



NATIONAL TECHNICAL UNIVERSITY OF ATHENS
SCHOOL OF ELECTRICAL AND COMPUTER ENGINEERING
DIVISION OF SIGNALS, CONTROL AND ROBOTICS

Morphological Diffusion for Handwritten Text Generation

DIPLOMA THESIS

of

Dimitrios Bakalis

Supervisor: Petros Maragos
Professor NTUA

Co-supervisors: Georgios Retsinas
Postdoctoral Researcher NTUA
Ioannis Kordonis
Postdoctoral Researcher NTUA

COMPUTER VISION, SPEECH COMMUNICATION AND SIGNAL PROCESSING GROUP
Athens, October 2023



National Technical University of Athens
School of Electrical and Computer Engineering
Division of Signals, Control and Robotics
Computer Vision, Speech Communication and Signal Processing Group

Morphological Diffusion for Handwritten Text Generation

DIPLOMA THESIS

of

Dimitrios Bakalis

Supervisor: Petros Maragos
Professor NTUA

Co-supervisor: Georgios Retsinas
Postdoctoral Researcher NTUA
Ioannis Kordonis
Postdoctoral Researcher NTUA

Approved by the examination committee on 20th October, 2023.

.....
Petros Maragos
Professor NTUA

.....
Gerasimos Potamianos
Associate Professor UTH

.....
Konstantinos Tzafestas
Associate Professor NTUA

Athens, October 2023

.....
DIMITRIOS BAKALIS
Graduate of Electrical and
Computer Engineering NTUA

Copyright © – All rights reserved Dimitrios Bakalis, 2023.

The copying, storage and distribution of this diploma thesis, all or part of it, is prohibited for commercial purposes. Reprinting, storage and distribution for nonprofit, educational or of a research nature is allowed, provided that the source is indicated and that this message is retained.

The content of this thesis does not necessarily reflect the views of the Department, the Supervisor, or the committee that approved it.

Περίληψη

Η Γενετική Τεχνητή Νοημοσύνη βρίσκεται στην πρώτη γραμμή της σύγχρονης τεχνολογίας, επεκτείνοντας τα όρια του τι μπορούν να δημιουργήσουν και να φανταστούν οι υπολογιστικές μηχανές. Το συγκεκριμένο πεδίο αντιπροσωπεύει έναν επαναστατικό συνδυασμό της επιστήμης των υπολογιστών, της μηχανικής μάθησης και των νευρωνικών δικτύων, επιτρέποντας στους υπολογιστές να δημιουργούν πρωτότυπο περιεχόμενο, που μιμείται την ανθρώπινη δημιουργικότητα. Η ανάπτυξη της γενετικής τεχνητής νοημοσύνης έχει προκαλέσει μια επανάσταση σε διάφορους τομείς, από τη δημιουργία περιεχομένου και την ψυχαγωγία έως την υγεία και τη χρηματοοικονομία.

Σε αυτήν τη διπλωματική, χρησιμοποιούμε μοντέλα διάχυσης, για να αντιμετωπίσουμε το περίπλοκο πρόβλημα της Παραγωγής Χειρόγραφου Κειμένου, εστιάζοντας στο να το συνθέσουμε βάσει περιεχομένου κειμένου και στυλ γραφής. Εμπνεόμενοι από πρόσφατες εργασίες στον τομέα των γενικευμένων διαχύσεων, παρουσιάζουμε μία νέα μη γραμμική διαδικασία διάχυσης βασισμένη σε έναν θεμελιώδη τελεστή των μορφολογικών μαθηματικών, την διαστολή.

Αρχικά, παρουσιάζουμε τα πειράματά μας στα σύνολα δεδομένων MNIST και CIFAR-10, παρέχοντας ένα βασικό proof-of-concept για την προσέγγισή μας. Έπειτα, συγκρίνουμε τη μεθοδολογία μας με πρόσφατες εξελίξεις στις γενικευμένες διαχύσεις, αναδεικνύοντας μία ανταγωνιστική απόδοση. Επιπλέον, προτείνουμε μια διαδικασία δύο σταδίων, συμπληρωμένη από την ενσωμάτωση ενός Παραγωγικού Ανταγωνιστικού Δικτύου (GAN), ώστε να επιτρέψουμε τη παραγωγή εικόνων υπό συνθήκη για το σύνολο δεδομένων MNIST. Η αξία της συγκεκριμένης προσέγγισης αναδεικνύεται ιδιαίτερα από το γεγονός ότι υπερβαίνει τις κλασικές διαδικασίες διάχυσης, όταν αποτελούνται από έναν περιορισμένο αριθμό βημάτων.

Στη συνέχεια, εστιάζουμε την προσοχή μας στο περίπλοκο πρόβλημα της Παραγωγής Χειρόγραφου Κειμένου. Θα επιχειρήσουμε να παρέχουμε περαιτέρω βελτιστοποιήσεις, ενισχύοντας το ήδη υπάρχον state-of-the-art μοντέλο με πιο αποδοτικούς αλγόριθμους παραγωγής εικόνων, όπως αναφέρεται στη βιβλιογραφία. Επιπλέον, βελτιστοποιούμε την αποτελεσματικότητα του μοντέλου με την εισαγωγή της έννοιας της μορφολογικής διάχυσης. Συγκεκριμένα, αποκλίνουμε από το συμβατικό γκαουσιανό πλαίσιο και τροποποιούμε τη συνάρτηση καταστροφής εικόνων της διαδικασίας διάχυσης, υιοθετώντας τη μορφολογική διάχυση. Αυτή η μετατροπή παρέχει ανταγωνιστικά αποτελέσματα σε σύγκριση με τα κορυφαία μοντέλα όσον αφορά την ποιότητα, ενώ, ταυτόχρονα, μειώνει σημαντικά τις υπολογιστικές απαιτήσεις κατά τη διάρκεια, τόσο της εκπαίδευσης, όσο και της διαδικασίας παραγωγής εικόνων.

Λέξεις Κλειδιά — Βαθιά Μάθηση, Γενετική Τεχνητή Νοημοσύνη, Μορφολογικά Μαθηματικά, Μοντέλα Διάχυσης, Μορφολογική Διάχυση, Παραγωγή Χειρόγραφου Κειμένου, Λανθάνουσα Διάχυση, Ψυχρή Διάχυση

Abstract

Generative Artificial Intelligence, often referred to simply as "Generative AI", stands at the forefront of modern technology, pushing the boundaries of what machines can create and imagine. This remarkable field represents a ground breaking fusion of computer science, machine learning, and neural networks, enabling computers to generate original content that mimics human creativity. The emergence of generative AI has sparked a revolution in various domains, from content creation and entertainment to healthcare and finance. It has given birth to powerful applications, such as natural language generation, style transfer in images, and autonomous creative agents that can inspire, inform, and entertain.

In this thesis, we use Diffusion Models to address the intricate challenge of Handwritten Text Generation (HTG), with a focus on conditioning it on textual content and writing style. Drawing inspiration from recent breakthroughs in the realm of generalized diffusions, we introduce a novel non-linear diffusion process rooted in a fundamental operation of morphological mathematics, specifically, the dilation.

We initially present our baseline experiments conducted on the MNIST and CIFAR-10 datasets, serving as a foundational proof-of-concept for our novel approach. We compare our methodology with recent advancements in generalized diffusions, shedding light on its comparative performance. Furthermore, we advocate for a two-stage approach, complemented by the inclusion of a Generative Adversarial Network (GAN), to facilitate conditional generation within the MNIST dataset. This approach proves its mettle by outperforming classic diffusion frameworks, when operating within a constrained number of timesteps.

Subsequently, we pivot our focus towards the intricate task of Handwritten Text Generation. In a quest for optimization, we enhance the existing state-of-the-art model with more efficient sampling algorithms, as documented in the bibliography. Furthermore, we streamline the model's efficiency by introducing the concept of morphological diffusion. Specifically, we deviate from the conventional Gaussian framework and modify the degradation function within the latent diffusion process to embrace morphological diffusion. This transformation yields competitive results, rivalling the state-of-the-art, all while significantly reducing the computational demands imposed during both training and sampling procedures.

Keywords — Deep Learning, Generative AI, Morphological Mathematics, Diffusion Models, Morphological Diffusion, Handwritten Text Generation, Latent Diffusion, Cold Diffusion

Ευχαριστίες

Θα ήθελα, αρχικά, να ευχαριστήσω θερμά τον Καθηγητή Πέτρο Μαραγκό για την εμπιστοσύνη που μου έδειξε, ώστε να εκπονήσω τη διπλωματική μου εργασία στο Εργαστήριο Όρασης Υπολογιστών, Επικοινωνίας Λόγου και Επεξεργασίας Σημάτων.

Στη συνέχεια, θα ήθελα, επίσης, να ευχαριστήσω τους συνεπιβλέποντες, Δρ. Γεώργιο Ρετσινά και Δρ. Ιωάννη Κορδώνη, για τις συμβουλές, την καθοδήγηση και την συνεισφορά τους στην ολοκλήρωση της συγκεκριμένης εργασίας.

Τέλος, οφείλω ένα μεγάλο ευχαριστώ στους γονείς μου, Νίκο και Στέλλα, για την υποστήριξη και την κατανόησή τους καθ'όλη τη διάρκεια των σπουδών μου, καθώς και στους υπόλοιπους κοντινούς μου ανθρώπους, με τους οποίους που μοιραστήχαμε αυτά τα χρόνια.

Μπακάλης Δημήτριος
Οκτώβριος 2023

Contents

Contents	xiii
List of Figures	xv
List of Tables	xviii
1 Introduction	29
1.1 Morphological Mathematics	30
1.2 Diffusion Models	31
1.3 Handwritten Text Recognition and Generation	31
1.4 Outline	32
2 Theoretical Background	35
2.1 Introduction to Deep Learning	36
2.1.1 Architectures of Deep Neural Networks	36
2.1.2 Training	43
2.1.3 Introduction to Generative Models	46
2.2 Introduction to Morphological Mathematics	48
2.2.1 Structuring Elements	48
2.2.2 Morphological Operations	49
2.3 Introduction to Diffusion Models	51
2.3.1 Background	51
2.3.2 Training-Sampling	52
2.3.3 Improvements	53
2.3.4 Score-based generative modeling with stochastic differential equations	54
2.4 Introduction to Handwritten Text Recognition and Generation	56
2.4.1 Handwritten Text Recognition	56
2.4.2 Handwritten Text Generation	57
3 Related Work	59
3.1 Cold Diffusion	60
3.1.1 Introduction	60
3.1.2 Sampling	60
3.1.3 Generalized Diffusions	61
3.2 Latent Diffusion Models	64
3.2.1 Introduction	64
3.2.2 Latent Diffusion - Conditioning Mechanisms	64
3.2.3 Applications	67
3.3 Handwritten-Text Generation	69
3.3.1 GANWriting	69
3.3.2 SmartPatch	71
3.3.3 WordStylist	74
4 Proposed Method	77
4.1 Morphological Diffusion	78

4.1.1	Motivation	78
4.1.2	Image Reconstruction	79
4.1.3	Conditional Generation	79
4.2	Extensions in the WordStylist model	83
4.2.1	WordStylist Sampling Limitations	83
4.2.2	DDIM Sampling	83
4.2.3	PNDM Sampling	84
4.3	Morphological Diffusion for Handwritten-Text Generation	86
5	Experimental Results	89
5.1	Datasets	90
5.1.1	MNIST	90
5.1.2	CIFAR-10	90
5.1.3	IAM	90
5.2	Evaluation Metrics	91
5.2.1	Fréchet Inception Distance	91
5.2.2	Structural Similarity	91
5.2.3	Root Mean Square Error	92
5.3	Implementation	92
5.3.1	Morphological Diffusion	92
5.3.2	Extensions in the WordStylist model	95
5.3.3	Morphological Diffusion for Handwritten-Text Generation	95
5.4	Results	95
5.4.1	Morphological Diffusion	95
5.4.2	Extensions in the WordStylist model	100
5.4.3	Morphological Diffusion for Handwritten-Text Generation	101
6	Conclusion and Future Work	107
6.1	Conclusion	108
6.2	Future Work	108
A	Bibliography	111

List of Figures

0.0.1	Η αρχιτεκτονική ενός VAE. Από εδώ	3
0.0.2	Η αρχιτεκτονική ενός Παραγωγικού Ανταγωνιστικού Δικτύου. Από εδώ	3
0.0.3	Βασικοί Μορφολογικοί Τελεστές. Από [17]	4
0.0.4	Η διαδικασία διάχυσης σχηματικά. Από [20]	5
0.0.5	Δύο διαφορετικά χρονοδιαγράμματα (γραμμικό και συνημιτονικό) για τη διαδικασία διάχυσης. Από [21]	5
0.0.6	Σύγκριση των δύο αλγορίθμων. Από [33]	8
0.0.7	Αρχιτεκτονική Μοντέλων Λανθάνουσας Διάχυσης. Από [32]	8
0.0.8	Η αρχιτεκτονική του μοντέλου GANWriting. Από [51]	9
0.0.9	GANWriting Artifacts. Από [30]	10
0.0.10	Η αρχιτεκτονική του SmartPatch. Από [30]	10
0.0.11	Η αρχιτεκτονική του Wordstylist. Από [31]	11
0.0.12	Η μορφολογία του MNIST.	12
0.0.13	Η μορφολογία του IAM.	12
0.0.14	Παράδειγμα διάχυσης για κάθε ψηφίο του MNIST.	13
0.0.15	Παράδειγμα διάχυσης για κάθε κλάση του CIFAR-10.	13
0.0.16	Η επίδραση του closing στις παραγόμενες εικόνες του cGAN.	14
0.0.17	Παραγωγή εικόνων με διαφορετικό αριθμό βημάτων από 100 έως 1000. Από [31]	14
0.0.18	Denosing Diffusion Implicit Models. Από [68]	15
0.0.19	Αρχικά προεπεξεργασμένα δείγματα του IAM.	16
0.0.20	Τα 4 κανάλια από τις λανθάνουσες αναπαραστάσεις των δειγμάτων του IAM.	16
0.0.21	Τα 4 κανάλια από τις λανθάνουσες αναπαραστάσεις των δειγμάτων του IAM μετά από διαστολή.	16
0.0.22	Ανακατασκευή εικόνων του MNIST.	19
0.0.23	Ανακατασκευή εικόνων του CIFAR-10.	20
0.0.24	Παραγωγή εικόνων του MNIST υπό συνθήκη.	21
0.0.25	Αποτελέσματα με εφαρμογή του αλγορίθμου DDIM. Κάθε σειρά περιέχει 5 αποτελέσματα για τη ίδια λέξη ('what') με 5 τυχαία στυλ γραφής και αριθμό βημάτων a) 20, b) 50, c) 100, d) 200.	23
0.0.26	Σύγκριση αποτελεσμάτων για διαφορετικούς αλγορίθμους παραγωγής εικόνων για 100 βήματα. Κάθε σειρά περιέχει 5 αποτελέσματα για τη λέξη ('what') με 5 τυχαία στυλ γραφής και αλγόριθμο a) default, b) DDIM, c) f-PNDM, d) s-PNDM.	23
0.0.27	Ποιοτικά αποτελέσματα για την ανακατασκευή εικόνων του IAM.	24
0.0.28	Out-Of-Vocabulary αποτελέσματα για τη λέξη "dance"	25
0.0.29	Αποτελέσματα για διαφορετικά στυλ γραφής της λέξης "what"	25
1.1.1	Basic Morphological Operations. From [17]	30
1.2.1	Diffusion Process. From here	31
1.3.1	An example of an HTR system. From here	32
2.1.1	CNN Architecture. From here	36
2.1.2	Convolution on a 7×7 image with a 3×3 kernel. From here	37
2.1.3	The effect of max-pooling (left) and average-pooling (right) layers. From here	37
2.1.4	The effect of dropout in the training. From here	38
2.1.5	The Architecture of ResNet-12. From here	39
2.1.6	Residual Block. From [77]	40
2.1.7	The Architecture of U-Net. From [60]	40
2.1.8	The Architecture of an unrolled RNN. From here	41

2.1.9	The Architecture of an LSTM. From here	42
2.1.10	The Architecture of a GRU. From here	42
2.1.11	The Sigmoid, ReLU, SiLU and GELU activation functions.	45
2.1.12	Variational Autoencoder. From here	46
2.1.13	Generative Adversarial Networks. From here	47
2.2.1	An example of how morphological operations are capable of generating outcomes that convolution-based filtering is not. From [88]	48
2.2.2	Flat and Non-Flat Structuring Elements. From here	49
2.2.3	Morphological Operations with a 5×5 square SE. From [88]	50
2.3.1	The directed graphical model. From [20]	51
2.3.2	Sampled from linear (top) and cosine (bottom) schedules. From [21]	54
2.3.3	$\bar{\alpha}_t$ throughout diffusion in the linear and cosine schedules. From [21]	54
2.3.4	FID improvement by replacing Addition + GroupNorm [20] with AdaGN [59]. From [59]	54
2.3.5	Solving a reverse-time SDE yields a score-based generative model. From [44]	55
2.3.6	Overview of score-based generative modeling through SDEs. From [44]	55
2.4.1	Online Handwritten Text. From [25]	56
2.4.2	Offline Handwritten Text. From here	56
3.1.1	Comparison of algorithms 1 and 2. From [33]	62
3.1.2	Comparison of deblurred images to the degraded and the original ones. From [33]	62
3.1.3	Comparison of reconstructed images from inpainting to the degraded and the original ones. From [33]	63
3.1.5	Comparison of reconstructed images from snowification to the degraded and the original ones. From [33]	63
3.1.4	Comparison of reconstructed images from downsampling to the degraded and the original ones. From [33]	64
3.2.1	Latent Diffusion Models. From Improving Diffusion Models as an Alternative To GANs, Part 2	65
3.2.2	Conditional (via concatenation or cross-attention) Latent Diffusion Models. From [32]	65
3.2.3	Object removal with the inpainting LDM. From [32]	67
3.2.4	Upsampling from 64 to 256 results in ImageNet with the super-resolution LDM. From [32]	68
3.3.1	GANWriting Architecture. From [51]	70
3.3.2	Word Recognizer Architecture. From [51]	70
3.3.3	Image generation: a) In-Vocabulary - Seen style, b) In-Vocabulary - Unseen style, c) Out-Of-Vocabulary - Seen style, d) Out-Of-Vocabulary - Unseen Style. From [51]	71
3.3.4	GANWriting Artifacts. From [30]	71
3.3.5	SmartPatch Architecture. From [30]	72
3.3.6	SmartPatch Methods. From [30]	73
3.3.7	Comparison of generated images from SmartPatch and GANWriting [51]. From [30]	73
3.3.8	Wordstylist Architecture. From [31]	74
3.3.9	Comparison of generated images from Wordstylist, SmartPatch [30] and GANWriting [51]. From [31]	75
4.1.1	MNIST Morphology.	78
4.1.2	IAM Morphology.	78
4.1.3	MNIST degraded examples for each digit.	79
4.1.4	CIFAR-10 degraded examples for each class.	79
4.1.5	MNIST degraded examples for each digit.	80
4.1.6	Dilated noise as the first image of sampling.	81
4.1.7	The effect of closing on the cGAN generated samples.	81
4.1.8	cGAN samples.	82
4.1.9	cGAN samples after closing.	82
4.2.1	Sampling with different number of timesteps ranging from 100 to 1000. From [31]	83
4.2.2	Graphical models for diffusion and non-markovian inference models. From [68]	83
4.3.1	Original preprocessed IAM samples.	86
4.3.2	The 4 channels of the latent representations of IAM samples.	86
4.3.3	The 4 channels of the dilated latent representations of IAM samples.	87

5.1.1	MNIST instances	90
5.1.2	CIFAR-10 instances	90
5.1.3	IAM preprocessed instances	91
5.3.1	The architecture of the U-net used for Image Reconstruction.	93
5.3.2	The architecture of the U-net used for Conditional Generation. * Spatial Transformer Blocks were only in the first Block group for Up and Down Blocks.	94
5.3.3	The architecture of the GAN used for generating the initial masks.	94
5.4.1	Image Reconstruction Results for MNIST.	96
5.4.2	Image Reconstruction Results for CIFAR-10.	97
5.4.3	Conditional Generation Results for MNIST.	98
5.4.4	WordStylist Results with DDIM Sampling. Each row contains 5 results of the same word ('what') generated with 5 random writing styles with different number sampling steps a) with 20 sampling steps, b) with 50 sampling steps, c) with 100 sampling steps, d) with 200 sampling steps.	100
5.4.5	WordStylist Results with f-PNDM Sampling. Each row contains 5 results of the same word ('what') generated with 5 random writing styles with different number sampling steps a) with 20 sampling steps, b) with 50 sampling steps, c) with 100 sampling steps, d) with 200 sampling steps.	100
5.4.6	WordStylist Results with s-PNDM Sampling. Each row contains 5 results of the same word ('what') generated with 5 random writing styles with different number sampling steps a) with 20 sampling steps, b) with 50 sampling steps, c) with 100 sampling steps, d) with 200 sampling steps.	101
5.4.7	Comparison of WordStylist Sampling Results with 100 steps. Each row contains 5 results of the same word ('what') generated with 5 random writing styles with different sampling algorithm a) with default sampling, b) with DDIM sampling, c) with f-PNDM sampling, d) with s-PNDM sampling.	101
5.4.8	Qualitative results for reconstructed IAM samples.	102
5.4.9	Out-Of-Vocabulary results for "dance"	103
5.4.10	Out-Of-Vocabulary results for "troll"	103
5.4.11	Out-Of-Vocabulary results for "waist"	103
5.4.12	Out-Of-Vocabulary results of lower quality	104
5.4.13	Results for different styles of "what"	104

List of Tables

0.1	Comparison of Metrics for the Proposed Diffusions of [33] and ours for MNIST [62].	22
0.2	Comparison of Metrics for the Proposed Diffusions of [33] and ours for CIFAR-10 [64].	22
0.3	Comparison of Metrics for Different Conditional Generation Models with 10 sampling steps for MNIST [62].	22
0.4	FID score comparison for our model with state-of-the-art.	25
0.5	Style Adaptation comparison for our model with state-of-the-art.	26
2.1	Properties of Morphological Operations	50
3.1	Metrics for quality of deblurred images . From [33]	62
3.2	Metrics for quality of reconstructed images from inpainting. From [33]	63
3.3	Metrics for quality of reconstructed images from downsampling. From [33]	64
3.4	Comparison of different methods using various metrics. From [32]	66
3.5	Comparison of quantitative results for the ipainting model with the state-of-the-art. †recomputed on [32] test set, since the one used in [100] was not available. From [32]	67
3.6	Upsampling results (4×) using the super-resolution LDM. †FID on validation set, ‡FID on training set. From [32]	68
3.7	FID score comparison for the models. Results from [31]	75
3.8	Style Adaptation results for the models. From [31]	75
5.1	Comparison of Metrics for the Proposed Diffusions of [33] and ours for MNIST [62].	99
5.2	Comparison of Metrics for the Proposed Diffusions of [33] and ours for CIFAR-10 [64].	99
5.3	Comparison of Metrics for Different Conditional Generation Models with 10 sampling steps for MNIST [62].	100
5.4	FID score comparison for our model with state-of-the-art.	105
5.5	Style Adaptation comparison for our model with state-of-the-art.	105

Εκτεταμένη Περίληψη στα Ελληνικά

Εισαγωγή

Στη συγκεκριμένη διπλωματική θα ασχοληθούμε με το πρόβλημα της παραγωγής εικόνων χειρόγραφου κειμένου. Προκειμένου να αντιμετωπίσουμε τις δυσκολίες του συγκεκριμένου προβλήματος θα χρησιμοποιήσουμε ένα μοντέλο λανθάνουσας διάχυσης, το οποίο θα επεκτείνουμε, ώστε η διάχυση να πραγματοποιείται με μορφολογική διαστολή, λόγω της αντιστοιχίας του συγκεκριμένου τελεστή με το σύνολο δεδομένων IAM. Η συγκεκριμένη περίληψη θα χωριστεί στις ακόλουθες ενότητες:

- Εισαγωγή
- Θεωρητικό Υπόβαθρο
- Σχετική Βιβλιογραφία
- Προτεινόμενη Μέθοδος
- Πειραματικά Αποτελέσματα
- Συμπεράσματα και Μελλοντικές Επεκτάσεις

Θεωρητικό Υπόβαθρο

Εισαγωγή στη Βαθιά Μάθηση

Συνελικτικά Νευρωνικά Δίκτυα (CNNs)

Τα Συνελικτικά Νευρωνικά Δίκτυα (CNNs) αποτελούν έναν από τους βασικούς πυλώνες της βαθιάς μάθησης, ιδιαίτερα σε εργασίες που σχετίζονται με τα οπτικά δεδομένα. Εμπνεόμενα από το ανθρώπινο οπτικό σύστημα και τον τρόπο με τον οποίο αντιλαμβάνομαστε τα αντικείμενα γύρω μας, τα CNNs έχουν ξαναορίσει τα όρια της αναγνώρισης και ταξινόμησης εικόνων.

Σε αντίθεση με τα παραδοσιακά νευρωνικά δίκτυα, τα οποία επεξεργάζονται τις εισόδους με πλήρως συνδεδεμένο τρόπο, τα CNNs εκμεταλλεύονται τη χωρική φύση των εικόνων. Αναγνωρίζουν ότι τα pixels, που βρίσκονται κοντά το ένα στο άλλο σε μια εικόνα συχνά σχετίζονται περισσότερο από τα pixels που απέχουν περισσότερο. Αυτή η κατανόηση επιτρέπει στα CNNs να αποτυπώνουν τοπικά μοτίβα, όπως οι ακμές ή οι υφές, στα αρχικά στρώματα, συναρμολογώντας τα σταδιακά σε πιο περίπλοκες δομές, όπως σχήματα ή αντικείμενα, σε βαθύτερα στρώματα [1].

Όπως υπονοείται και από το όνομά τους, η κύρια λειτουργία των CNNs είναι η συνέλιξη. Πρόκειται για ένα εξειδικευμένο είδος γραμμικής λειτουργίας, όπου ένα μικρό φίλτρο ή πυρήνας "γλιστρά" πάνω από τα εισαγόμενα δεδομένα (όπως μια εικόνα), για να παράγει έναν χάρτη χαρακτηριστικών, μετασχηματίζοντας αποτελεσματικά τα δεδομένα με βάση το μοτίβο του φίλτρου. Αυτή η λειτουργία βοηθά το δίκτυο να επικεντρωθεί σε τοπικά χαρακτηριστικά.

Επαναληπτικά Νευρωνικά Δίκτυα (RNNs)

Στον τομέα των νευρωνικών δικτύων, τα Επαναληπτικά Νευρωνικά Δίκτυα (RNNs) [1] ξεχωρίζουν ως το πλέον κατάλληλο μοντέλο για ακολουθιακά δεδομένα. Εν αντιθέσει με τα παραδοσιακά feedforward νευρωνικά δίκτυα,

που επεξεργάζονται τις εισόδους αυτόνομα, τα RNNs διαθέτουν τη μοναδική ικανότητα να διατηρούν μια μνήμη προηγούμενων εισόδων στην εσωτερική τους κατάσταση. Αυτό το χαρακτηριστικό τα καθιστά ιδιαίτερα κατάλληλα για εργασίες, όπου οι χρονικές δυναμικές και το πλαίσιο από προηγούμενα βήματα είναι ουσιαστικά, όπως η πρόβλεψη χρονοσειρών, η αναγνώριση ομιλίας και η επεξεργασία φυσικής γλώσσας.

Η βασική ιδέα πίσω από τα RNNs είναι η εισαγωγή βρόχων μέσα στο δίκτυο, επιτρέποντας στην πληροφορία να διατηρείται. Σε κάθε χρονικό βήμα, ένα RNN δέχεται μια νέα είσοδο μαζί με την προηγούμενη κατάστασή του (state) (που περιέχει πληροφορία από προηγούμενα χρονικά βήματα) για να παράγει μια έξοδο και να ενημερώσει την κατάστασή του.

Ωστόσο, τα RNNs έχουν κάποια σημαντικά ελαττώματα. Αντιμετωπίζουν προβλήματα με μακροπρόθεσμες εξαρτήσεις [2], λόγω προβλημάτων που είναι γνωστά ως vanishing και exploding gradients. Ως αποτέλεσμα, έχουν εισαχθεί πιο προηγμένες αρχιτεκτονικές RNN, όπως τα LSTMs [3] και τα GRUs [4].

Εκπαίδευση Νευρωνικών Δικτύων

Η διαδικασία εκπαίδευσης των νευρωνικών δικτύων περιλαμβάνει τη διάδοση των δεδομένων εισόδου μέσα από το δίκτυο, τον υπολογισμό της απώλειας (ή του σφάλματος) συγκρίνοντας την έξοδο του δικτύου με τις πραγματικές ετικέτες, και την αναδιάδοση του σφάλματος πίσω μέσα από το δίκτυο για να ενημερώσει τα βάρη.

Αρχικά, τα δεδομένα εισόδου διαδίδονται μέσω του νευρωνικού δικτύου σε μια διαδικασία που αποκαλείται εμπρόσθια διάδοση (feedforward pass). Καθώς τα δεδομένα εισόδου περνούν από κάθε στρώμα, οι νευρώνες επεξεργάζονται τα δεδομένα χρησιμοποιώντας τα τρέχοντα βάρη και τις συναρτήσεις ενεργοποίησης.

Η έξοδος του δικτύου στη συνέχεια συγκρίνεται με τις πραγματικές ετικέτες, για να υπολογιστεί η απώλεια με τη βοήθεια μιας συνάρτησης απώλειας (loss function). Η απώλεια αυτή αποτελεί ένα μέτρο της διαφοράς μεταξύ της προβλεπόμενης και της πραγματικής ετικέτας.

Στη συνέχεια, εφαρμόζεται η αλγόριθμος προς τα πίσω διάδοσης (backpropagation), για να διαδοθεί το σφάλμα πίσω στο δίκτυο. Κατά την προς τα πίσω διάδοση, τα βάρη του δικτύου ενημερώνονται, βάσει του βελτιστοποιητή (optimizer), ώστε να μειωθεί το σφάλμα στην επόμενη επανάληψη της εμπρόσθιας διάδοσης.

Συναρτήσεις Απώλειας

Όπως αναφέρθηκε και παραπάνω, η συνάρτηση απώλειας ποσοτικοποιεί πόσο καλά ένα μοντέλο προβλέπει σε σύγκριση με την πραγματικότητα. Παρέχει ένα αριθμητικό μέτρο της απόκλισης μεταξύ των προβλεπόμενων και των πραγματικών τιμών, λειτουργώντας ως πυξίδα που καθοδηγεί τη βελτιστοποίηση των παραμέτρων του μοντέλου. Όσο χαμηλότερη είναι η τιμή της συνάρτησης απώλειας, τόσο καλύτερη είναι η απόδοση του μοντέλου. Τρεις από τις πιο κοινές συναρτήσεις απώλειας είναι οι εξής:

- **L1 loss (MAE)**
- **L2 loss (MSE)**
- **Cross Entropy Loss**

Συναρτήσεις Ενεργοποίησης

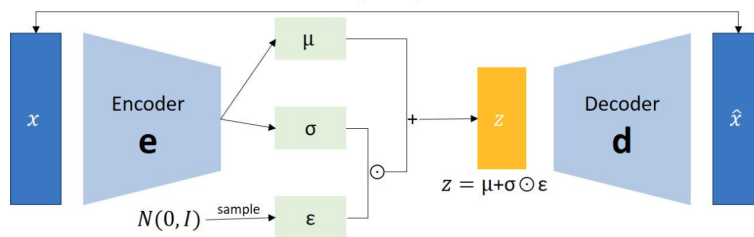
Οι συναρτήσεις ενεργοποίησης εισάγουν μη γραμμικότητα στο δίκτυο, επιτρέποντας του να μοντελοποιεί και να μαθαίνει πολύπλοκες, μη γραμμικές σχέσεις μεταξύ των εισόδων και των εξόδων. Χωρίς αυτές, ανεξάρτητα από το πόσο βαθύ ή ευρύ είναι το δίκτυο, θα λειτουργούσε απλώς ως ένας γραμμικός παλινδρομητής, περιορίζοντας δραστικά την ικανότητά του να προσεγγίζει περίπλοκες συναρτήσεις. Κάποιες δημοφιλείς συναρτήσεις ενεργοποίησης που θα χρησιμοποιηθούν σε αυτή τη διπλωματική είναι οι εξής:

- **Sigmoid**
- **Softmax**
- **ReLU**
- **SiLU**
- **GELU**

Εισαγωγή στα Παραγωγικά Μοντέλα

Οι Variational Autoencoders, που είναι ευρέως γνωστοί ως VAEs, έχουν αναδυθεί ως ένα από τα πιο δημοφιλή και ισχυρά εργαλεία στον κόσμο της γεννητικής μοντελοποίησης. Εισήχθησαν το 2013 [5] και γεφυρώνουν το χάσμα μεταξύ δύο κύριων προσεγγίσεων στη μηχανική μάθηση: της βαθιάς μάθησης και της Μπεϋζιανής συμπερασματολογίας.

Οι autoencoders είναι νευρωνικά δίκτυα που σχεδιάστηκαν, για να ανακατασκευάζουν την είσοδό τους. Το επιτυγχάνουν αυτό συμπιέζοντας την είσοδο σε μια συμπαγή, λανθάνουσα αναπαράσταση μέσω ενός κωδικοποιητή (encoder) και, στη συνέχεια, αναπτύσσοντας αυτή τη λανθάνουσα αναπαράσταση πίσω στον αρχικό χώρο δεδομένων μέσω ενός αποκωδικοποιητή (decoder). Ωστόσο, οι VAEs προσθέτουν μια πιθανοτική περιστροφή σε αυτή τη διαδικασία. Αντί να κωδικοποιούν μια είσοδο ως ένα απλό σημείο στον λανθάνοντα χώρο, την κωδικοποιούν ως μια κατανομή. Αυτή η εγγενής τυχαιότητα επιτρέπει στους VAEs να δημιουργούν νέα, παρόμοια δεδομένα, δειγματοληπώντας από αυτή την κατανομή.

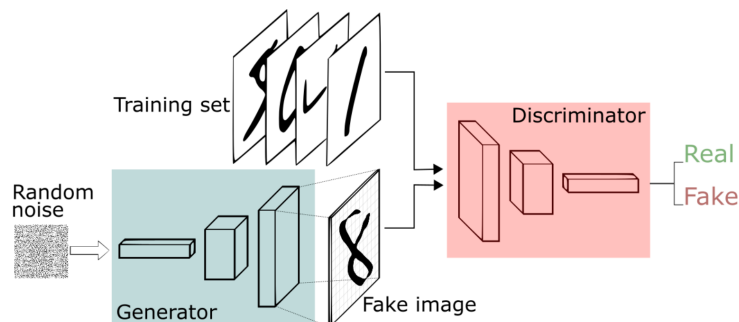


Σχήμα 0.0.1: Η αρχιτεκτονική ενός VAE. Από [εδώ](#)

Τα Παραγωγικά Ανταγωνιστικά Δίκτυα (GANs) παρουσιάστηκαν το 2014 [6] και από τότε έχουν κερδίσει την προσοχή, όχι μόνο για τη μαθηματική τους κομψότητα, αλλά και για την πρακτική τους ικανότητα να δημιουργούν ρεαλιστικά αποτελέσματα, ποικίλλοντας από εικόνες έως και ήχους.

Η κύρια ιδέα πίσω από τα GANs αφορά την εκπαίδευση δύο δικτύων, του γεννήτορα (generator) και του διευκρινιστή (discriminator), μέσω ενός παιχνιδιού μηδενικού αθροίσματος. Ειδικότερα, ο στόχος του γεννήτορα είναι να δημιουργεί ρεαλιστικά δείγματα, των οποίων η κατανομή είναι κοντά στην αντίστοιχη των πραγματικών δεδομένων, ενώ ο στόχος του διευκρινιστή είναι να μπορεί να διακρίνει τις πραγματικές εικόνες από τις συνθετικές (που δημιουργήθηκαν από τον γεννήτορα). Η συνολική αρχιτεκτονική ενός GAN απεικονίζεται στο παρακάτω σχήμα.

Επομένως, η εκπαίδευση μπορεί να διατυπωθεί μαθηματικά ως ένα παιχνίδι minimax, όπου ο γεννήτορας προσπαθεί να μεγιστοποιήσει τις πιθανότητες να ξεγελάσει τον διευκρινιστή, ενώ ο τελευταίος προτίθεται να ελαχιστοποιήσει αυτή την πιθανότητα.



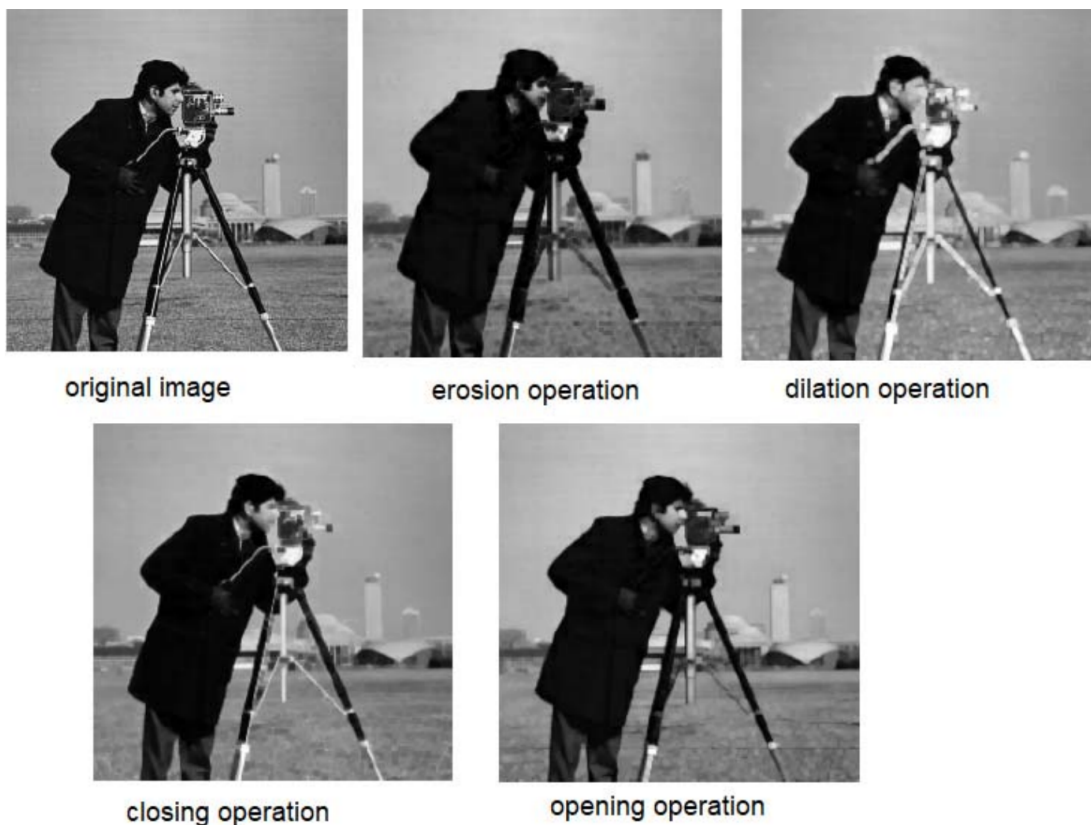
Σχήμα 0.0.2: Η αρχιτεκτονική ενός Παραγωγικού Ανταγωνιστικού Δικτύου. Από [εδώ](#)

Εισαγωγή στα Μορφολογικά Μαθηματικά

Τα Μορφολογικά Μαθηματικά αποτελούν ένα πλαίσιο εντός των τομέων της ανάλυσης εικόνας και της επεξεργασίας σήματος, που επικεντρώνεται στον μορφολογικό χειρισμό των δομών εντός των δεδομένων. Αναπτύχθηκε, αρχικά, για την ανάλυση δυαδικών εικόνων, αλλά από τότε έχει επεκταθεί σε πιο περίπλοκες μορφές δεδομένων, όπως γκριζες εικόνες, τρισδιάστατα δεδομένα, ακόμη και μη χωρικά σήματα.

Η θεωρία των μορφολογικών μαθηματικών εισήχθη από τον Matheron [7] και τον Serra [8] και βασίστηκε στη θεωρία των συνόλων, τη θεωρία του πλέγματος και την τοπολογία. Οι κύριες λειτουργίες, δηλαδή, η διαστολή (dilation), η διάβρωση (erosion), το άνοιγμα (opening) και το κλείσιμο (closing), άντλησαν από τα Minkowski set operations. Τα τελευταία χρόνια, τα μορφολογικά μαθηματικά έχουν εφαρμοστεί σε ποικιλία εργασιών υπολογιστικής όρασης όπως η ανάλυση εικόνας [9, 10], η ταξινόμηση [11, 12], το φιλτράρισμα [13], η κατάτμηση [14, 15], η ανίχνευση ακμών [16] κ.λπ.

Τα μορφολογικά μαθηματικά, λόγω των μη γραμμικών τους ιδιοτήτων, μπορούν να αποτυπώσουν χαρακτηριστικά, τα οποία δεν μπορούν να διατηρηθούν από άλλες γραμμικές προσεγγίσεις. Οι βασικοί μορφολογικοί τελεστές παρουσιάζονται στην παρακάτω εικόνα.



Σχήμα 0.0.3: Βασικοί Μορφολογικοί Τελεστές. Από [17]

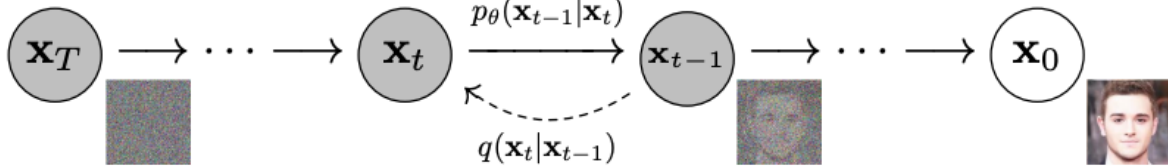
Εισαγωγή στα Μοντέλα Διάχυσης

Τα μοντέλα διάχυσης, που ανήκουν στην ευρεία κατηγορία των γεννητικών μοντέλων, αποκτούν όλο και περισσότερη φήμη για τις εντυπωσιακές τους δυνατότητες στην παραγωγή υψηλής ποιότητας, ρεαλιστικών δειγμάτων. Τα θεωρητικά θεμέλια των μοντέλων διάχυσης είναι βαθιά ριζωμένα στις αρχές της μη ισορροπημένης θερμοδυναμικής και των στοχαστικών διαδικασιών [18].

