SRDiff: Single image super-resolution with diffusion probabilistic models", *Neurocomputing*, vol. 479, pp. 47–59, 2022.

[92] Y. Li, M.-C. Chang, and S. Lyu, *In ictu oculi: Exposing AI generated fake face videos by detecting eye blinking*, 2018. arXiv: 1806.02877 [cs.CV].

[93] Y. Li, X. Yang, P. Sun, H. Qi, and S. Lyu, "Celeb-DF: A large-scale challenging dataset for deepfake forensics", in *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[94] A. Lindt, P. Barros, H. Siqueira, and S. Wermter, "Facial expression editing with continuous emotion labels", in *IEEE Int. Conf. on Automatic Face & Gesture Recognition (FG)*, 2019.

[95] L. Liu, Y. Ren, Z. Lin, and Z. Zhao, "Pseudo numerical methods for diffusion models on manifolds", in *Int. Conf. on Learning Representations (ICLR)*, 2022.

[96] M. Liu *et al.*, "STGAN: A unified selective transfer network for arbitrary image attribute editing", in *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[97] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild", in *Proc. IEEE/CVF Int. Conf. on Computer Vision (ICCV)*, 2015.

[98] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization", in *Int. Conf. on Learning Representations (ICLR)*, 2019.

[99] Y. Lu, J. Chai, and X. Cao, "Live speech portraits: Real-time photorealistic talking-head animation", *ACM Trans. on Graphics (ToG)*, vol. 40, pp. 1–17, 2021.

[100] A. Lugmayr, M. Danelljan, A. Romero, F. Yu, R. Timofte, and L. Van Gool, "RePaint: Inpainting using denoising diffusion probabilistic models", in *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[101] C. Luo, *Understanding diffusion models: A unified perspective*, 2022. arXiv: 2208.11970 [cs.LG].

[102] X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. Paul Smolley, "Least squares generative adversarial networks", in *Proc. IEEE/CVF Int. Conf. on Computer Vision (ICCV)*, 2017.

[103] A. Mehrabian, J. Russell, J. Russell, and J. Russell, *An Approach to Environmental Psychology*. M.I.T. Press, 1974.

[104] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "NeRF: Representing scenes as neural radiance fields for view synthesis", in *Proc. Eur. Conf. on Computer Vision (ECCV)*, 2020.

[105] Y. Mirsky and W. Lee, "The creation and detection of deepfakes: A survey", *ACM Computing Surveys (CSUR)*, vol. 54, pp. 1–41, 2021.

[106] S. Mohamed and B. Lakshminarayanan, *Learning in implicit generative models*, 2016. arXiv: 1610.03483 [stat.ML].

[107] A. Mollahosseini, B. Hasani, and M. H. Mahoor, "AffectNet: A database for facial expression, valence, and arousal computing in the wild", *IEEE Trans. on Affective Computing*, vol. 10, pp. 18–31, 2017.

[108] J. C. Neves, R. Tolosana, R. Vera-Rodriguez, V. Lopes, H. Proença, and J. Fierrez, "Ganprintr: Improved fakes and evaluation of the state of the art in face manipulation detection", *IEEE Journal of Selected Topics in Signal Processing*, vol. 14, pp. 1038–1048, 2020.

[109] A. Q. Nichol and P. Dhariwal, "Improved denoising diffusion probabilistic models", in *Int. Conf. on Machine Learning (ICML)*, 2021.

[110] A. Q. Nichol *et al.*, "GLIDE: Towards photorealistic image generation and editing with text-guided diffusion models", in *Int. Conf. on Machine Learning (ICML)*, 2022.

[111] V. Nitsch and M. Popp, "Emotions in robot psychology", *Biological Cybernetics*, vol. 108, pp. 621–629, 2014.

[112] A. van den Oord, Y. Li, and O. Vinyals, *Representation learning with contrastive predictive coding*, 2019. arXiv: 1807.03748 [cs.LG].

[113] G. Papamakarios, E. T. Nalisnick, D. J. Rezende, S. Mohamed, and B. Lakshminarayanan, "Normalizing flows for probabilistic modeling and inference", *Journal of Machine Learning Research (JMLR)*, vol. 22, pp. 1–64, 2021.

[114] G. Parisi, "Correlation functions and computer simulations", *Nuclear Physics B*, vol. 180, pp. 378–384, 1981.