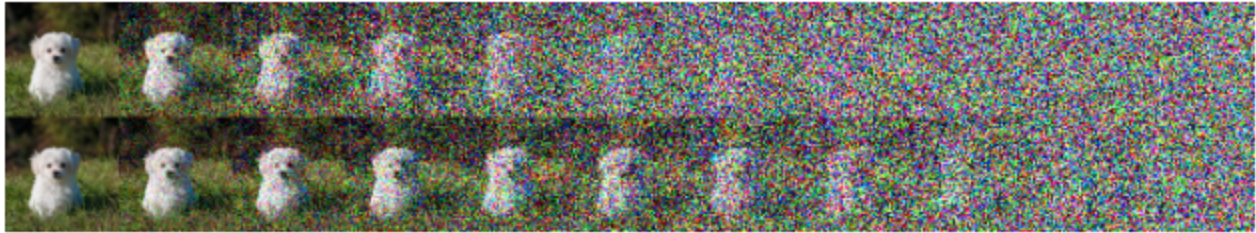
Η εκπαίδευση ενός μοντέλου διάχυσης ισοδυναμεί με τη μάθηση για την αντιστροφή αυτής της διαδικασίας διάχυσης προς τα εμπρός. Από την αρχική κατανομή δεδομένων, προστίθεται θόρυβος σταδιακά μέχρι τα δεδομένα

να μεταμορφώνονται σε μια απλή κατανομή θορύβου. Ο στόχος του μοντέλου είναι να μάθει την αντίστροφη λειτουργία - να απομακρύνει τον θόρυβο και έτσι να ανακτήσει την αρχική κατανομή. Αυτή η αρχή είναι παρόμοια με τους denoising autoencoders, οι οποίοι επίσης επικεντρώνονται στην εξάλειψη του θορύβου από τα δεδομένα. Ωστόσο, το κύριο διακριτικό σημείο προκύπτει από το γεγονός ότι τα μοντέλα διάχυσης εισάγουν σταδιακά θόρυβο σε αρκετά βήματα, ενώ οι denoising autoencoders ενσωματώνουν θόρυβο σε ένα βήμα [19].

Παρακάτω, παρουσιάζεται η διαδικασία διάχυσης τόσο σχηματικά, όσο και ποιοτικά (για δύο διαφορετικά χρονοδιαγράμματα), καθώς και δύο πίνακες που περιέχουν τους αλγορίθμους εκπαίδευσης και παραγωγής εικόνων για τα μοντέλα διάχυσης.



Σχήμα 0.0.4: Η διαδικασία διάχυσης σχηματικά. Από [20]



Σχήμα 0.0.5: Δύο διαφορετικά χρονοδιαγράμματα (γραμμικό και συνημιτονικό) για τη διαδικασία διάχυσης. Από [21]

Algorithm 1: Training. From [20]

- 1: **repeat**
 - 2: $x_0 \sim q(x_0)$
 - 3: $t \sim \text{Uniform}(\{1, \dots, T\})$
 - 4: $\varepsilon \sim \mathcal{N}(0, I)$
 - 5: Take gradient descent step on $\|\varepsilon - \varepsilon_\theta(\sqrt{\alpha_t}x_0 + \sqrt{1 - \alpha_t}\varepsilon, t)\|^2$
 - 6: **until** converged
-

Algorithm 2: Sampling. From [20]

- 1: $x_T \sim \mathcal{N}(0, I)$
 - 2: **for** $t = T$ to 1 **do**
 - 3: $z \sim \mathcal{N}(0, I)$ if $t > 1$, else $z = 0$
 - 4: $x_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(x_t - \sqrt{\frac{1 - \alpha_t}{1 - \alpha_{t-1}}} \varepsilon_\theta(x_t, t) \right) + \sigma_t z$
 - 5: **end for**
 - 6: **return** x_0
-

Εισαγωγή στην Αναγνώριση και Παραγωγή Χειρόγραφου Κειμένου

Η Αναγνώριση Χειρόγραφου Κειμένου (HTR) επικεντρώνεται κυρίως στη μετατροπή του χειρόγραφου κειμένου σε κείμενο που είναι κωδικοποιημένο από τον υπολογιστή. Πρόκειται για ένα ιδιαίτερα δύσκολο πρόβλημα, λόγω της μεταβλητότητας στα επί μέρους στυλ γραφής, της ασυνεπούς απόστασης μεταξύ των χαρακτήρων και των διαφορετικών τύπων γραφής [22]. Υπάρχουν δύο κύριες μέθοδοι στις οποίες βασίζονται τα συστήματα HTR, το online και το offline. Αυτό που διαφοροποιεί αυτές τις δύο μεθόδους είναι τα δεδομένα που χρησιμοποιούνται για την εκπαίδευση των μοντέλων HTR.

Η Δημιουργία Χειρόγραφου Κειμένου (HTG), από την άλλη πλευρά, στοχεύει στη δημιουργία ρεαλιστικού χειρόγραφου κειμένου δεδομένου ενός κειμένου που είναι κωδικοποιημένο από τον υπολογιστή και ενός συγκεκριμένου στυλ γραφής. Αυτό μπορεί να είναι χρήσιμο για τη δημιουργία συνθετικών δεδομένων για την εκπαίδευση

μοντέλων μηχανικής μάθησης, την καλλιτεχνική δημιουργία κειμένου και περισσότερο. Επίσης, ανάλογα με το χρησιμοποιούμενο σύνολο δεδομένων, το HTG διαιρείται, επίσης, σε online HTG και offline HTG.

Online Αναγνώριση και Παραγωγή Χειρόγραφου Κειμένου

Το online HTR περιλαμβάνει δεδομένα που παράγονται από τη θέση της πέννας, ενώ ένα έγγραφο ή μια φράση γράφεται. Αντίστοιχα, το online HTG αφορά τη σύνθεση χειρόγραφου κειμένου με βάση τις χωρικές συντεταγμένες μιας πέννας καθώς συμβαίνει το γράψιμο. Ο Graves [23] πρότεινε την πρώτη προσέγγιση με ενθαρρυντικά αποτελέσματα, τα οποία βασίζονταν σε δίκτυα Long Short-Term Memory (LSTM) [3] και τον μηχανισμό προσοχής. Η επόμενη εργασία [24] παρείχε περαιτέρω βελτιώσεις με την εισαγωγή των Conditional Variational Recurrent Neural Networks (CVRNNs). Στο DeepWriting [25], αυτό προσεγγίζεται διαχωρίζοντας το περιεχόμενο του κειμένου από το στυλ του, ενώ βελτίωσαν ακόμη περισσότερο τα αποτελέσματα αντικαθιστώντας τα CVRNNs με Stochastic Temporal Convolutional Neural Networks (STCNNs) [26].

Offline Αναγνώριση και Παραγωγή Χειρόγραφου Κειμένου

Από την άλλη, οι εφαρμογές offline HTR σχετίζονται με δεδομένα που παράγονται από τη σάρωση αρχικών εγγράφων. Προφανώς, το offline HTR αποτελεί μια πολύ πιο πολύπλοκη εργασία, καθώς τα σύνολα δεδομένων για online HTR χαρακτηρίζονται συχνά από πολύ καλύτερη ποιότητα. Ωστόσο, τα offline σύνολα δεδομένων είναι, προφανώς, ευκολότερο να δημιουργηθούν, ενώ οι εφαρμογές τους είναι επίσης ευρύτερες, συμπεριλαμβανομένης της ψηφιοποίησης ιστορικών εγγράφων, της αυτοματοποίησης της ταξινόμησης ταχυδρομικής αλληλογραφίας και ακόμη και της αναγνώρισης μαθηματικών εξισώσεων [27].

Αντίστοιχα, το offline HTG, του οποίου οι εφαρμογές αποτελούν το κύριο αντικείμενο αυτής της διπλωματικής, σχετίζεται με τη σύνθεση χειρόγραφου κειμένου χρησιμοποιώντας δεδομένα από σαρωμένα έγγραφα. Οι πρόσφατες τεράστιες βελτιώσεις στον τομέα των γεννητικών μοντέλων για τη σύνθεση εικόνων έχουν επηρεάσει ευρέως τον τομέα του offline HTG, καθώς τα παρεχόμενα μοντέλα χαρακτηρίζονται από προηγμένες δυνατότητες, όπως φαίνεται και στις παραγόμενες εικόνες τους.

Ειδικότερα, οι περισσότερες πρόσφατες εργασίες προσεγγίζουν αυτό το πρόβλημα με τα Παραγωγικά Ανταγωνιστικά Δίκτυα (GANs) [6]. Το ScrabbleGAN [28] είναι μία από τις πρώτες προσεγγίσεις που παράγει πειστικά αποτελέσματα. Το GANWriting εισήχθη ως μια επέκταση του ScrabbleGAN, επιτυγχάνοντας ακόμη υψηλότερης ποιότητας σύνθεση γραφής, ενώ, επίσης, προσαρμόστηκε το μοντέλο στα στυλ γραφής κάθε λέξης.

Ο συνδυασμός ενός GAN και ενός autoencoder, που προτάθηκε στο [29], οδήγησε σε περαιτέρω πρόοδο στον τομέα του HTG, αλλά, αντίθετα με τις άλλες μεθόδους, αυτή η εργασία επικεντρώθηκε στη δημιουργία ολόκληρων προτάσεων χειρόγραφου κειμένου. Εμπνευσμένοι από αυτήν την εργασία, οι Mattick, Mayr, Seuret, Maier και Christlein [30] παρουσίασαν το SmartPatch, που είναι, επίσης, μια προσέγγιση βασισμένη σε GAN (για τη δημιουργία χειρόγραφων λέξεων) και ξεπέρασε τα προηγούμενα μοντέλα παράγοντας κορυφαία αποτελέσματα.

Αντίθετα με τις προηγούμενες περιπτώσεις, το WordStylist [31] προτάθηκε, για να αντιμετωπίσει το πρόβλημα του HTG με μοντέλα λανθάνουσας διάχυσης (LDMs) [32]. Ειδικότερα, ένα LDM χρησιμοποιείται για την παραγωγή εικόνων χειρόγραφων λέξεων, λαμβάνοντας πληροφορία από το περιεχόμενο και τα στυλ γραφής. Όσον αφορά την ποιότητα της εικόνας, το WordStylist ανταγωνίζεται τα αποτελέσματα του SmartPatch, ενώ στην υιοθέτηση συγκεκριμένων στυλ γραφής αποδίδει εμφανώς καλύτερα από όλες τις άλλες προσεγγίσεις.

Σχετική Βιβλιογραφία

Ψυχρή Διάχυση

Η Ψυχρή Διάχυση [33] προτάθηκε για να αμφισβητήσει την αναγκαιότητα του γκαουσιανού θορύβου, ή οποιασδήποτε τυχαιότητας, για την πρακτική εφαρμογή των μοντέλων διάχυσης. Πρόκειται για γενικευμένα μοντέλα διάχυσης που εκτείνονται πέρα από τα θεωρητικά όρια που αρχικά τέθηκαν για αυτά τα μοντέλα. Αντί να επικεντρώνονται αποκλειστικά σε μοντέλα που στηρίζονται στον γκαουσιανό θόρυβο, εξετάζονται μοντέλα που κατασκευάζονται γύρω από τυχαίους μετασχηματισμούς εικόνας. Σκοπός είναι να εκπαιδευτεί ένα δίκτυο, για να αντιστρέψει αυτές τις παραμορφώσεις, ελαχιστοποιώντας μια συνάρτηση απώλειας l_p .

Ειδικότερα, συχνά χρησιμοποιούνται επαναληπτικά νευρωνικά μοντέλα για την αντιμετώπιση αντίστροφων προβλημάτων [34, 35], ενώ πρόσφατα, τα μοντέλα διάχυσης έχουν προσαρμοστεί επίσης σε αυτά, ειδικά για ζητήματα όπως το deblurring [36], η αποθορυβοποίηση [37] και η υπερ-ανάλυση [38].

Η συγκεκριμένη εργασία επιχειρεί να δώσει μία βελτιωμένη λύση σχετικά με τον αλγόριθμο παραγωγής εικόνων για τα μοντέλα με ντετερμινιστική διάχυση. Ο αλγόριθμος αυτός, όπως και η αντίστοιχη απλούστερη εκδοχή του αλγορίθμου συνοψίζονται στα παρακάτω πινακάκια.

Algorithm 3: Naive Sampling

Require: A degraded sample x_t

```

for  $s = t, t - 1, \dots, 1$  do
   $\hat{x}_0 \leftarrow R(x_s, s)$ 
   $x_{s-1} \leftarrow D(\hat{x}_0, s - 1)$ 
end for
return  $x_0$ 

```

Algorithm 4: Improved Sampling for Cold Diffusion

Require: A degraded sample x_t

```

for  $s = t, t - 1, \dots, 1$  do
   $\hat{x}_0 \leftarrow R(x_s, s)$ 
   $x_{s-1} \leftarrow x_s - D(\hat{x}_0, s) + D(\hat{x}_0, s - 1)$ 
end for
return  $x_0$ 

```

Ο προτεινόμενος αλγόριθμος επιδεικνύει σημαντική ανθεκτικότητα σε σφάλματα του μοντέλου αποθορυβοποίησης R για μικρές τιμές των x και s . Για να κατανοήσουμε το γιατί, τα ακόλουθα χαρακτηριστικά του προτεινόμενου αλγορίθμου επισημαίνονται στο [33]:

Υποθέτοντας μια γραμμική συνάρτηση καταστροφής που παίρνει τη μορφή

$$D(x, s) \approx x + s \cdot e \quad (0.0.1)$$

για ένα συγκεκριμένο διάνυσμα e . Παρόλο που φαίνεται περιοριστικό, οποιαδήποτε ομαλή συνάρτηση $D(x, s)$, μέσω της σειράς Taylor γύρω από $x = x_0, s = 0$, παίρνει την παρακάτω μορφή:

$$D(x, s) \approx x + s \cdot e + HOT \quad (0.0.2)$$

όπου HOT αντιπροσωπεύει όρους υψηλότερης τάξης. Ο σταθερός/μηδενικής τάξης όρος σε αυτή τη σειρά Taylor είναι μηδέν, διότι $D(x_0, 0) = x_0$.

Δεδομένης μιας συνάρτησης της παραπάνω μορφής και οποιουδήποτε μοντέλου R , η ενημέρωση που χρησιμοποιείται στον προτεινόμενο αλγόριθμο μπορεί να γραφτεί ως εξής:

$$\begin{aligned}
 x_{s-1} &= x_s - D(R(x_s, s), s) + D(R(x_s, s), s - 1) \\
 &= D(x_0, s) - D(R(x_s, s), s) + D(R(x_s, s), s - 1) \\
 &= x_0 + s \cdot e - R(x_s, s) - s \cdot e + R(x_s, s) + (s - 1) \cdot e \\
 &= x_0 + (s - 1) \cdot e \\
 &= D(x_0, s - 1)
 \end{aligned}$$

Ως αποτέλεσμα, συμπεραίνουμε ότι ο αλγόριθμος παράγει το $x_s = D(x_0, s)$ για όλα τα $s < t$, ανεξαρτήτως της επιλογής του R . Δηλαδή, η επανάληψη συμπεριφέρεται ακριβώς όπως θα έκανε αν το R ήταν μια τέλεια αντιστροφή της υποβάθμισης D . Από την άλλη, ο αφελής αλγόριθμος δεν εμφανίζει αυτή την ιδιότητα. Η σημασία της χρήσης του προτεινόμενου αλγορίθμου φαίνεται και στην παρακάτω εικόνα.

Λανθάνουσα Διάχυση

Τα τελευταία χρόνια έχει παρατηρηθεί εντυπωσιακή ανάπτυξη στον τόμενο της όρασης υπολογιστών. Ενώ η σύνθεση υψηλής ανάλυσης αποτελείται κυρίως από μοντέλα πιθανότητας με autoregressive transformers [39, 40] που μπορεί να έχουν δισεκατομμύρια παραμέτρους, η χρησιμότητα των Παραγωγικών Ανταγωνιστικών Δικτύων (GANs) [41, 42, 43] έχει περιορισμούς λόγω της δυσκολίας τους στο να μοντελοποιήσουν σύνθετες κατανομές δεδομένων.



Σχήμα 0.0.6: Σύγκριση των δύο αλγορίθμων. Από [33]

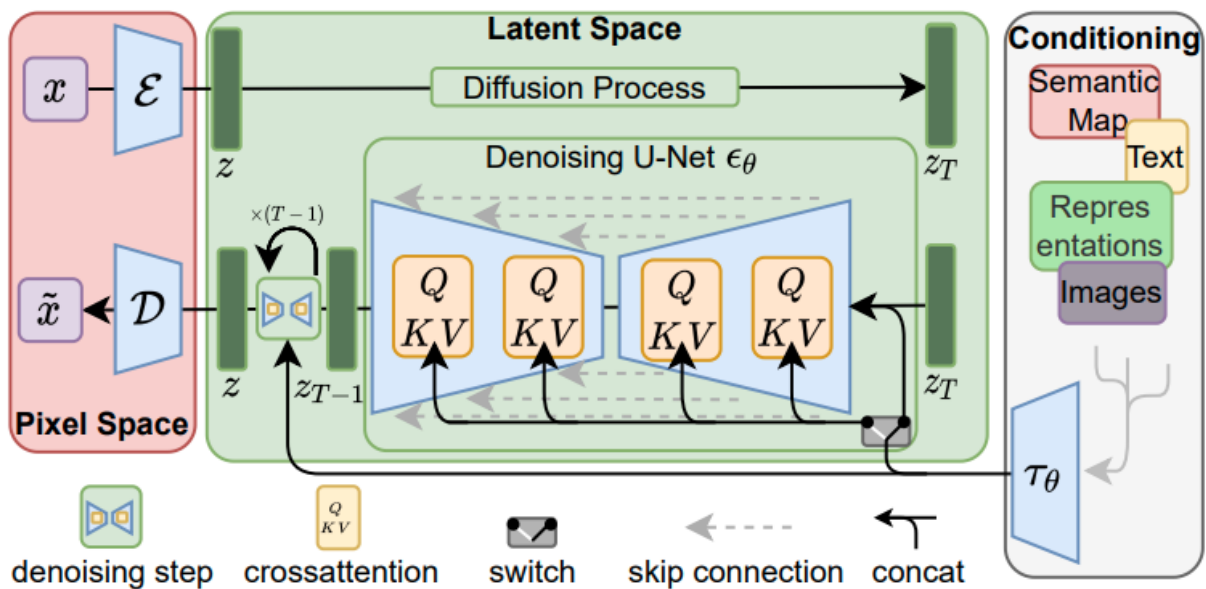
Τα Μοντέλα Διάχυσης έχουν πρόσφατα επιδείξει ανώτερα αποτελέσματα στη σύνθεση εικόνων [20, 44] και άλλες εργασίες [45, 46, 47]. Ωστόσο, ένα σημαντικό μειονέκτημα των μοντέλων διάχυσης είναι η πολυπλοκότητά τους, καθώς, τόσο η εκπαίδευσή, όσο και η παραγωγή εικόνων απαιτεί τεράστιους υπολογιστικούς πόρους.

Για να αντιμετωπίσουν αυτό το πρόβλημα, οι Rombach, Blattmann, Lorenz, Esser και Ommer [32] προτείνουν τα μοντέλα λανθάνουσας διάχυσης (LDMs). Ειδικότερα, τα θεμέλια αυτής της προσέγγισης είναι τα δύο κύρια στάδια της μάθησης:

- **Αντιληπτική συμπίεση**, που σχετίζεται με τη διαχείριση λεπτομερειών υψηλών συχνοτήτων.
- **Σημασιολογική συμπίεση**, που επικεντρώνεται στις έννοιες των δεδομένων.

Σύμφωνα με τα παραπάνω, η κύρια ιδέα είναι η μετάβαση σε έναν υπολογιστικά αποτελεσματικό χώρο για την εκπαίδευση Μοντέλων Διάχυσης, με σκοπό τη σύνθεση εικόνων υψηλής ανάλυσης. Ακολουθώντας προηγούμενες σχετικές εργασίες [48, 49, 50, 39, 40], οι Rombach, Blattmann, Lorenz, Esser και Ommer [32] παρουσιάζουν μια διαδικασία εκπαίδευσης που αποτελείται από δύο στάδια: Πρώτον, εκπαιδεύεται ένας autoencoder, για να παρέχει έναν χώρο αναπαράστασης μικρότερης διάστασης, ο οποίος τελικά χρησιμοποιείται για την εκπαίδευση Μοντέλων Διάχυσης.

Μία ακόμα σημαντική συνεισφορά της συγκεκριμένης εργασίας είναι ο τρόπος που προτείνεται, προκειμένου να ενσωματώνεται στο μοντέλο η υπό συνθήκη πληροφορία, καθώς η χρήση μηχανισμού διασταυρούμενης προσοχής (cross attention mechanism) επιτρέπει την εξαγωγή αποτελεσματικών αναπαραστάσεων για πολλά διαφορετικά είδη συνθηκών (για παράδειγμα κείμενο ή εικόνες). Η συνολική αρχιτεκτονική ενός μοντέλου λανθάνουσας διάχυσης παρουσιάζεται στην παρακάτω εικόνα.



Σχήμα 0.0.7: Αρχιτεκτονική Μοντέλων Λανθάνουσας Διάχυσης. Από [32]

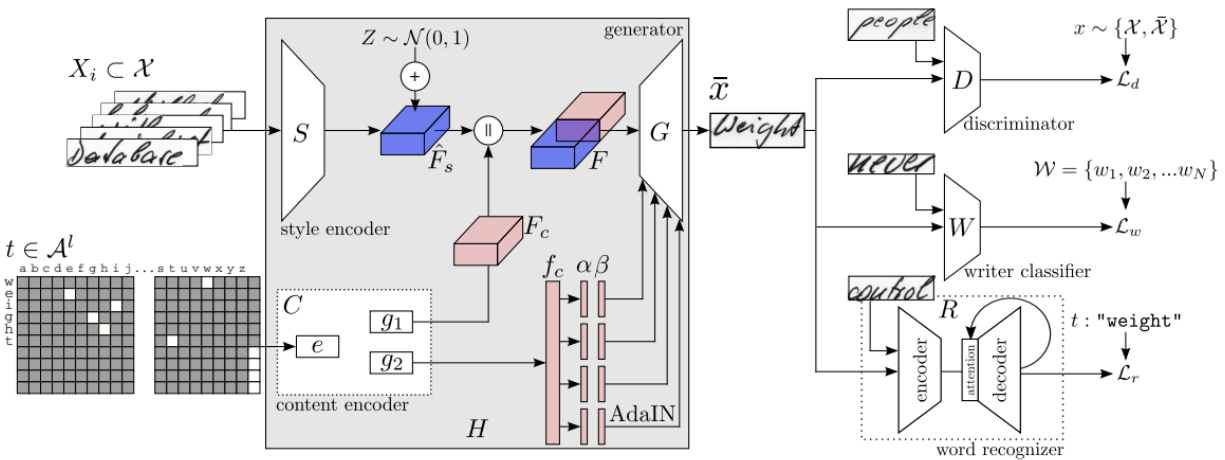
Παραγωγή χειρόγραφου κειμένου

GANWriting

Η συγκεκριμένη εργασία [51] προτείνει μια μέθοδο για τη δημιουργία ρεαλιστικών χειρόγραφων λέξεων, με βάση το στυλ γραφής και το περιεχόμενο. Τα αποτελέσματα του μοντέλου χαρακτηρίζονται από μεγάλη ποικιλία, καθώς το μοντέλο δεν περιορίζεται στις λέξεις που ήδη υπάρχουν στο σύνολο δεδομένων και μπορεί να αντιγράψει γρήγορα το στυλ ενός δεδομένου συγγραφέα.

Η αρχιτεκτονική του συνολικού δικτύου είναι η ακόλουθη:

- Παραγωγικό Δίκτυο
 - **Γεννήτορας:** Ο γεννήτορας λαμβάνει ως είσοδο τις συγκατανεταμένες αναπαραστάσεις του περιεχομένου και του στυλ και αποτελείται από δύο residual blocks [52] με κανονικοποίηση AdaIN [53]. Η τελική εικόνα \bar{x} παράγεται μετά από τέσσερα ενότητες συνέλιξης και μια ενεργοποίηση tanh.
 - **Κωδικοποιητής Στυλ:** $\hat{F}_s = S(X_i) + Z$, όπου S είναι ένα VGG-19-BN [54] και $Z \sim \mathcal{N}(0, 1)$.
 - **Κωδικοποιητής Περιεχομένου:** Η τελική αναπαράσταση είναι η γραμμική μετασχηματισμός των διανυσμάτων ονε-ηοτ των συμβολοσειρών με 2 MLPs g_1, g_2 (κάθε μία με 3 επίπεδα με ενεργοποιήσεις ReLU και batch normalization [55]) για την κωδικοποίηση χαρακτήρα και συμβολοσειράς, αντίστοιχα.
- Στόχοι Μάθησης
 - **Διευκρινιστική Απώλεια:** Η αρχιτεκτονική του Διευκρινιστή περιλαμβάνει ένα επίπεδο συνέλιξης και έξι residual blocks [52] με ενεργοποιήσεις LeakyReLU και average poolings.
 - **Απώλεια Στυλ:** Ο ταξινομητής συγγραφέα ακολουθεί την ίδια αρχιτεκτονική με τον Διευκρινιστή, ενώ, σε αυτή την περίπτωση, χρησιμοποιείται ως συνάρτηση απώλειας η διασταυρούμενη εντροπίας.
 - **Απώλεια Περιεχομένου:** Ο αναγνώστης λέξεων αποτελείται από ένα VGG-19-BN [54] και ένα Bidirectional Gated Recurrent Unit (B-GRU) [56]. Η απώλεια που χρησιμοποιείται είναι το Kullback-Leibler divergence.



Σχήμα 0.0.8: Η αρχιτεκτονική του μοντέλου GANWriting. Από [51]

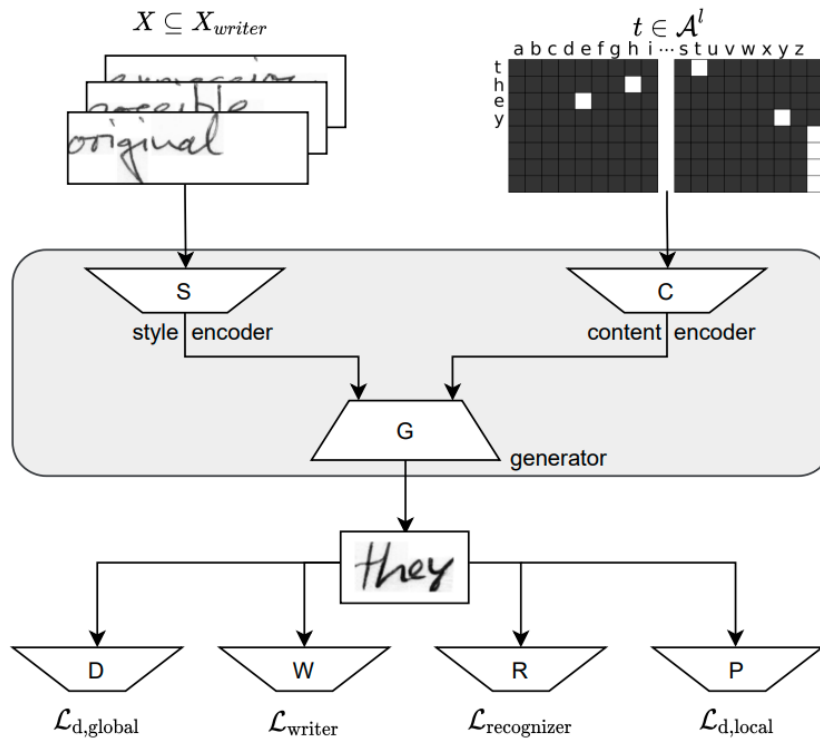
SmartPatch

Περαιτέρω βελτιώσεις στη παραγωγή χειρόγραφου κειμένου παρέχονται στο [30] με την εισαγωγή του SmartPatch. Η αρχιτεκτονική του προτεινόμενου μοντέλου απεικονίζεται παρακάτω. Όπως φαίνεται, η αρχιτεκτονική είναι έντονα εμπνευσμένη από το GANWriting [51], ενώ οι βελτιώσεις σχετίζονται με μια επιπλέον συνιστώσα

απώλειας, τον τοπικό διευκρινιστή (local discriminator) $\mathcal{L}_{d,local}$. Το κίνητρο πίσω από αυτή την προσθήκη, είναι η εμφάνιση κάποιων artifacts στο GANWriting, όπως φαίνεται στην ακόλουθη εικόνα.



Σχήμα 0.0.9: GANWriting Artifacts. Από [30]



Σχήμα 0.0.10: Η αρχιτεκτονική του SmartPatch. Από [30]

Αξίζει να αναφερθεί πως στη συγκεκριμένη δουλειά δοκιμάστηκαν τρία διαφορετικά είδη τοπικών διευκρινιστών:

- **NaivePatch:** Πρόκειται για τη βασική προσέγγιση, η οποία ακολουθεί την αρχιτεκτονική Pix2Pix [57] και απλώς διαχωρίζει την εικόνα σε επικαλυπτόμενα τετράγωνα (με επικάλυψη $\frac{square_dim}{2}$), εμπνευσμένη από το ScrabbleGAN [28].
- **CenteredPatch:** Αυτή η προσέγγιση εκμεταλλεύεται το σύστημα αναγνώρισης κειμένου, προκειμένου να προσδιορίσει τα patches των δημιουργημένων δειγμάτων. Πιο συγκεκριμένα, η υπερ-δειγματοληψία του διανύσματος προσοχής (από τον χάρτη προσοχής που δημιουργείται στον λανθάνοντα χώρο του γεννήτορα [58]) στο πλήρες πλάτος της εικόνας οδηγεί σε ένα κεντρικό παράθυρο για κάθε χαρακτήρα. Τόσο η αρχιτεκτονική όσο και η συνάρτηση απώλειας αυτού του τοπικού διευκρινιστή παραμένουν ίδιες με το NaivePatch.
- **SmartPatch:** Σε αυτή την περίπτωση, το CenteredPatch βελτιώνεται περαιτέρω μεταβιβάζοντας τις κλάσεις χαρακτήρων c_{real}, c_{fake} στο μοντέλο (μετά την προβολή των κλάσεων one-hot σε έναν λανθάνοντα χώρο C_{enc}).

WordStylist

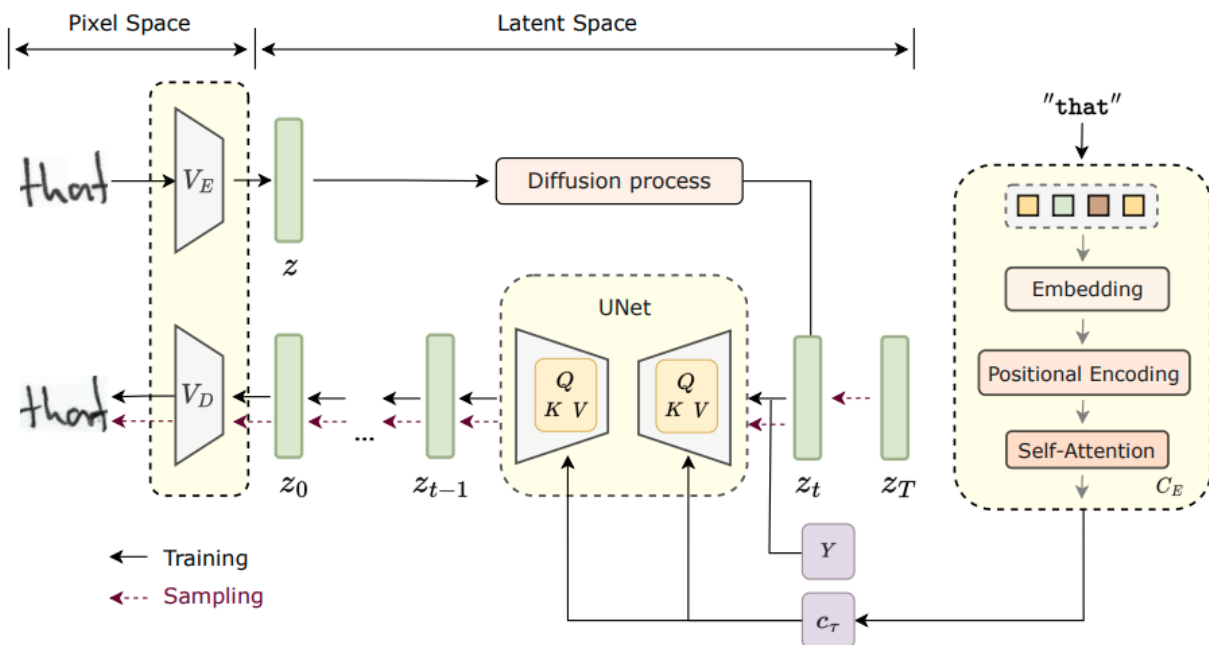
Εν αντιθέσει με τις προηγούμενες εργασίες [51, 30], το WordStylist [31] αντιμετωπίζει την παραγωγή εικόνων χειρόγραφου κειμένου με τη χρήση Μοντέλων Διάχυσης [18, 20, 21, 59] αντί για GANs [6]. Πιο συγκεκριμένα, χρησιμοποιούνται μοντέλα λανθάνουσας διάχυσης [32] για τη μείωση του υπολογιστικού κόστους που απαιτείται για την εκπαίδευση και τη παραγωγή εικόνων, ενώ το μοντέλο προσαρμόζεται τόσο στο περιεχόμενο του κειμένου όσο και στο στυλ γραφής των εικόνων λέξεων, παρόμοια με προηγούμενες αντίστοιχες προσεγγίσεις [51, 30].

Η αρχιτεκτονική του προτεινόμενου μοντέλου είναι σε μεγάλο βαθμό εμπνευσμένη από το αντίστοιχο μοντέλο της [32]. Ειδικότερα, για τη διαδικασία προόδου της εκπαίδευσης και της παραγωγής εικόνων, η μετάβαση στον λανθάνοντα χώρο πραγματοποιείται χρησιμοποιώντας έναν προεκπαιδευμένο autoencoder από το Hugging Face, ενώ ως ο προβλέπτης θορύβου για το μοντέλο λανθάνουσας διάχυσης χρησιμοποιείται ένα UNet [60].

Σχετικά με τις συνθήκες, το κείμενο περνάει στο δίκτυο μέσω ενός μηχανισμού διασταυρούμενης προσοχής [61] όπως προτείνεται στο [32], ενώ η ενσωμάτωση του στυλ γραφής απλά προστίθεται στο χρονικό βήμα. Τα χρονικά βήματα κωδικοποιούνται από ένα sinusoidal positional embedding όπως προτείνεται στο [61], τα στυλ γραφής κωδικοποιούνται σε ένα embedding layer, ενώ η αναπαράσταση του κειμένου αποτελείται από τα εξής στάδια:

- Tokenization
- Embedding layer
- Positional Encoding (όπως προτείνεται στο [61])
- Self-Attention [61]

Η συνολική αρχιτεκτονική απεικονίζεται στο παρακάτω σχήμα.



Σχήμα 0.0.11: Η αρχιτεκτονική του Wordstylist. Από [31]

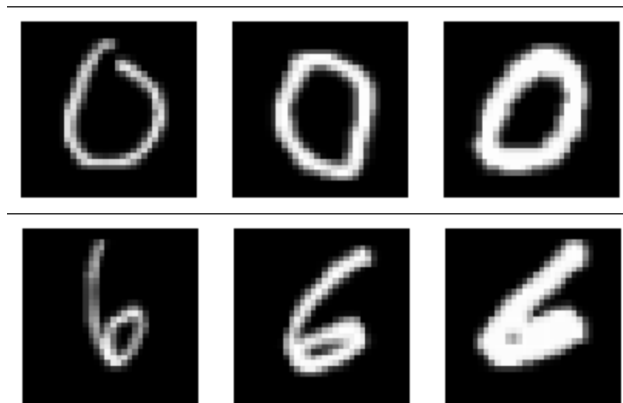
Προτεινόμενη Μέθοδος

Μορφολογική Διάχυση

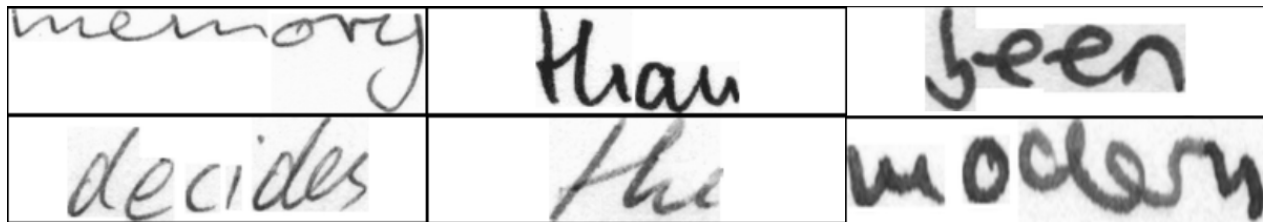
Κίνητρο

Η ψυχρή διάχυση [33] ήταν η πρώτη προσέγγιση που πρότεινε γενικευμένες διαχύσεις με ντετερμινιστικές συναρτήσεις καταστροφής εικόνων. Όπως συζητήθηκε και προηγουμένως, η συγκεκριμένη εργασία εισήγαγε έναν βελτιωμένο αλγόριθμο παραγωγής εικόνων για τέτοιες συναρτήσεις. Παρά την δοκιμή του προτεινόμενου αλγόριθμου για μια ποικιλία διαδικασιών διάχυσης, εκτός από το snowification (για το οποίο δεν αναφέρθηκαν ποσοτικά αποτελέσματα), όλες οι άλλες διαχύσεις βασίζονται στο γκαουσιανό blur (blur, inpainting, downsampling), επομένως δεν ανταποκρίνονται πλήρως στον όρο 'Γενικευμένες Διαχύσεις'. Σε αυτήν τη διπλωματική, θα πειραματιστούμε με τη μορφολογική διαστολή (morphological dilation) ως τη διαδικασία διάχυσης, η οποία διαφέρει σημαντικά από τις διαχύσεις του [33], καθώς όχι μόνο δεν βασίζεται στο γκαουσιανό blur, αλλά καθιστά και μία από τις πρώτες προσπάθειες με μη γραμμική συνάρτηση καταστροφής εικόνων.

Η λογική πίσω από την επιλογή της διαστολής ως συνάρτηση καταστροφής βασίζεται στη μορφολογία των εικόνων των συνόλων δεδομένων MNIST [62] και IAM [63]. Τα ψηφία και το χειρόγραφο κείμενο, αντίστοιχα, ποικίλουν ανάλογα με τον συγγραφέα ή ακόμη και την πένα, προκαλώντας ελαφρές αποκλίσεις στο πώς γράφεται μια λέξη ή ένας αριθμός (λεπτότερες ή παχύτερες γραμμές για παράδειγμα). Αυτές οι παραλλαγές άπτονται μερικών μορφολογικών λειτουργιών όπως η διάβρωση (erosion) ή η διαστολή (dilation), τις οποίες θα χρησιμοποιήσουμε για την δημιουργία μιας διαδικασίας διάχυσης που είναι πιο κατάλληλη για αυτά τα σύνολα δεδομένων. Επιπλέον, θα προσθέσουμε και την απόδοση της μορφολογικής διάχυσης για το CIFAR-10 [64], όχι μόνο για σκοπούς σύγκρισης με το [33], αλλά και για πιθανές μελλοντικές εργασίες για τέτοια σύνολα δεδομένων.



Σχήμα 0.0.12: Η μορφολογία του MNIST.



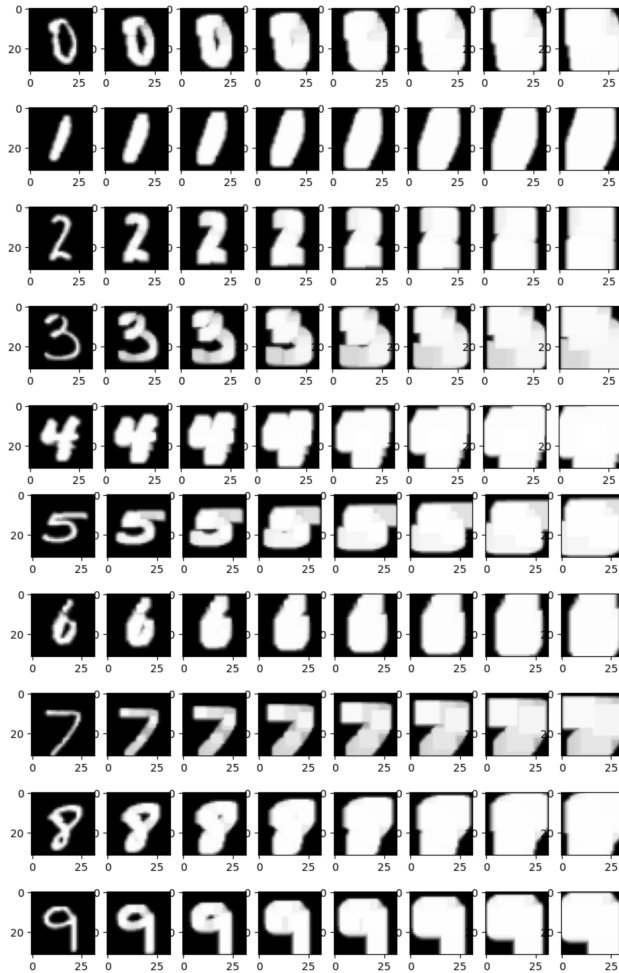
Σχήμα 0.0.13: Η μορφολογία του IAM.

Ανακατασκευή εικόνων

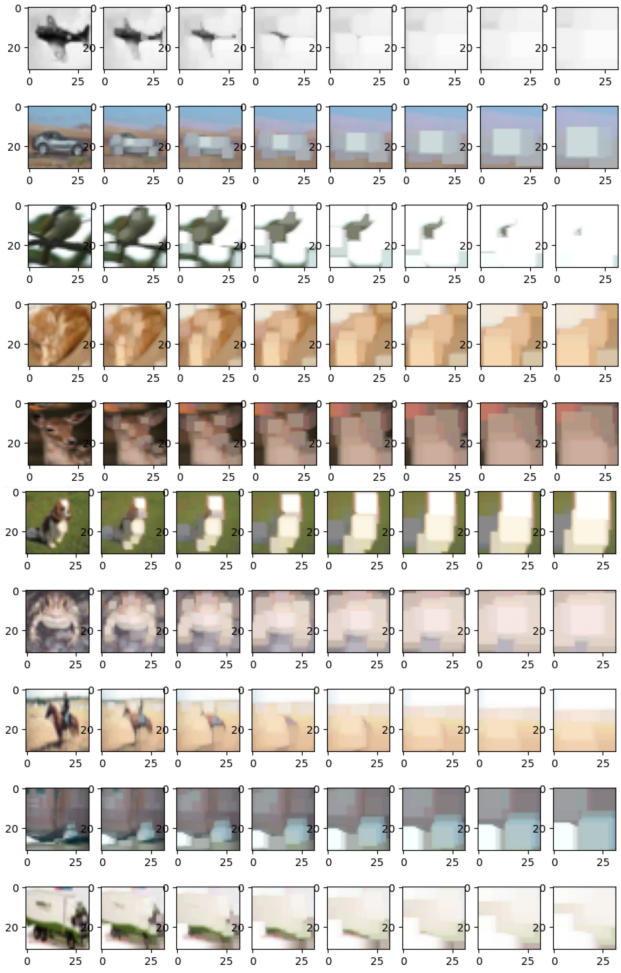
Για την ανακατασκευή εικόνων, θα χρησιμοποιήσουμε τα σύνολα δεδομένων MNIST και CIFAR-10, προκειμένου να είμαστε σε θέση να συγκρίνουμε τα αποτελέσματά μας με αυτά του [33]. Η προτεινόμενη διαδικασία

διάχυσης για το έργο ανακατασκευής εικόνας αποτελείται από 8 βήματα. Κάθε βήμα πραγματοποιείται με ένα τετραγωνικό δομικό στοιχείο (square structuring element) 3×3 .

Οι παρακάτω εικόνες απεικονίζουν τις διαδικασίες διάχυσης του MNIST και CIFAR-10 αντίστοιχα, για ένα τυχαίο δείγμα κάθε κλάσης των συνόλων δεδομένων:



Σχήμα 0.0.14: Παράδειγμα διάχυσης για κάθε ψηφίο του MNIST.

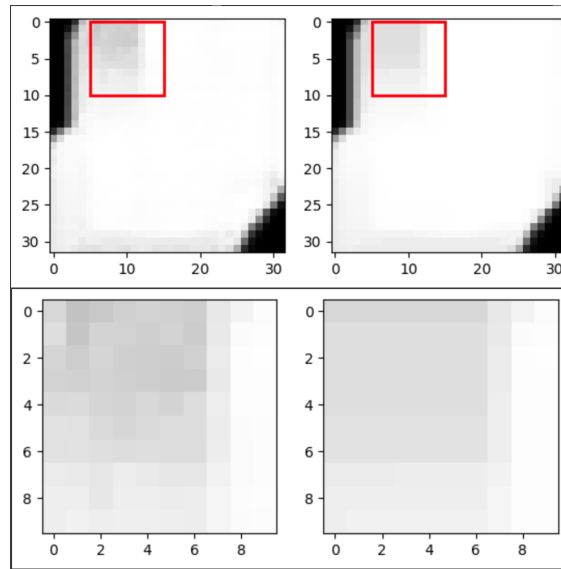


Σχήμα 0.0.15: Παράδειγμα διάχυσης για κάθε κλάση του CIFAR-10.

Υπό συνθήκη παραγωγή εικόνων

Αντίθετα, για την περίπτωση της υπό συνθήκη παραγωγής εικόνων, αυξάνουμε τα βήματα διάχυσης από 8 σε 11, καθώς, σε αυτή την περίπτωση, θέλουμε το μοντέλο να καθοδηγείται κυρίως από την κλάση και όχι από την αρχική (κατεστραμμένη) εικόνα. Ωστόσο, η πρώτη εικόνα της διαδικασίας παραγωγής εικόνων παραμένει πρόβλημα, καθώς η μορφολογική διάχυση διατηρεί τη γεωμετρία κάθε ψηφίου (σε ορισμένες περιπτώσεις ακόμα και στο τελευταίο βήμα), σε αντίθεση με την τυπική γκαουσιανή διάχυση. Αυτό δεν επηρεάζει μόνο την ποιότητα των παραγόμενων εικόνων, αλλά και την ποικιλία τους, καθώς οι περισσότερες εικόνες, κατά την διάχυση, καταλήγουν σε απλές άσπρες εικόνες (όλα τα pixels λαμβάνουν την μέγιστη τιμή της αρχικής εικόνας). Προκειμένου να αντιμετωπίσουμε αυτό το πρόβλημα, αρχικά, θα πειραματιστούμε ορίζοντας την διαστολή απλού θορύβου ως την αρχική εικόνα (σαν benchmark). Στη συνέχεια, θα εκμεταλλευτούμε τα πλεονέκτημα διαφορετικών γεννητικών μοντέλων, για να βελτιστοποιήσουμε την προτεινόμενη διαδικασία. Πιο συγκεκριμένα, βασισμένοι και σε άλλες προσεγγίσεις που χρησιμοποιούν δύο διαφορετικά μοντέλα στο ίδιο συνολικό δίκτυο [32, 48, 49, 40, 65, 66, 67] θα χρησιμοποιήσουμε ένα υπό συνθήκη παραγωγικό ανταγωνιστικό δίκτυο (cGAN), για να παράγουμε τις αρχικές εικόνες. Έπειτα, παρατηρώντας ότι το συγκεκριμένο cGAN μαθαίνει αποτελεσματικά τη γεωμετρία

μίας εικόνας μετά από διαστολή, προσθέτοντας, ωστόσο, λίγο γκαουσιανό θόρυβο, θα επιχειρήσουμε να βελτιώσουμε περαιτέρω τα αποτελέσματά μας πραγματοποιώντας ένα μορφολογικό κλείσιμο (morphological closing) στις εικόνες που παράγει το cGAN. Η επίδραση αυτού του closing συνοψίζεται στην παρακάτω εικόνα.

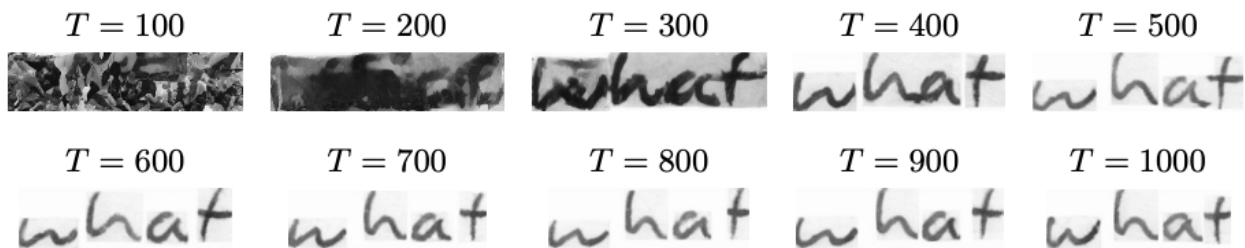


Σχήμα 0.0.16: Η επίδραση του closing στις παραγόμενες εικόνες του cGAN.

Επεκτάσεις στο μοντέλο WordStylist

Το WordStylist [31] είναι η μόνη προσέγγιση που βασίζεται στα μοντέλα διάχυσης για την παραγωγή χειρόγραφου κειμένου. Το συγκεκριμένο μοντέλο παράγει αποτελέσματα κορυφαίας ποιότητας παρόμοια με αυτά του [30], ενώ υπερτερεί αρκετά όλων των άλλων μοντέλων στην προσαρμογή στο στυλ γραφής. Γενικά, μπορεί να θεωρηθεί ότι είναι το μοντέλο με την καλύτερη απόδοση, καθώς τα ποσοτικά αποτελέσματά του σε ό,τι αφορά την ποιότητα των παραγόμενων εικόνων είναι παρομοίου επιπέδου με το SmartPatch [30] (τα FIDs είναι 22.74 και 22.55 αντίστοιχα), ενώ η απόδοσή του στην προσαρμογή στο στυλ γραφής είναι σημαντικά υψηλότερη.

Ωστόσο, όπως αναφέρεται στο [31], ο κύριος περιορισμός αυτής της προσέγγισης είναι η υπολογιστική πολυπλοκότητα της διαδικασίας παραγωγής εικόνων. Παρά τη σημαντική μείωση των απαιτήσεων σε υπολογιστικούς πόρους με τη χρήση των μοντέλων λανθάνουσας διάχυσης (latent diffusion models) [32] και τη παραγωγή εικόνων με 600 βήματα αντί για 1000 (όπως στην εκπαίδευση), η συγκεκριμένη διαδικασία απαιτεί περίπου 12 δευτερόλεπτα για την παραγωγή μιας μόνο εικόνας. Όπως φαίνεται στην παρακάτω εικόνα, η περαιτέρω μείωση των βημάτων οδηγεί σε σημαντική απώλεια ποιότητας, ιδίως όταν τα βήματα είναι 300 ή λιγότερα.

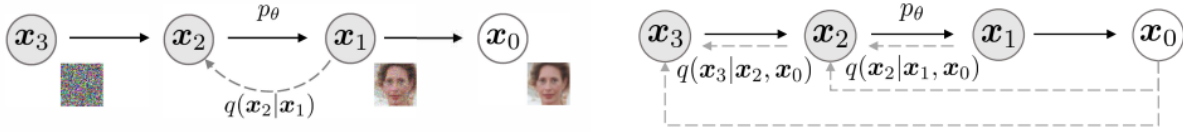


Σχήμα 0.0.17: Παραγωγή εικόνων με διαφορετικό αριθμό βημάτων από 100 έως 1000. Από [31]

Προκειμένου να αντιμετωπίσουμε αυτό το πρόβλημα, αρχικά, θα εφαρμόσουμε πιο αποτελεσματικούς αλγόριθμους για την παραγωγή εικόνων. Πιο συγκεκριμένα, οι αλγόριθμοι που θα εφαρμόσουμε είναι οι ακόλουθοι:

- Denoising Diffusion Implicit Models (DDIM) Sampling [68]
- Pseudo-Numerical Methods for Diffusion Models (PNDM) sampling [69]

Όσον αφορά τα DDIMs, οι Song, Meng και Ermon [68] αμφισβήτησαν την εξάρτηση της απώλειας ενός Μοντέλου Διάχυσης από τα περιθωριακά (marginals) $q(x_T|x_0)$ μόνο, όπως φαίνεται στην εξίσωση 2.3.13. Ειδικότερα, ο στόχος τροποποιήθηκε ώστε να εξαρτάται και από τα ενδιάμεσα στάδια $q(x_{1:T}|x_0)$ (όπως φαίνεται και στην παρακάτω εικόνα). Ως αποτέλεσμα, εξετάζονται μη-Μαρκοβιανά inference processes, τα οποία διαφοροποιούν στον στόχο των DDPMs [20]. Ο συγκεκριμένος αλγόριθμος συνοψίζεται στο ακόλουθο πινακάκι.



Σχήμα 0.0.18: Denoising Diffusion Implicit Models. Από [68]

Algorithm 5: DDIMs. From [69]

```

1:  $x_T \sim N(0, I)$ 
2: for  $t = T - 1, \dots, 1, 0$  do
3:    $x_t = \phi(x_{t+1}, \varepsilon_\theta(x_{t+1}, t + 1), t + 1, t)$ 
4: end for
5:
6: return  $x_0$ 

```

Οι Liu, Ren, Lin και Zhao [69] βελτιώνουν περαιτέρω στο την διαδικασία παραγωγής εικόνων εισάγοντας τους αλγόριθμους PNDM, οι οποίοι αποδεικνύεται ότι είναι γενικότεροι του DDIM (ο DDIM είναι μια ειδική περίπτωση των PNDMs). Ειδικότερα, προτείνονται δύο παρόμοιοι αλγόριθμοι, f-PNDM και s-PNDM, οι οποίοι βασίζονται σε αριθμητικές μεθόδους πρώτης και δεύτερης τάξης, αντίστοιχα.

Και οι δύο αλγόριθμοι θεωρούν τη διαδικασία αποθορυβοποίησης ως την Κανονική Διαφορική Εξίσωση (ODE) που παρουσιάζεται στην εξίσωση 2.3.20, αλλά χρησιμοποιούν διαφορετικές αριθμητικές μεθόδους για να την λύσουν.

Ακολουθώντας την προτεινόμενη συντόμηση:

- Pseudo Linear Multi-Step method: PLMS
- Pseudo Runge-Kutta method: PRK
- Pseudo Second-Order Linear Multi-Step method: PLMS'
- Pseudo Improved Euler method: PIE

οι δύο αλγόριθμοι μπορούν να συνοψιστούν στους παρακάτω πίνακες:

Algorithm 6: f-PNDMs. From [69]

```

1:  $x_T \sim N(0, I)$ 
2: for  $t = T - 1, T - 2, T - 3$  do
3:    $x_t, e_t = \text{PRK}(x_{t+1}, t + 1, t)$ 
4: end for
5: for  $t = T - 4, \dots, 1, 0$  do
6:    $x_t, e_t = \text{PLMS}(x_{t+1}, \{e_p\}_{p>t}, t + 1, t)$ 
7: end for
8: return  $x_0$ 

```

Algorithm 7: s-PNDMs. From [69]

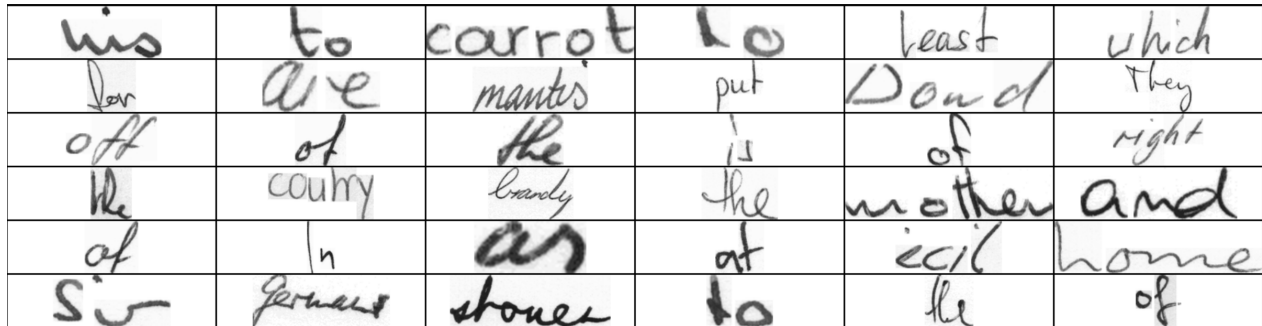
```

1:  $x_T \sim N(0, I)$ 
2: for  $t = T - 1$  do
3:    $x_t, e_t = \text{PIE}(x_{t+1}, t + 1, t)$ 
4: end for
5: for  $t = T - 2, \dots, 1, 0$  do
6:    $x_t, e_t = \text{PLMS}'(x_{t+1}, \{e_p\}_{p>t}, t + 1, t)$ 
7: end for
8: return  $x_0$ 

```

Μορφολογική Διάχυση για Παραγωγή Χειρόγραφου Κειμένου

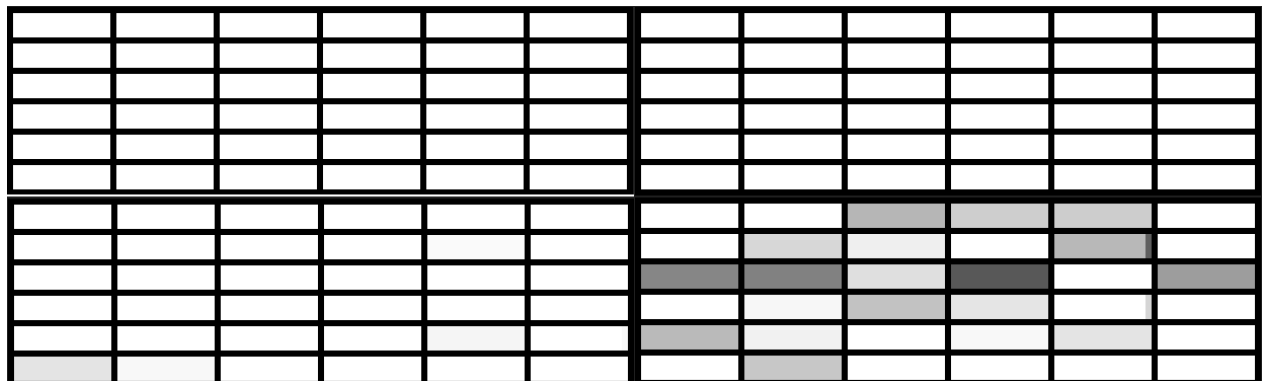
Σε αυτή την υποενότητα, θα χρησιμοποιήσουμε την προτεινόμενη διαδικασία διάχυσης των βημάτων μορφολογικής διαστολής για να βελτιώσουμε περαιτέρω την αποδοτικότητα του τυπικού πλαισίου λανθάνουσας διάχυσης που χρησιμοποιείται στο WordStylist [31]. Ειδικότερα, μετά την κωδικοποίηση των αρχικών εικόνων στον λανθάνοντα χώρο μέσω ενός προεκπαιδευμένου Autoencoder, θα χρησιμοποιήσουμε 30 βήματα διαστολής για να τις επιδεινώσουμε. Όπως και στις προηγούμενες περιπτώσεις, θα χρησιμοποιηθεί ένα τετράγωνο 3×3 ως δομικό τοιχείο για κάθε βήμα. Ο λανθάνων χώρος αποτελείται από 4 κανάλια, όπως φαίνεται παρακάτω, τα οποία υφίστανται υποδειγματοληψία κατά παράγοντα 8 σε κάθε διάσταση, μια μείωση που οδηγεί σε σημαντική σύμπτυξη της αρχικής πληροφορίας από (3, 64, 256) (pixel space) σε έναν πιο συμπαγή και διαχειρίσιμο (4, 8, 32) (latent space).



Σχήμα 0.0.19: Αρχικά προεπεξεργασμένα δείγματα του IAM.



Σχήμα 0.0.20: Τα 4 κανάλια από τις λανθάνουσες αναπαραστάσεις των δειγμάτων του IAM.



Σχήμα 0.0.21: Τα 4 κανάλια από τις λανθάνουσες αναπαραστάσεις των δειγμάτων του IAM μετά από διαστολή.

Πειραματικά Αποτελέσματα

Σύνολα Δεδομένων

Θα πραγματοποιήσουμε τα πειράματά μας στα ακόλουθα σύνολα δεδομένων:

- **MNIST [62]**: Το σύνολο δεδομένων MNIST είναι μια συλλογή χειρόγραφων ψηφίων που έχει γίνει πρότυπο για την αξιολόγηση των αλγορίθμων επεξεργασίας εικόνων και μηχανικής μάθησης. Περιλαμβάνει 60.000 εικόνες εκπαίδευσης και 10.000 εικόνες δοκιμής σε ασπρόμαυρο χρώμα, όλες διαστάσεων 28×28 . Λόγω της απλότητας του και της καθιερωμένης φήμης του, συχνά λειτουργεί ως σημείο εκκίνησης για εκείνους που εισέρχονται στον τομέα της μηχανικής μάθησης, ειδικά στον τομέα της ταξινόμησης εικόνων.
- **CIFAR-10 [64]**: Το σύνολο δεδομένων CIFAR-10 αποτελεί μια ένα από τα πιο συχνά χρησιμοποιούμενα σύνολα δεδομένων για την αξιολόγηση των μοντέλων μηχανικής μάθησης που αφιερώνονται στην αναγνώριση αντικειμένων. Αποτελείται από 60.000 έγχρωμες εικόνες που διαιρούνται σε 10 διακριτές κατηγορίες, όπως αεροπλάνα, πουλιά και αυτοκίνητα, με κάθε κατηγορία να περιέχει 6.000 εικόνες. Αυτές οι εικόνες, με ανάλυση 32×32 , προσφέρουν μια πιο δύσκολη εργασία ταξινόμησης από το MNIST, λόγω της πολυπλοκότητας και της ποικιλομορφίας τους.
- **IAM [63]**: Το συγκεκριμένο σύνολο δεδομένων αποτελεί έναν κεντρικό πόρο για έρευνες που σχετίζονται με τη αναγνώριση ή παραγωγή χειρόγραφου κειμένου. Προσφέρει μια πλούσια συλλογή χειρόγραφου αγγλικού κειμένου που έχει συνταχθεί από πάνω από 600 συγγραφείς. Στην περίπτωση μας, θα χρησιμοποιήσουμε το υποσύνολο Aachen split, με το μήκος των λέξεων να κυμαίνεται από 2 έως 10 χαρακτήρες, με αποτέλεσμα 44,412 εικόνες με 339 στυλ γραφής. Θα υιοθετήσουμε, επίσης, την ίδια διαδικασία προεπεξεργασίας όπως προτείνεται στο [31], προκειμένου να αλλάξουμε το μέγεθος των εικόνων σε ένα σταθερό μέγεθος εικόνας 64×256 .

Μετρικές Επίδοσης

Σε αυτό το σημείο, θα αναλύσουμε σύντομα τις μετρικές που θα χρησιμοποιήσουμε στη συνέχεια, για να επαληθεύσουμε τα εκπαιδευμένα μοντέλα μας. Οι μετρικές επίδοσης είναι οι ακόλουθες:

- **Fréchet Inception Distance [70]**: Το FID είναι μια μετρική που προτάθηκε, αρχικά, για την αξιολόγηση των GANs [6]. Ωστόσο, τελευταία, χρησιμοποιείται ευρέως στους τομείς της Υπολογιστικής Όρασης και της Γενετικής Τεχνητής Νοημοσύνης, για την αξιολόγηση της ποιότητας των παραγόμενων εικόνων. Το FID παρουσιάζεται ως μια βελτίωση του Inception Score [71], καθώς ο υπολογισμός του απαιτεί, τόσο πραγματικά, όσο και συνθετικά δείγματα. Υπολογίζεται από τον παρακάτω τύπο:

$$d^2((m, C), (m_w, C_w)) = \|m - m_w\|_2^2 + \text{Tr}(C + C_w - 2\sqrt{CC_w})$$

- **Structural Similarity [72]**: Το Structural Similarity (SSIM) παρουσιάστηκε ως μια βελτίωση του Universal Quality Index (UQI). Υπήρξε σημαντική μετρική για την αξιολόγηση της ποιότητας των εικόνων, αν και τα τελευταία χρόνια δεν χρησιμοποιείται τόσο συχνά λόγω της εμφάνισης νέων, πιο ισχυρών μετρικών όπως το FID. Υπολογίζεται μέσω του ακόλουθου τύπου:

$$\text{SSIM}(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)}$$

- **Root Mean Square Error**: Θα χρησιμοποιήσουμε, επίσης, και το RMSE ως μετρική, το οποίο ορίζεται ως:

$$\text{RMSE}(I_1, I_2) = \sqrt{\frac{1}{M \times N} \sum_{i=1}^M \sum_{j=1}^N (I_1(i, j) - I_2(i, j))^2}$$

Υλοποίηση

Ανακατασκευή εικόνων

Για τα μοντέλα ανακατασκευής εικόνων μας τόσο για το MNIST [62] όσο και για το CIFAR-10 [64], οι λεπτομέρειες υλοποίησης είναι οι ίδιες, λόγω της αλλαγής του μεγέθους των δειγμάτων του MNIST σε 32×32 (όσο και του CIFAR-10), όπως στο [33]. Πιο συγκεκριμένα, πραγματοποιήσαμε εκπαίδευση 500 εποχών, με batch size 32, χρησιμοποιώντας τον βελτιστοποιητή Adam [73] με ρυθμό μάθησης 2×10^{-5} . Όσον αφορά τη διαδικασία διάχυσης, κάθε εικόνα διαστέλλεται 8 φορές με ένα τετραγωνικό δομικό στοιχείο 3×3 .

Το μοντέλο που χρησιμοποιείται για την αποθορυβοποίηση είναι ένα U-net [60], ενώ το τελικό μοντέλο που χρησιμοποιούμε για τη παραγωγή εικόνων είναι ένας εκθετικός κινούμενος μέσος όρος (Exponential Moving Average - EMA) του εκπαιδευμένου μοντέλου (ρυθμός υποβάθμισης 0,995), προκειμένου να αυξήσουμε τη σταθερότητα της σύγκλισης του μοντέλου μας. Η αρχιτεκτονική του U-net είναι η ίδια με αυτήν που προτείνεται στο [33], ώστε να υπάρχει συνέπεια όσον αφορά τα πειραματικά αποτελέσματα.

Παραγωγή εικόνων υπό συνθήκη

Για το υπό συνθήκη μοντέλο μας για το MNIST [62], οι λεπτομέρειες υλοποίησης είναι οι εξής: Πραγματοποιήσαμε μια εκπαίδευση 100 εποχών, με batch size 128 χρησιμοποιώντας τον βελτιστοποιητή AdamW [74] με ρυθμό μάθησης 10^{-4} . Όσον αφορά τη διαδικασία διάχυσης, κάθε εικόνα διαστέλλεται 11 φορές με ένα τετραγωνικό δομικό στοιχείο 3×3 .

Το μοντέλο που χρησιμοποιείται για την αποθορυβοποίηση είναι, και σε αυτή την περίπτωση, ένα U-net [60], ενώ το τελικό μοντέλο που χρησιμοποιούμε για τη δειγματοληψία είναι πάλι ένα εκθετικός κινούμενος μέσος όρος (Exponential Moving Average - EMA) του εκπαιδευμένου μοντέλου (ρυθμός υποβάθμισης 0,995) για να αυξήσουμε τη σταθερότητα της σύγκλισης του μοντέλου μας. Αντίστοιχα, η αρχιτεκτονική του U-net είναι η ίδια με αυτή που προτείνεται από την [31], ώστε να υπάρχει συνέπεια στα πειραματικά αποτελέσματα. Οι συνθήκες για τις κλάσεις (class conditions) του συνόλου δεδομένων MNIST περνούν στο μοντέλο μέσω ενός στοιχείου διασταυρούμενης προσοχής (cross attention) [61] μετά από ένα απλό embedding layer.

Όπως έχει ήδη συζητηθεί, το πρώτο στάδιο της διαδικασίας δειγματοληψίας είναι ένα απλό υπό συνθήκη GAN (conditional GAN) [75]. Η εκπαίδευσή του διήρκεσε 50 εποχές με batch size 128 και βελτιστοποιητή Adam [73] με ρυθμό μάθησης 10^{-4} . Οι συνθήκες για τις κλάσεις κωδικοποιήθηκαν σε one-hot και στη συνέχεια συνενώθηκαν με τον αρχικό θόρυβο που δόθηκε στον Γεννήτορα (generator).

Επεκτάσεις στο μοντέλο WordStylist

Αυτή η υποενότητα αφορά την εφαρμογή των αλγορίθμων DDIM [68], f-PNDM [69] και s-PNDM [69], οπότε δεν υλοποιήθηκε κάποια διαδικασία εκπαίδευσης. Για να πραγματοποιήσουμε τα πειράματα, χρησιμοποιήσαμε τα βάρη του εκπαιδευμένου μοντέλου WordStylist [31] από [εδώ](#).

Παραγωγή χειρόγραφου κειμένου

Για το υπό συνθήκη μοντέλο μας για το σύνολο δεδομένων IAM [63], οι περισσότερες λεπτομέρειες υλοποίησης σχετικά με τη διαδικασία εκπαίδευσης και τις υπερπαραμέτρους είναι ίδιες με αυτές στο αντίστοιχο μοντέλο για το MNIST. Ωστόσο, κάναμε μια τροποποίηση στον μηχανισμό της συνθήκης (conditioning mechanism), για να ανταποκριθεί στο κείμενο και το στυλ γραφής του συνόλου δεδομένων. Επιπλέον, μεταβίβαμε από τη χρήση της συνάρτησης απώλειας L_2 στη συνάρτηση απώλειας L_1 . Αυτή η αλλαγή οφείλεται στο εύρος τιμών του λανθάνοντος χώρου (latent space) στο μοντέλο μας. Συγκεκριμένα, οι λανθάνουσες αναπαραστάσεις ήταν εντός ενός εύρους από -4 έως 4. Η χρήση της απώλειας L_2 σε αυτό το πλαίσιο οδήγησε σε υπερβολικά υψηλές απώλειες, οδηγώντας σε σημαντικά βήματα κλίσης (gradient steps) και επακόλουθη υπερπροσαρμογή στο σύνολο δεδομένων (overfitting). Η υιοθέτηση της συνάρτησης απώλειας L_1 αντιμετώπισε αυτό το πρόβλημα, προωθώντας μια πιο σταθερή και αποτελεσματική διαδικασία εκπαίδευσης. Ο μηχανισμός συνθήκης που ακολούθησαμε είναι αυτός που προτείνεται στο Wordstylist [31].

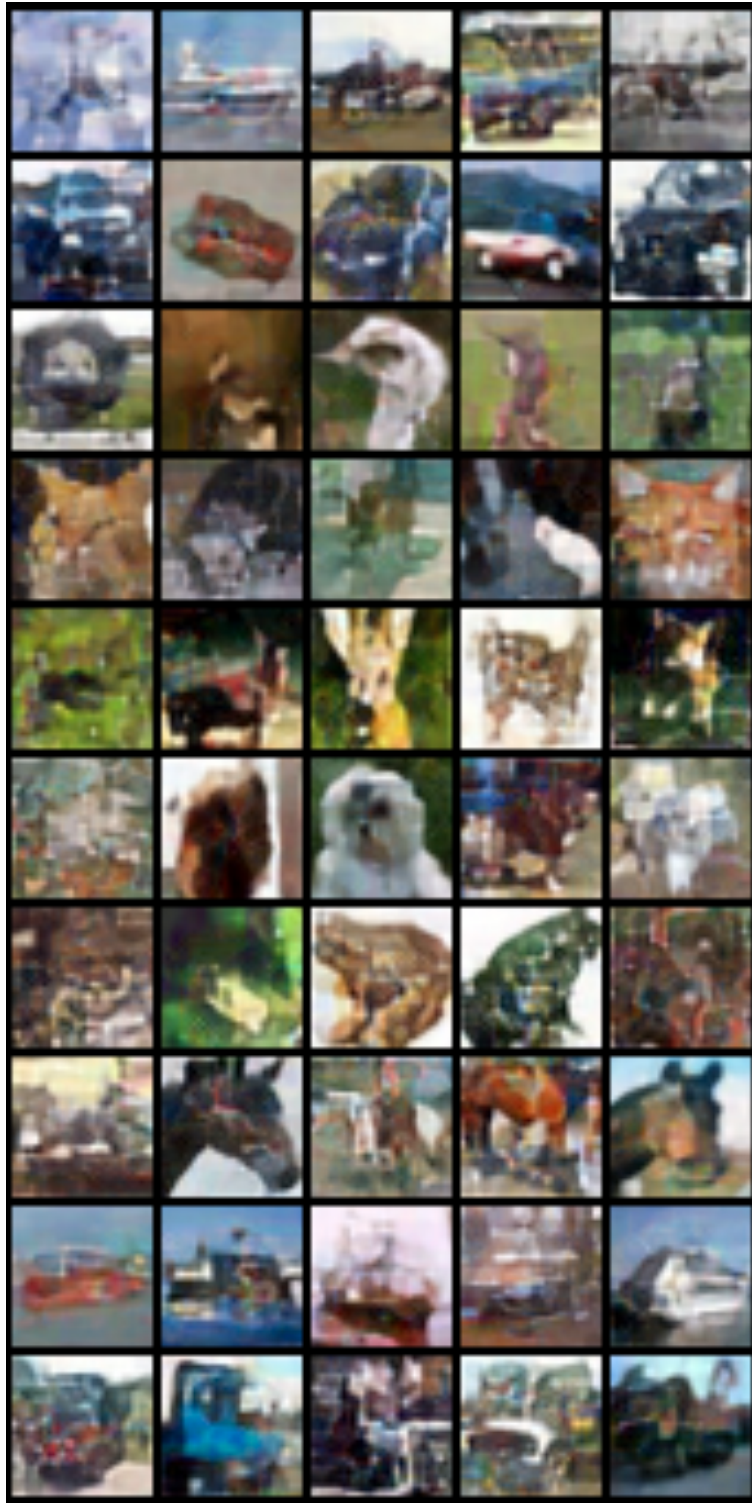
Αποτελέσματα

Μορφολογική Διάχυση

Τα ποιοτικά αποτελέσματα του μοντέλου μας για την ανακατασκευή εικόνων του MNIST και του CIFAR-10, καθώς και υπό συνθήκη παραγωγή εικόνων του MNIST παρουσιάζονται στις παρακάτω εικόνες.



Σχήμα 0.0.22: Ανακατασκευή εικόνων του MNIST.



Σχήμα 0.0.23: Ανακατασκευή εικόνων του CIFAR-10.



Σχήμα 0.0.24: Παραγωγή εικόνων του MNIST υπό συνθήκη.

Στη συνέχεια, θα παρουσιάσουμε τα αποτελέσματα μας, συγκρίνοντας τα με τα αντίστοιχα του [33]. Όπως φαίνεται παρακάτω, τόσο στο MNIST, όσο και στο CIFAR-10, με τη μορφολογική διάχυση επιτυγχάνεται παρόμοια και σε πολλές περιπτώσεις καλύτερη επίδοση από ότι με τις μεθόδους της ψυχρής διάχυσης.

Table 0.1: Comparison of Metrics for the Proposed Diffusions of [33] and ours for MNIST [62].

Degradation	Degraded			Sampled		
	FID	SSIM	RMSE	FID	SSIM	RMSE
Gaussian blur	438.59	0.287	0.287	4.69	0.718	0.154
Inpainting	108.48	0.490	0.262	1.61	0.941	0.068
Downsampling	368.56	0.178	0.231	4.33	0.820	0.115
Dilation (our approach)	314.87	0.005	0.78	2.46	0.96	0.05

Table 0.2: Comparison of Metrics for the Proposed Diffusions of [33] and ours for CIFAR-10 [64].

Degradation	Degraded			Sampled		
	FID	SSIM	RMSE	FID	SSIM	RMSE
Gaussian blur	298.60	0.315	0.136	80.08	0.773	0.075
Inpainting	40.83	0.615	0.143	8.92	0.859	0.068
Downsampling	358.99	0.279	0.146	152.76	0.411	0.155
Dilation (our approach)	217.81	0.11	0.39	52.42	0.63	0.11

Ο επόμενος πίνακας δείχνει τα αποτελέσματα για την παραγωγή εικόνων του MNIST υπό συνθήκη. Επειδή δεν υπάρχει ακόμη κάποια άλλη προσέγγιση που αφορά τη γενικευμένη, ντετερμινιστική διάχυση για την κλασική παραγωγή υπό συνθήκη, τα αποτελέσματα των πειραμάτων μας θα συγκριθούν με εκείνα των τυπικών μοντέλων διάχυσης γκαουσιανής, τα οποία έχουν ορισμένες ομοιότητες με την προσέγγισή μας, όσον αφορά την πολυπλοκότητα της διαδικασίας εκπαίδευσης και παραγωγής εικόνων.

Πιο συγκεκριμένα, θα εκπαιδεύσουμε πρώτα ένα μοντέλο διάχυσης με 10 βήματα και θα μετρήσουμε το FID του με το αρχικό σύνολο δεδομένων, ενώ τα άλλα μοντέλα γκαουσιανής διάχυσης θα εκπαιδευτούν για 100 βήματα αλλά θα χρησιμοποιηθούν μόλις 10 για την παραγωγή εικόνων, μέσω των αλγορίθμων DDIM [68], f-PNDM [69] και s-PNDM [69]. Όπως φαίνεται, όλες αυτές οι μέθοδοι αποτυγχάνουν προφανώς να παράγουν αξιοπρεπή δείγματα, λόγω του μικρού αριθμού των βημάτων. Αντίθετα, η προσέγγισή μας, που βασίζεται στη μορφολογική διάχυση, φαίνεται να παράγει αποτελέσματα που είναι πολύ καλύτερα, όπως υποδεικνύει και το αντίστοιχο FID.

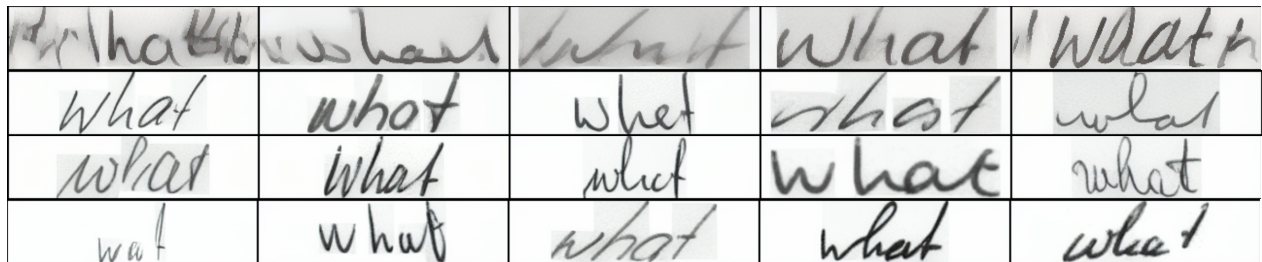
Table 0.3: Comparison of Metrics for Different Conditional Generation Models with 10 sampling steps for MNIST [62].

Model	FID
Gaussian Diffusion (100 steps)	4.31
Gaussian Diffusion (100 steps) with 10 DDIM [68] sampling steps	163.90
Gaussian Diffusion (100 steps) with 10 f-PNDM [69] sampling steps	293.76
Gaussian Diffusion (100 steps) with 10 s-PNDM [69] sampling steps	226.02
Gaussian Diffusion (10 steps)	123.31
Morphological Diffusion (10 steps) with dilated noise	41.30
Morphological Diffusion (10 steps) with GAN masks	19.45
Morphological Diffusion (10 steps) with GAN masks after morphological closing	15.53

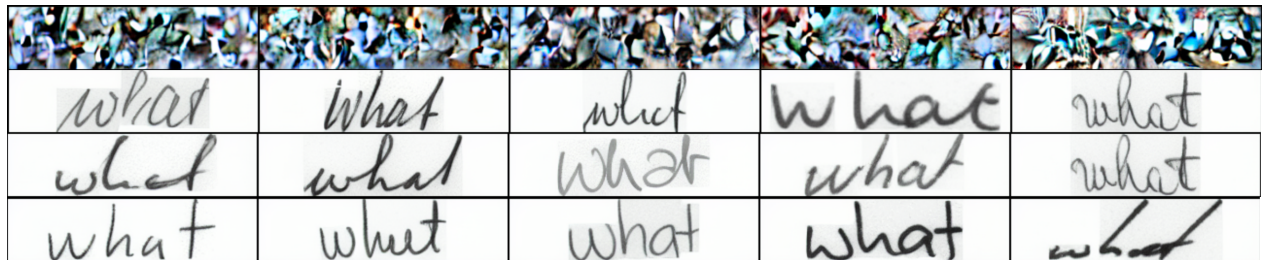
Επεκτάσεις στο μοντέλο WordStylist

Σε αυτό το σημείο, θα παρουσιάσουμε τα ποιοτικά αποτελέσματα μετά τη χρήση των αλγορίθμων DDIM [68], f-PNDM [69] και s-PNDM [69] στο μοντέλο WordStylist [31]. Οι παρακάτω εικόνες απεικονίζουν αυτά τα αποτελέσματα για έναν αριθμό βημάτων από 20 έως 200, για τον αλγόριθμο DDIM (τα αποτελέσματα των τριών αλγορίθμων είναι αρκετά παρόμοια), καθώς και μία σύγκριση όλων των αλγορίθμων (συμπεριλαμβανομένης της μη χρήση κάποιου αλγορίθμου) για 100 βήματα. Παρατηρούμε ξεκάθαρα ότι με την εισαγωγή αυτών των αλγορίθμων μπορούμε να μειώσουμε δραματικά την ανάγκη για πολλά βήματα, καθώς ακόμη και 50 βήματα μπορούν να είναι επαρκή για τη διατήρηση της ποιότητας των παραγόμενων εικόνων. Η χρήση λιγότερων από 50 βημάτων δειγματοληψίας έχει όμως αρνητικό αποτέλεσμα στην ποιότητα της δειγματοληψίας, όπως φαίνεται στην πρώτη σειρά των αποτελεσμάτων για τον αλγόριθμο DDIM.

Έτσι, αντιμετωπίζουμε μία από τις κύριες περιορισμούς του μοντέλου WordStylist, που αφορά την υπολογιστική του πολυπλοκότητα, καθώς τα 600 βήματα δειγματοληψίας που χρειάζονταν για τη γεννήτρια δεν είναι πλέον απαραίτητα και η ποιότητα των αποτελεσμάτων παραμένει σχεδόν ίδια.



Σχήμα 0.0.25: Αποτελέσματα με εφαρμογή του αλγορίθμου DDIM. Κάθε σειρά περιέχει 5 αποτελέσματα για τη ίδια λέξη ('what') με 5 τυχαία στυλ γραφής και αριθμό βημάτων a) 20, b) 50, c) 100, d) 200.



Σχήμα 0.0.26: Σύγκριση αποτελεσμάτων για διαφορετικούς αλγορίθμους παραγωγής εικόνων για 100 βήματα. Κάθε σειρά περιέχει 5 αποτελέσματα για τη λέξη ('what') με 5 τυχαία στυλ γραφής και αλγόριθμο a) default, b) DDIM, c) f-PNDM, d) s-PNDM.

Μορφολογική Διάχυση για Παραγωγή Χειρόγραφου Κειμένου

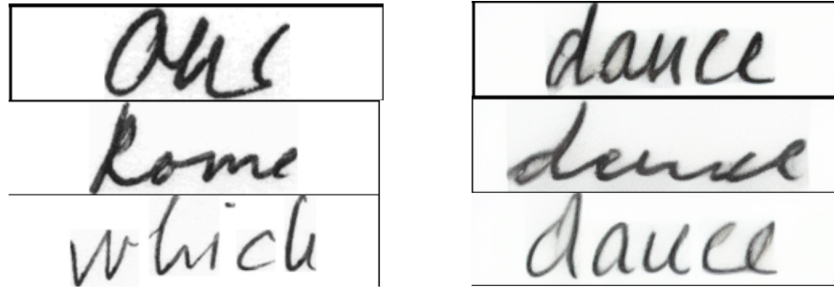
Σε αυτή την ενότητα, καταπιαστήκαμε με μια εξονυχιστική εξέταση των ποιοτικών αποτελεσμάτων που προέκυψαν από τα πειράματά μας. Πέρα από τις ποσοτικές μετρήσεις και τις στατιστικές αναλύσεις, είναι κρίσιμο να αξιολογήσουμε την ποιότητα και τα χαρακτηριστικά των αποτελεσμάτων με έναν πιο περιγραφικό και ερμηνευτικό τρόπο.

Ξεκινάμε αυτή την εξέταση παρουσιάζοντας οπτικά ανακατασκευασμένα δείγματα που εξήχθησαν από το σύνολο δεδομένων IAM. Η παρακάτω εικόνα απεικονίζει καθαρά την επάρκεια του μοντέλου στη συλλογή, τόσο των συνθηκών κειμένου, όσο και των ατομικών στυλ γραφής που σχετίζονται με κάθε χειρόγραφη λέξη. Παρά το γεγονός ότι εκπαιδεύτηκε για μόλις 250 εποχές και 30 χρονικά βήματα, το μοντέλο επιδεικνύει αρκετά καλή απόδοση σε ένα ποικίλο φάσμα στυλ γραφής και λέξεων, υπογραμμίζοντας την πολυμορφία και την αποτελεσματικότητά του.

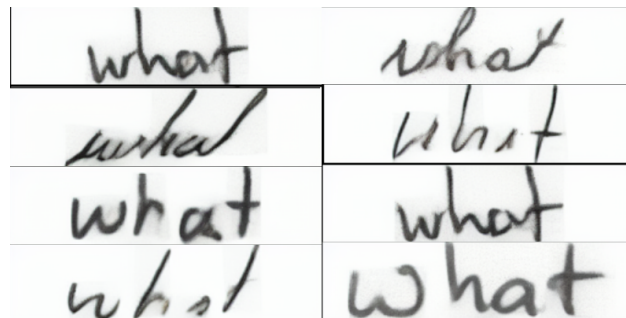
four	the	about	of	undo	was
the	which	not	four	to	oppose
to	would	he	power	then	of
but	in	to	the	face	is
in	the	we	the	are	cut
read	Britain	well,	body	the	is
of	less	ever	of	sound	here
love	she	men	many	today	too
of	she	Cyprus	Spanish	to	going
rapid	that	enable	we	The	than
the	his	and	the	by	free
the	his	very	well	and	rough
promis	being	labors	been	at	is
with	and	many	enable	barley	started
the	and	the	and	six	the
doctor	and	practice	at	is	part
stitch	she	only	Africa	check	Both
and	for	of	to	up	the
said	the	Mauro	on	ful	leap
do	three	he	This	olast	the
last	end	she	best	script	promis
rooms	the	home	that	ceramic	of
been	three	being	and	was	just
all	his	thread	classes	hid	how
already	you	agency	city	loot	new
to	in	your	first	then	the
Burdett	little	face	eight	came	of
just	working	which	all	days	the
work	output	the	than	to	when
who	last	and	NEWS	of	of

Σχήμα 0.0.27: Ποιοτικά αποτελέσματα για την ανακατασκευή εικόνων του IAM.