[115] O. Patashnik, Z. Wu, E. Shechtman, D. Cohen-Or, and D. Lischinski, "StyleCLIP: Text-driven manipulation of StyleGAN imagery", in *Proc. IEEE/CVF Int. Conf. on Computer Vision (ICCV)*, 2021.

[116] G. Perarnau, J. van de Weijer, B. Raducanu, and J. M. Álvarez, *Invertible conditional GANs for image editing*, 2016. arXiv: 1611.06355 [cs.CV].

[117] R. Plutchik, "The nature of emotions: Human emotions have deep evolutionary roots, a fact that may explain their complexity and provide tools for clinical practice", *American Scientist*, vol. 89, pp. 344–350, 2001.

[118] B. Poole, A. Jain, J. T. Barron, and B. Mildenhall, "DreamFusion: Text-to-3D using 2D Diffusion", in *Int. Conf. on Learning Representations (ICLR)*, 2023.

[119] K. Preechakul, N. Chatthee, S. Wizadwongsa, and S. Suwajanakorn, "Diffusion Autoencoders: Toward a meaningful and decodable representation", in *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[120] A. Pumarola, A. Agudo, A. M. Martinez, A. Sanfeliu, and F. Moreno-Noguer, "GANimation: Anatomically-aware facial animation from a single image", in *Proc. Eur. Conf. on Computer Vision (ECCV)*, 2018.

[121] A. Radford *et al.*, "Learning transferable visual models from natural language supervision", in *Int. Conf. on Machine Learning (ICML)*, 2021.

[122] A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, and M. Chen, *Hierarchical text-conditional image generation with CLIP latents*, 2022. arXiv: 2204.06125 [cs.CV].

[123] A. Razavi, A. Van den Oord, and O. Vinyals, "Generating diverse high-fidelity images with VQ-VAE-2", in *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[124] D. Rezende and S. Mohamed, "Variational inference with normalizing flows", in *Int. Conf. on Machine Learning (ICML)*, 2015.

[125] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models", in *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[126] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation", in *Int. Conf. on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, 2015.

[127] A. Rossler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Niessner, "FaceForensics++: Learning to detect manipulated facial images", in *Proc. IEEE/CVF Int. Conf. on Computer Vision (ICCV)*, 2019.

[128] A. Rössler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Nießner, *FaceForensics: A large-scale video dataset for forgery detection in human faces*, 2018. arXiv: 1803.09179 [cs.CV].

[129] J. Russell and A. Mehrabian, "Evidence for a three-factor theory of emotions", *Journal of Research in Personality*, vol. 11, pp. 273–294, 1977.

[130] C. Saharia *et al.*, "Palette: Image-to-image diffusion models", in *ACM SIGGRAPH*, 2022.

[131] C. Saharia *et al.*, "Photorealistic text-to-image diffusion models with deep language understanding", in *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[132] M. S. Sajjadi, O. Bachem, M. Lucic, O. Bousquet, and S. Gelly, "Assessing generative models via precision and recall", in *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

[133] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans", in *Advances in Neural Information Processing Systems (NIPS)*, 2016.

[134] T. Salimans, A. Karpathy, X. Chen, and D. P. Kingma, "PixelCNN++: A PixelCNN implementation with discretized logistic mixture likelihood and other modifications", in *Int. Conf. on Learning Representations (ICLR)*, 2017.

[135] A. V. Savchenko, "Facial expression and attributes recognition based on multi-task learning of lightweight neural networks", in *IEEE Int. Symp. on Intelligent Systems and Informatics (SISY)*, 2021.

[136] A. V. Savchenko, "Video-based frame-level facial analysis of affective behavior on mobile devices using EfficientNets", in *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2022.

[137] A. V. Savchenko, L. V. Savchenko, and I. Makarov, "Classifying emotions and engagement in online learning based on a single facial expression recognition neural network", *IEEE Trans. on Affective Computing*, vol. 13, pp. 2132–2143, 2022.

[138] S. Schneider, A. Baevski, R. Collobert, and M. Auli, "wav2vec: Unsupervised pre-training for speech recognition", in *Proc. ISCA Interspeech*, 2019.

[139] C. Schuhmann *et al.*, *LAION-400M: Open dataset of CLIP-filtered 400 Million image-text pairs*, 2021. arXiv: 2111.02114 [cs.CV].