Προκειμένου να αποκτήσουμε μια ακόμα καλύτερη εικόνα των δυνατοτήτων του μοντέλου μας, θα παρουσιάσουμε, επιπλέον, κάποια ποιοτικά αποτελέσματα για λέξεις (dance σε αυτή την περίπτωση) που δεν υπήρχαν στο αρχικό σύνολο δεδομένων (out-of-vocabulary), καθώς και κάποια ποιοτικά αποτελέσματα για συγκεκριμένη λέξη (what) με διαφορετικά στυλ γραφής. Αξίζει να σημειωθεί πως και στις δύο περιπτώσεις, οι εικόνες από τις οποίες ξεκίνησε η παραγωγή των τελικών εικόνων ήταν τυχαίες κατεστραμμένες εικόνες άλλων λέξεων του IAM, γεγονός που προσδίδει τυχαιότητα στην διαδικασία παραγωγής εικόνων.



Σχήμα 0.0.28: Out-Of-Vocabulary αποτελέσματα για τη λέξη "dance"



Σχήμα 0.0.29: Αποτελέσματα για διαφορετικά στυλ γραφής της λέξης "what"

Σε αυτό το σημείο, πρέπει να σημειωθεί πως οι παραπάνω εικόνες δεν αντικατοπτρίζουν απόλυτα την επίδοση του μοντέλου, καθώς υπάρχουν και περιπτώσεις, όπου τα αποτελέσματα είναι χαμηλότερης ποιότητας, ανάλογα και με τον συνδυασμό λέξης και στυλ γραφής.

Στρέφοντας τώρα την προσοχή μας στην ποσοτική αξιολόγηση του μοντέλου μας, θα παρουσιάσουμε μια σύγκριση των FIDs, ώστε να αποκτήσουμε ένα αριθμητικό μέτρο της ποιότητας και της ποικιλομορφίας των εικόνων που δημιούργησε το μοντέλο μας.

Ο παρακάτω πίνακας παρουσιάζει μια σύγκριση των FIDs του μοντέλου μας έναντι τριών σύγχρονων κορυφαίων μεθόδων: GANWriting [51], SmartPatch [30], και WordStylist [31]. Το μοντέλο μας καταγράφει σκορ 27.85, παρουσιάζοντας την ανταγωνιστικότητά του, σε σύγκριση με τις μετρικές απόδοσης των παραπάνω μοντέλων.

Model	FID ↓
GANWriting [51]	29.94
SmartPatch [30]	22.55
WordStylist [31]	22.74
Ours	27.85

Table 0.4: FID score comparison for our model with state-of-the-art.

Όπως αναφέρεται και στο [31], παρόλο που το FID είναι μια συχνά χρησιμοποιούμενη μετρική για την αξιολόγηση παραγωγικών μοντέλων, η καταλληλότητά του μπορεί να αμφισβητηθεί για εργασίες που δεν ασχολούνται με φυσικές εικόνες παρόμοιες με αυτές του ImageNet [76], το σύνολο δεδομένων στο οποίο εκπαιδεύτηκε αρχικά το δίκτυο.

Εξαιτίας του παραπάνω περιορισμού της μετρικής FID, επεκτείνουμε την αξιολόγησή μας, για να αξιολογήσουμε και την προσαρμογή στο στυλ γραφής του μοντέλου μας, χρησιμοποιώντας το πλαίσιο αξιολόγησης που περιγράφεται στο [31]. Ειδικότερα, θα χρησιμοποιήσουμε ένα CNN ResNet18 [77], για να κατηγοριοποιήσουμε τα στυλ γραφής των παραγόμενων δειγμάτων μας. Δεδομένου ότι ο επιλεγμένος ταξινομητής μας έχει προεκπαιδευτεί στο ImageNet [76], θα πραγματοποιηθεί ένα fine-tuning στο σύνολο δεδομένων IAM πριν προχωρήσουμε στην ταξινόμηση. Τα αποτελέσματα αυτής της ανάλυσης παρουσιάζονται λεπτομερώς στον παρακάτω πίνακα.

Test Set	Accuracy (%) \uparrow
GANWriting [51]	4.81
SmartPatch [30]	4.09
WordStylist [31]	70.6
Ours	26.7

Table 0.5: Style Adaptation comparison for our model with state-of-the-art.

Ο πίνακας αποκαλύπτει ότι τα παραγόμενα δείγματα του μοντέλου μας επιτυγχάνουν ακρίβεια 26.7%, υπερβαίνοντας σημαντικά το GANwriting [51] και το SmartPatch [30], χωρίς όμως να φτάνουν την απόδοση που παρουσιάζει το WordStylist [31]. Ενώ το 26.7% μπορεί αρχικά να φαίνεται μέτριο, είναι ενδεικτικό μιας αρκετά καλής απόδοσης στη μίμηση των στυλ γραφής, ειδικά αν ληφθεί υπόψη η παρουσία 339 διαφορετικών στυλ και η σημαντική ανισορροπία στο σύνολο δεδομένων IAM, όσον αφορά την αναπαράσταση κάθε συγκεκριμένου στυλ.

Συνεισφορές και μελλονικές προεκτάσεις

Σε αυτή τη διπλωματική, αντιμετωπίσαμε το πρόβλημα της παραγωγής χειρόγραφου κειμένου χρησιμοποιώντας ένα latent diffusion model (LDM) με μορφολογική διάχυση. Αρχικά, παρουσιάσαμε το θεωρητικό υπόβαθρο, στο οποίο βασίζεται η προσέγγισή μας, καθώς και τη σχετική βιβλιογραφία, που αποτέλεσε την έμπνευση για την αρχιτεκτονική, την εκπαίδευση και τις λειτουργίες των μοντέλων μας, συμπεριλαμβανομένης της ψυχρής διάχυσης (cold diffusion) για γενικευμένες διαδικασίες διάχυσης [33], λανθάνουσας διάχυσης [32] και των state-of-the-art αρχιτεκτονικών για τη παραγωγή χειρόγραφου κειμένου [31, 51, 30].

Αντιμετωπίσαμε μια ευρεία ποικιλία προβλημάτων που σχετίζονται με την προτεινόμενη διαδικασία μορφολογικής διάχυσης. Για αρχή, αυτή είναι μία από τις πρώτες, αν όχι η πρώτη, προσπάθεια να οριστεί μια μη γραμμική διαδικασία διάχυσης, η οποία συνεπάγεται σημαντικούς περιορισμούς, καθώς τα χαρακτηριστικά της λειτουργίας διάχυσης επέτρεψαν έναν πολύ συγκεκριμένο αριθμό χρονικών βημάτων, ανάλογα με το μέγεθος της εικόνας. Επιπλέον, οι πλήρως κατεστραμμένες (degraded) εικόνες και των δύο συνόλων δεδομένων MNIST [62] και IAM [63] (τα οποία ήταν τα κύρια σύνολα δεδομένων με τα οποία πειραματιστήκαμε) είχαν την τάση να είναι μια εντελώς 'λευκή' εικόνα, επηρεάζοντας έτσι αρνητικά τη διαφοροποίηση και την ποικιλία των παραγόμενων δειγμάτων των μοντέλων μας. Επιπλέον, αντίθετα με άλλες εργασίες με γενικευμένες αποφασιστικές διαδικασίες διάχυσης, παρείχαμε μια προσέγγιση για παραγωγή εικόνων υπό συνθήκη (conditional generation) χρησιμοποιώντας ντετερμινιστικές συναρτήσεις καταστροφής εικόνων με την εκπαίδευση ενός υπό συνθήκη παραγωγικού ανταγωνιστικού δικτύου (conditional generative adversarial network) για τον μετασχηματισμό του καθαρού θορύβου σε μια εικόνα που ταιριάζει με την κατανομή των κατεστραμμένων εικόνων.

Μετά την ολοκλήρωση των πειραμάτων στα σύνολα δεδομένων MNIST και CIFAR-10 και τη σύγκριση των αποτελεσμάτων μας με τα αντίστοιχα της εργασίας της ψυχρής διάχυσης [33], πραγματοποιήσαμε τα κύρια πειράματά μας με το σύνολο δεδομένων IAM με σκοπό τη παραγωγή χειρόγραφου κειμένου. Αρχικά, παρείχαμε κάποιες βελτιστοποιήσεις σχετικά με τη διαδικασία παραγωγής εικόνων του μοντέλου WordStylist [31], χρησιμοποιώντας τις μεθόδους των μοντέλων Denoising Diffusion Implicit Models (DDIMs) [68] και τις ψευδοαριθμητικές μεθόδους για μοντέλα διάχυσης (Pseudo Numerical methods for Diffusion Models) [69], καταλλήγοντας σε σημαντική μείωση του απαιτούμενου χρόνου παραγωγής εικόνων, καθώς διατηρήσαμε παρόμοια ποιότητα των παραγόμενων

εικόνων μετά τη μείωση των συνολικών βημάτων από 600 σε μόλις 50 ($12\times$ επιτάχυνση). Με αυτόν τον τρόπο, αντιμετωπίσαμε έναν από τους κύριους περιορισμούς αυτής του μοντέλου WordStylist χωρίς να επηρεάσουμε την ποιότητα και τη ποικιλία των παραγόμενων δειγμάτων.

Χρησιμοποιώντας το πλαίσιο μορφολογικής διάχυσης που περιγράφηκε παραπάνω, προσπαθήσαμε να μειώσουμε περαιτέρω τις υπολογιστικές ανάγκες για την παραγωγή χειρόγραφου κειμένου, εισάγοντας ένα μοντέλο που εκπαιδεύτηκε για 30 βήματα. Εφόσον, σε αυτή την περίπτωση, τόσο η εκπαίδευση, όσο και η παραγωγή εικόνων απαιτούν 30 βήματα (σε αντίθεση με την προηγούμενη μέθοδο που απαιτούσε 50 βήματα για παραγωγή εικόνων, αλλά η εκπαίδευση πραγματοποιήθηκε για 1000 βήματα), μειώνοντας έτσι όχι μόνο τον χρόνο και τους πόρους που απαιτούνται για αποτελεσματική σύνθεση εικόνων αλλά και για επαρκή εκπαίδευση (250 εποχές). Όπως συζητήθηκε στα πειραματικά αποτελέσματα, το μοντέλο μας σημείωσε αξιοπρεπή επίδοση και ήταν σε θέση να γενικεύσει, καθώς ήταν ικανό να παράγει λέξεις εκτός λεξιλογίου (Out-Of-Vocabulary). Παρόλο που η απόδοση του δεν έφτασε το επίπεδο του WordStylist [31], παρείχαμε ένα proof-of-concept ότι, λαμβάνοντας υπόψη τις πρόσφατες προόδους στις γενικευμένες διαδικασίες διάχυσης, η επιλογή και η προσαρμογή μιας κατάλληλης διαδικασίας διάχυσης σε ένα σύνολο δεδομένων μπορεί να έχει θετικό αντίκτυπο στην απόδοση του μοντέλου. Στην περίπτωσή μας, αυτή η επίδραση ήταν ένα trade-off μεταξύ της ποιότητας της εικόνας και της χρονικής πολυπλοκότητας για την παραγωγή εικόνων, αλλά, εφόσον πρόκειται για έναν πρόσφατα ανακαλυφθέντα τομέα έρευνας, η περαιτέρω έρευνα σε τέτοιους τύπους διαδικασιών διάχυσης μπορεί να οδηγήσει σε ακόμη καλύτερα αποτελέσματα.

Παρά τα ενθαρρυντικά αποτελέσματά μας, υπάρχουν ακόμη διάφοροι περιορισμοί στην προσέγγισή μας, γεγονός που μπορεί να ωθήσει περαιτέρω μελλοντική έρευνα στον τομέα των γενικευμένων διαδικασιών διάχυσης και της παραγωγής χειρόγραφου κειμένου. Τα παρακάτω bullets παρουσιάζουν κάποιες πιθανές μελλοντικές ερευνητικές περιοχές για την αντιμετώπιση των τρέχοντων περιορισμών της προσέγγισής μας.

- **Διαδικασία Διάχυσης:** Κατά τη διάρκεια των πειραμάτων μας, εφαρμόστηκε μια μέθοδος μορφολογικής διάχυσης, όπου η καταστροφή των εικόνων σε κάθε χρονικό βήμα ήταν σταθερή και ανεπηρέαστη από αυτό το χρονικό βήμα. Χρησιμοποιήσαμε ένα τετραγωνικό δομικό στοιχείο (square structuring element) 3×3 για τη διάχυση εικόνων σε κάθε χρονικό βήμα. Αυτό οδήγησε σε μια απότομη διαδικασία διάχυσης, υπονομεύοντας την αποτελεσματικότητα της σύλληψης της κατανομής δεδομένων από το μοντέλο μας. Σε αντίθεση με αυτό, οι κλασικές μέθοδοι διάχυσης εντείνουν την υποβάθμιση με κάθε αυξανόμενο χρονικό βήμα. Η προσαρμογή και βελτιστοποίηση ενός νέου προγράμματος καταστροφής εικόνων, που ταιριάζει σε μη γραμμικές συναρτήσεις αποτελεί μια πιθανή περιοχή για μελλοντική έρευνα.
- **Αλγόριθμος παραγωγής εικόνων:** Όπως φαίνεται από την ψυχρή διάχυση [33], ο αλγόριθμος παραγωγής εικόνων που χρησιμοποιήθηκε στην έρευνά μας βασίζεται σε χαρακτηριστικά των γραμμικών συναρτήσεων. Συνεπώς, ένας τομέας για μελλοντική έρευνα περιλαμβάνει την ενίσχυση αυτού του αλγορίθμου, προκειμένου να ανταποκρίνεται αποτελεσματικά και σε μη γραμμικές συναρτήσεις καταστροφής.
- **Προσθήκη ενός pix2pix μοντέλου για την βελτίωση της ποιότητας των παραγόμενων εικόνων:** Δεδομένης της μειωμένης ποιότητας των δειγμάτων που παράχθηκαν, μια πιθανή βελτίωση θα μπορούσε να περιλαμβάνει την ενσωμάτωση ενός μοντέλου pix2pix [57]. Αυτό το μοντέλο θα μπορούσε να τοποθετηθεί, για να επεξεργαστεί τις εικόνες εξόδου από το μοντέλο διάχυσής μας, βελτιώνοντας την ποιότητά τους. Αυτή η προσέγγιση θα ήταν παρόμοια με τη μεθοδολογία που περιγράφηκε στην ανάλυση της προτεινόμενης μεθόδου μας, αλλά έχει επίκεντρο, όχι στη δημιουργία αρχικών δειγμάτων εικόνας, αλλά στην εφαρμογή μιας διαδικασίας υπερ-ανάλυσης (super-resolution) στις ήδη παραγόμενες εικόνες. Δεδομένου ότι το pix2pix χρησιμοποιεί ένα υ-νετ για τον γεννήτορα (generator), η ενσωμάτωσή του θα ταίριαζε στην συνολική αρχιτεκτονική του μοντέλου, καθώς θα απαιτούσε ένα επιπλέον βήμα στη διαδικασία δειγματοληψίας για να βελτιώσει σημαντικά την ποιότητα των παραγόμενων εικόνων.

Chapter 1

Introduction

1.1	Morphological Mathematics	30
1.2	Diffusion Models	31
1.3	Handwritten Text Recognition and Generation	31
1.4	Outline	32

1.1 Morphological Mathematics

Morphological mathematics plays a pivotal role in digital image processing, offering a toolkit of operations to analyze and manipulate the geometrical structures within images. It is a branch of set theory that deals with the transformation and analysis of spatial structures, and although its applications were mainly focused on processing binary images and grayscale images, they have recently been expanded to RGB images as well. Morphological operations are fundamental in a plethora of applications, including edge detection, noise reduction, image enhancement, and object recognition. The main operations which are commonly used are the following:

- Dilation
- Erosion
- Opening
- Closing

Intuitively, dilation tends to brighten the original image to an extent determined by a defined structuring element, while erosion, as the dual operation of dilation, darkens the original image. Additionally, closing operation is performed by applying an erosion after a dilation, and its main effect can be described as bridging gaps between disparate segments of objects and removing small holes. Likewise, opening is performed by applying a dilation after an erosion and results in separating objects that are closely intertwined. These morphological operations have proved indispensable across an array of disciplines, from computer vision and artificial intelligence to biomedical imaging and geographical information systems and their effect can be visualized in Figure 1.1.1.

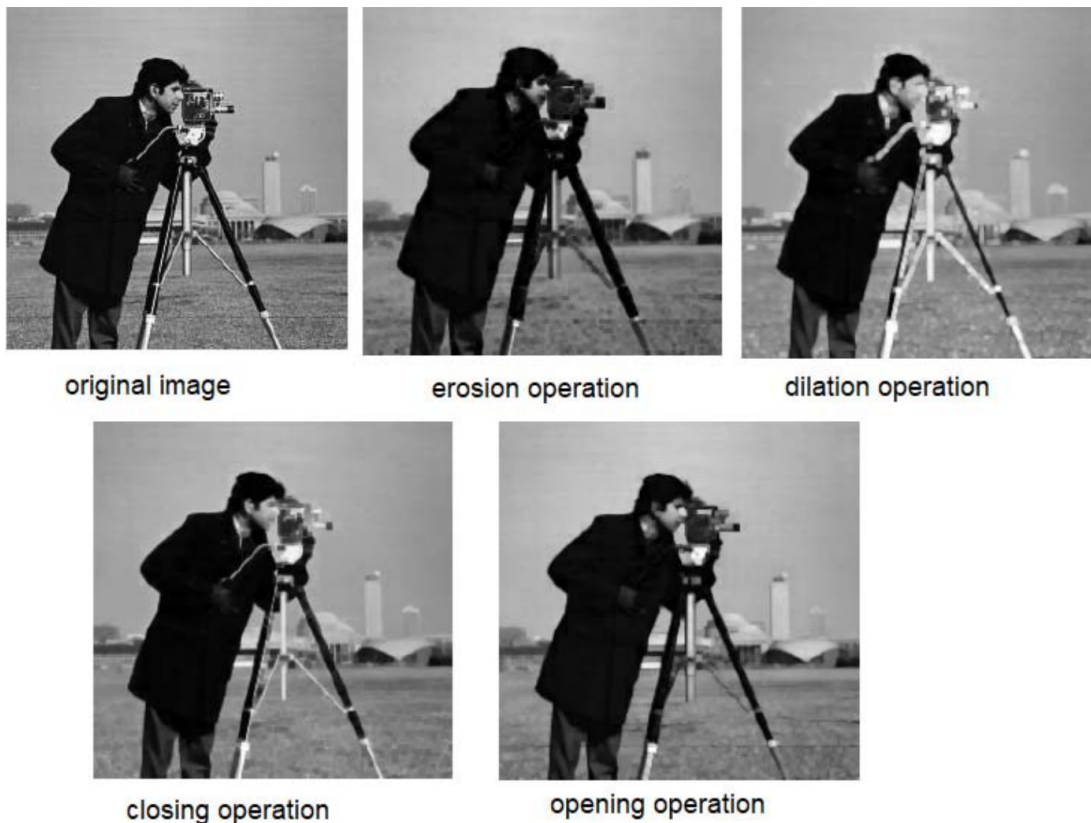


Figure 1.1.1: Basic Morphological Operations. From [17]

1.2 Diffusion Models

In the intricate domain of deep learning, diffusion models [18, 20, 21, 59] have recently surged into prominence, emerging as a powerful tool for generating complex and high-quality samples. Originally inspired by statistical physics and probabilistic modeling, these models interpret the data generation process as a diffusion process, a continuous-time Markov process, that gradually transforms a simple distribution, like Gaussian noise, into the target data distribution.

Diffusion models rest upon the principle of reverse-time diffusion. They model the data generation mechanism as a process of corrupting the original data sample step-by-step, with Gaussian noise until it transforms into pure noise. The learning task is to reverse this process: starting with noise, the model iteratively refines the sample, reducing the noise at each step until a meaningful sample from the data distribution is obtained.

One of the profound advantages of diffusion models is their flexibility. Unlike some generative models that can struggle with issues like mode collapse, diffusion models showcase a robust performance across varied types of data. They are also amenable to a range of architectural and training innovations, accommodating improvements and refinements in model architecture, training techniques, and sampling algorithms.

The diffusion process can be summarized in Figure 1.2.1

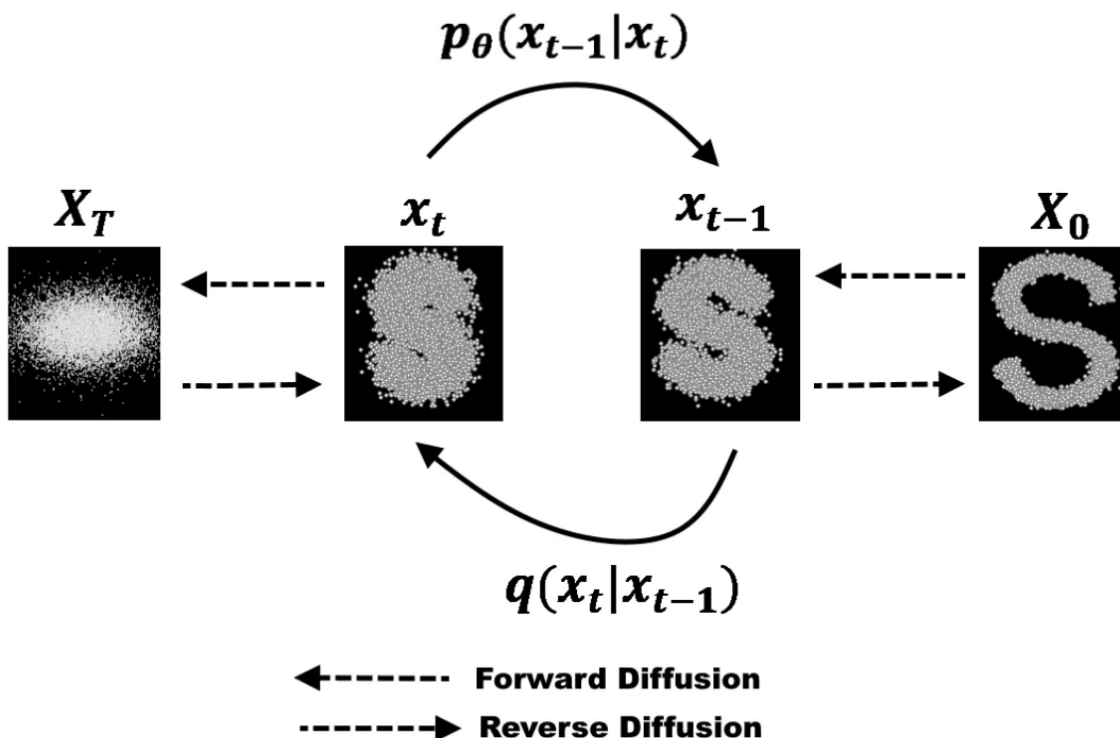


Figure 1.2.1: Diffusion Process. From [here](#)

1.3 Handwritten Text Recognition and Generation

In the interface between the analog expressiveness of handwriting and the digital precision of computational technologies, Handwritten Text Recognition (HTR) and Handwritten Text Generation (HTG) have emerged as twin pillars enabling a seamless transition and interaction between these two worlds.

HTR plays a quintessential role in deciphering, interpreting, and converting handwritten content into digital text. It addresses the complexities arising from the inherent variability and uniqueness in individual handwriting styles, capitalizing on advances in artificial intelligence and machine learning to enhance accuracy and

efficiency. HTR finds applications in sectors ranging from archival digitization where historical documents are converted into machine-readable formats, to healthcare where patient notes and prescriptions are digitally recorded and managed.

Complementing HTR, HTG is an equally potent technology focused on emulating and generating human-like handwritten content. Utilizing generative models embedded with deep learning algorithms, HTG creates handwritten texts that encapsulate the stylistic nuances and aesthetics associated with individual or generalized handwriting styles.

Figure 1.3.1 illustrates an example of deep learning based system for Handwritten Text Recognition, which consists of some Convolutional Neural Network (CNN) layers, some Recurrent Neural Network (RNN) layers and a Connectionist Temporal Classification (CTC) layer.

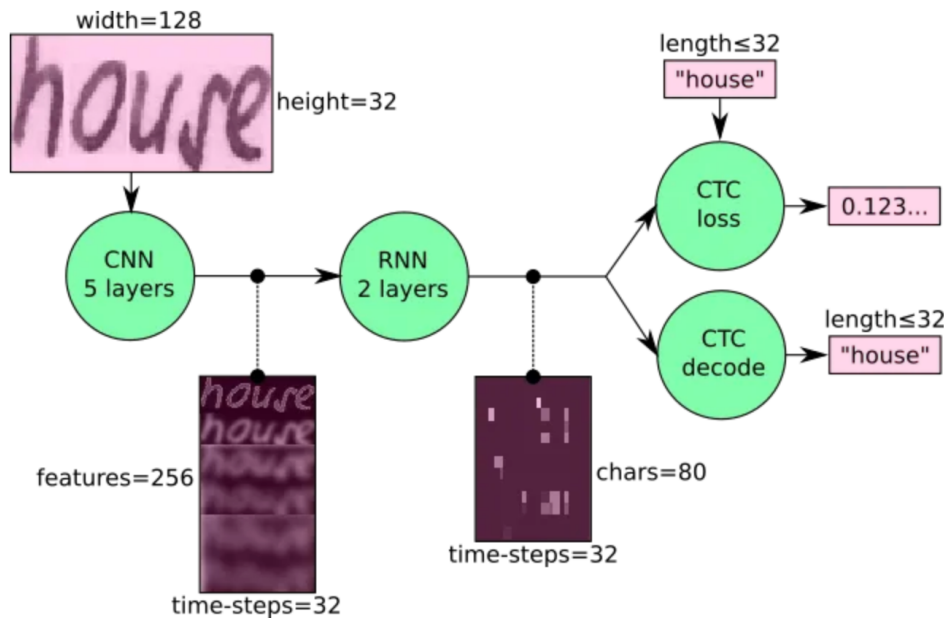


Figure 1.3.1: An example of an HTR system. From [here](#)

1.4 Outline

This thesis consists of 6 chapters, each one of which is summarised in the following outline:

- Chapter 1: We provided an introduction to the main topics which are discussed in this thesis and in which we base the theoretical foundations of our proposed method and experimental setup, namely, morphological mathematics, diffusion models and handwritten text recognition and generation.
- Chapter 2: We present the theoretical background in which the work of this thesis is founded. An introduction to deep learning as well as an more extended introcution to the topics discussed in chapter 1 are included in this chapter.
- Chapter 3: We summarize the related work to this thesis, by introducing the state-of-the-art approaches to the methods in which our work is based on, such as generalised diffusion processes and latent diffusion along with the most popular models for HTG.
- Chapter 4: We analyze our proposed method regarding morphological diffusion starting from easier datasets to HTG along with an extension to the current state-of-the-art model for HTG.
- Chapter 5: We discuss our experimental setup including the examined datasets, the evaluation metrics, our implementation details concerning our models' archutecture and training hyperparameters and we report both qualitative and quantitative results for our experiments.

- Chapter 6: We summarize the conclusions and the results of this thesis and provide an overview of possible future work that was not included.

Chapter 2

Theoretical Background

2.1	Introduction to Deep Learning	36
2.1.1	Architectures of Deep Neural Networks	36
2.1.2	Training	43
2.1.3	Introduction to Generative Models	46
2.2	Introduction to Morphological Mathematics	48
2.2.1	Structuring Elements	48
2.2.2	Morphological Operations	49
2.3	Introduction to Diffusion Models	51
2.3.1	Background	51
2.3.2	Training-Sampling	52
2.3.3	Improvements	53
2.3.4	Score-based generative modeling with stochastic differential equations	54
2.4	Introduction to Handwritten Text Recognition and Generation	56
2.4.1	Handwritten Text Recognition	56
2.4.2	Handwritten Text Generation	57

2.1 Introduction to Deep Learning

Deep learning is a transformative technology that sits at the intersection of artificial intelligence and machine learning. In particular, deep learning is a subset of machine learning, which itself is a branch of artificial intelligence. What sets deep learning apart is its use of artificial neural networks, especially those with a deep structure. These networks are computational models inspired by the human brain's interconnected neurons. By processing data through multiple layers of neurons, deep learning algorithms can discern intricate patterns and make decisions based on them.

The true power of deep neural networks lies in their ability to learn hierarchically. In other words, as data passes through each layer of the network, the model gradually extracts increasingly abstract and complex features from the raw input. This layered, hierarchical approach enables deep learning models to automatically and adaptively improve their performance as they are exposed to more data.

One of the most captivating aspects of deep learning is its vast array of applications. From image and speech recognition to predicting stock market trends, deep learning is reshaping industries by offering sophisticated solutions to longstanding challenges.

2.1.1 Architectures of Deep Neural Networks

Convolutional Neural Networks

Convolutional Neural Networks (CNNs) stand as one of the cornerstones of deep learning, particularly in tasks related to visual data. Drawing inspiration from the human visual system and how we perceive objects in our surroundings, CNNs have redefined the frontier of image recognition and classification.

Unlike traditional neural networks, which process inputs in a fully connected manner, CNNs leverage the spatial nature of images. They recognize that pixels close to each other in an image are often more related than pixels far apart. This understanding allows CNNs to capture local patterns, such as edges or textures, in the initial layers, gradually assembling them into more complex structures, like shapes or objects, in deeper layers [1].

The architecture of a typical Convolutional Neural Network is depicted in Figure 2.1.1

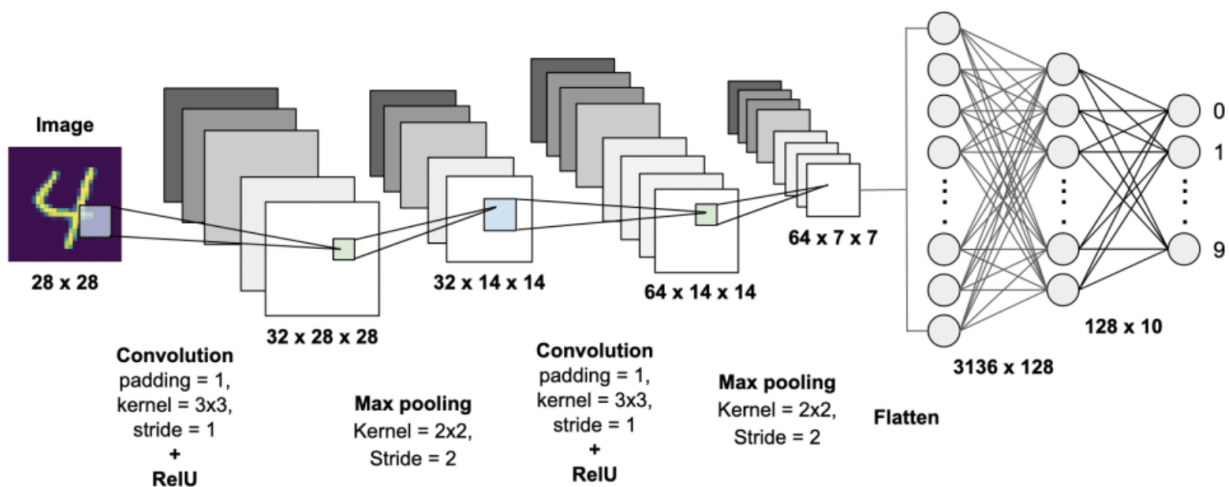
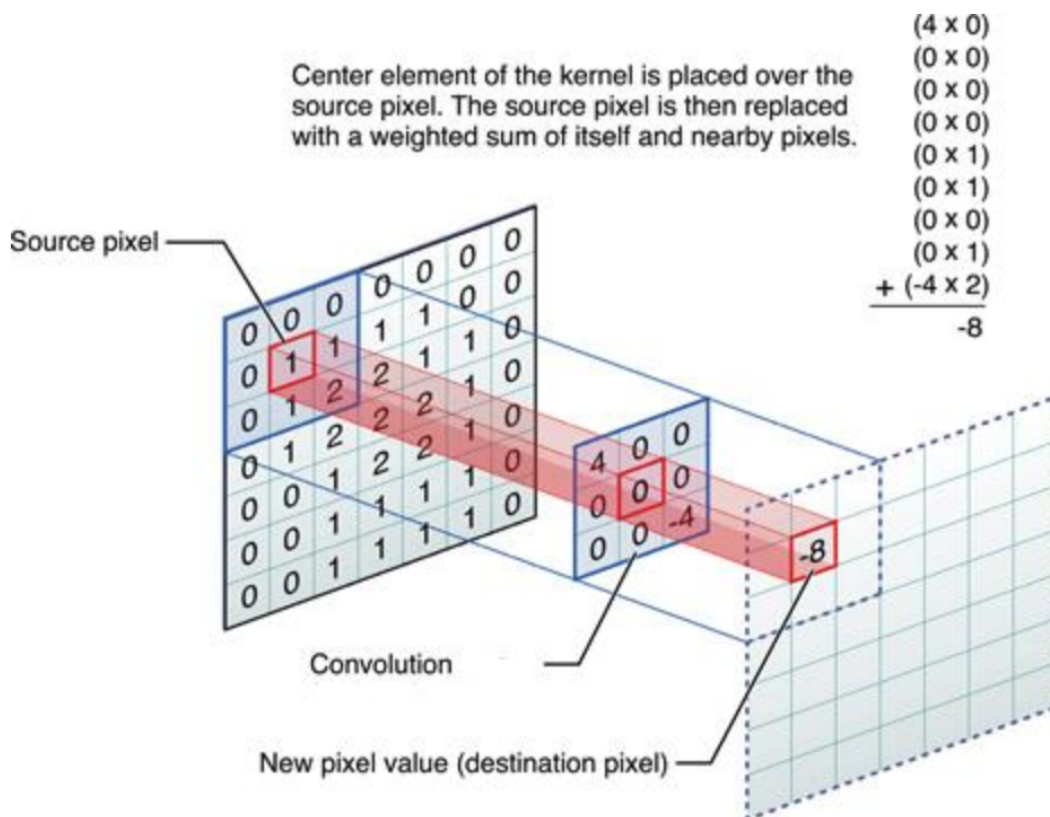
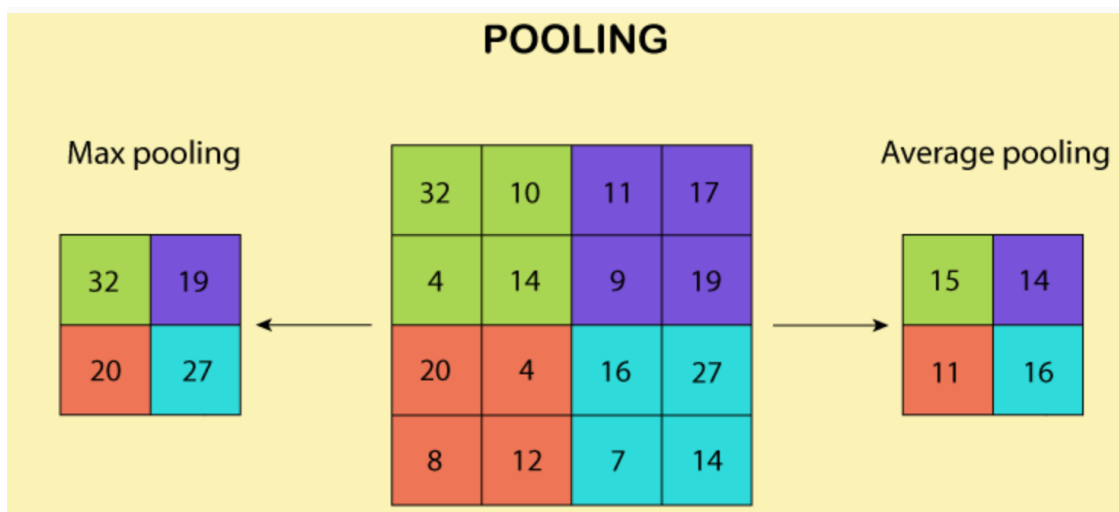


Figure 2.1.1: CNN Architecture. From [here](#)

As implied by their name, CNNs' key operation is the convolution. This is a specialized kind of linear operation, where a small filter or kernel slides across the input data (like an image) to produce a feature map, effectively transforming the data based on the filter's pattern. This operation helps the network focus on localized features. Convolution is visualised in Figure 2.1.2.

Figure 2.1.2: Convolution on a 7×7 image with a 3×3 kernel. From [here](#)

As shown in Figure 2.1.1 another significant component is the pooling layer. These layers are usually placed after the convolutional layers in order to aggregate the feature maps into lower dimensions. Hence, they do not only reduce the learning parameters and the computational efficiency of the model, but they are also critical to avoid overfitting. The most common pooling layers are the max-pooling and the average-pooling which can be summarized in Figure 2.1.3

Figure 2.1.3: The effect of max-pooling (left) and average-pooling (right) layers. From [here](#)

Another layer which is frequently used in Convolutional Neural Networks is the Batch Normalization [55]. This method leads to a faster and more stable training of CNNs, while it also prevents overfitting. Consider the values x of a mini-batch $\mathcal{B} = \{x_{1\dots m}\}$ and γ, β the learnable parameters. The proposed the normalization [55] is defined as:

$$y_i = \gamma \hat{x}_i + \beta \quad (2.1.1)$$

where,

$$\hat{x}_i = \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} \quad (2.1.2)$$

$$\mu_{\mathcal{B}} = \frac{1}{m} \sum_{i=1}^m x_i \quad (2.1.3)$$

$$\sigma_{\mathcal{B}}^2 = \frac{1}{m} \sum_{i=1}^m (x_i - \mu_{\mathcal{B}})^2 \quad (2.1.4)$$

and ϵ is a small constant used for numerical stability.

As can be seen from Figure 2.1.1 another layer that is commonly used in CNNs is Dropout [78]. Similar to pooling and batch normalization, this method is also used widely to prevent overfitting, due to its simplicity and effectiveness. More specifically, Dropout is a regularization technique which involves randomly "dropping out" or deactivating a subset of neurons in a layer at each iteration during training. By doing so, the network becomes less reliant on any single neuron, encouraging a more distributed and robust representation.

In essence, dropout can be seen as training a collection of "thinned" networks with shared weights. At test time, all neurons are used, but their outputs are typically scaled down by a factor equal to the dropout rate to compensate for the larger active network.

Dropout can be summarized in Figure 2.1.4

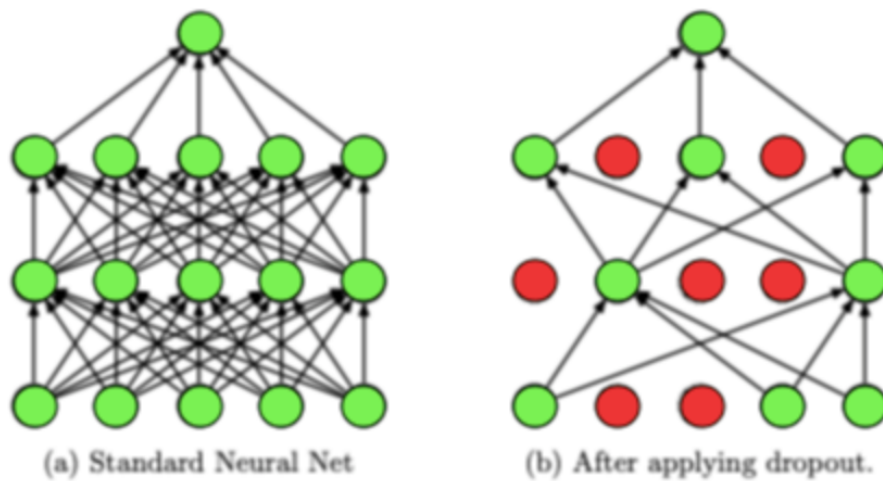


Figure 2.1.4: The effect of dropout in the training. From [here](#)

Convolutional Neural Networks (CNNs) have undergone a rapid evolution since their inception, with a variety of more specific architectures emerging, each tailored to address specific challenges or improve performance in visual tasks. The early days of CNNs were marked by simple architectures like LeNet [62], while soon other more complex and deep architectures such as AlexNet [79], VGGNet [54] and GoogLeNet (or Inception-V1) [80] arose to provide even better results in substantially harder datasets.

Two of the most important such architectures which will be discussed and used later for the scope of this thesis are the ResNet [77] and the U-net [60].

In the realm of deep learning, especially for tasks related to visual data, achieving deeper neural networks often promises better performance, capturing more intricate patterns from data. However, training very deep networks can be challenging due to problems like vanishing or exploding gradients. ResNet, or Residual Network, emerged as a solution to this depth dilemma.

The main idea behind ResNet is the introduction of residual connections that bypass one or more layers. Instead of aiming to learn the direct underlying mapping from inputs to outputs, ResNets try to learn the residual or difference between the two. By doing so, ResNets effectively ease the training process by letting the gradients flow and allow for the construction of much deeper networks.

A residual block is illustrated in Figure 2.1.6, while the entire architecture of ResNet-12 [77] is shown in Figure 2.1.5.

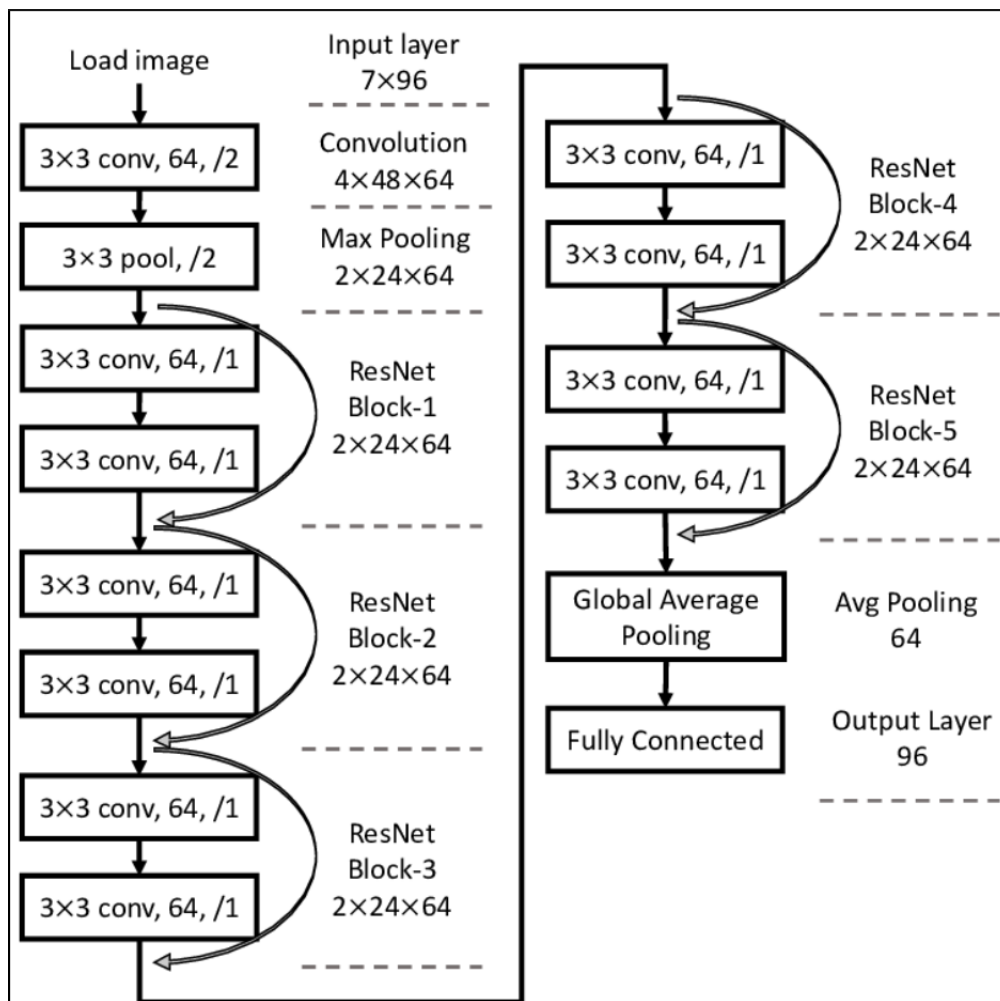


Figure 2.1.5: The Architecture of ResNet-12. From [here](#)

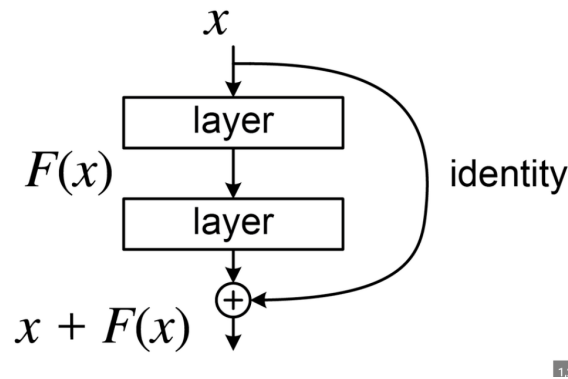


Figure 2.1.6: Residual Block. From [77]

On the other hand, U-Net [60] emerged initially as a very powerful network in the domain of medical image segmentation, before being used for a variety of tasks. U-Net's architecture, as shown in Figure 2.1.7, is distinctively shaped like a "U", consisting of a contracting (downsampling) path and an expansive (upsampling) path.

The contracting path captures context and reduces the spatial dimensions using convolutional and pooling layers, while the expansive path focuses on precise localization, using up-convolutions to upscale feature maps. What makes U-Net particularly effective is its use of skip connections between corresponding layers of the contracting and expansive paths. These connections ensure that detailed spatial information lost during downsampling is reintroduced during upsampling.

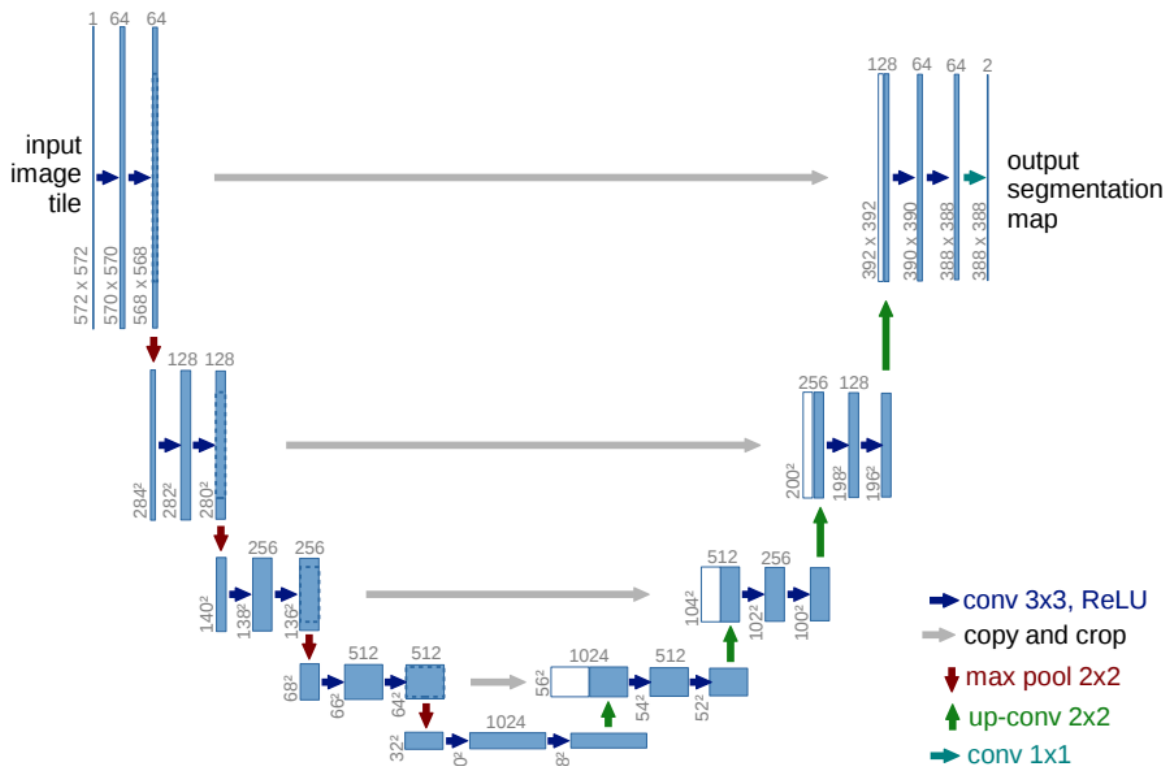


Figure 2.1.7: The Architecture of U-Net. From [60]

Recurrent Neural Networks

In the intricate domain of neural networks, Recurrent Neural Networks (RNNs) [1] stand out as the most suitable model for sequential data. Unlike traditional feedforward neural networks, which process inputs in isolation, RNNs possess the unique capability to maintain a memory of past inputs in their internal state. This characteristic makes them particularly suited for tasks where temporal dynamics and context from earlier steps are essential, such as time series forecasting, speech recognition, and natural language processing.

The core idea behind RNNs is the introduction of loops within the network, allowing information to persist. At each time step, an RNN takes in a new input along with its previous state (which contains information from prior time steps) to produce an output and update its state.

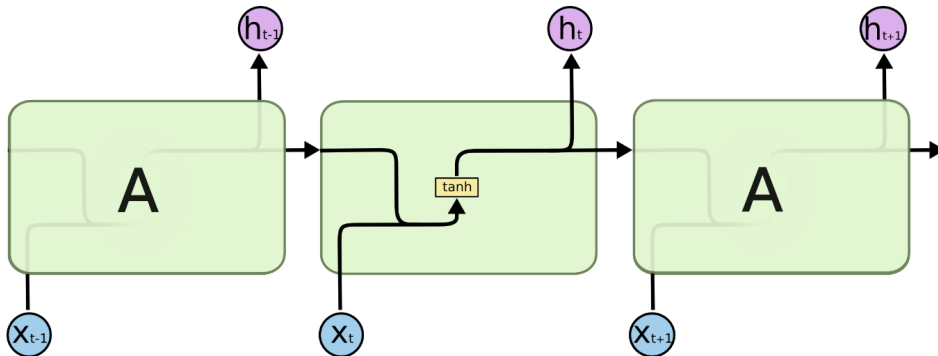


Figure 2.1.8: The Architecture of an unrolled RNN. From [here](#)

As shown in Figure 2.1.8, the output of an RNN at each timestep is defined as

$$h_t = \tanh(W h_{t-1} + U x_t + b) \tag{2.1.5}$$

where W, U are the learnable parameters and b the bias.

However, vanilla RNNs are not without flaws. They struggle with long-range dependencies [2] due to issues known as the vanishing and exploding gradient problems. As a result, more advanced RNN architectures have been introduced.

Long Short-Term Memory Networks (LSTMs) [3] are a clear example of such architectures, as they are designed to remember patterns over long durations. They achieve this through a system of gates (input, forget, and output gates) that regulate the flow of information, ensuring that the network retains only relevant context and discards unnecessary data. The overall architecture of LSTMs is depicted in Figure 2.1.9

The functionality of each gate is the following:

- **Forget Gate:** This gate determines which information is going to be kept in the cell state C_t and which is going to be thrown away (values close to 0 indicate that information will be thrown away, while values close to 1 indicate that information will be kept). The formulation of this gate is the following:

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f)$$

- **Input Gate:** This gate determines which new information will be updated and, combined with \tilde{C} (which uses a tanh layer to determine the values of the new information), they update the cell state. The input gate as well as the \tilde{C} are defined as:

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i)$$

$$\tilde{C}_i = \tanh(W_c[h_{t-1}, x_t] + b_c)$$

- **Cell State Update:** The update of the cell state is the following:

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

- **Output Gate:** The output gate decides which information will go to the output. This information will be a limited version of the cell state which will be determined by a sigmoid layer as shown in the equations below:

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o)$$

$$h_t = o_t \odot \tanh(C_t)$$

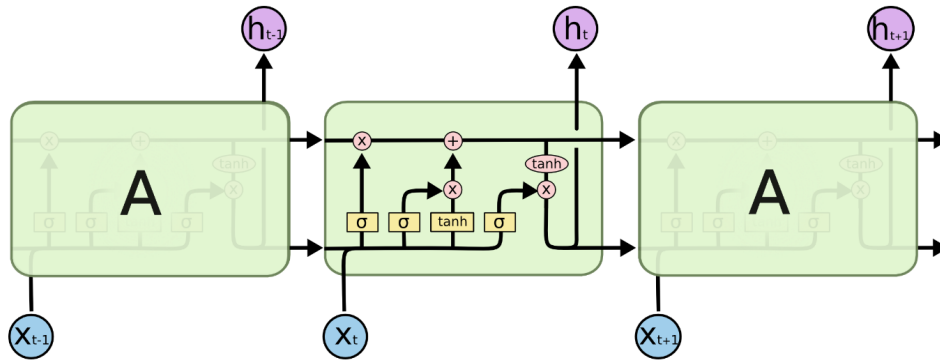


Figure 2.1.9: The Architecture of an LSTM. From [here](#)

Another widely used approach to address the vanishing and exploding gradients problem is the Gated Recurrent Unit (GRU) [4]. Similarly to LSTMs, GRUs use the reset and update gates to process the information flow. Their architecture is summarized in Figure 2.1.10

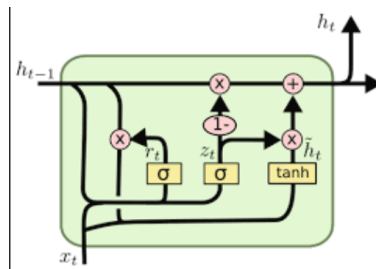


Figure 2.1.10: The Architecture of a GRU. From [here](#)

The final type of RNN which is going to be examined is the Bidirectional Recurrent Neural Network [81]. Bidirectional RNNs offer a significant enhancement by processing data from two directions: from the beginning to the end and from the end to the beginning. By doing this, they ensure that the information at any given time step is influenced by both past and future context, providing a more comprehensive view of the sequence.

This dual approach is achieved by stacking two separate RNNs. The first processes the sequence in the regular order (forward), while the second processes it in reverse (backward). The outputs of these two RNNs are typically concatenated at each time step, resulting in a combined representation that is then fed into subsequent layers or used for predictions.

The strength of Bidirectional RNNs shines especially in tasks where future context can inform current interpretations. This makes them exceptionally valuable in various applications, especially in natural language processing tasks like machine translation, named entity recognition, and speech recognition.

2.1.2 Training

Backpropagation

Neural networks, with their intricate architectures and vast number of parameters, possess the capability to approximate a wide range of functions. However, to transform a neural network from a random initializer into a powerful predictor, it's essential to adjust its weights appropriately. The driving force behind this adjustment is an optimization algorithm called backpropagation [82].

Backpropagation, short for "backward propagation of errors," is a supervised learning algorithm used for minimizing the error in neural network predictions. Conceptually, it can be understood as a two-step process:

- **Feedforward:** An input is passed through the network to produce an output. This output is then compared with the true label to compute an error or loss. For a specific layer of the network, the forward pass can be formulated as:

$$x_i = f(W_i \cdot x_{i-1} + b_i)$$

where x_i is the output of the i -th layer, W_i the trainable weights, b_i the bias and f the activation function as will be discussed in the following subsections.

- **Backward pass:** The gradient of this loss is computed with respect to each weight in the network by applying the chain rule of calculus. This gradient signifies how much each weight contributed to the error. The weights are then adjusted in the direction that decreases the error. Considering the loss function as L its gradient with respect to W_i in the above example is the following:

$$\frac{\partial L}{\partial W_i} = \frac{\partial L}{\partial x_i} \frac{\partial x_i}{\partial W_i}$$

Loss Functions

The loss function quantifies how well a prediction of a model aligns with the actual truth. It provides a numerical measure of the discrepancy between predicted values and actual values, serving as a compass guiding the optimization of model parameters. The lower the value of the loss function, the better the performance of the model. Three of the most common loss functions are the following:

- **L1 loss (MAE):** In the landscape of loss functions, the L1 loss, often referred to as the Mean Absolute Error (MAE), stands out for its straightforward interpretation and computational simplicity. It treats each deviation linearly, ensuring that large discrepancies don't disproportionately impact the model's learning, thus making it suitable for tasks where outliers are common and their influence should be limited. Given true values y_i and predicted values \hat{y}_i for $i = 1, 2, \dots, N$ where N is the number of samples:

$$L_1(y, \hat{y}) = \frac{1}{N} \sum_{i=1}^N |y_i - \hat{y}_i|$$

- **L2 loss (MSE):** Commonly used in regression tasks, L2 loss calculates the average of squared differences between predictions and actual values. This emphasizes larger errors over smaller ones, leading to a robust performance metric. Given true values y_i and predicted values \hat{y}_i for $i = 1, 2, \dots, N$ where N is the number of samples:

$$L_2(y, \hat{y}) = \frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)^2$$

- **Cross Entropy Loss:** Predominantly used in classification tasks, this function measures the dissimilarity between the predicted probability distribution and the true distribution. It's particularly suitable when modeling probabilistic outputs, like in logistic regression or deep neural networks for classification. It is defined as:

$$H(y, p) = -\frac{1}{N} \sum_{i=1}^N \sum_{c=1}^C y_{i,c} \log(p_{i,c})$$

where $y_{i,c}$ is a binary indicator for the class (usually one-hot encoding) and $p_{i,c}$ the predicted probability that observation i belongs to class c .

Optimization

At its core, the goal of an optimizer is to find the optimal set of parameters that results in the lowest possible loss for a given model on a specific task. To achieve this, optimizers utilize gradients (computed by the backpropagation algorithm as described above), which are essentially the directions and magnitudes of changes in parameters that lead to the steepest decrease in the loss. The most fundamental optimizer is the Gradient Descent. It adjusts model parameters iteratively in the direction of the negative gradient. However, vanilla gradient descent can be slow and might not always find the best possible solution. Two of the most common optimizers which will be used in this thesis are Adam [73] and AdamW [74].

- **Adam:** Adam (Adaptive Moment Estimation) is a method for efficient stochastic optimization which combines the benefits of AdaGrad [83] and RMSProp [84]. Adam operates by computing adaptive learning rates for each parameter, offering faster convergence in many tasks. This is achieved by maintaining an exponentially decaying average of past gradients and the square of these gradients. The update rule of Adam is summarized in the following steps:

1. $g_t \leftarrow \nabla_{\theta} f_t(\theta_{t-1})$
2. $m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$
3. $v_t \leftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$
4. $m_{b_t} \leftarrow \frac{m_t}{1 - \beta_1^t}$
5. $v_{b_t} \leftarrow \frac{v_t}{1 - \beta_2^t}$
6. $\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \frac{m_{b_t}}{\sqrt{v_{b_t} + \epsilon}}$

- **AdamW:** AdamW was introduced to address the limitation of Adam regarding its interaction with weight decay regularization. This is rectified by decoupling the weight decay from the optimization steps, making it more compatible with the adaptive nature of Adam. In terms of mathematical formulation the only difference from the Adam optimizer can be seen in the last step of the update rule which becomes:

$$\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \left(\frac{m_{b_t}}{\sqrt{v_{b_t} + \epsilon}} + \lambda \theta_{t-1} \right)$$

Some typical values for the above hyperparameters are the following: $\alpha = 0.001, \beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 10^{-8}, \lambda \in \mathbb{R}$

Activation Functions

Activation functions introduce non-linearity into the network, which allows it to model and learn complex, non-linear relationships between inputs and outputs. Without them, no matter how deep or wide the network, it would behave merely as a linear regressor, drastically limiting its capacity to approximate intricate functions. Some popular activation functions which are going to be used in this thesis are the following:

- **Sigmoid:** Defined in the range between 0 and 1, the sigmoid function was historically popular due to its clear interpretation as a probability. However, it has since waned in usage for deep networks due to issues like vanishing gradients. It is defined as:

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

- **Softmax:** As a generalization of the sigmoid activation, Softmax is often used in the output layer of classification tasks, in order to transform a vector into a probability distribution over multiple classes.

$$\text{softmax}(x_i) = \frac{e^{x_i}}{\sum_{j=1}^N e^{x_j}}$$

where x_i is the i th component of the vector x and N is the dimensionality of x .

- **ReLU:** A simple yet powerful function that outputs the input if positive, otherwise zero. Due to its efficiency and effectiveness, it has become the default choice for many deep learning tasks.

$$\text{ReLU}(x) = \max(0, x)$$

- **SiLU:** Often referred to as the Swish function, SiLU [85] adaptively gates the activations, providing a smooth, bounded, and non-monotonic curve, while experiments have shown that SiLU often outperforms the more traditional ReLU, especially in deeper architectures.

$$\text{SiLU}(x) = x \cdot \sigma(x)$$

where σ is the sigmoid function.

- **GELU:** The GELU [86] function finds its inspiration in the Gaussian distribution. Its mathematical form closely resembles the cumulative distribution function (CDF) of a Gaussian. This smooth activation function has shown promising results, especially in transformer architectures, where it has been favored for its ability to handle the training dynamics of such models.

$$\text{GELU}(x) = 0.5x \left(1 + \tanh \left(\sqrt{\frac{2}{\pi}} (x + 0.044715x^3) \right) \right)$$

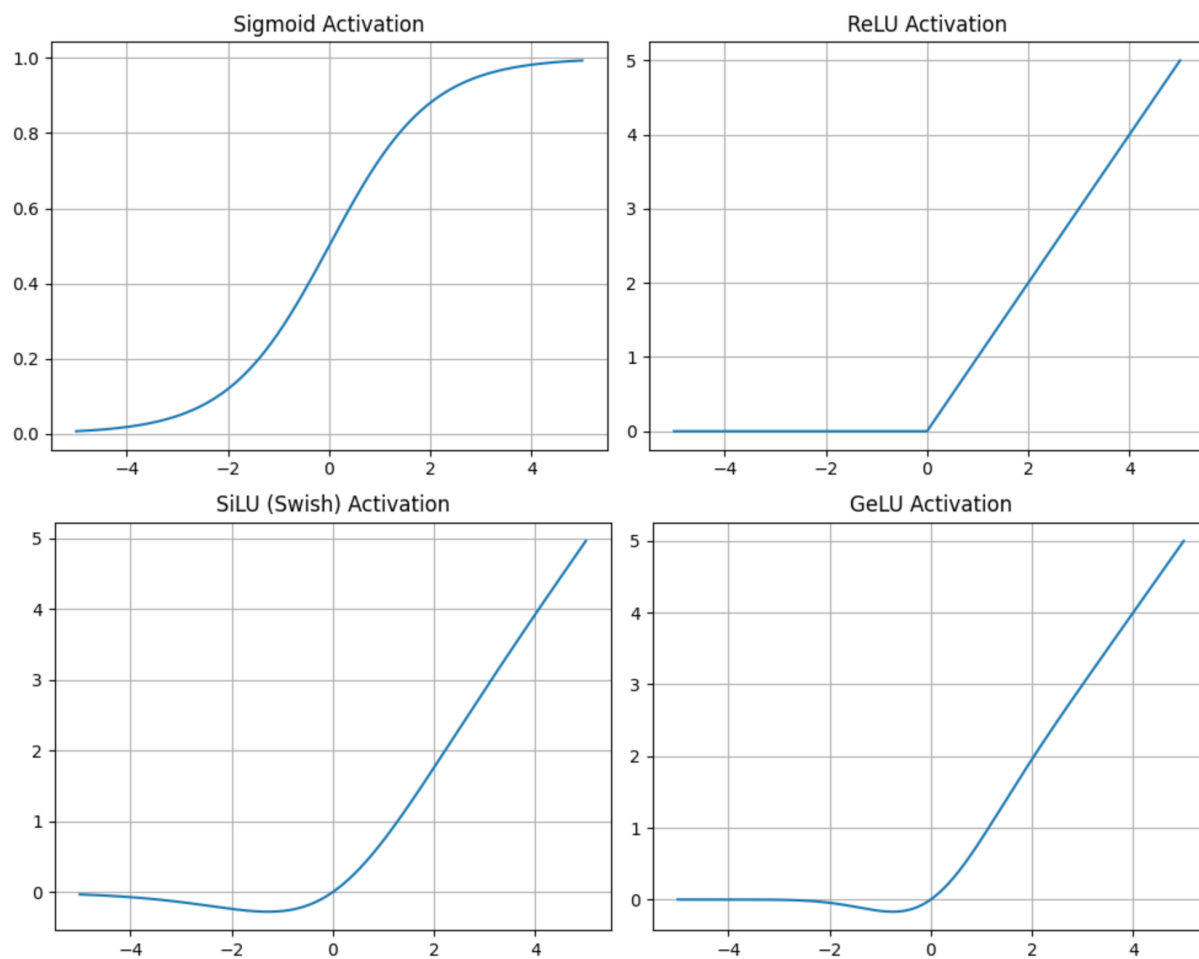


Figure 2.1.11: The Sigmoid, ReLU, SiLU and GELU activation functions.

2.1.3 Introduction to Generative Models

Introduction to Variational Autoencoders

Variational Autoencoders, commonly known as VAEs, have emerged as one of the most popular and powerful tools in the world of generative modeling. Introduced in 2013 [5], they bridge the gap between two key approaches in machine learning: deep learning and Bayesian inference.

At its core, a VAE is an autoencoder. Autoencoders are neural networks designed to reconstruct their input. They do this by compressing the input into a compact, latent representation through an encoder, and then expanding this latent representation back into the original data space through a decoder. However, VAEs add a probabilistic twist to this process. Instead of encoding an input as a single point in the latent space, VAEs encode it as a distribution. This inherent randomness allows VAEs to generate new, similar data points by sampling from this distribution. In total, the architecture of VAEs involves an encoder and a decoder:

- **Encoder:** The encoder's role is to take an input data point and produce parameters (typically means and variances) of a proposed latent space probability distribution. In essence, it encodes the data into a set of parameters from which we can sample latent representations. The effect of the encoder can also be formulated as the extraction of a posterior distribution $q(z|x)$, where x is the initial distribution and z the latent representation, respectively.
- **Decoder:** The decoder's role is to reconstruct the original input data from this latent representation. Essentially, it decodes the latent space back to the original data space. Likewise, the role of the decoder can be interpreted as a conditional distribution $q(x|z)$ which produces a distribution with respect to the latent samples z .

Additionally, the objective of VAEs consists of two losses, one that indicates the quality of image reconstruction and another one which measures the similarity of the latent space distribution with a prior one:

- **Reconstruction Loss:** As mentioned above, this component estimates the quality of the reconstructed image compared with the original one, similar to traditional autoencoders. Cross entropy or MSE are some of the most common functions used for this purpose.
- **Kullback-Leibler Divergence:** This term ensures that the latent space distributions are kept close to a prior, usually a standard normal distribution. It acts as a regularizer, promoting smoothness and continuity in the latent space. KL divergence is defined as:

$$D_{KL}(P||Q) = \sum_x P(x) \log \frac{P(x)}{Q(x)}$$

where P is the true probability distribution and Q the model probability distribution.

Although the above loss combination constitutes a powerful tool for image generation from random noise, the balance between these two components can sometimes be tricky, as there is a risk of the model focusing too much on either term, leading to issues like over-regularization or poor reconstruction [87].

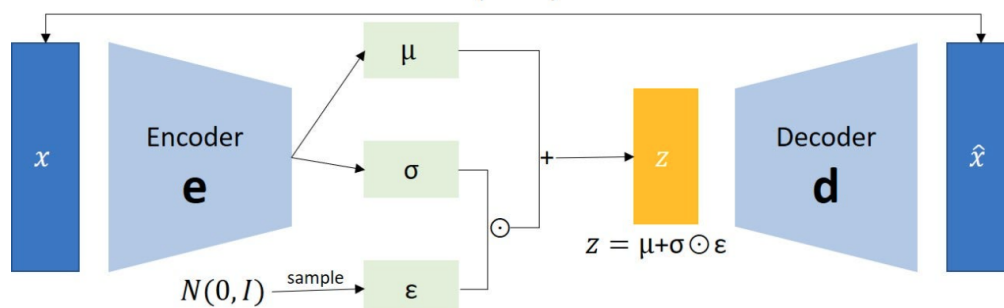


Figure 2.1.12: Variational Autoencoder. From [here](#)

Introduction to Generative Adversarial Networks

Generative Adversarial Networks (GANs) were introduced in 2014 [6] and have since garnered attention not just for their mathematical elegance, but for their practical ability to generate realistic outputs, varying from images to sounds.

The main idea behind GANs is to train two networks, namely, generator and discriminator through a zero-sum game. More specifically, the aim of the generator is to generate realistic samples, the distribution of which is close to the respective of the real data, whereas the goal of the discriminator is to be able to distinguish real images from synthetic (generated by the generator). The total architecture of a GAN is depicted in Figure 2.1.13.

Therefore, the training can be mathematically formulated as a minimax game, where the generator tries to maximize the chances of fooling the discriminator, while the latter intends to minimize this chance, as shown in the equation below:

$$V(D, G) = \mathbb{E}_{x \sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z)))] \tag{2.1.6}$$

where the first term ($\mathbb{E}_{x \sim p_{data}(x)}[\log D(x)]$) represents the expected value over real samples x (it maximizes when the discriminator correctly identifies real samples) and the second term ($\mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z)))]$) is the expected value over noise vectors z (it maximizes when the discriminator correctly identifies fake samples).

The ideal state of GAN convergence is achieved when the generator produces samples that come from the same distribution as the real data, rendering the discriminator’s task equivalent to a coin toss, i.e., assigning a probability of 0.5 to each sample (real or fake).

Formally, this state corresponds to the Nash equilibrium of the minimax game, where neither player (Generator or Discriminator) has an incentive to change their strategy given that the other player’s strategy remains fixed. The Global Nash equilibrium for the GAN game is achieved when the generated distribution, p_g , matches the real data distribution, p_{data} . However, achieving this equilibrium in practice is non-trivial, and GANs are notorious for the challenges encountered during training. Some of the most common challenges are the following:

- Mode Collapse
- Vanishing Gradients
- Oscillations
- Sensitivity to Hyperparameters
- Overpowering Discriminator

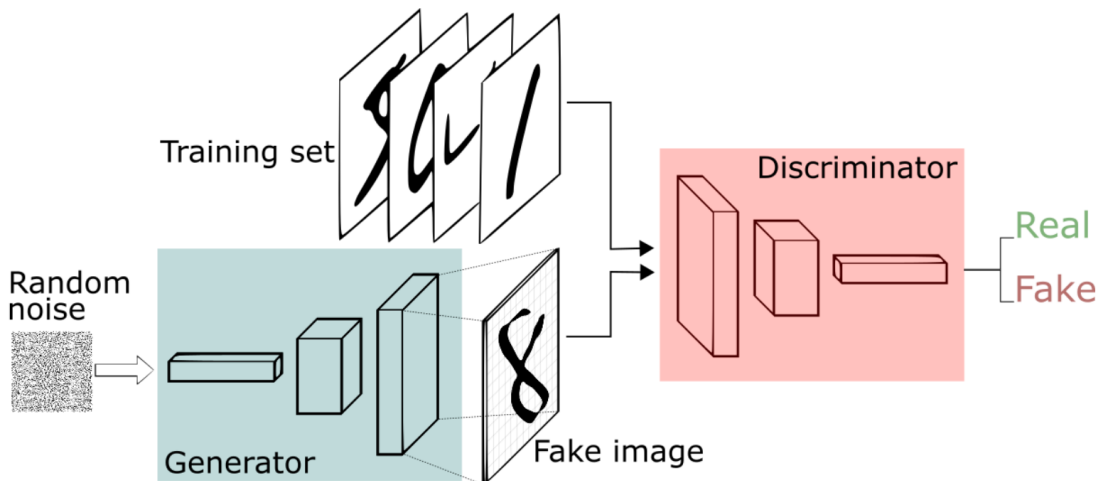


Figure 2.1.13: Generative Adversarial Networks. From [here](#)

2.2 Introduction to Morphological Mathematics

Morphological Mathematics, also known simply as mathematical morphology, is a framework within the fields of image analysis and signal processing that focuses on the shape-based manipulation of structures within data. Developed initially for binary image analysis, it has since been extended to more complex data forms such as gray-scale images, 3D data, and even non-spatial signals.

The theory of morphological mathematics was introduced by Matheron [7] and Serra [8] and it was based on set theory, lattice theory, and topology. The main operations, namely, dilation, erosion, opening and closing, were inspired by Minkowski set operations. In the past few years, morphological mathematics have been applied to a variety of computer vision tasks such as image analysis [9, 10], classification [11, 12], filtering [13], segmentation [14, 15], edge detection [16], etc.

Morphological mathematics, due to their non-linear properties, are able to capture features which cannot be preserved by other linear approaches. Figure 2.2.1 clearly illustrates the need of such operations.

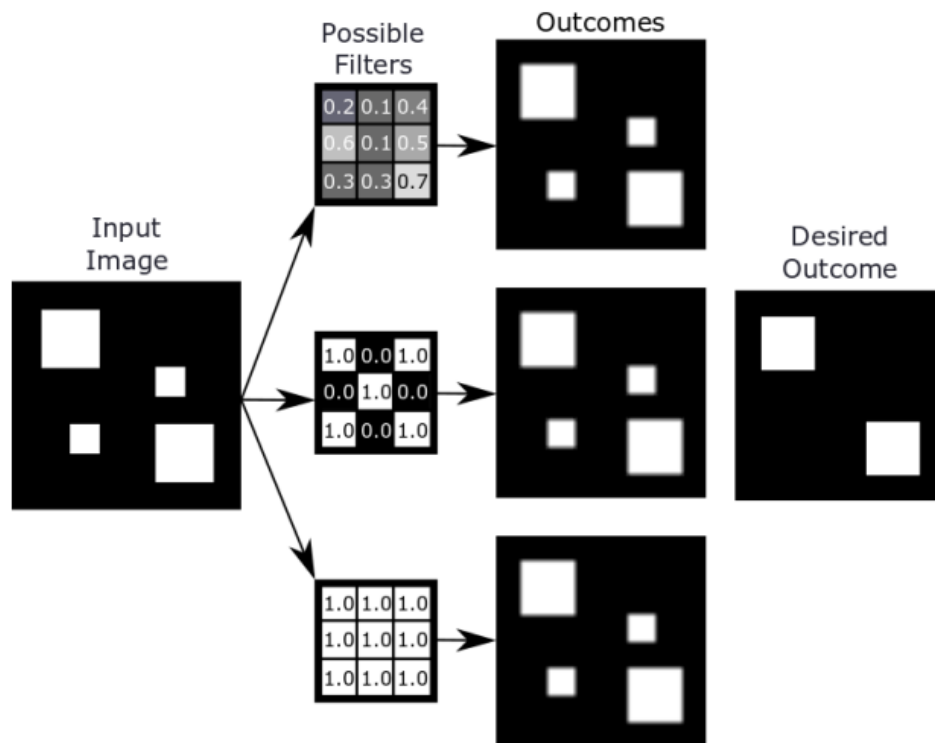
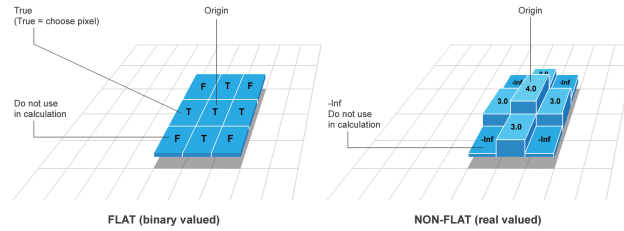


Figure 2.2.1: An example of how morphological operations are capable of generating outcomes that convolution-based filtering is not. From [88]

2.2.1 Structuring Elements

Structuring Elements (SE) are essential for every morphological operation and can be defined as a function to determine how an image is processed (similar to kernels in convolutions). Depending on the values that they take, SEs can be divided into two categories:

- **Flat Structuring Elements:** Flat SEs are binary and they indicate which pixel values should be included in order to determine the final value of the processed pixel.
- **Non-Flat Structuring Elements:** The values Non-Flat SEs are usually float numbers and they indicate not only which pixels considered for the final result, but also the extent to which each pixel in the neighborhood should be taken into account.


 Figure 2.2.2: Flat and Non-Flat Structuring Elements. From [here](#)

2.2.2 Morphological Operations

The morphological operations which will be analyzed below are the main operations, dilation, erosion, opening and closing, as well as top-hat transformations and geodesic reconstruction. Additionally, Table 2.1 provides the main properties (for an operator ψ), which will be matched to each of the morphological operations. Let us consider an image A and a structuring element B , the morphological operations can be defined as demonstrated below.

Dilation

Dilation is a core operation in the field of morphological mathematics and its qualitative effect is that it brightens the darker areas contained in the neighborhood that is specified by the selected SE. It is an increasing and extensive operation which is defined as:

$$\delta(A, B) = A \oplus B(x) = \sup_{y \in \mathbb{R}^n} \{A(x - y) + B(y)\} \quad (2.2.1)$$

Erosion

On the contrary, erosion, as the dual operation of dilation, darkens the bright areas of each neighborhood. It is an increasing and anti-extensive operation and it is formulated as follows:

$$\epsilon(A, B) = A \ominus B(x) = \inf_{y \in \mathbb{R}^n} \{A(x + y) - B(y)\} \quad (2.2.2)$$

Opening

Opening operation is the result of performing a dilation right after an erosion for an image with same SE. It is an increasing, idempotent and anti-extensive operation defined as:

$$\gamma(A, B) = A \circ B = (A \ominus B) \oplus B \quad (2.2.3)$$

Closing

On the other hand, performing an erosion right after a dilation results in the closing operator, an increasing, idempotent and extensive operation which is the following:

$$\phi(A, B) = A \bullet B = (A \oplus B) \ominus B \quad (2.2.4)$$

Top-Hat Transformations

Top-Hat transformations constitute another useful set morphological operations and they are separated into:

- White Top-Hat Transformation:

$$\mathcal{T}^w(A, B) = A - A \circ B \quad (2.2.5)$$

- Black Top-Hat Transformation:

$$\mathcal{T}^b(A, B) = A \bullet B - A \quad (2.2.6)$$

Geodesic Reconstruction

The final morphological operation which is going to be examined is the Geodesic Reconstruction. This transformation is also divided into two types, the geodesic reconstruction by erosion and by dilation. In the first case, the input is an image A and a SE B and by this operation we intend to reconstruct the A from $A \oplus B$ with consecutive erosions as shown in Equation 2.2.7. Apparently, geodesic reconstruction by dilation is the exact opposite and is defined in Equation 2.2.8.

These two equations are the following

- Geodesic Reconstruction by Erosion:

$$\rho^\epsilon = \mathcal{R}^\epsilon(B', \delta(A, B)) \quad (2.2.7)$$

- Geodesic Reconstruction by Dilation:

$$\rho^\delta = \mathcal{R}^\delta(B', \epsilon(A, B)) \quad (2.2.8)$$

Table 2.1: Properties of Morphological Operations

Property	Definition
Increasing	$A \leq B \Rightarrow \psi(A) \leq \psi(B)$
Extensive	$\psi \geq A$
Anti-Extensive	$\psi \leq A$
Idempotent	$\psi^2 = \psi$

The effect of the above morphological operations is illustrated in Figure 2.2.3.

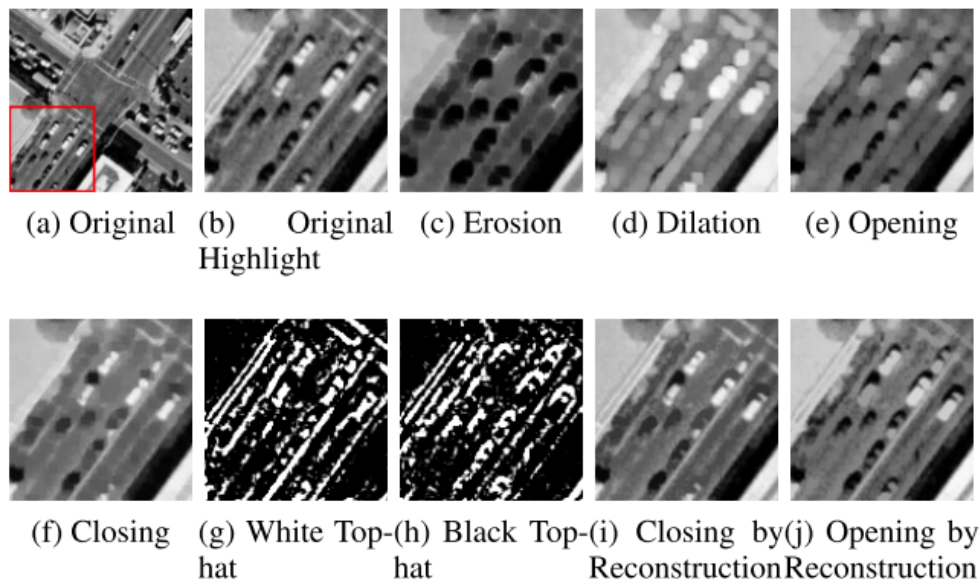


Figure 2.2.3: Morphological Operations with a 5×5 square SE. From [88]

2.3 Introduction to Diffusion Models

Diffusion models, part of the broad class of generative models, have gained substantial attention for their impressive capabilities in generating high-quality, realistic samples. The theoretical foundation of diffusion models is deeply rooted in the principles of nonequilibrium thermodynamics and stochastic processes [18].

Training a diffusion model equates to learning to invert this forward diffusion process. From the observed data distribution, noise is slowly added until the data transforms into a simple noise distribution. The model's aim is to learn the reverse operation - to denoise the data and thereby recover the original distribution. This principle draws parallels with denoising autoencoders, which also focus on eliminating noise from the data. However, the key distinction arises from the fact that diffusion models gradually introduce noise across several steps, whereas denoising autoencoders incorporate noise all at once [19].

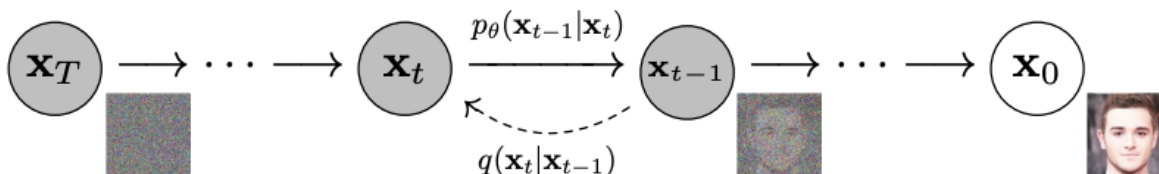


Figure 2.3.1: The directed graphical model. From [20]

2.3.1 Background

Assume we have a data distribution denoted as \mathbf{x}_0 , following the distribution $q(\mathbf{x}_0)$. We establish a forward noise addition process, symbolized as q , that results in latent variables \mathbf{x}_1 to \mathbf{x}_T . This process adds Gaussian noise at each time step t with variance β_t falling within the range $(0, 1)$. The equation is expressed as:

$$q(x_1, \dots, x_T | x_0) := \prod_{t=1}^T q(x_t | x_{t-1}) \quad (2.3.1)$$

where

$$q(x_t | x_{t-1}) := \mathcal{N}\left(x_t; \sqrt{1 - \beta_t} x_{t-1}, \beta_t \mathbf{I}\right) \quad (2.3.2)$$

For a sufficiently large T and an appropriately managed schedule of β_t , the resulting latent x_T is approximately an isotropic Gaussian distribution. Hence, if we can identify the accurate reverse distribution $q(x_{t-1} | x_t)$, we can sample x_T from a standard Gaussian distribution $\mathcal{N}(0, \mathbf{I})$ and execute the process in reverse to get a sample from $q(x_0)$.

As $q(x_{t-1} | x_t)$ depends on the entire data distribution, we approximate it using a neural network as follows:

$$p_\theta(x_{t-1} | x_t) = \mathcal{N}\left(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t)\right) \quad (2.3.3)$$

As established by Ho, Jain and Abbeel, in 2020, [20] the noising process articulated in Equation (2) provides the capability to sample any step from the noised latent variables directly conditioned on the input x_0 . With $\alpha_t := 1 - \beta_t$ and $\bar{\alpha}_t := \prod_{s=0}^t \alpha_s$, the marginal $q(x_t | x_0)$ can be written as:

$$q(x_t | x_0) = \mathcal{N}\left(x_t; \sqrt{\bar{\alpha}_t} x_0, (1 - \bar{\alpha}_t) \mathbf{I}\right) \quad (2.3.4)$$

and

$$x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \bar{\alpha}_t} \epsilon \quad (2.3.5)$$

where $\epsilon \sim \mathcal{N}(0, \mathbf{I})$. Here, $1 - \bar{\alpha}_t$ denotes the variance of the noise at any arbitrary time step, offering an alternative definition for the noise schedule instead of β_t .

By applying Bayes' theorem, the posterior $q(x_{t-1}|x_t, x_0)$ can be computed in terms of $\tilde{\beta}_t$ and $\tilde{\mu}_t(x_t, x_0)$ as follows:

$$\tilde{\beta}_t = \frac{1 - \bar{\alpha}_{t-1}}{1 - \bar{\alpha}_t} \beta_t \quad (2.3.6)$$

$$\tilde{\mu}_t(x_t, x_0) = \frac{\sqrt{\bar{\alpha}_{t-1}\beta_t}}{1 - \bar{\alpha}_t} x_0 + \frac{\sqrt{\alpha_t(1 - \bar{\alpha}_{t-1})}}{1 - \bar{\alpha}_t} x_t \quad (2.3.7)$$

yielding

$$q(x_{t-1}|x_t, x_0) = \mathcal{N}\left(x_{t-1}; \tilde{\mu}(x_t, x_0), \tilde{\beta}_t \mathbf{I}\right) \quad (2.3.8)$$

Since the combination of q and p is a variational auto-encoder, we can train the model such that $p(\mathbf{x}_0)$ learns the true data distribution $q(\mathbf{x}_0)$, by optimizing the following variational lower bound:

$$L_{\text{vlb}} = L_0 + L_1 + \dots + L_{T-1} + L_T \quad (2.3.9)$$

$$L_0 = -\log p_\theta(x_0|x_1) \quad (2.3.10)$$

$$L_{t-1} = D_{KL}(q(x_{t-1}|x_t, x_0)||p_\theta(x_{t-1}|x_t)) \quad (2.3.11)$$

$$L_T = D_{KL}(q(x_T|x_0)||p(x_T)) \quad (2.3.12)$$

2.3.2 Training-Sampling

The objective given in Equation 2.3.9 is a sum of independent components, denoted by L_{t-1} . An efficient way of sampling from any step of the forward noising process and estimating L_{t-1} is provided by Equation 2.3.5, using the posterior (Equation 2.3.8) and prior (Equation 2.3.3). Consequently, a random sample of t can be taken and the expectation $E_{t, x_0, \epsilon}[L_{t-1}]$ can be used to approximate L_{vlb} . According to Ho, Jain and Abbeel (2020) [20], an effective strategy is to uniformly sample t for each image in a mini-batch.