[140] S. Shen *et al.*, "DiffTalk: Crafting diffusion models for generalized audio-driven portraits animation", in *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[141] C. Sheng *et al.*, *Deep learning for visual speech analysis: A survey*, 2022. arXiv: 2205.10839 [cs.CV].

[142] B. Shi, W.-N. Hsu, K. Lakhotia, and A. Mohamed, "Learning audio-visual speech representation by masked multimodal cluster prediction", in *Int. Conf. on Learning Representations (ICLR)*, 2022.

[143] B. Shi, W.-N. Hsu, and A. Mohamed, *Robust self-supervised audio-visual speech recognition*, 2022. arXiv: 2201.01763 [cs.SD].

[144] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition", in *Int. Conf. on Learning Representations (ICLR)*, 2015.

[145] U. Singer *et al.*, "Make-A-Video: Text-to-video generation without text-video data", in *Int. Conf. on Learning Representations (ICLR)*, 2023.

[146] J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli, "Deep unsupervised learning using nonequilibrium thermodynamics", in *Int. Conf. on Machine Learning (ICML)*, 2015.

[147] C. K. Sønderby, T. Raiko, L. Maaløe, S. K. Sønderby, and O. Winther, "Ladder variational autoencoders", in *Advances in Neural Information Processing Systems (NIPS)*, 2016.

[148] J. Song, C. Meng, and S. Ermon, "Denoising diffusion implicit models", in *Int. Conf. on Learning Representations (ICLR)*, 2021.

[149] Y. Song and S. Ermon, "Generative modeling by estimating gradients of the data distribution", in *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[150] Y. Song and S. Ermon, "Improved techniques for training score-based generative models", in *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[151] Y. Song, S. Garg, J. Shi, and S. Ermon, "Sliced score matching: A scalable approach to density and score estimation", in *Uncertainty in Artificial Intelligence*, 2020.

[152] Y. Song and D. P. Kingma, *How to train your energy-based models*, 2021. arXiv: 2101.03288 [cs.LG].

[153] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, "Score-based generative modeling through stochastic differential equations", in *Int. Conf. on Learning Representations (ICLR)*, 2021.

[154] M. Stypułkowski, K. Vougioukas, S. He, M. Zięba, S. Petridis, and M. Pantic, *Diffused Heads: Diffusion models beat GANs on talking-face generation*, 2023. arXiv: 2301.03396 [cs.CV].

[155] S. Suwajanakorn, S. M. Seitz, and I. Kemelmacher-Shlizerman, "Synthesizing Obama: Learning lip sync from audio", *ACM Trans. on Graphics (ToG)*, vol. 36, pp. 1–13, 2017.

[156] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision", in *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[157] J. Thies, M. Zollhofer, M. Stamminger, C. Theobalt, and M. Nießner, "Face2Face: Real-time face capture and reenactment of RGB videos", in *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[158] J. Thies, M. Zollhöfer, and M. Nießner, "Deferred neural rendering: Image synthesis using neural textures", *ACM Trans. on Graphics (ToG)*, vol. 38, pp. 1–12, 2019.

[159] J. Thies, M. Zollhöfer, M. Nießner, L. Valgaerts, M. Stamminger, and C. Theobalt, "Real-time expression transfer for facial reenactment.", *ACM Trans. on Graphics (ToG)*, vol. 34, pp. 183–1, 2015.

[160] J. Thies, M. Zollhöfer, C. Theobalt, M. Stamminger, and M. Nießner, "HeadOn: Real-time reenactment of human portrait videos", *ACM Trans. on Graphics (ToG)*, vol. 37, pp. 1–13, 2018.

[161] R. Tolosana, R. Vera-Rodriguez, J. Fierrez, A. Morales, and J. Ortega-Garcia, "Deepfakes and beyond: A survey of face manipulation and fake detection", *Information Fusion*, vol. 64, pp. 131–148, 2020.

[162] S. Tripathy, J. Kannala, and E. Rahtu, "FACEGAN: Facial attribute controllable reenactment GAN", in *Proc. IEEE/CVF Winter Conf. on Appl. of Computer Vision (WACV)*, 2021.

[163] S. Tripathy, J. Kannala, and E. Rahtu, "ICface: Interpretable and controllable face reenactment using GANs", in *Proc. IEEE/CVF Winter Conf. on Appl. of Computer Vision (WACV)*, 2020.