Despite the sound reasoning behind the aforementioned objective, Ho, Jain and Abbeel [20] discovered that a distinct objective yielded superior practical samples. Specifically, they didn't directly parameterize $\mu_\theta(x_t, t)$ using a neural network. Instead, they trained a model $\epsilon_\theta(x_t, t)$ to predict ϵ from Equation 2.3.5. This leads to the definition of a simplified objective as follows:

$$L_{\text{simple}} = E_{t, x_0, \epsilon}[|\epsilon - \epsilon_\theta(x_t, t)|^2] \quad (2.3.13)$$

L_{simple} , the reweighted form of L_{vlb} (excluding the terms that impact Σ_θ), led to better sample quality when optimized instead of directly optimizing L_{vlb} . This surprising result is explained by drawing parallels to generative score matching. One point of nuance is that L_{simple} provides no learning signal for $\Sigma_\theta(x_t, t)$, since Ho, Jain and Abbeel (2020) [20] obtained their best results by setting the variance to a fixed value of $\sigma_t^2 \mathbf{I}$ rather than learning it. They found comparable sample quality using either $\sigma_t^2 = \beta_t$ or $\sigma_t^2 = \tilde{\beta}_t$, which represent the upper and lower bounds of the variance given that $q(x_0)$ can be either isotropic Gaussian noise or a delta function.

However, Nichol and Dwariwal [21] argue that the above method does not perform optimally when sampling with fewer diffusion steps. They suggest an alternative approach, which is to parameterize $\Sigma_\theta(x_t, t)$ using a neural network. The output of this neural network, denoted as v , is then interpolated as follows:

$$\Sigma_\theta(x_t, t) = \exp(v \log \beta_t + (1 - v) \log \tilde{\beta}_t) \quad (2.3.14)$$

Additionally, Nichol and Dhariwal [21] then put forward a hybrid training objective that optimizes both $\varepsilon_\theta(x_t, t)$ and $\Sigma_\theta(x_t, t)$. They use a weighted sum:

$$L_{\text{hybrid}} = L_{\text{simple}} + \lambda L_{\text{vib}} \quad (2.3.15)$$

to achieve this. By learning the reverse process variances using their proposed hybrid objective, they found it possible to perform sampling with fewer steps, without a significant reduction in the quality of the samples. For their experiments, λ was set to the value of 0.001, in order to prevent L_{vib} from overwhelming L_{simple} .

Algorithm 8: Training. From [20]

```

1: repeat
2:    $x_0 \sim q(x_0)$ 
3:    $t \sim \text{Uniform}(\{1, \dots, T\})$ 
4:    $\varepsilon \sim \mathcal{N}(0, I)$ 
5:   Take gradient descent step on
      $\nabla_\theta \|\varepsilon - \varepsilon_\theta(\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1 - \bar{\alpha}_t}\varepsilon, t)\|^2$ 
6: until converged

```

Algorithm 9: Sampling. From [20]

```

1:  $x_T \sim \mathcal{N}(0, I)$ 
2: for  $t = T$  to 1 do
3:    $z \sim \mathcal{N}(0, I)$  if  $t > 1$ , else  $z = 0$ 
4:    $x_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left( x_t - \sqrt{\frac{1 - \alpha_t}{1 - \bar{\alpha}_t}} \varepsilon_\theta(x_t, t) \right) + \sigma_t z$ 
5: end for
6: return  $x_0$ 

```

2.3.3 Improvements

Cosine schedule

Nichol and Dhariwal [21] proposed cosine schedule, as they found that, although the linear noise schedule proposed by Ho, Jain and Abbeel (2020) [20] was effective for high-resolution images, it underperformed with images of lower resolution, specifically 64 x 64 and 32 x 32. One of the reasons is that the later stages of the forward noising process are excessively noisy, which marginally contributes to the quality of the samples, as can be observed in Figure 2.3.3.

To rectify this, a new noise schedule in terms of the cumulative noise scale $\bar{\alpha}_t$ was proposed:

$$\bar{\alpha}_t = \frac{f(t)}{f(0)}, \quad f(t) = \cos \left(\frac{t/T + s}{1 + s} \cdot \frac{\pi}{2} \right)^2 \quad (2.3.16)$$

The variances β_t can then be determined from this definition as:

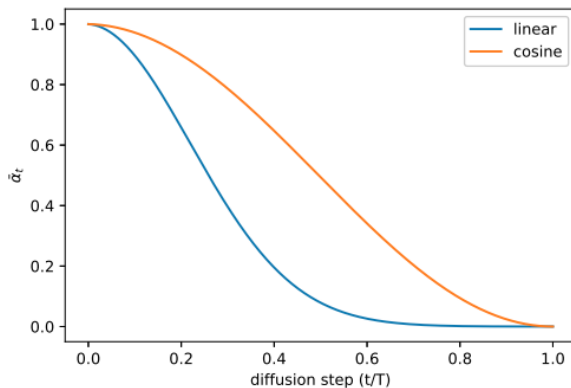
$$\beta_t = 1 - \frac{\bar{\alpha}_t}{\bar{\alpha}_{t-1}} \quad (2.3.17)$$

The cosine schedule is designed to maintain a consistent linear decrease of $\bar{\alpha}_t$ during the midpoint of the process, while avoiding drastic changes near the extreme points $t = 0$ and $t = T$ to evade abrupt shifts in noise levels. Figure 2.3.4 illustrates the progression of $\bar{\alpha}_t$ for both schedules. It becomes apparent that the linear schedule diminishes towards zero much faster, resulting in the rapid loss of information.

Furthermore, a small offset is introduced, denoted by s , to keep β_t from being excessively small close to $t = 0$ as we observed that having minuscule noise levels at the process's inception complicates the network's prediction of ε . More specifically, s was selected such that $\sqrt{\beta_0}$ was slightly smaller than the pixel bin size $1/127.5$, which provides $s = 0.008$.



Figure 2.3.2: Sampled from linear (top) and cosine (bottom) schedules. From [21]

Figure 2.3.3: $\bar{\alpha}_t$ throughout diffusion in the linear and cosine schedules. From [21]

Adaptive Group Normalization

Nichol and Dwariwal [59] also examine the effect of implementing a layer known as Adaptive Group Normalization (AdaGN). This layer brings together the timestep and class embeddings into every residual block, following the execution of a group normalization operation [89]. This approach echoes the tactics employed in Adaptive Instance Normalization [43] and FiLM [90]. The layer is structured as $\text{AdaGN}(h, y) = y_s \text{GroupNorm}(h) + y_b$, where h refers to the intermediate activations that follow the first convolution in the residual block, while $y = [y_s, y_b]$ is derived from a linear projection of the timestep and class embeddings. The use of AdaGN results in the following FID [70] improvement:

Operation	FID
AdaGN	13.06
Addition + GroupNorm	15.08

Figure 2.3.4: FID improvement by replacing Addition + GroupNorm [20] with AdaGN [59]. From [59]

2.3.4 Score-based generative modeling with stochastic differential equations

A different perspective on Denoising Diffusion Probabilistic Models (DDPMs) was offered in [44]. It is suggested viewing the diffusion process as a solution to a specific stochastic differential equation, which is represented as:

$$dx = (\sqrt{1 - \beta(t)} - 1)x(t)dt + \beta(t)dw \quad (2.3.18)$$

In this equation, the rate of change of the system (dx) is determined by a function of the system's current

state $(x(t))$, and a noise term $(\beta(t)dw)$, where $\beta(t)$ signifies the noise level at time t and dw indicates a random, small variation in the noise.

By beginning with samples from the distribution $p(T)$ denoted by $x_p(T)$, we can reverse the process to obtain samples $x_p(0)$ from the distribution $p(0)$. Anderson (1982) [91] presented an insightful result stating that reversing a diffusion process simply results in another diffusion process, albeit one that operates backwards in time, and can be described by the reverse-time Stochastic Differential Equation (SDE):

$$dx = \left((\sqrt{1 - \beta(t)} - 1)x(t) - \beta(t)\epsilon_\theta(x(t), t) \right) dt + \sqrt{\beta(t)}d\bar{w} \tag{2.3.19}$$

This refers to Variance Preserving Stochastic Differential Equations (VP-SDEs). If we alter the domain of t from $[1, N]$ to $[0, 1]$, as N approaches infinity, the sequences $\{\beta_i\}_{i=1}^N$ and $\{x_i\}_{i=1}^N$ become continuous functions $\beta(t)$ and $x(t)$ over the interval $[0, 1]$. This work [44] further demonstrates that this equation corresponds to an Ordinary Differential Equation (ODE) variant with the same marginal probability density as that of Equation 2.3.19:

$$dx = \left((\sqrt{1 - \beta(t)} - 1)x(t) - \frac{1}{2}\beta(t)\epsilon_\theta(x(t), t) \right) dt \tag{2.3.20}$$

The equation 2.3.20 is named as the probability flow ODE in [44]. When the score function is estimated by a model based on time, often a neural network, it exemplifies a neural ordinary differential equation, as discussed in [92].

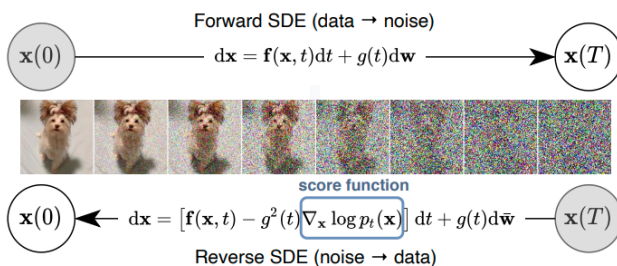


Figure 2.3.5: Solving a reverse-time SDE yields a score-based generative model. From [44]

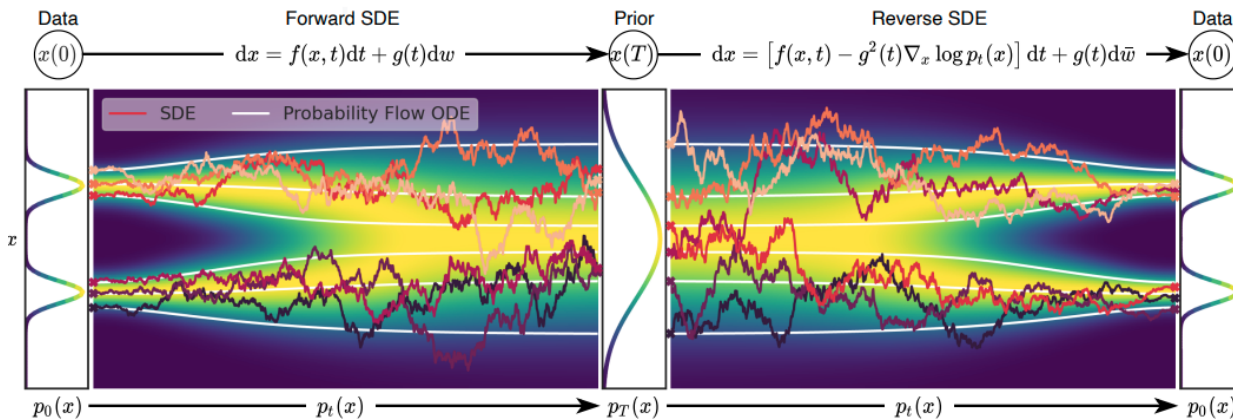


Figure 2.3.6: Overview of score-based generative modeling through SDEs. From [44]

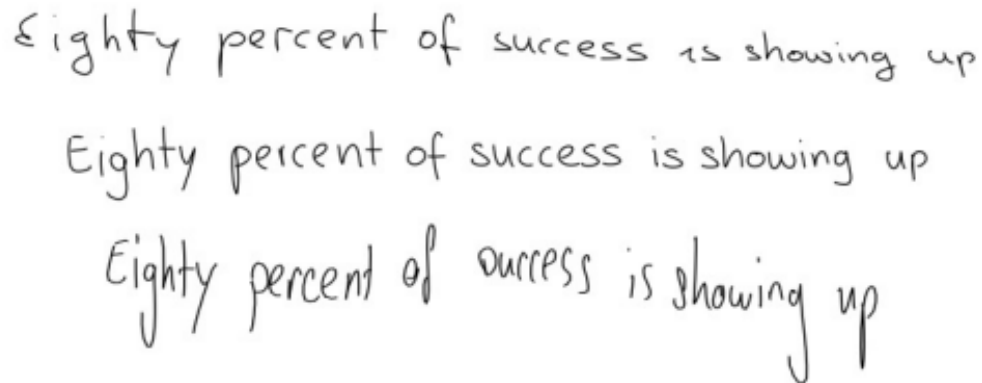
2.4 Introduction to Handwritten Text Recognition and Generation

2.4.1 Handwritten Text Recognition

Handwritten Text Recognition (HTR) primarily focuses on converting handwritten text into machine-encoded text. This is a particularly challenging problem due to the variability in individual handwriting styles, inconsistent spacing between characters, and different types of script (cursive, print, etc.) [22]. There are two main methods in which HTR systems are based on, online and offline. What differentiates these two methods is the data that are used to train the HTR models.

Online HTR

In particular, online HTR involves data, which are generated by the pen's location, while a document or a document or a phrase is being written. An example of online handwritten text is shown below in Figure 2.4.1.

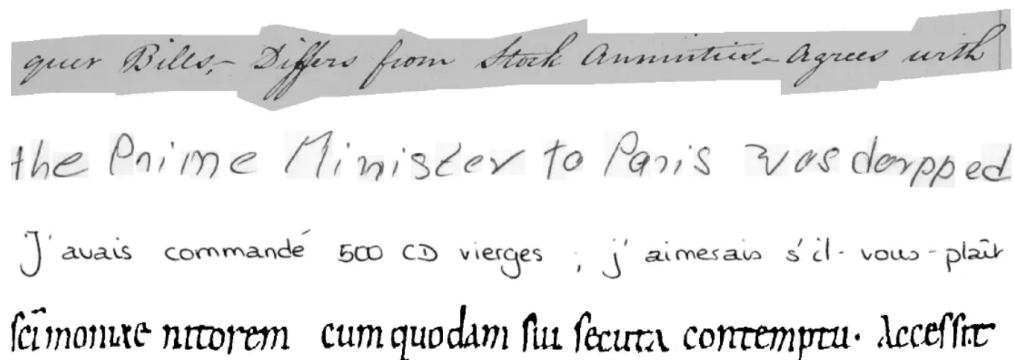


Eighty percent of success is showing up
 Eighty percent of success is showing up
 Eighty percent of success is showing up

Figure 2.4.1: Online Handwritten Text. From [25]

Offline HTR

On the other hand, offline HTR applications are related to data which are generated by scanning original documents. Apparently, offline HTR constitutes a much more complex task, since datasets for online HTR are often characterised by a much better quality. Nonetheless, offline datasets are clearly easier to create, while their applications are also wider including digitizing historical documents, automating postal mail sorting, and even recognizing mathematical equations [27]. Figure 2.4.2 provides an example of offline handwritten text:



quer Bills. Differs from Stock Announced. Agrees with
 the Prime Minister to Paris was dropped
 J'avais commandé 500 CD vierges ; j'aimerais s'il-vous-plait
 sc̄imonare nitorem cum quodam sui secuta contemptu. accessit

Figure 2.4.2: Offline Handwritten Text. From [here](#)

2.4.2 Handwritten Text Generation

Handwritten Text Generation (HTG), on the other hand, aims to create realistic handwritten text given a machine-encoded text and a specific writing style. This can be useful in generating synthetic data for training machine learning models, artistic text generation, and more. Likewise, depending on the used dataset, HTG is also divided into online HTG and offline HTG.

Online HTG

As mentioned above, online HTG concerns the synthesis of handwriting text based on the spatial coordinates of a pen as the writing occurs. Graves [23] proposed the first approach with promising results which was based in Long Short-Term Memory (LSTM) networks [3] and the attention mechanism. Later work [24] provided further improvements in this task by introducing Conditional Variational Recurrent Neural Networks (CVRNNs). In DeepWriting [25], this is approached by differentiating the content of the text from its style, while they improved even more the results by replacing CVRNNs with Stochastic Temporal Convolutional Neural Networks (STCNNs) [26].

Offline HTG

On the contrary, offline HTG, the applications of which are the main scope of this thesis, is related to the synthesis of handwritten text using data from scanned documents. The recent vast improvements in the field of generative models for image synthesis have widely influenced the field of offline HTG, since the provided models are characterised by advanced capabilities which can be obtained in their generated images. In particular, most recent works approach this task with Generative Adversarial Networks (GANs) [6]. ScrabbleGan [28] is one the first approaches which produces convincing results. GANWriting was introduced as an extension to ScrabbleGAN, achieving even higher-quality handwriting synthesis, while they also conditioned the model in the writing styles of each word. The combination of a GAN and an autoencoder, which was proposed in [29] led to further advancement in the field of HTG, but contrary to the other methods, this work focused on generating entire lines of handwritten text. Inspired from this work, Mattick, Mayr, Seuret, Maier and Christlein [30] introduced SmartPatch, which is also a GAN-based approach (for handwritten word generation) which outperformed the previous models producing state-of-the-art results. Unlike previous works, WordStylist [31] was proposed to address the HTG problem with Latent Diffusion Models (LDMs) [32]. More specifically, a LDM is used to produce handwritten words and, according to previous works, it is conditioned on the content and the writing styles. In terms of image quality, WordStylist produces competitive results to SmartPatch, while in the task of adopting a specific writing style it shows superior performance to all other approaches.

The following models are going to be further discussed in Section 3.3:

- GANWriting
- SmartPatch
- WordStylist

Chapter 3

Related Work

3.1	Cold Diffusion	60
3.1.1	Introduction	60
3.1.2	Sampling	60
3.1.3	Generalized Diffusions	61
3.2	Latent Diffusion Models	64
3.2.1	Introduction	64
3.2.2	Latent Diffusion - Conditioning Mechanisms	64
3.2.3	Applications	67
3.3	Handwritten-Text Generation	69
3.3.1	GANWriting	69
3.3.2	SmartPatch	71
3.3.3	WordStylist	74

3.1 Cold Diffusion

3.1.1 Introduction

Recently, diffusion models have gained prominence as effective instruments for generative modeling [93]. Although diffusion models have various forms, they all revolve around the notion of eradicating random noise. These models train a denoising network capable of receiving a Gaussian noise-infected image and producing a cleaned version. At the testing phase, the network transforms pure Gaussian noise into a photorealistic image via an alternating update rule involving the denoiser and the addition of Gaussian noise. The proper series of updates can reveal intricate generative behavior.

Cold Diffusion [33] was proposed to question the necessity of Gaussian noise, or any randomness at all, for the practical application of diffusion models. They contemplate generalized diffusion models that venture beyond the theoretical boundaries originally set for these models. Instead of solely focusing on models centered around Gaussian noise, models constructed around random image transformations are explored. A restoration network is trained to reverse these deformations using a basic l_p loss.

In particular, iterative neural models have been used to address inverse problems [34, 35], while, recently, diffusion models have also been adapted to these, especially for issues such as deblurring [36], denoising [37], super-resolution [38], and compressive sensing [94].

3.1.2 Sampling

Let x_0 be an image from the real-valued space \mathbb{R}^N and consider the degradation of x_0 through an operator D with severity level t , expressed as $x_t = D(x_0, t)$. The output distribution $D(x_0, t)$ resulting from the degradation should demonstrate continuous variation with respect to t . The operator should fulfill the following condition:

$$D(x_0, 0) = x_0 \tag{3.1.1}$$

In the classic diffusion framework, the operator D introduces Gaussian noise with a variance which depends on t . In the following work [33], D is selected to perform different transformations, namely, blurring, pixel masking, downsampling, and snowification, with severity depending on t .

In order to reverse the degradation D , we also need a restoration operator, which is denoted as R . This operator holds the characteristic:

$$R(x, t) \approx x_0 \tag{3.1.2}$$

In practical terms, this operator is realized via a neural network characterized by parameters θ . The restoration network’s training is achieved through the minimization problem:

$$\min_{\theta} \mathbb{E}_{x \sim X} \|R_{\theta}(D(x, t), t) - x\| \tag{3.1.3}$$

where x represents a random image sampled from the distribution X and $\|\cdot\|$ represents a norm (which is chosen to be the l_1 norm in the paper’s experiments). R_{θ} has been used to stress the dependence of R on θ during training, but θ will be discarded for simplicity.

The degradation operator D and restoration model R can be used together to invert severe image degradations using methods from diffusion literature. For slight degradations, a single application of R suffices. But for larger degradations, the restoration can be blurry. Diffusion models solve this by iteratively denoising and adding decreasing amounts of noise back to the image as shown in Algorithm 1. For imperfect restoration operators, this can result in deviations and inaccuracies. In this work [33] a new sampling algorithm (Algorithm 2) is proposed that performs better for smooth, cold degradations and can provide exact reconstructions even when R fails to perfectly invert D .

Algorithm 10: Naive Sampling

Require: A degraded sample x_t **for** $s = t, t - 1, \dots, 1$ **do** $\hat{x}_0 \leftarrow R(x_s, s)$ $x_{s-1} \leftarrow D(\hat{x}_0, s - 1)$ **end for****return** x_0

Algorithm 11: Improved Sampling for Cold Diffusion

Require: A degraded sample x_t **for** $s = t, t - 1, \dots, 1$ **do** $\hat{x}_0 \leftarrow R(x_s, s)$ $x_{s-1} \leftarrow x_s - D(\hat{x}_0, s) + D(\hat{x}_0, s - 1)$ **end for****return** x_0

If the restoration operator is an exact reverse of the degradation operator, both Algorithms accurately recreate the iterator $x_s = D(x_0, s)$ for all $s < t$. In this segment, it is interesting how these algorithms react to inaccuracies in the restoration operator.

The proposed algorithm displays great resilience against restoration operator R errors for minimal values of x and s . To understand why, the following properties of proposed sampling algorithm (Algorithm 2) are pointed out in [33]:

Think of a model issue with a linear degradation function that takes the form

$$D(x, s) \approx x + s \cdot e \quad (3.1.4)$$

for a certain vector e . Despite appearing restrictive, any smooth degradation $D(x, s)$'s Taylor expansion around $x = x_0, s = 0$ takes the following form:

$$D(x, s) \approx x + s \cdot e + HOT \quad (3.1.5)$$

where HOT represents higher order terms. The constant/zeroth-order term in this Taylor series is zero because of Equation 3.1.1.

Given a degradation of the above form and any restoration operator R , Algorithm 2's update can be written out as follows:

$$\begin{aligned} x_{s-1} &= x_s - D(R(x_s, s), s) + D(R(x_s, s), s - 1) \\ &= D(x_0, s) - D(R(x_s, s), s) + D(R(x_s, s), s - 1) \\ &= x_0 + s \cdot e - R(x_s, s) - s \cdot e + R(x_s, s) + (s - 1) \cdot e \\ &= x_0 + (s - 1) \cdot e \\ &= D(x_0, s - 1) \end{aligned}$$

As a result, we deduce that the algorithm generates the value $x_s = D(x_0, s)$ for all $s < t$, irrespective of the selection of R . In other words, the iteration behaves identically to how it would if R was a perfect reverse of the degradation D . On the other hand, Algorithm 1 does not exhibit this property. In fact, if R does not perfectly reverse D , x_0 is not even a constant point of the update rule in Algorithm 1 since $x_0 \neq D(R(x_0, 0), 0) = R(x_0, 0)$. Figure 2 clearly illustrates the difference of the stability of Algorithms 1 and 2.

3.1.3 Generalized Diffusions

In the following section, both qualitative and quantitative results from the experiments of the cold diffusion work [33] are presented for the transformations that were mentioned above. The datasets which are examined in this work are MNIST [62], CIFAR-10 [64] and Celeb-A [95], while the metrics in which the experiments are based are the Fréchet Inception Distance (FID) [70], the Structural Similarity (SSIM) [72] and the Root Mean Square Error (RMSE).

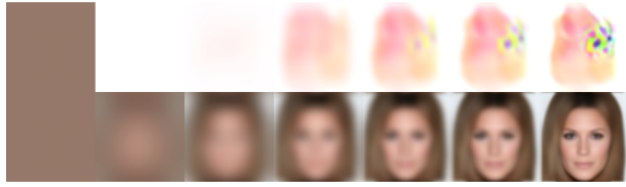


Figure 3.1.1: Comparison of algorithms 1 and 2. From [33]

Deblurring

After training a deblurring model, following the diffusion process:

$$x_t = G_t * \dots * G_1 * x_0 = D(x_0, t) \quad (3.1.6)$$

the following results were obtained:



Figure 3.1.2: Comparison of deblurred images to the degraded and the original ones. From [33]

Table 3.1: Metrics for quality of deblurred images . From [33]

Dataset	Degraded			Sampled			Direct		
	FID	SSIM	RMSE	FID	SSIM	RMSE	FID	SSIM	RMSE
MNIST	438.59	0.287	0.287	4.69	0.718	0.154	5.10	0.757	0.142
CIFAR-10	298.60	0.315	0.136	80.08	0.773	0.075	83.69	0.775	0.071
CelebA	382.81	0.254	0.193	26.14	0.568	0.093	36.37	0.607	0.083

Inpainting

The diffusion process of the inpainting transformation is the following:

$$D(x_0, t) = x_0 \cdot \prod_{i=1}^t z_{\beta_i} \quad (3.1.7)$$

where z_{β_i} are $n \times n$ Gaussian masks and β_i the variances of these masks. The results of the inpainting models are shown below:

Super-Resolution

The diffusion process of the Super-Resolution transformation is downsampling the original image to a 4x4 images in MNIST and CIFAR-10, and 2x2 images in Celeb-A. The results are the following:

Snowification

Bansal et al [33] also experiment with the snowification transform in order to provide an even more general transformation that is not characterized by the blur operators' properties.



Figure 3.1.3: Comparison of reconstructed images from inpainting to the degraded and the original ones.
From [33]

Table 3.2: Metrics for quality of reconstructed images from inpainting. From [33]

Dataset	Degraded			Sampled			Direct		
	FID	SSIM	RMSE	FID	SSIM	RMSE	FID	SSIM	RMSE
MNIST	108.48	0.490	0.262	1.61	0.941	0.068	2.24	0.948	0.060
CIFAR-10	40.83	0.615	0.143	8.92	0.859	0.068	9.97	0.869	0.063
CelebA	127.85	0.663	0.155	5.73	0.917	0.043	7.74	0.922	0.039

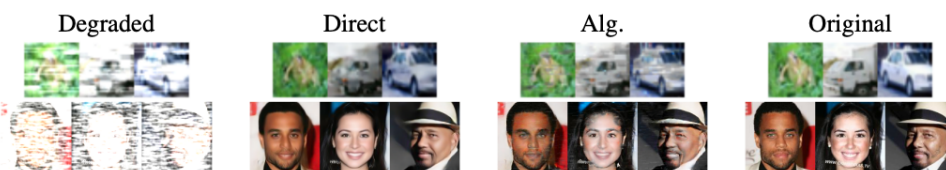


Figure 3.1.5: Comparison of reconstructed images from snowification to the degraded and the original ones.
From [33]



Figure 3.1.4: Comparison of reconstructed images from downsampling to the degraded and the original ones. From [33]

Table 3.3: Metrics for quality of reconstructed images from downsampling. From [33]

Dataset	Degraded			Sampled			Direct		
	FID	SSIM	RMSE	FID	SSIM	RMSE	FID	SSIM	RMSE
MNIST	368.56	0.178	0.231	4.33	0.820	0.115	4.05	0.823	0.114
CIFAR-10	358.99	0.279	0.146	152.76	0.411	0.155	169.94	0.420	0.152
CelebA	349.85	0.335	0.225	96.92	0.381	0.201	112.84	0.400	0.196

3.2 Latent Diffusion Models

3.2.1 Introduction

The field of image synthesis in computer vision has seen impressive growth recently. While high-resolution synthesis is driven by likelihood-based models with autoregressive transformers [39, 40] that can have billions of parameters, the utility of Generative Adversarial Networks (GANs) [41, 42, 43] has limitations due to its difficulty in modeling complex data distributions.

Diffusion Models have recently demonstrated superior results in image synthesis [20, 44] and other tasks [45, 46, 47]. However, Diffusion Models are resource-intensive, since training them requires vast computational resources.

To address this problem, Rombach, Blattmann, Lorenz, Esser and Ommer [32] propose Latent Diffusion Models (LDMs). In particular, the foundations of this approach are the two main stages of learning:

- **Perceptual compression**, which is about handling high-frequency details.
- **Semantic compression**, which focuses on the conceptual aspects of data.

Accordingly, the main idea is to transition to a computationally efficient space for training Diffusion Models for high-resolution image synthesis. Following previous relative work [48, 49, 50, 39, 40], Rombach, Blattmann, Lorenz, Esser and Ommer [32] demonstrate a training procedure which consists of two stages: First, an autoencoder is trained to provide a lower-dimensional representational space, which is finally used to train Diffusion Models, as shown in Figure 3.2.1.

3.2.2 Latent Diffusion - Conditioning Mechanisms

As mentioned above, the proposed perceptual compression model is an autoencoder trained to minimize a perceptual loss [96] and a patch-based [57] adversarial objective [97, 49, 67]. In particular, the autoencoder receives an RGB image as an input and encodes it to a latent representation. The importance of this encoding is substantial, since the latent dimensions are downsampled by a factor $f = 2^m$, with $m \in \mathbb{N}$, hence reducing the computational complexity of training and sampling, compared to a Diffusion Model.

Additionally, Rombach, Blattmann, Lorenz, Esser and Ommer [32] propose a method for more general conditional generation with Diffusion Models. In the realm of image creation, the integration of Diffusion Models' (DMs) generative capabilities with conditionings beyond just class labels [59] or blurred image inputs [38] remains a relatively untouched research topic. The versatility of DMs as conditional image creators is enhanced by enriching their foundational UNet [60] structure with a cross-attention technique [61].

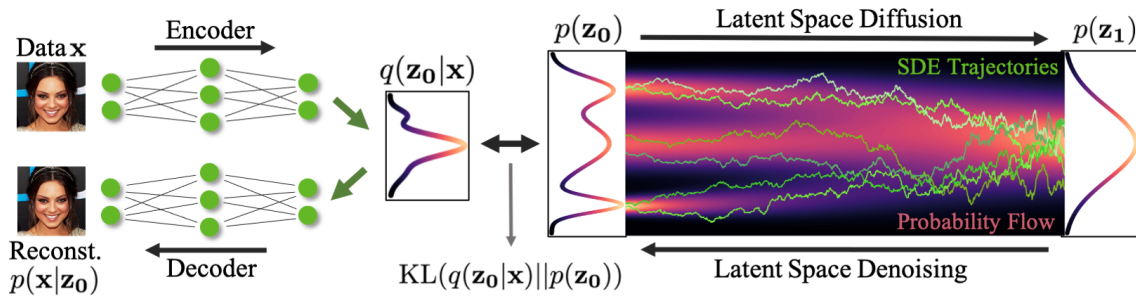


Figure 3.2.1: Latent Diffusion Models. From [Improving Diffusion Models as an Alternative To GANs, Part 2](#)

For preprocessing various input types, a specialized encoder τ_θ is employed in order to convert y into a midway representation. This representation then interacts with the UNet's intermediate layers through a cross-attention layer, defined by the formula:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d}}\right) \cdot V \quad (3.2.1)$$

$$Q = W_Q^{(i)} \cdot \phi_i(z_t)$$

$$K = W_K^{(i)} \cdot \tau_\theta(y)$$

$$V = W_V^{(i)} \cdot \tau_\theta(y)$$

The overall architecture of Latent Diffusion Models is illustrated below in Figure 3.2.2

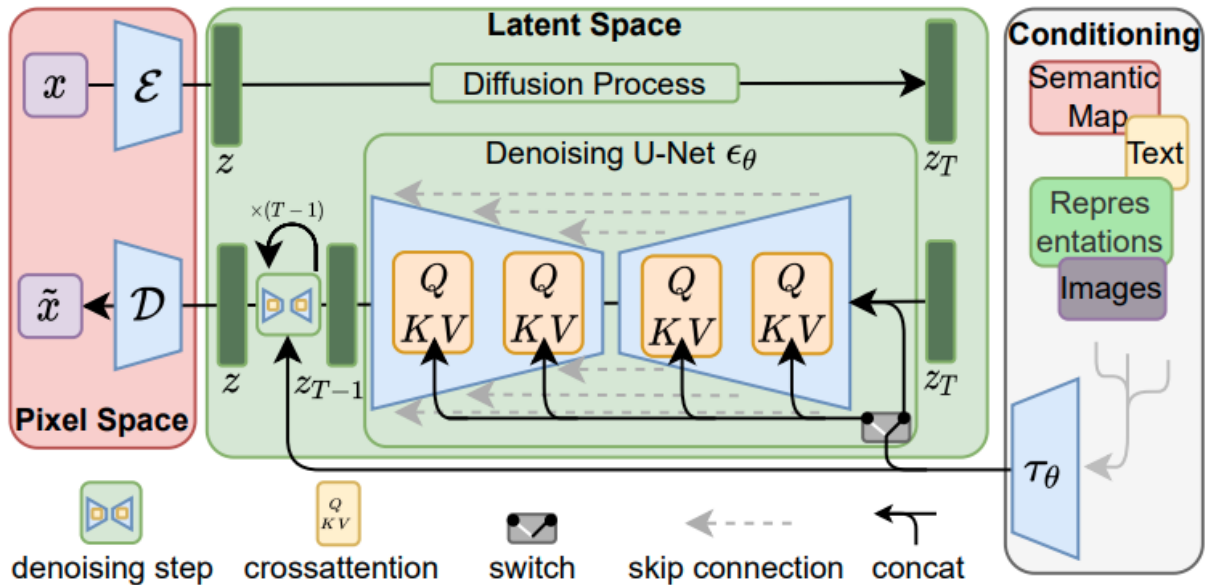


Figure 3.2.2: Conditional (via concatenation or cross-attention) Latent Diffusion Models. From [32]

More precisely, Rombach, Blattmann, Lorenz, Esser and Ommer [32] parameterize τ_θ as a transformer-encoder model [61] using the Bidirectional Encoder Representations from Transformers (BERT) tokenizer [98], while

the application of classifier-free diffusion guidance provides further improvements leading to state-of-the-art results [99].

The class-conditional ImageNet [76] models which perform best ($f \in (4, 8)$) are presented in the following table compared with other state-of-the-art models.

Table 3.4: Comparison of different methods using various metrics. From [32]

Method	FID	IS	Precision	Recall	Nparams
BigGAN-deep [42]	6.95	203.6±2.6	0.87	0.28	340M
ADM [59]	10.94	100.98	0.69	0.63	554M (250 DDIM [68] steps)
ADM-G [59]	4.59	186.7	0.82	0.52	608M (250 DDIM [68] steps)
LDM-4 [32]	10.56	103.49±1.24	0.71	0.62	400M (250 DDIM [68] steps)
LDM-4-G [32]	3.60	247.67±5.59	0.87	0.48	400M (250 steps, c.f.g [32], s = 1.5)

3.2.3 Applications

Inpainting

As described in section 3.1.3, Inpainting involves filling in masked or corrupted areas of an image with new content, or replacing existing unwanted content. The performance of LDMs is evaluated through a comparison with other specialized state-of-the-art methods for inpainting. Using the LaMa [100] protocol as a benchmark (a model employing Fast Fourier Convolutions [101]), Latent Diffusion Models achieve at least a $2.7\times$ speed-up and $1.6\times$ FID score improvement compared to pixel-based models. However, as shown in the quantitative results in the table below [32], Learned Perceptual Image Patch Similarity (LPIPS) [96] of LDMs is a bit higher than LaMa’s [100].



Figure 3.2.3: Object removal with the inpainting LDM. From [32]

Method	40-50% masked		All samples	
	FID ↓	LPIPS ↓	FID ↓	LPIPS ↓
LDM-4 [32] (big, w/ ft)	9.39	0.246 ± 0.042	1.50	0.137 ± 0.080
LDM-4 [32] (big, w/o ft)	12.89	0.257 ± 0.047	2.40	0.142 ± 0.085
LDM-4 [32] (w/ attn)	11.87	0.257 ± 0.042	2.15	0.144 ± 0.084
LDM-4 [32] (w/o attn)	12.60	0.259 ± 0.041	2.37	0.145 ± 0.084
LaMa [100] †	12.31	0.243 ± 0.038	2.23	0.134 ± 0.080
LaMa [100]	12.0	0.24	2.21	0.14
CoModGAN [102]	10.4	0.26	1.82	0.15
RegionWise [103]	21.3	0.27	4.75	0.15
DeepFill v2 [104]	22.1	0.28	5.20	0.16
EdgeConnect [105]	30.5	0.28	8.37	0.16

Table 3.5: Comparison of quantitative results for the inpainting model with the state-of-the-art.

†recomputed on [32] test set, since the one used in [100] was not available. From [32]

Super-Resolution

Latent Diffusion Models can be trained effectively for super-resolution by directly using low-resolution images as conditions. In an experiment that follows the methodology of SR3 [38] and uses a $4\times$ bicubic downsampling, the performance of the super-resolution model outperforms SR3 in the FID score. However, SR3 achieves a higher Inception Score (IS) [71]. On the other hand, even though a simple image regression model can achieve top PSNR and SSIM scores, the capabilities of such metrics are not the appropriate concerning human perception [96].



Figure 3.2.4: Upsampling from 64 to 256 results in ImageNet with the super-resolution LDM. From [32]

Method	FID ↓	IS ↑	PSNR ↑	SSIM ↑	Nparams
Image Regression [38]	15.2	121.1	27.9	0.801	625M
SR3 [38]	5.2	180.1	<u>26.4</u>	<u>0.762</u>	625M
LDM-4 [32] (100 steps)	<u>2.8</u> [†] / <u>4.8</u> [‡]	166.3	24.4 ± 3.8	0.69 ± 0.14	169M
LDM-4 [32] (big, 100 steps)	2.4 [†] / 4.3 [‡]	174.9	24.7 ± 4.1	0.71 ± 0.15	552M
LDM-4 [32] (50 steps, guiding)	4.4 [†] /6.4 [‡]	153.7	25.8 ± 3.7	0.74 ± 0.12	<u>184M</u>

Table 3.6: Upsampling results ($4\times$) using the super-resolution LDM. [†]FID on validation set, [‡]FID on training set. From [32]

3.3 Handwritten-Text Generation

3.3.1 GANWriting

Architecture-Training

This work [51] proposes a method for creating realistic artificial handwriting, conditioned on style and content. The model aims for lifelike images, mimicking specific styles and accurately depicting text. It’s versatile, not limited by vocabulary, and can emulate a given writer’s style quickly.

Let $\{X, Y, W\}$ be a multi-writer handwritten word dataset, containing grayscale word images X , their content (text) Y , and their writer ids $W = \{w_i\}_{i=1}^N$. Let $X_i = \{x_{w_i,j}\}_{j=1}^K \subset X$ be a random subset of K word images from the same given writer $w_i \in W$. Additionally, let A be the character-level vocabulary (alphabet), while A^l being a subset of A , which includes only strings of length l . Based on the above, the generative model H can be formulated as follows:

$$\bar{x} = H(t, X_i) = H(t, \{x_1, \dots, x_K\}) \quad (3.3.1)$$

where \bar{x} is the generated image.

The proposed architecture in [51], as shown in Figure 3.3.1, consists of the following components:

- Generative Network

- **Generator:** The generator receives as an input the concatenated representations of the content and the style and consists of two residual blocks [52] with AdaIN [53] normalization:

$$\text{AdaIN}(z, \alpha, \beta) = \alpha \left(\frac{z - \mu(z)}{\sigma(z)} \right) + \beta \quad (3.3.2)$$

where μ and σ are the channel-wise mean and standard deviations.

The final image \bar{x} is produced after four convolutional modules and a tanh activation.

- **Style encoder:** $\hat{F}_s = S(X_i) + Z$, where S is a VGG-19-BN [54] and $Z \sim \mathcal{N}(0, 1)$.
- **Content encoder:** The final representation is the linear transformation of the one-hot vectors of the strings with 2 MLPs g_1, g_2 (3 layers each with ReLU activations and batch normalization [55]) for character-wise and global string encoding, respectively.

- Learning Objectives

- **Discriminative loss:** The Discriminator’s architecture includes a convolutional layer and six residual blocks [52] with LeakyReLU activations and average poolings. The discriminative loss is the following:

$$\mathcal{L}_d(H, D) = \mathbb{E}_{x \sim X} [\log(D(x))] + \mathbb{E}_{\bar{x} \sim \bar{X}} [\log(1 - D(\bar{x}))] \quad (3.3.3)$$

where D is the discriminator, and \bar{X} the distribution of the output images.

- **Style loss:** The writer classifier follows the same architecture as the discriminator, while, in this case, cross entropy loss is used, so the formulation is the following:

$$\mathcal{L}_w(H, W) = -E_{x \sim \{X, \bar{X}\}} \left[\sum_{i=1}^{|W|} w_i \log(\hat{w}_i) \right] \quad (3.3.4)$$

where \hat{w} is the predicted distribution of writing styles.

- **Content loss:** The word recognizer consists of a VGG-19-BN [54] and Bi-directional Gated Recurrent Unit (B-GRU) [56]. The loss used is the Kullback-Leibler divergence:

$$\mathcal{L}_r(H, R) = -E_{x \sim \{X, \bar{X}\}} \left[\sum_{i=0}^l \sum_{j=0}^{|A|} t_{i,j} \log \left(\frac{t_{i,j}}{\hat{t}_{i,j}} \right) \right] \quad (3.3.5)$$

The total loss for the end-to-end training of GANWriting model is the sum of the above three loss functions:

$$\mathcal{L}(H, D, W, R) = \mathcal{L}_d(H, D) + \mathcal{L}_w(H, W) + \mathcal{L}_r(H, R) \quad (3.3.6)$$

The overall model architecture, the detailed word recognizer architecture, as well as the training procedure, are illustrated in Figure 3.3.1, Figure 3.3.2 and Algorithm 12, respectively.

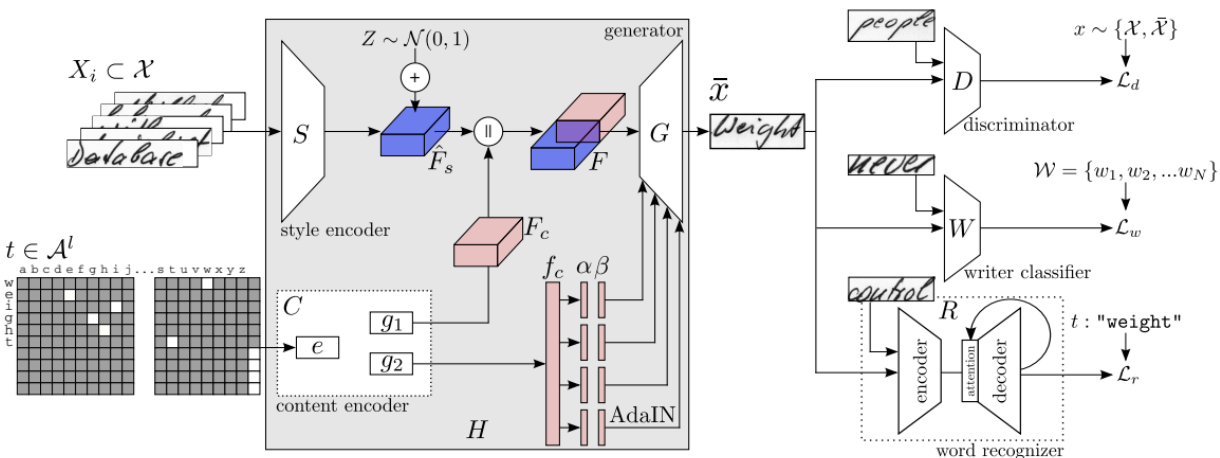


Figure 3.3.1: GANWriting Architecture. From [51]

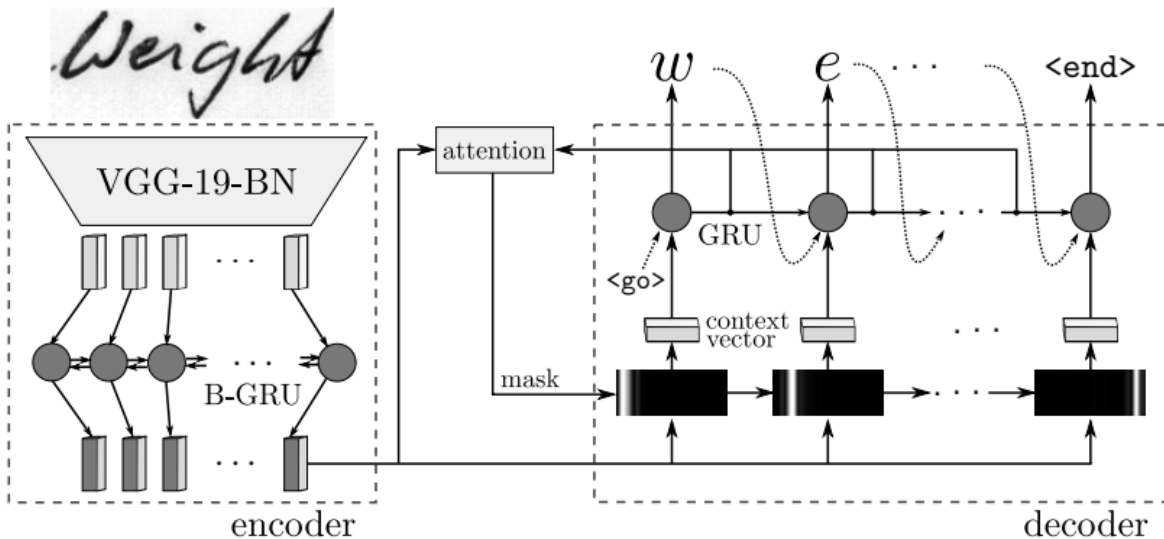


Figure 3.3.2: Word Recognizer Architecture. From [51]

Image Generation

The following qualitative results (Figure 3.3.3) were obtained, demonstrating a performance, which in Out-Of-Vocabulary words or unseen writing styles is competitive to the respective performance in In-Vocabulary

Algorithm 12: Training algorithm for GANWriting. From [51]

Require: Input data $\{X, Y, W\}$; alphabet A ; max training iterations T

Ensure: Networks parameters $\{\Theta_H, \Theta_D, \Theta_W, \Theta_R\}$

```

1: repeat
2:   Get style and content mini-batches  $\{X_i, w_i\}_{i=1}^{NB}$  and  $\{t_i\}_{i=1}^{NB}$ 
3:    $L_d \leftarrow$  Eq. 3.3.3 . Real and generated samples  $x \sim \{X, \bar{X}\}$ 
4:    $L_{w,r} \leftarrow$  Eq. 3.3.4 + Eq. 3.3.5 . Real samples  $x \sim X$ 
5:    $\Theta_D \leftarrow \Theta_D + \Gamma(\nabla_{\Theta_D} L_d)$ 
6:    $\Theta_{W,R} \leftarrow \Theta_{W,R} - \Gamma(\nabla_{\Theta_{W,R}} L_{w,d})$ 
7:    $L \leftarrow$  Eq. 3.3.6 . Generated samples  $x \sim \bar{X}$ 
8:    $\Theta_H \leftarrow \Theta_H - \Gamma(\nabla_{\Theta_H} L)$ 
9: until Max training iterations  $T$ 

```

and seen writing styles. The quantitative results of this work will be discussed in the next subsections for comparison purposes.

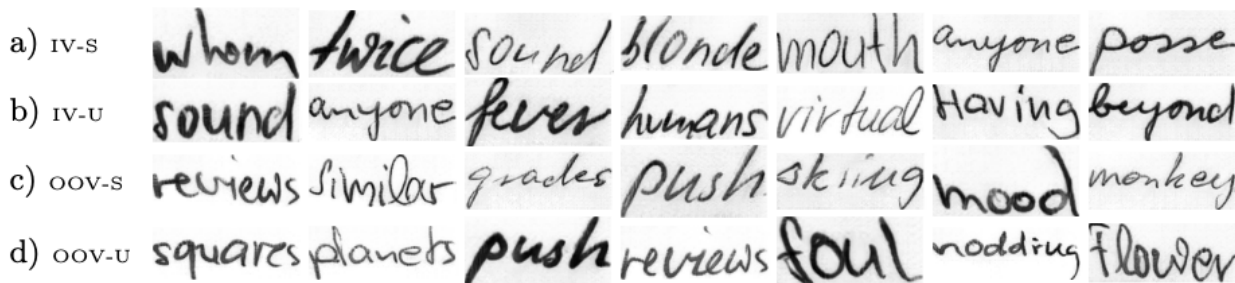


Figure 3.3.3: Image generation: a) In-Vocabulary - Seen style, b) In-Vocabulary - Unseen style, c) Out-Of-Vocabulary - Seen style, d) Out-Of-Vocabulary - Unseen Style. From [51]

3.3.2 SmartPatch

Architecture

Further improvements in handwritten-text generation were provided in [30] by introducing SmartPatch. The architecture of the proposed model is illustrated in Figure 3.3.5. As can be seen, the architecture is highly inspired by GANWriting [51], while the improvements are related to an additional loss component, the local discriminator $\mathcal{L}_{d,\text{local}}$. The motivation behind this addition, is the appearance of some artifacts in GANWriting, as shown in Figure 3.3.4.



Figure 3.3.4: GANWriting Artifacts. From [30]

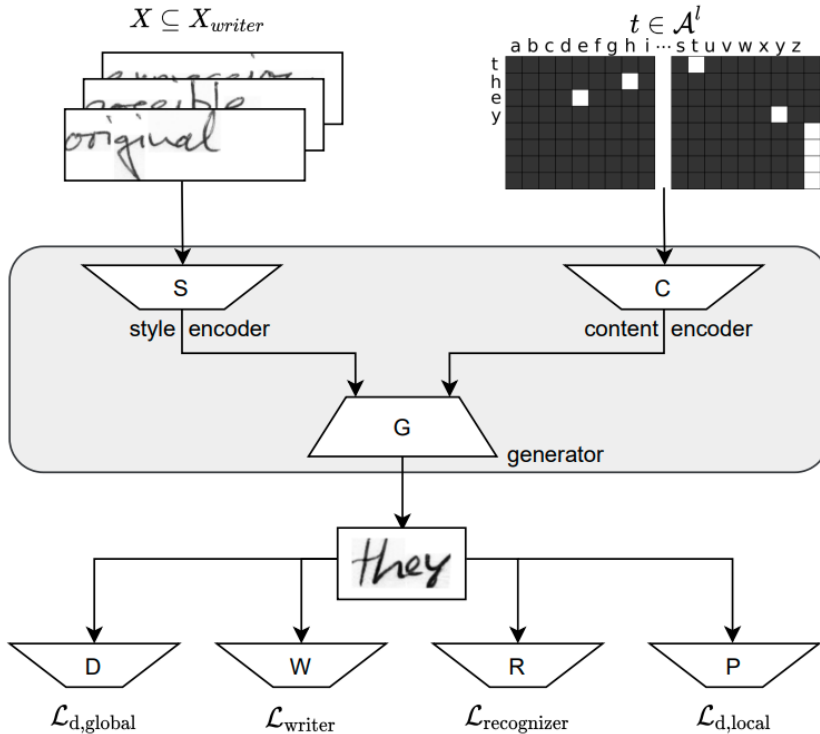


Figure 3.3.5: SmartPatch Architecture. From [30]

Local Discriminators

- **NaivePatch:** This is the baseline approach, which follows the Pix2Pix architecture [57] and just separates the image into overlapping squares ($\frac{\text{square_dim}}{2}$ overlap), inspired by ScrabbleGAN [28]. Accordingly, the total loss becomes:

$$\mathcal{L}(G, D, W, R) = \mathcal{L}_{d, \text{glob}}(G, D) + \mathcal{L}_w(G, W) + \mathcal{L}_r(G, R) + \mathcal{L}_{d, \text{loc}}(p, q) \quad (3.3.7)$$

where $q \in P_{\text{real}}$ and $p \in P_{\text{fake}}$ are the sets of patches.

The loss function of the NaivePatch local discriminator is the following:

$$\mathcal{L}_{d, \text{loc}}(p, q) = \mathbb{E}[\log D_{\text{loc}}(q)] + \mathbb{E}[1 - \log D_{\text{loc}}(p)] \quad (3.3.8)$$

- **CenteredPatch:** This approach leverages the text recognition system, in order to identify the patches of the generated samples. More specifically, up-sampling the attention-vector (from the attention map which is generated in the generator's latent space [58]) to the image's full width leads to a centered window for each character. Both the architecture and the loss function of this local discriminator remain the same as in NaivePatch.
- **SmartPatch:** CenteredPatch is further improved by passing the character-labels $c_{\text{real}}, c_{\text{fake}}$ to the model (after projecting the one-hot character-labels to a latent space C_{enc}). The updated loss function is defined as:

$$\mathcal{L}_{d, \text{loc}}(p, q, c_{\text{real}}, c_{\text{fake}}) = \mathbb{E}[\log D_{\text{loc}}(q, c_{\text{real}})] - \mathbb{E}[1 - \log D_{\text{loc}}(p, c_{\text{fake}})] \quad (3.3.9)$$

The architectural overview of these three different methods is depicted in Figure 3.3.6.

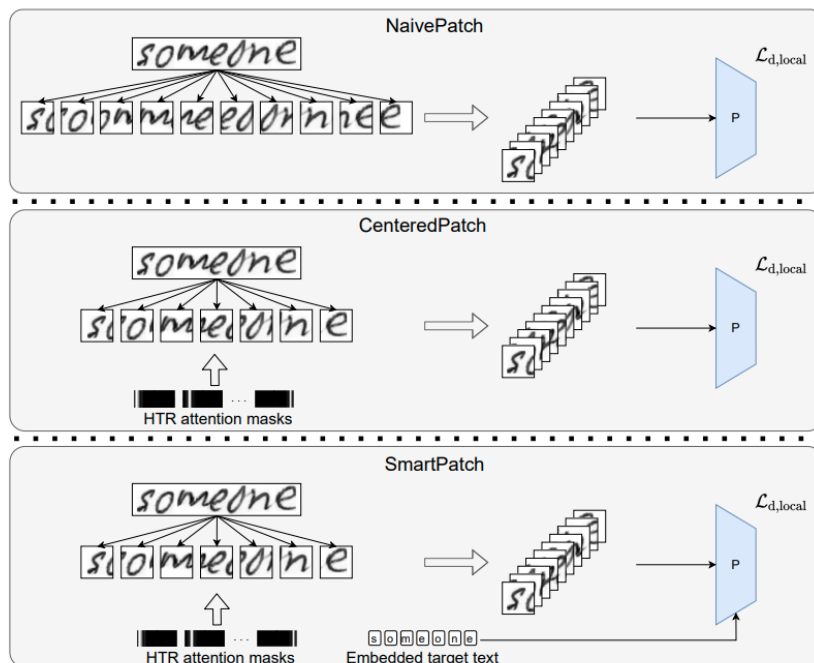


Figure 3.3.6: SmartPatch Methods. From [30]

Image Generation

The following qualitative results were obtained, outperforming the GANWriting [51] model. The quantitative results of this work will be discussed in the next subsection for comparison purposes.

Real IAM	GANwriting	lineGen	NaivePatch (ours)	CenteredPatch (ours)	SmartPatch (ours)

Figure 3.3.7: Comparison of generated images from SmartPatch and GANWriting [51]. From [30]

3.3.3 WordStylist

Contrary to the previous works [51, 30], WordStylist [31] work approaches the generation of handwritten-text images with Diffusion Models [18, 20, 21, 59] instead of GANs [6]. More specifically, Latent Diffusion Models [32] are used in order to reduce the computational cost that is needed for training and sampling, while the model is conditioned to both textual content and writing style of the word images similarly to [51, 30].

Architecture

The architecture of the proposed model is highly inspired by the respective model of [32]. In particular, for the forward process of training and sampling, the departure to latent space is performed using a pretrained autoencoder from Hugging Face ¹, while a UNet [60] is utilized as the noise predictor for the Latent Diffusion Model. Regarding the conditions, the textual content is passed to the network through a cross-attention module [61] as proposed in [32], while the writing style embedding is just added to the time step embedding. Timesteps are encoded by a sinusoidal position embedding as proposed in [61] and writing styles are encoded in an embedding layer, whereas the representation of the text consists of the following stages:

- Tokenization
- Embedding layer
- Positional Encoding (as proposed in [61])
- Self-Attention [61]

The overall architecture is depicted in Figure 3.3.8.

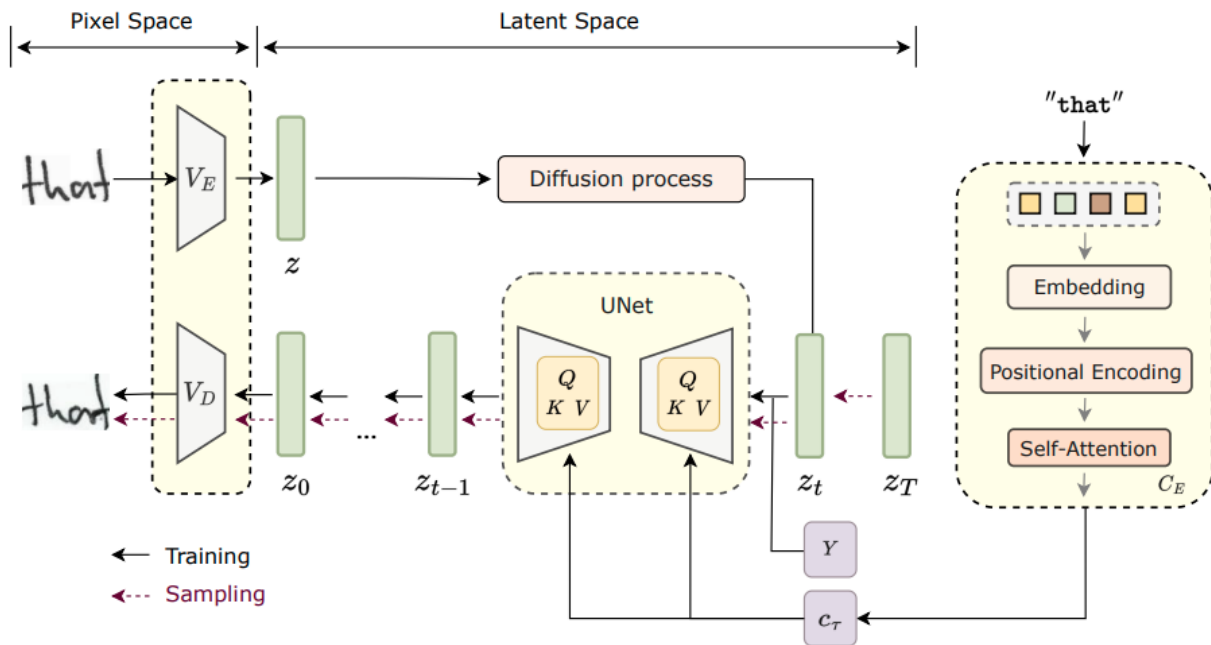


Figure 3.3.8: Wordstylist Architecture. From [31]

Image Generation and Model Comparison

The following results were obtained, regarding the FID score of the trained model. Apart from the qualitative results illustrated in Figure 3.3.9, the table below 5.4 shows that WordStylist [31] clearly outperforms GANWriting [51], while its score is slightly lower than the respective score of [30]. However, as mentioned in

¹<https://huggingface.co/CompVis/stable-diffusion>

[31], FID may not be the most appropriate metric when it comes to the evaluation Handwritten-Text Generation, since it is the Inception V3 [106] (which performs the features extraction of the images) is trained on ImageNet [76].

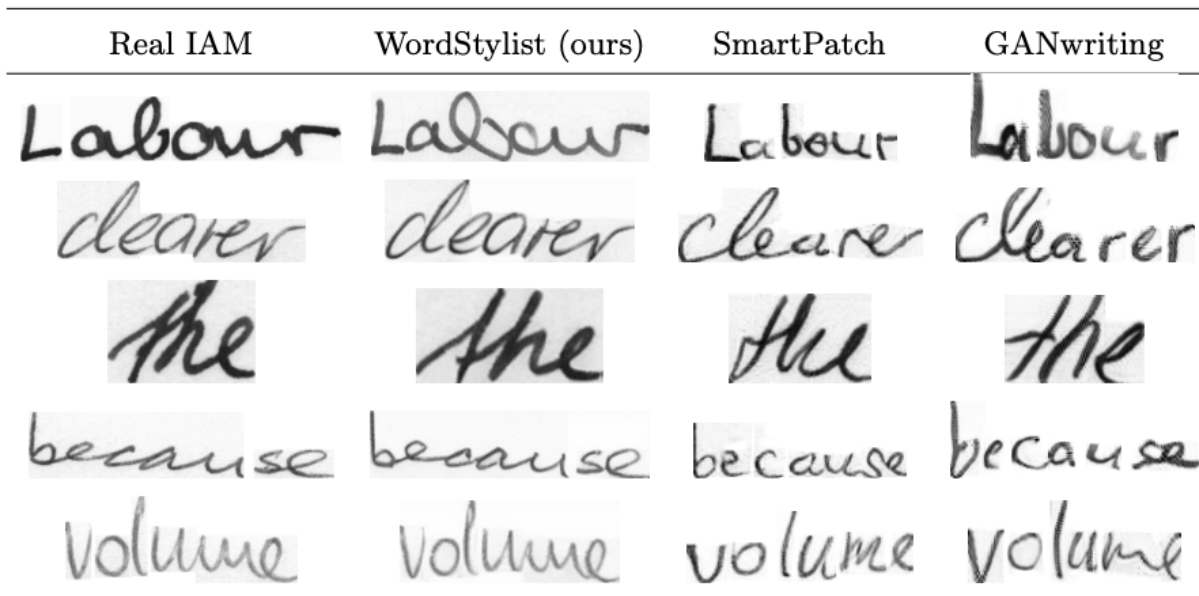


Figure 3.3.9: Comparison of generated images from Wordstylist, SmartPatch [30] and GANWriting [51]. From [31]

Model	FID ↓
GANWriting [51]	29.94
SmartPatch [30]	22.55
WordStylist [31]	22.74

Table 3.7: FID score comparison for the models. Results from [31]

On the other hand, Wordstylist achieves significantly superior results to both GANWriting and SmartPatch in the task of style adaptation. The table 5.5 below illustrates the accuracy of writer-classification performed by a finetuned ResNet18 [77] CNN pretrained on ImageNet.

Test Set	Accuracy (%) ↑
GANWriting [51]	4.81
SmartPatch [30]	4.09
WordStylist [31]	70.67

Table 3.8: Style Adaptation results for the models. From [31]

Chapter 4

Proposed Method

4.1	Morphological Diffusion	78
4.1.1	Motivation	78
4.1.2	Image Reconstruction	79
4.1.3	Conditional Generation	79
4.2	Extensions in the WordStylist model	83
4.2.1	WordStylist Sampling Limitations	83
4.2.2	DDIM Sampling	83
4.2.3	PNDM Sampling	84
4.3	Morphological Diffusion for Handwritten-Text Generation	86

4.1 Morphological Diffusion

In this section, we will introduce the main idea regarding the diffusion process for specific tasks along with some components that improved the results that are going to be discussed in the next chapter.

4.1.1 Motivation

Cold Diffusion [33] was the first approach to propose generalised diffusions with deterministic image degradations. As discussed in section 3.1, this work proposed an improved sampling algorithm for such degradations. Despite providing a proof-of-concept for a variety of diffusions processes, apart from snowification (for which quantitative results were not provided in the paper), all the other diffusions are based on gaussian blur (blurring, inpainting, downsampling), hence not fully aligning with the term 'Generalised Diffusions'. In this thesis, we will experiment with morphological dilation as the diffusion process, which differs significantly from the diffusions of [33], since it is not only a non blur based approach, but also one of the first attempts with non-linear degradation function.

The intuition behind the selection of dilation as the degradation function is founded in the morphology of the images of MNIST [62] and IAM [63] datasets. Handwritten text or digits vary depending on the writer or even the pen, which causes slight deviations on how a word or a number (thinner or thicker lines for instance) is written. These variations clearly align with morphological operations such as erosion or dilation, which we will use to perform a diffusion process that is more suitable for these datasets. Additionally, we will demonstrate the performance of morphological diffusion for CIFAR-10 [64] not only for comparison purposes with [33] but also for potential future work for such datasets.

The following figures (4.1.1 and 4.1.2) consist of some examples from both MNIST and IAM datasets that illustrate the idea described above. As can be seen, both dataset's morphology is clear, due to the variety of writing styles which correspond to the same digit or letter, respectively.

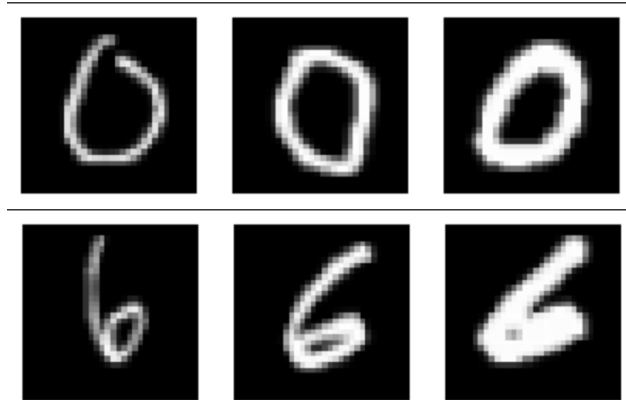


Figure 4.1.1: MNIST Morphology.

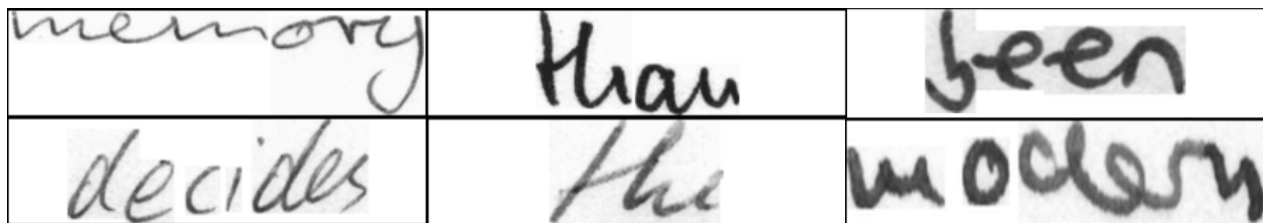


Figure 4.1.2: IAM Morphology.

Our experiments concern the following tasks:

- Image Reconstruction
- Conditional Generation

4.1.2 Image Reconstruction

For the image reconstruction task, we will use both MNIST and CIFAR-10 datasets in order to be able to compare our results with the respective of [33]. The proposed diffusion process for the task of image reconstruction consists of 8 dilation steps. Each dilation step is performed with a 3×3 square structuring element.

Figures 4.1.3 and 4.1.4 depict the diffusion processes of MNIST and CIFAR-10, respectively, for a random instance of each class of the datasets:

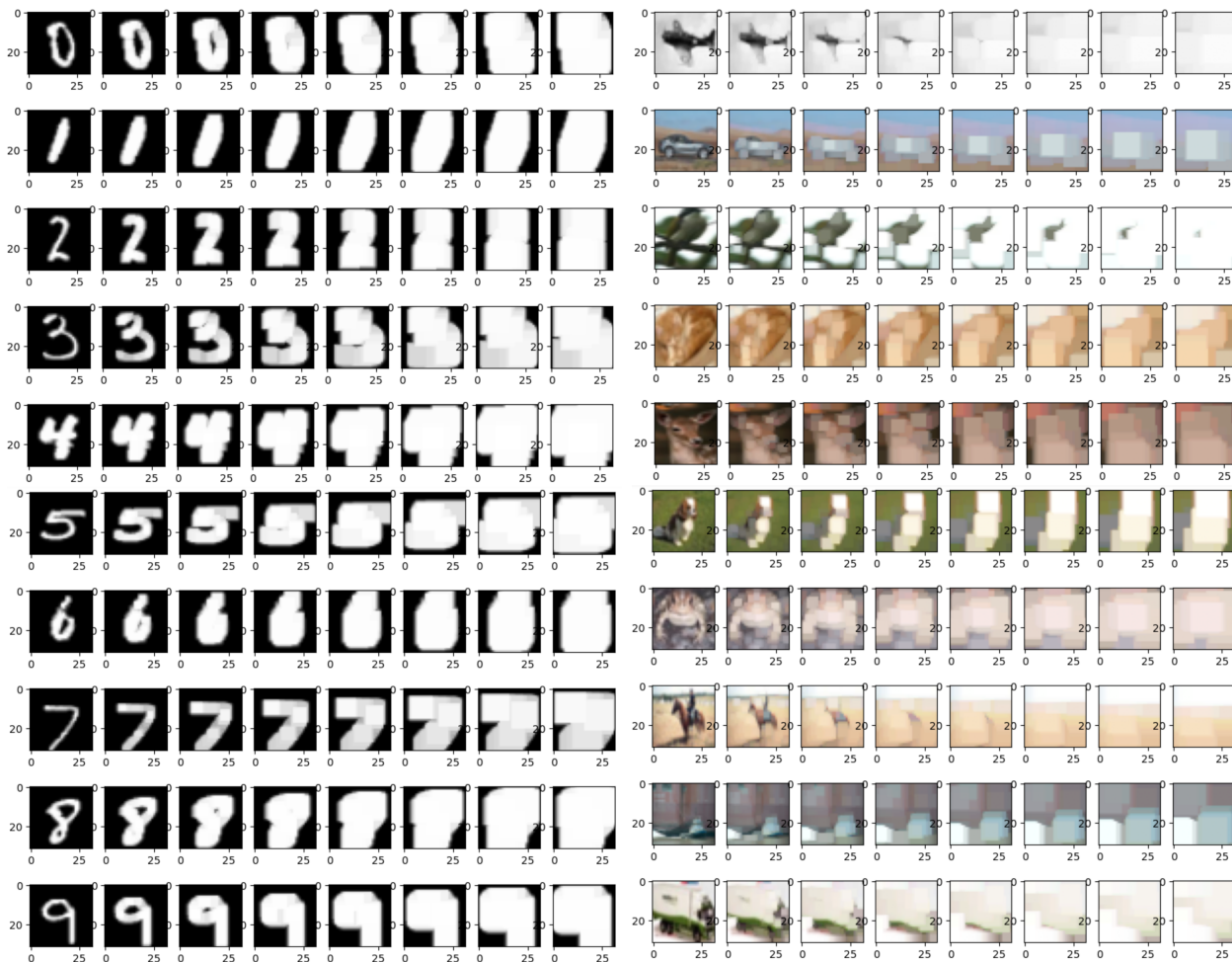


Figure 4.1.3: MNIST degraded examples for each digit.



Figure 4.1.4: CIFAR-10 degraded examples for each class.

4.1.3 Conditional Generation

On the contrary, as shown in Figure 4.1.5, we increase the diffusion steps from 8 to 11, since, in this case, we want the model to be guided mainly by the class condition rather than the degraded image. However, the first image of the sampling process remains a problem, since the morphological diffusion maintains the geometry of each digit (even in some cases in the final time step) unlike the typical gaussian diffusion.

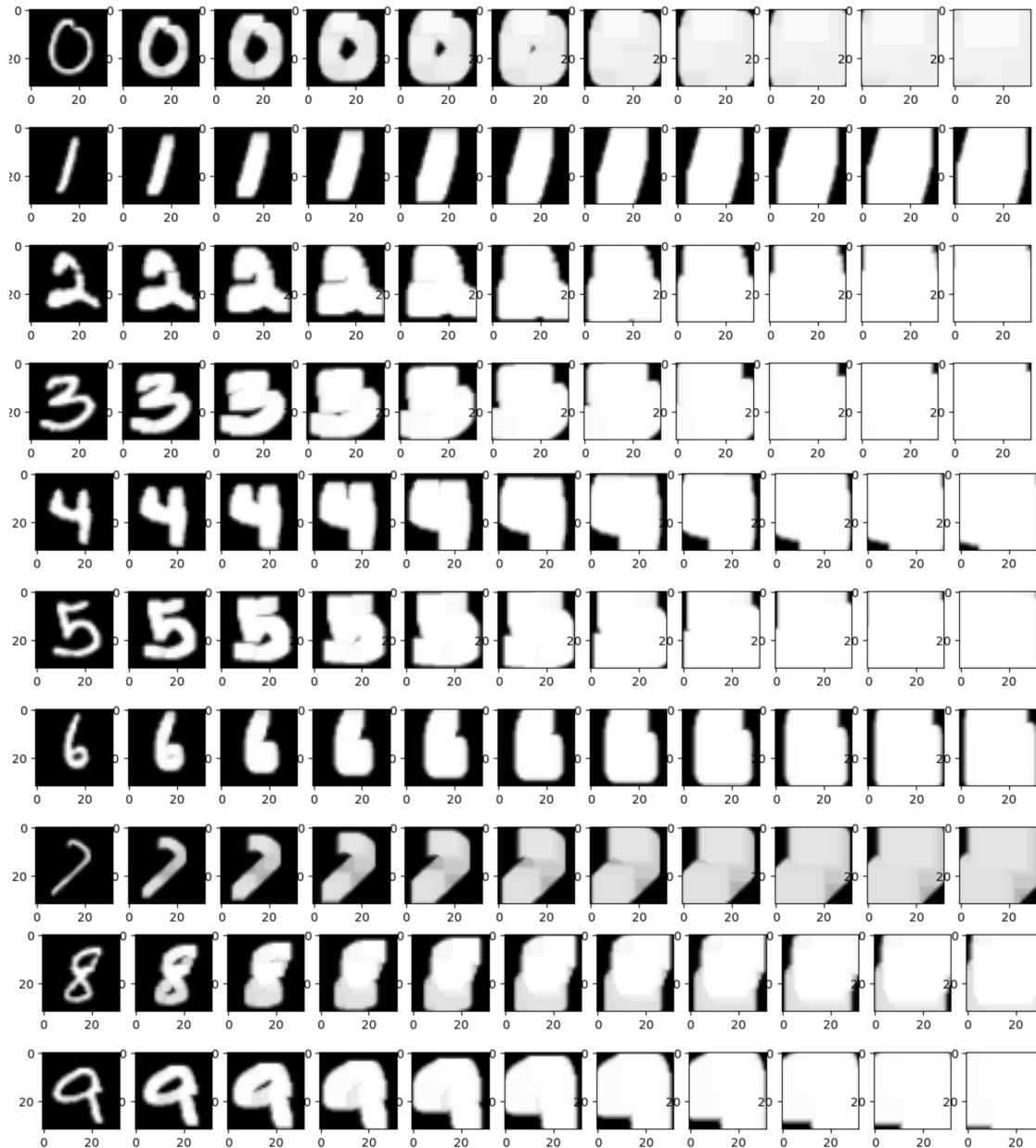


Figure 4.1.5: MNIST degraded examples for each digit.

This problem does not limit just the quality of sampling (since the quality of the final image certainly depends on the first one), but also the diversity of our generated images. This is a result of the non linear properties of dilation, since, in a finite number of steps, each image is degraded to another (constant image), each pixel values of which is the maximum value of the initial image (most the times this value is 1.0). A similar problem is obtained in the case of blurring diffusion [33], where the degraded images are also constant, but their value is the mean of all the pixels as stated in [33], which provides better diversity.

To address this, we will first experiment with dilated gaussian noise, in order to have a benchmark for our next approaches. In particular, after the generation of a random noisy image z , where $z \sim \mathcal{N}(0, 1)$, we clip and normalize the values to $(-1, 1)$ and then dilate it with a 17×17 square structuring element to achieve a dilated digit-like image. Figure 4.1.6 illustrates examples of this process.

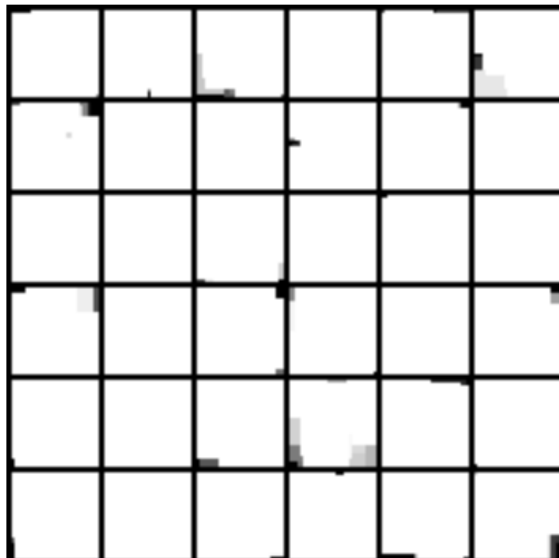


Figure 4.1.6: Dilated noise as the first image of sampling.

In the past years, a lot of research has been based in combining the advantages of two different generative models to substantially increase the performance of the whole network [32, 48, 49, 40, 65, 66, 67]. We will further extend our approach by introducing a two stage class conditional image generation. In particular, the first stage consists of a conditional Generative Adversarial Network (cGAN) [75], while the second stage is a morphological diffusion model as described above. Figure 4.1.8 illustrates five examples of generated images for each class.

As can be seen from this figure, while the trained cGAN captures efficiently the geometry of the dilated digits, it fails to learn the non linear properties of these images, as it introduces some gaussian noise. We will address this problem by zero-padding the image and then performing a closing with a large structuring element (11×11 square) in order to increase the quality of the generated samples. Figure 4.1.9 depicts some examples of the produced samples by the cGAN after the closing, while Figure 4.1.7 demonstrates the effect of this morphological operation.

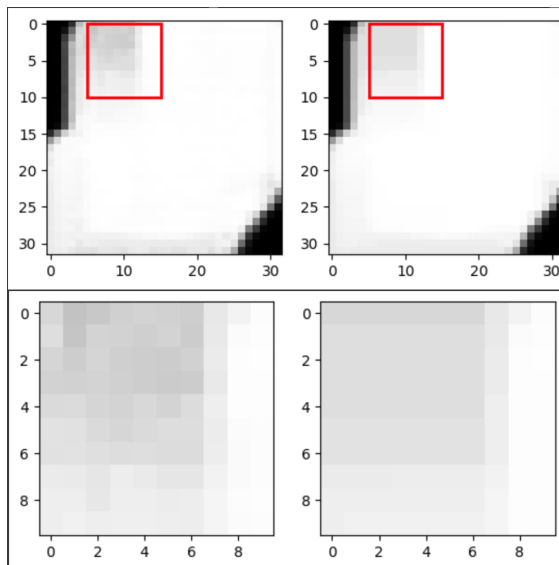


Figure 4.1.7: The effect of closing on the cGAN generated samples.

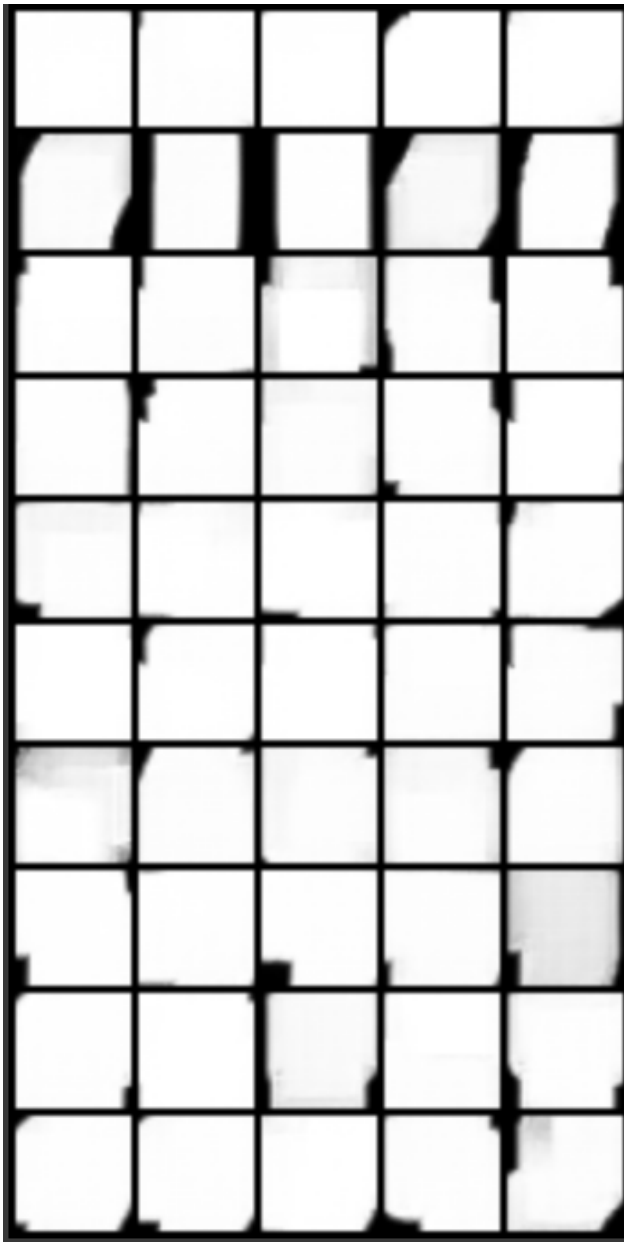


Figure 4.1.8: cGAN samples.

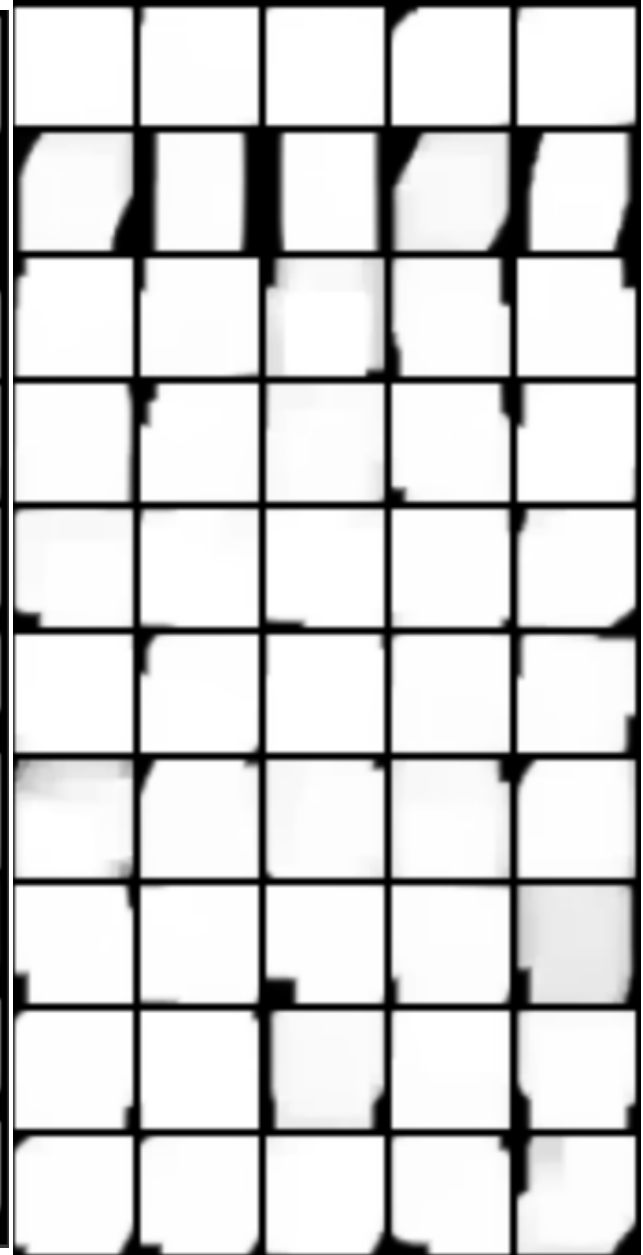


Figure 4.1.9: cGAN samples after closing.

It is important to highlight the simplicity of the cGAN that is used for this purpose. Since one of the major drawbacks of diffusion models concerns their computational needs for efficient training and sampling, we choose a very basic generator, which consists of only 3 convolutional layers with batch normalization [55] and ReLU activations, in order to prevent the sampling process from becoming even more time and resource consuming. Additionally, given the fact that the training lasted just 50 epochs makes cGAN as a minor computational addition in our whole network. More details about its implementation will be discussed at section 5.3.1.

4.2 Extensions in the WordStylist model

4.2.1 WordStylist Sampling Limitations

WordStylist [31] is the only diffusion based approach to Handwritten-Text Generation. As seen in Section 3.3.3 this model produces state-of-the-art results similar to [30], while it clearly outperforms the other models in the task of style adaptation. Overall, it is arguably the best performing model, since its quantitative results in terms of the quality of the generated images is slightly less than SmartPatch [30] (FID scores of 22.74 and 22.55), while its performance in style adaptation is substantially higher as shown in Table 5.5.

However, as mentioned in [31] the main limitation of this approach is the computational efficiency of the sampling process. Despite significantly reducing the computational resource needs by using Latent Diffusion Models [32] and sampling images with 600 steps instead of 1000 (like the training), the sampling process still requires ~ 12 sec to generate a single image. As shown in Figure 4.2.1, simply reducing the sampling steps leads to a notable quality loss, especially when the sampling consists of 300 steps or below.