[164] A. Ulhaq, N. Akhtar, and G. Pogrebna, *Efficient diffusion models for vision: A survey*, 2022. arXiv: 2210.09292 [cs.CV].

[165] B. Uria, I. Murray, and H. Larochelle, "RNADE: The real-valued neural autoregressive density-estimator", in *Advances in Neural Information Processing Systems (NIPS)*, 2013.

[166] A. Van den Oord, N. Kalchbrenner, L. Espeholt, O. Vinyals, A. Graves, *et al.*, "Conditional image generation with PixelCNN decoders", in *Advances in Neural Information Processing Systems (NIPS)*, 2016.

[167] A. Van Den Oord, O. Vinyals, *et al.*, "Neural discrete representation learning", in *Advances in Neural Information Processing Systems (NIPS)*, 2017.

[168] A. Vaswani *et al.*, "Attention is all you need", in *Advances in Neural Information Processing Systems (NIPS)*, 2017.

[169] C. Villani, *Optimal transport: Old and new*. Springer, 2009, vol. 338.

[170] P. Vincent, "A connection between score matching and denoising autoencoders", *Neural Computation*, vol. 23, pp. 1661–1674, 2011.

[171] H. Wang *et al.*, "CosFace: Large margin cosine loss for deep face recognition", in *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[172] K. Wang *et al.*, "MEAD: A large-scale audio-visual dataset for emotional talking-face generation", in *Proc. Eur. Conf. on Computer Vision (ECCV)*, 2020.

[173] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity", *IEEE Trans. on Image Processing*, vol. 13, pp. 600–612, 2004.

[174] L. N. Wasserstein, "Markov processes over denumerable products of spaces, describing large systems of automata", *Problemy Peredachi Informatsii*, vol. 5, pp. 64–72, 1969.

[175] J. Wolleb, R. Sandkühler, F. Bieder, P. Valmaggia, and P. C. Cattin, "Diffusion models for implicit image segmentation ensembles", in *Int. Conf. on Medical Imaging with Deep Learning (MIDL)*, 2022.

[176] Z. Xiao, K. Kreis, and A. Vahdat, "Tackling the generative learning trilemma with denoising diffusion GANs", in *Int. Conf. on Learning Representations (ICLR)*, 2022.

[177] L. Xie and Z.-Q. Liu, "Realistic mouth-synching for speech-driven talking face using articulatory modelling", *IEEE Trans. on Multimedia (TMM)*, vol. 9, pp. 500–510, 2007.

[178] D. Yi, Z. Lei, S. Liao, and S. Z. Li, *Learning face representation from scratch*, 2014. arXiv: 1411.7923 [cs.CV].

[179] J. Yu *et al.*, "Vector-quantized image modeling with improved VQGAN", in *Int. Conf. on Learning Representations (ICLR)*, 2022.

[180] E. Zakharov, A. Shysheya, E. Burkov, and V. Lempitsky, "Few-shot adversarial learning of realistic neural talking head models", in *Proc. IEEE/CVF Int. Conf. on Computer Vision (ICCV)*, 2019.

[181] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric", in *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[182] Y. Zhang, H. Jiang, Y. Miura, C. D. Manning, and C. P. Langlotz, *Contrastive learning of medical visual representations from paired images and text*, 2020. arXiv: 2010.00747 [cs.CV].

[183] J. Zhao, M. Mathieu, and Y. LeCun, *Energy-based generative adversarial network*, 2016. arXiv: 1609.03126 [cs.LG].

[184] M. Zhao, F. Bao, C. Li, and J. Zhu, "EGSDE: Unpaired image-to-image translation via energy-guided stochastic differential equations", *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[185] H. Zhou, Y. Liu, Z. Liu, P. Luo, and X. Wang, "Talking face generation by adversarially disentangled audio-visual representation", in *Proc. AAAI Conf. on Artificial Intelligence*, 2019.

[186] Y. Zhou, X. Han, E. Shechtman, J. Echevarria, E. Kalogerakis, and D. Li, "MakeItTalk: Speaker-aware talking-head animation", *ACM Trans. on Graphics (ToG)*, vol. 39, pp. 1–15, 2020.

[187] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks", in *Proc. IEEE/CVF Int. Conf. on Computer Vision (ICCV)*, 2017.