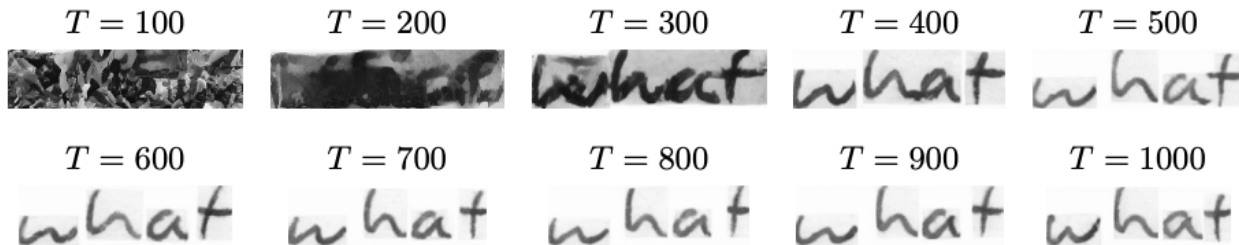


Figure 4.2.1: Sampling with different number of timesteps ranging from 100 to 1000. From [31]

To address this problem we will, first, explore different and more efficient ways of sampling. More specifically, we will use the following sampling algorithms to lower the sampling timesteps with minor effects in the quality of the generated images:

- Denoising Diffusion Implicit Models (DDIM) Sampling [68]
- Pseudo-Numerical Methods for Diffusion Models (PNDM) sampling [69]

4.2.2 DDIM Sampling

In this work, Song, Meng and Ermon [68] questioned the dependence of the loss of a Diffusion Model by the marginals $q(x_T|x_0)$ only as shown in equation 2.3.13 of Section 2.3.2. In particular, the objective was altered in order to depend on the joint $q(x_{1:T}|x_0)$. As a result, non-Markovian inference processes are examined (Figure 4.2.2), which lead to the objective of DDPMs [20].

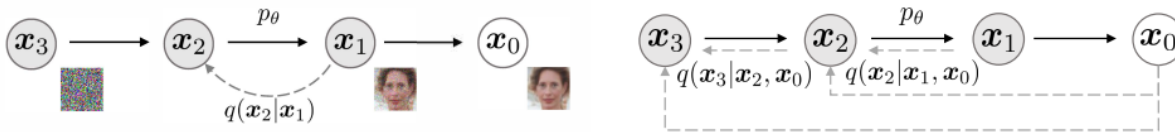


Figure 4.2.2: Graphical models for diffusion and non-markovian inference models. From [68]

The proposed sampling method is defined as:

$$x_{t-1} = \sqrt{\alpha_{t-1}} \left(\frac{x_t - \sqrt{1 - \alpha_t} \epsilon_\theta^{(t)}(x_t)}{\sqrt{\alpha_t}} \right) + \sqrt{1 - \alpha_{t-1} - \sigma_t^2} \cdot \epsilon_\theta^{(t)}(x_t) + \sigma_t \epsilon_t \quad (4.2.1)$$

As mentioned in [68], when $\sigma_t = \sqrt{(1 - \alpha_{t-1})/(1 - \alpha_t)} \sqrt{(1 - \alpha_t/\alpha_{t-1})}$, the forward process becomes markovian, which means that DDPM training is suitable for generating samples, while, when $\sigma_t = 0$ the process becomes deterministic and the new model an implicit probabilistic model [107]. The latter models are named Denoising Diffusion Implicit Models by Song, Meng and Ermon [68]. Following the formulation of PNDMs [69], if we define $\phi()$ as:

$$\phi(x_t, \epsilon_t, t, t - \delta) = \frac{\sqrt{\bar{\alpha}_{t-\delta}}}{\sqrt{\bar{\alpha}_t}} x_t - \frac{(\bar{\alpha}_{t-\delta} - \bar{\alpha}_t)}{\sqrt{\bar{\alpha}_t} \left(\sqrt{(1 - \bar{\alpha}_{t-\delta})\bar{\alpha}_t} + \sqrt{(1 - \bar{\alpha}_t)\bar{\alpha}_{t-\delta}} \right)} \epsilon_t \quad (4.2.2)$$

DDIM sampling can be summarised in the following algorithm:

Algorithm 13: DDIMs. From [69]

```

1:  $x_T \sim N(0, I)$ 
2: for  $t = T - 1, \dots, 1, 0$  do
3:    $x_t = \phi(x_{t+1}, \epsilon_\theta(x_{t+1}, t + 1), t + 1, t)$ 
4: end for
5:
6: return  $x_0$ 

```

4.2.3 PNDM Sampling

Liu, Ren, Lin and Zhao [69] provide further sampling improvements by introducing PNDMs, which are proved to be more general than DDIMs (DDIMs are a special case of PNDMs). More specifically, two similar algorithms, namely, f-PNDM and s-PNDM, are proposed, which are based on numerical methods.

Both algorithms consider the denoising process as the Ordinary Differential Equation (ODE) presented in equation 2.3.20 of Section 2.3.4, but use different numerical methods to solve it.

f-PNDM

In the case of f-PNDM, the proposed solution is the linear multi-step method [108], which is defined as:

$$x_{t+\delta} = x_t + \frac{\delta}{24} (55f_t - 59f_{t-\delta} + 37f_{t-2\delta} - 9f_{t-3\delta}), \quad f_t = f(x_t, t) \quad (4.2.3)$$

Likewise the sampling becomes:

$$\begin{cases} e_t = \epsilon_\theta(x_t, t) \\ e'_t = \frac{1}{24} (55e_t - 59e_{t-\delta} + 37e_{t-2\delta} - 9e_{t-3\delta}) \\ x_{t+\delta} = \phi(x_t, e'_t, t, t + \delta) \end{cases} \quad (4.2.4)$$

where $\phi()$ is the equation 4.2.2.

Since, this numerical method requires the results of the first three sampling timesteps, another numerical method is used, the Runge-Kutta [108]:

$$\begin{cases} e_t^1 = \theta(x_t, t) \\ x_t^1 = \phi(x_t, e_t^1, t, t + \frac{\delta}{2}) \\ e_t^2 = \theta(x_t^1, t + \frac{\delta}{2}) \\ x_t^2 = \phi(x_t, e_t^2, t, t + \frac{\delta}{2}) \\ e_t^3 = \theta(x_t^2, t + \frac{\delta}{2}) \\ x_t^3 = \phi(x_t, e_t^3, t, t + \delta) \\ e_t^4 = \theta(x_t^3, t + \delta) \\ e_t^0 = \frac{1}{6}(e_t^1 + 2e_t^2 + 2e_t^3 + e_t^4) \\ x_{t-\delta} = \phi(x_t, e_t^0, t, t + \delta) \end{cases} \quad (4.2.5)$$

s-PNDM

On the other hand, s-PNDM sampling is performed by using the improved Euler method [108] in equation 4.2.6:

$$\begin{cases} e_t^1 = \theta(x_t, t) \\ x_t^1 = \phi(x_t, e_t^1, t, t + \delta) \\ e_t^2 = \theta(x_t^1, t + \delta) \\ e_t^0 = \frac{1}{2}(e_t^1 + e_t^2) \\ x_{t+\delta} = \phi(x_t, e_t^0, t, t + \delta) \end{cases} \quad (4.2.6)$$

and the second-order linear multi-step method [108] in equation 4.2.7:

$$\begin{cases} e_t = \theta(x_t, t) \\ e_t^0 = \frac{1}{2}(3e_t - e_{t-\delta}) \\ x_{t+\delta} = \phi(x_t, e_t^0, t, t + \delta) \end{cases} \quad (4.2.7)$$

Following the proposed abbreviation:

- Pseudo Linear Multi-Step method: PLMS
- Pseudo Runge-Kutta method: PRK
- Pseudo Second-Order Linear Multi-Step method: PLMS'
- Pseudo Improved Euler method: PIE

the two sampling algorithms can be summarized in the following tables:

Algorithm 14: f-PNDMs. From [69]	Algorithm 15: s-PNDMs. From [69]
1: $x_T \sim N(0, I)$ 2: for $t = T - 1, T - 2, T - 3$ do 3: $x_t, e_t = \text{PRK}(x_{t+1}, t + 1, t)$ 4: end for 5: for $t = T - 4, \dots, 1, 0$ do 6: $x_t, e_t = \text{PLMS}(x_{t+1}, \{e_p\}_{p>t}, t + 1, t)$ 7: end for 8: return x_0	1: $x_T \sim N(0, I)$ 2: for $t = T - 1$ do 3: $x_t, e_t = \text{PIE}(x_{t+1}, t + 1, t)$ 4: end for 5: for $t = T - 2, \dots, 1, 0$ do 6: $x_t, e_t = \text{PLMS}'(x_{t+1}, \{e_p\}_{p>t}, t + 1, t)$ 7: end for 8: return x_0

4.3 Morphological Diffusion for Handwritten-Text Generation

In this subsection, we will use the proposed diffusion process of morphological dilation steps to further improve the efficiency of the standard latent diffusion framework used in WordStylist [31]. In particular, after encoding the original images to the latent space through a pretrained Autoencoder, we will use 30 dilation steps to degrade them. The Autoencoder which is used is the same as the one of the WordStylist work from the Hugging Face repository.

Similarly to Section 4.1, a flat 3×3 square will be used as a structuring element for the each step. A closer examination of the latent representations reveals a composition of four distinct channels. Each channel is imbued with a representation that, while bearing resemblance, is distinctively different from the initial image. A downsampling process by a factor of 8 is applied to each dimension, a reduction resulting in a marked contraction of the initial pixel space’s dimensionality from (3, 64, 256) to a more compact and manageable (4, 8, 32) within the latent space.

Figure 4.3.1 illustrates some original IAM samples, while Figures 4.3.2 and 4.3.3 depict the latent representations of these images and their degraded form after 30 dilation steps, respectively.

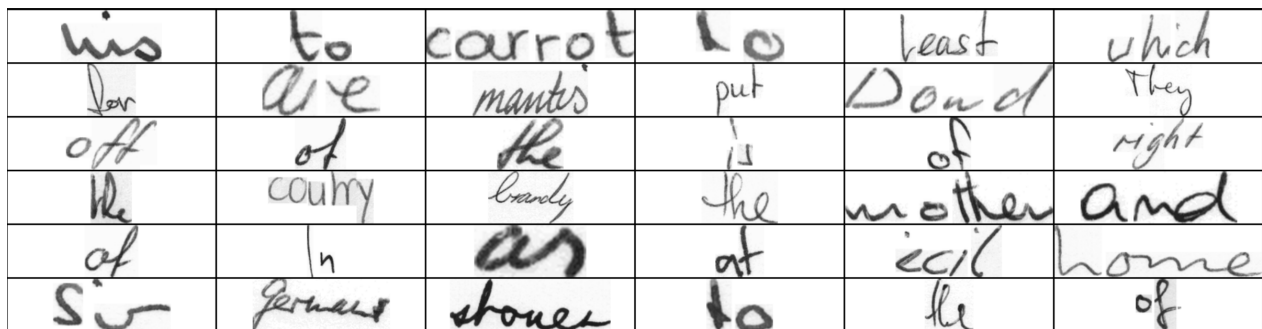


Figure 4.3.1: Original preprocessed IAM samples.



Figure 4.3.2: The 4 channels of the latent representations of IAM samples.

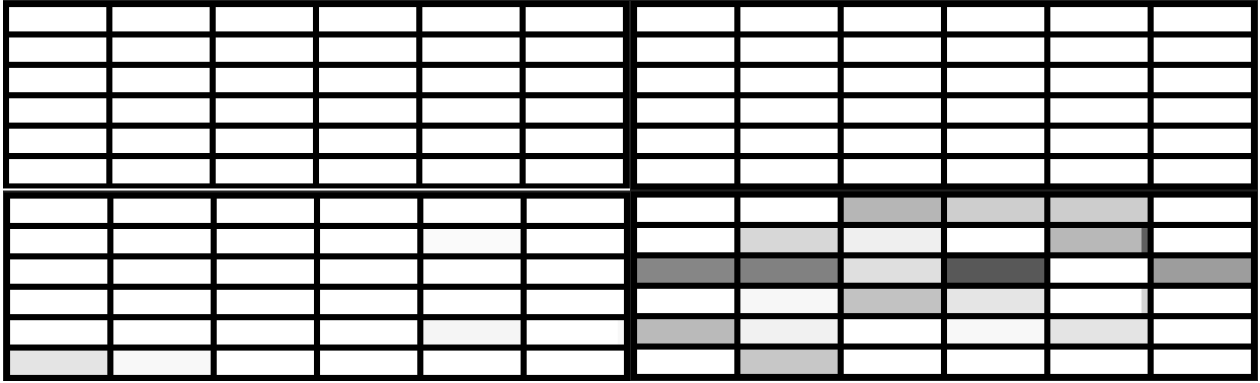


Figure 4.3.3: The 4 channels of the dilated latent representations of IAM samples.

Chapter 5

Experimental Results

5.1	Datasets	90
5.1.1	MNIST	90
5.1.2	CIFAR-10	90
5.1.3	IAM	90
5.2	Evaluation Metrics	91
5.2.1	Fréchet Inception Distance	91
5.2.2	Structural Similarity	91
5.2.3	Root Mean Square Error	92
5.3	Implementation	92
5.3.1	Morphological Diffusion	92
5.3.2	Extensions in the WordStylist model	95
5.3.3	Morphological Diffusion for Handwritten-Text Generation	95
5.4	Results	95
5.4.1	Morphological Diffusion	95
5.4.2	Extensions in the WordStylist model	100
5.4.3	Morphological Diffusion for Handwritten-Text Generation	101

5.1 Datasets

The datasets which are going to be used in our experiments are the following:

- MNIST [62]
- CIFAR-10 [64]
- IAM [63]

5.1.1 MNIST

The MNIST dataset [62] is a collection of handwritten digits that has become a standard for evaluating image processing and machine learning algorithms. Originating from the Modified National Institute of Standards and Technology, MNIST comprises 60,000 training and 10,000 testing grayscale images, all sized at 28×28 pixels. Given its simplicity and established reputation, it often serves as an entry point for those venturing into the field of machine learning, especially in the domain of image classification.

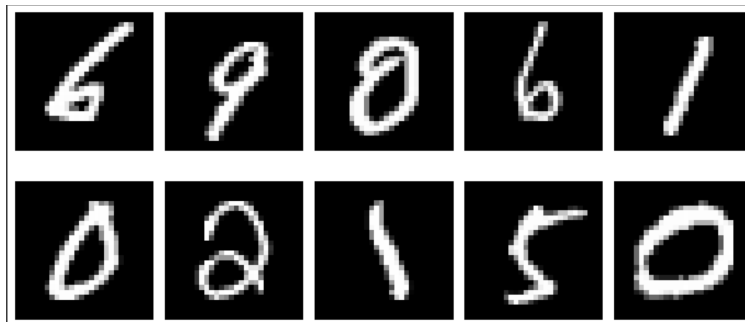


Figure 5.1.1: MNIST instances

5.1.2 CIFAR-10

The CIFAR-10 dataset [64], developed by the Canadian Institute For Advanced Research, is a cornerstone for the evaluation of machine learning models dedicated to object recognition. It consists of 60,000 colored images that are divided into 10 distinct classes, such as airplanes, birds, and cars, each class containing 6,000 images. These images, with a resolution of 32×32 pixels, offer a more challenging classification task than MNIST due to their complexity and diversity.

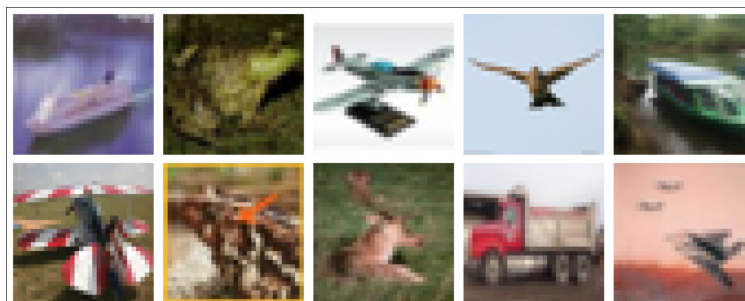


Figure 5.1.2: CIFAR-10 instances

5.1.3 IAM

The IAM Handwriting Database [63] stands as a pivotal resource for tasks linked to handwriting. It offers a rich array of handwritten English text penned down by over 600 unique writers. The diversity and depth of

this dataset, which totals 1,657 scanned pages of varied content, make it apt for Handwritten Text Recognition (HTR) and writer identification research

In our case, similarly to [31] we will use the Aachen split train set with the length of the words ranging from 2 to 10 characters, resulting in 44,412 images with 339 writing styles. We will also adopt the same preprocessing procedure as proposed in [31], in order to resize the images to a fixed image size 64×256 .

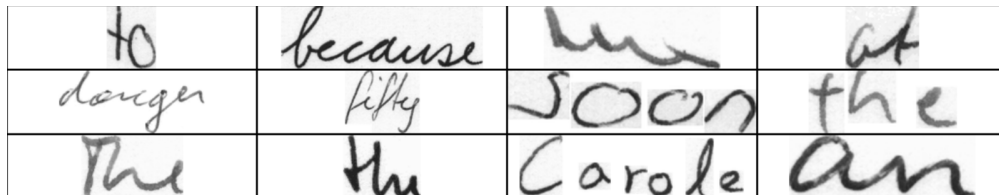


Figure 5.1.3: IAM preprocessed instances

5.2 Evaluation Metrics

In this section, the evaluation metrics of the experiments will be presented, which are the following:

- Fréchet Inception Distance [70]
- Structural Similarity [72]
- Root Mean Square Error

5.2.1 Fréchet Inception Distance

Fréchet Inception Distance (FID) [70] is a metric initially proposed to evaluate GANs [6]. However, recently, FID score is widely used in the fields of Computer Vision and Generative AI, in order to evaluate the quality of generated images. FID is presented as an improvement to Inception Score [71], since its calculation requires both real world and synthetic samples. In particular, first, an Inception model (Inception V3 [106] in our case) is used to perform a feature extraction for the real and synthetic images. Next, assuming that these distributions both follow a multidimensional Gaussian with different mean and variance, their distance can be computed by Fréchet Distance [109] (also known as Wassertein-2 distance [110]) as follows:

$$d^2((m, C), (m_w, C_w)) = \|m - m_w\|_2^2 + \text{Tr}(C + C_w - 2\sqrt{CC_w}) \quad (5.2.1)$$

where m, m_w are the means and C, C_w are the variances of the Gaussians.

5.2.2 Structural Similarity

Structural Similarity (SSIM) was introduced by Wang, Bovik, Sheikh and Simoncelli [72] as an improvement of Universal Quality Index (UQI) [111, 112]. SSIM has been a significant measure for image quality assessment, although in the last few years it is not used as frequently as in the past, due to rise of new, more powerful metrics like FID. It is defined as:

$$\text{SSIM}(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \quad (5.2.2)$$

where x, y are the images, μ_x, μ_y are the means of the images, σ_x, σ_y are the variances of the images, σ_{xy} is the cross-correlation of the images and C_1, C_2 two variables to avoid instabilities in the denominator ($C_1 = C_2 = 0$ leads to the UQI).

5.2.3 Root Mean Square Error

Finally, we will also use the Root Mean Square Error (RMSE) for our experiments which is formulated as:

$$\text{RMSE}(I_1, I_2) = \sqrt{\frac{1}{M \times N} \sum_{i=1}^M \sum_{j=1}^N (I_1(i, j) - I_2(i, j))^2} \quad (5.2.3)$$

where $I_1(i, j)$, $I_2(i, j)$ are the pixel values at the location (i, j) and $M \times N$ is the image size.

5.3 Implementation

5.3.1 Morphological Diffusion

Image Reconstruction

For our Image Reconstruction models for both MNIST [62] and CIFAR-10 [64] the implementation details are the same, since we resize MNIST samples to 32×32 (the same size as CIFAR-10 samples), similar to [33]. More precisely, we performed a training of 500 epochs and batch size 32 using Adam optimizer [73] with learning rate 2×10^{-5} . Regarding the diffusion process, each image is dilated 8 times with a 3×3 square structuring element.

The model which is used for denoising is a U-net [60], while the final model we use for sampling is an Exponential Moving Average of the trained model (decay rate 0.995) in order to increase the stability of our model’s convergence. The backbone of the U-net is the same with the proposed one of [33] for consistency purposes in terms of the experimental results.

The architecture of the U-net is shown in Figure 5.3.1

Conditional Generation

For our conditional model for MNIST [62], the implementation details are the following: We performed a training of 100 epochs and batch size 128 using AdamW optimizer [74] with learning rate 10^{-4} . Regarding the diffusion process, each image is dilated 11 times with a 3×3 square structuring element.

The model which is used for denoising is a U-net [60], while the final model we use for sampling is again an Exponential Moving Average of the trained model (decay rate 0.995) in order to increase the stability of our model’s convergence. The backbone of the U-net is the same with the proposed one of [31] for consistency purposes in terms of the experimental results. The class conditions of the MNIST dataset are passed to the model through a cross attention [61] component after a simple embedding layer.

As has already been discussed, the first stage of the sampling procedure is a simple conditional GAN [75]. Its training lasted 50 epochs with batch size 128 and Adam optimizer [73] with learning rate 10^{-4} . The conditions were simply transformed to one-hot encoding and then concatenated to the starting noise given to the Generator.

The architectures of the U-net and the GAN are shown in Figure 5.3.2 and Figure 5.3.3, respectively.

Block Group	# of Blocks	Type of Block
Down Blocks	4 x	<i>ConvNeXt Block</i>
		<i>ConvNeXt Block</i>
		<i>Attention Block + Normalization + Residual</i>
		<i>Downsampling</i>
Middle Blocks	1 x	<i>ConvNeXt Block</i>
		<i>Attention Block + Normalization + Residual</i>
		<i>ConvNeXt Block</i>
Up Blocks	4 x	<i>Concatenation + ConvNeXt Block</i>
		<i>ConvNeXt Block</i>
		<i>Attention Block + Normalization + Residual</i>
		<i>Upsampling</i>
Final Layer	1 x	<i>ConvNeXt Block</i>
		<i>2D Convolution</i>

Figure 5.3.1: The architecture of the U-net used for Image Reconstruction.

Block Group	# of Blocks	Type of Block
Down Blocks	4 x	<i>Residual Block</i>
		<i>Spatial Transformer Block *</i>
		<i>Downsampling</i>
Middle Blocks	1 x	<i>Residual Block</i>
		<i>Spatial Transformer Block</i>
		<i>Residual Block</i>
Up Blocks	4 x	<i>Concatenation + Residual Block</i>
		<i>Spatial Transformer Block *</i>
		<i>Upsampling</i>
Final Layer	1 x	<i>Normalization</i>
		<i>SiLU Activation</i>
		<i>2D Convolution</i>

Figure 5.3.2: The architecture of the U-net used for Conditional Generation. * Spatial Transformer Blocks were only in the first Block group for Up and Down Blocks.

Block Group	# of Blocks	Type of Block
Initial Block	1 x	<i>Concatenation of input noise and one hot label encodings</i>
Middle Blocks	3 x	<i>2D Transposed Convolution</i>
		<i>2D Batch Normalization</i>
		<i>ReLU Activation</i>
Output Block	1 x	<i>2D Transposed Convolution</i>
		<i>Tanh Activation</i>

Figure 5.3.3: The architecture of the GAN used for generating the initial masks.

In Figure 5.3.1, ConvNeXt Blocks correspond to the proposed blocks in [113], Attention Blocks are simple self-Attention components [61], Residual is a residual connection [77], while Downsampling and Upsampling are performed with 2D convolutions and 2D transposed convolutions, respectively.

On the other hand, in Figure 5.3.2, Residual Blocks are the blocks proposed in [77], Spatial Transformer Blocks correspond to the blocks proposed in [61], while Downsampling and Upsampling are performed with a 2D convolution and an nearest-neighbor interpolation followed by a 2D convolution, respectively.

5.3.2 Extensions in the WordStylist model

This section concerns the application of DDIM [68], f-PNDM [69] and s-PNDM [69] sampling algorithms, so there is no training procedure that was implemented. In order to perform the sampling experiments we used the weights of the trained WordStylist [31] model from [here](#).

5.3.3 Morphological Diffusion for Handwritten-Text Generation

For our conditional model for IAM [63] dataset, most implementation details regarding the training procedure and the hyperparameters are the same as in the respective model for MNIST. In our adaptation of the conditional model for the IAM dataset [63], we maintained most of the training and hyperparameter settings from the model originally developed for MNIST. However, we made a modification in the conditioning mechanism to cater to the textual content and writing style unique to the IAM dataset. Additionally, we transitioned from using the L_2 loss function to L_1 loss. This change was prompted by the latent space value range in our model. Specifically, the latent representations were within a range of -4 to 4. Utilizing the L_2 loss in this context resulted in excessively high losses, leading to substantial gradient steps and consequent overfitting. The adoption of the L_1 loss function mitigated this issue, promoting a more stable and effective training process.

The conditioning mechanism that we follow is the one proposed in Wordstylist [31], which consists of the following stages.

- Character-level tokenization
- Embedding layer
- Positional Encoding
- Self-Attention [61]

5.4 Results

5.4.1 Morphological Diffusion

Qualitative Results

In the following section, we will provide the qualitative results for the models described in section 4.1. More specifically, we will initially display the reconstruction results for the MNIST [62] and CIFAR-10 [64] datasets. More specifically, we will present 50 samples of each dataset in total, which contain 5 samples of each class in every row. Likewise, we will also visualize the results for the two-stage approach described again in section 4.1.

As can be seen, from Figures 5.4.1, 5.4.2, 5.4.3, these results indicate that the model has learnt the distributions of both datasets, while it is also able to further generalize by getting a random image generated from a conditional Generative Adversarial Network and producing decent results.



Figure 5.4.1: Image Reconstruction Results for MNIST.

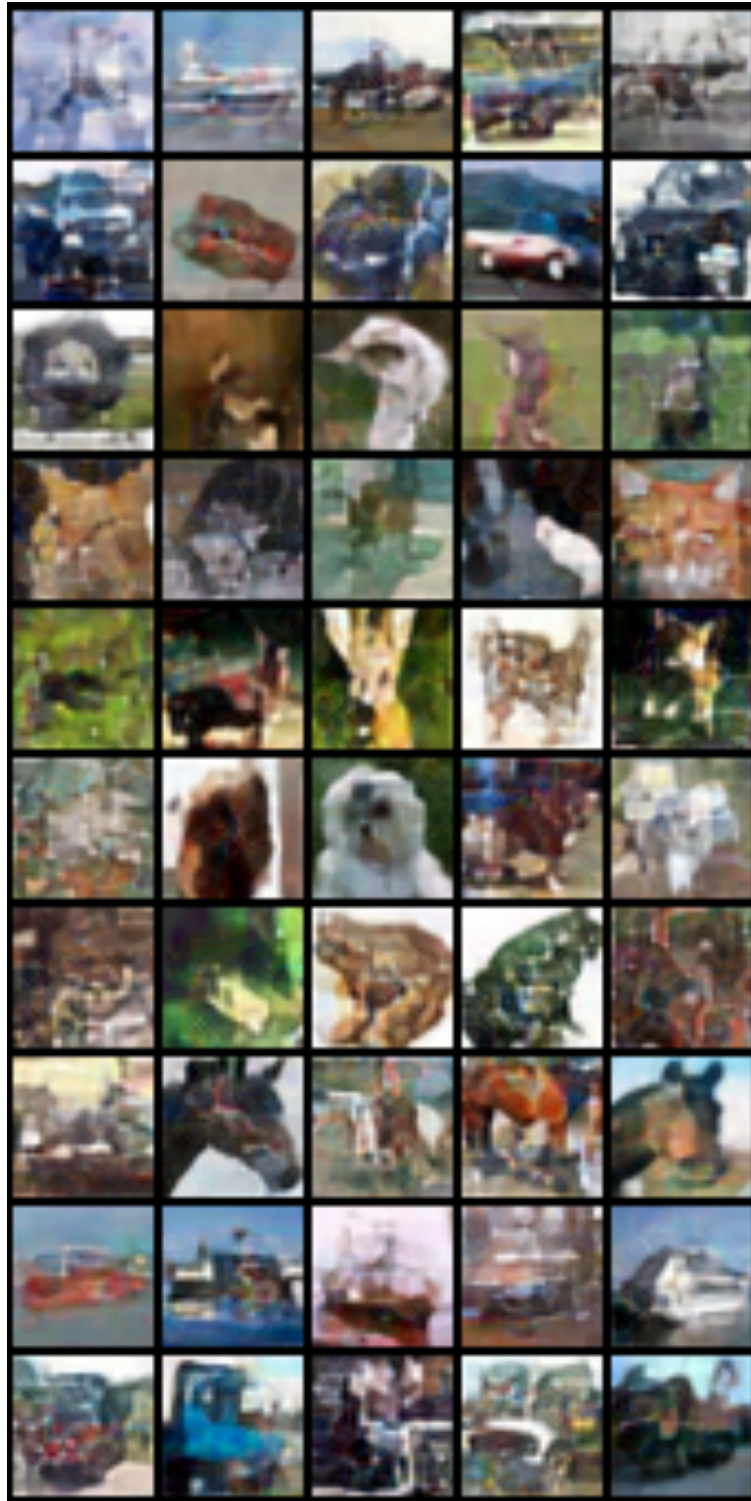


Figure 5.4.2: Image Reconstruction Results for CIFAR-10.



Figure 5.4.3: Conditional Generation Results for MNIST.

Quantitative Results

In this section, we will perform our quantitative analysis of the results of our previous models. For the reconstruction models, we will compare their results for MNIST [62] and CIFAR-10 [64] datasets with the respective results of [33].

Table 5.1: Comparison of Metrics for the Proposed Diffusions of [33] and ours for MNIST [62].

Degradation	Degraded			Sampled		
	FID	SSIM	RMSE	FID	SSIM	RMSE
Gaussian blur	438.59	0.287	0.287	4.69	0.718	0.154
Inpainting	108.48	0.490	0.262	1.61	0.941	0.068
Downsampling	368.56	0.178	0.231	4.33	0.820	0.115
Dilation (our approach)	314.87	0.005	0.78	2.46	0.96	0.05

Table 5.2: Comparison of Metrics for the Proposed Diffusions of [33] and ours for CIFAR-10 [64].

Degradation	Degraded			Sampled		
	FID	SSIM	RMSE	FID	SSIM	RMSE
Gaussian blur	298.60	0.315	0.136	80.08	0.773	0.075
Inpainting	40.83	0.615	0.143	8.92	0.859	0.068
Downsampling	358.99	0.279	0.146	152.76	0.411	0.155
Dilation (our approach)	217.81	0.11	0.39	52.42	0.63	0.11

As can be seen from Tables 5.1 and 5.2, our approach achieves superior results to the other trained models of [33] except from the one which was trained for the inpainting task. However, this was expected, since the inpainting degradation has a significantly lower impact to the initial compared with gaussian blur, downsampling and dilation.

This is also illustrated in the metrics of the degraded images of the two datasets. For instance, regarding the MNIST dataset, we can see that inpainted images are characterized by an FID of 108.48, which is almost the $\frac{1}{3}$ of the minimum FID of the other three degradations. In the CIFAR-10 dataset, the impact of the inpainting degradation is even lower, as the inpainted images have an FID, which is less than the reconstructed results from all the other transformations.

The next table shows the quantitative results for the conditional generation task for the MNIST dataset. Since, there is still no other approach which concerns generalized, deterministic diffusion for class conditional generation, the results of our experiments will be compared with those of standard gaussian diffusion models, which share some similarities with our approach, regarding the complexity of the training and sampling process.

More specifically, we will first train a diffusion model with 10 timesteps and measure its FID with the original dataset, while the other gaussian diffusion models will be trained for 100 steps but sampled with 10 steps, using DDIM [68], f-PNDM [69] and s-PNDM [69] sampling methods. As can be seen from the table, all these methods clearly fail to generate decent samples, due to the small number of timesteps. On the contrary, our morphological diffusion based approach seems to produce results which are much better as shown by the FID scores.

Additionally, these scores also indicate the effect of the GAN masks and the morphological closing, since they lead to a substantially reduced FID score compared with not only, the one achieved with dilated noise as input, but also the one that used only the GAN masks.

Table 5.3: Comparison of Metrics for Different Conditional Generation Models with 10 sampling steps for MNIST [62].

Model	FID
Gaussian Diffusion (100 steps)	4.31
Gaussian Diffusion (100 steps) with 10 DDIM [68] sampling steps	163.90
Gaussian Diffusion (100 steps) with 10 f-PNDM [69] sampling steps	293.76
Gaussian Diffusion (100 steps) with 10 s-PNDM [69] sampling steps	226.02
Gaussian Diffusion (10 steps)	123.31
Morphological Diffusion (10 steps) with dilated noise	41.30
Morphological Diffusion (10 steps) with GAN masks	19.45
Morphological Diffusion (10 steps) with GAN masks after morphological closing	15.53

5.4.2 Extensions in the WordStylist model

In this section we will present the qualitative results after using the DDIM [68], f-PNDM [69] and s-PNDM [69] sampling methods to the WordStylist model [31]. Figures 5.4.4, 5.4.6 and 5.4.5 illustrate these results for a number of timesteps that varies from 20 to 200. We can clearly see that by introducing these algorithms we can radically reduce the need of many sampling steps, since even 50 steps can be adequate to maintain the quality of the generated images. Using less than 50 sampling steps has a negative effect on the sampling quality though, as shown in the first row of the results for the 3 sampling methods.

Hence, we address one the main limitations of the WordStylist model concerning its computational complexity, since the 600 sampling steps which were needed for generation are no longer needed and the quality of results remains almost the same.

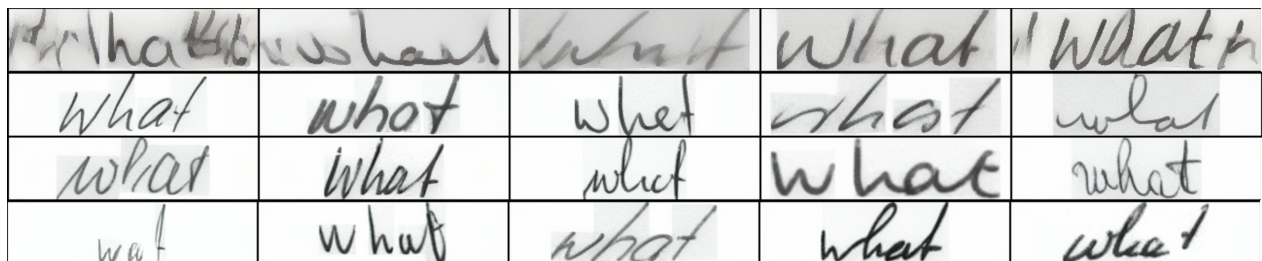


Figure 5.4.4: WordStylist Results with DDIM Sampling. Each row contains 5 results of the same word ('what') generated with 5 random writing styles with different number sampling steps a) with 20 sampling steps, b) with 50 sampling steps, c) with 100 sampling steps, d) with 200 sampling steps.

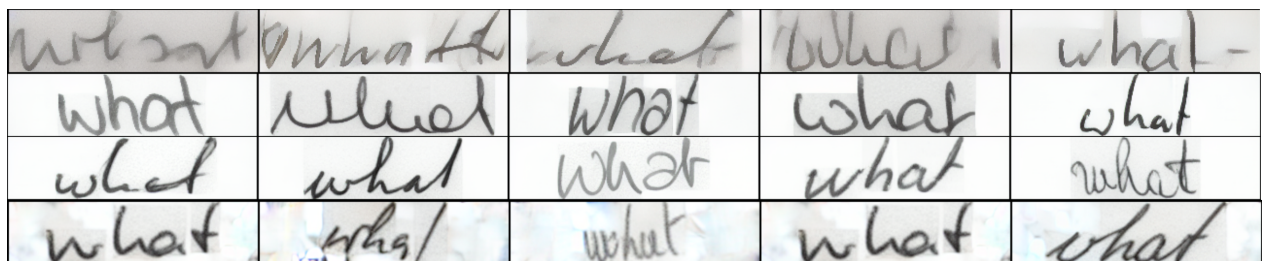


Figure 5.4.5: WordStylist Results with f-PNDM Sampling. Each row contains 5 results of the same word ('what') generated with 5 random writing styles with different number sampling steps a) with 20 sampling steps, b) with 50 sampling steps, c) with 100 sampling steps, d) with 200 sampling steps.

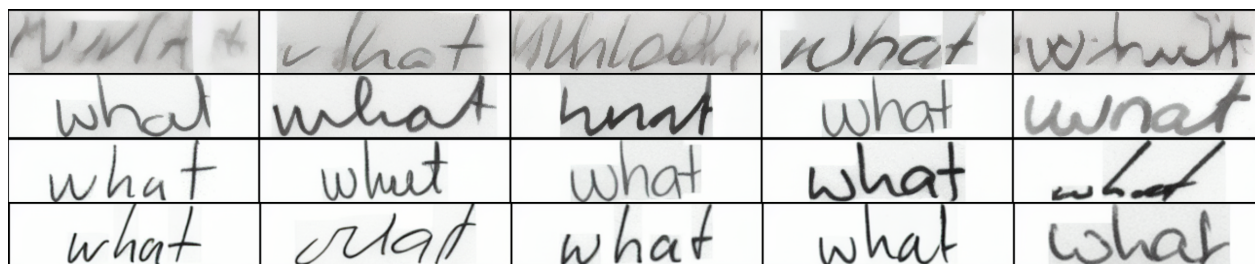


Figure 5.4.6: WordStylist Results with s-PNDM Sampling. Each row contains 5 results of the same word ('what') generated with 5 random writing styles with different number sampling steps a) with 20 sampling steps, b) with 50 sampling steps, c) with 100 sampling steps, d) with 200 sampling steps.

Additionally, Figure 5.4.7 illustrates the sampled images with 100 steps with default sampling (top row) and the 3 sampling methods (bottom 3 rows) and indicates the impact that these algorithms make, since default sampling with 100 steps leads to almost pure noise.

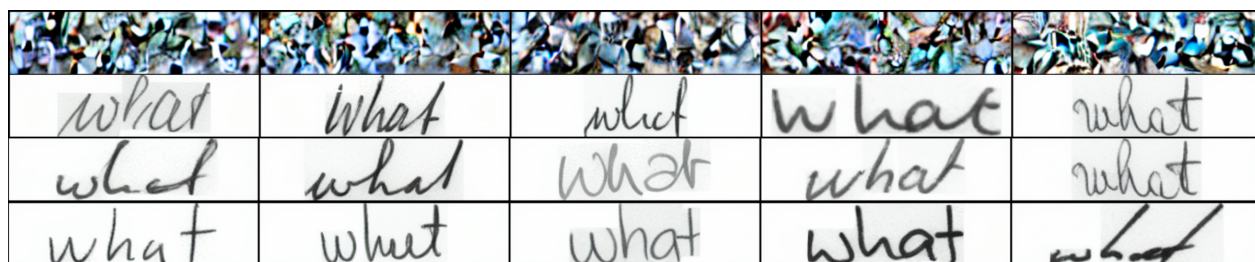


Figure 5.4.7: Comparison of WordStylist Sampling Results with 100 steps. Each row contains 5 results of the same word ('what') generated with 5 random writing styles with different sampling algorithm a) with default sampling, b) with DDIM sampling, c) with f-PNDM sampling, d) with s-PNDM sampling.

5.4.3 Morphological Diffusion for Handwritten-Text Generation

Qualitative Results

In this section, we delve into a comprehensive examination of the qualitative results derived from our experiments. Beyond the quantitative metrics and statistical analyses, it's pivotal to assess the quality and characteristics of the outcomes in a more descriptive and interpretative manner.

We initiate this examination by presenting visually reconstructed samples extracted from the IAM dataset. Figure 5.4.8 clearly illustrates the model's adeptness at encapsulating both the text conditions and the individual writing styles associated with each handwritten word. Despite being trained for a mere 250 epochs and 30 timesteps, the model exhibits commendable performance across a diverse array of writing styles and words, underscoring its versatility and effectiveness.

four	the	about	of	undo	was
the	which	not	four	to	oppose
to	would	he	power	then	of
but	in	to	the	face	is
in	the	we	the	are	cut
read	Britain	well,	body	the	is
of	less	ever	of	sound	here
love	she	men	need	today	too
of	she	Cyprus	Spanish	to	going
rapid	that	enable	we	The	than
the	his	and	the	by	free
the	his	very	well	and	rough
proms	being	labors	been	at	is
with	and	ing	enable	barley	started
the	and	the	and	six	the
doctor	and	practice	at	is	part
stitch	she	only	Africa	check	Both
and	for	of	to	up	the
said	the	Mauro	on	ful	leap
do	three	he	This	olast	the
last	end	she	best	script	proms
rooms	the	home	that	ceramic	of
been	three	being	and	was	just
all	his	thread	classes	hid	how
already	you	agency	city	lost	new
to	in	your	first	then	the
Burdett	little	face	eight	came	the
just	working	which	all	days	the
work	output	the	than	to	when
who	last	and	NEWS	of	of

Figure 5.4.8: Qualitative results for reconstructed IAM samples.

Subsequently, to provide a more nuanced assessment of our model’s capabilities, we will showcase examples of images generated for Out-Of-Vocabulary (OOV) words. This aims to demonstrate the model’s proficiency in grasping the dataset’s distribution without being explicitly attuned to the specific instances of the IAM dataset.

Unlike the aforementioned reconstruction task, we are not starting with a degraded image for the sampling process in this instance. To circumvent this challenge, we introduce a random degraded sample from the original IAM dataset as the initial image for each case. This addition infuses an element of randomization into the sampling process, enhancing the evaluation’s comprehensiveness.

Figures 5.4.9, 5.4.10, 5.4.11 depict qualitative results for the OOV words "dance", "troll", and "waist". In each figure, six images are displayed: the first three (on the left) represent the initial images, the degraded instances of which, are used as the starting point for sampling, and the latter three (on the right) represent the model’s generated images for the corresponding OOV words, maintaining consistency in style.

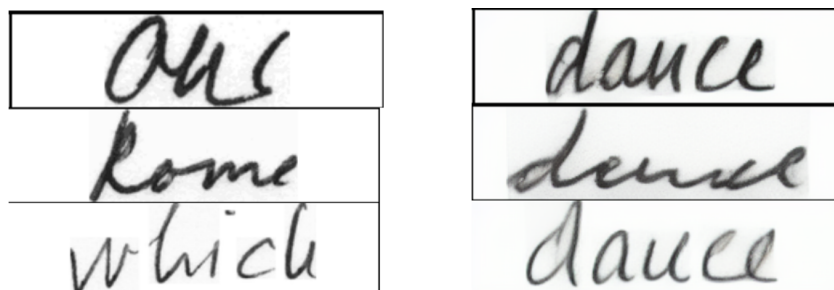


Figure 5.4.9: Out-Of-Vocabulary results for "dance"

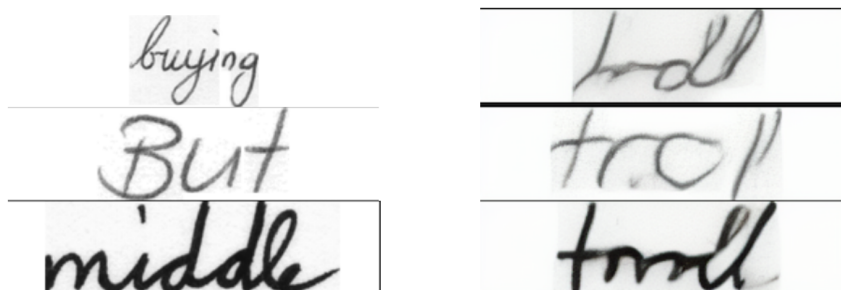


Figure 5.4.10: Out-Of-Vocabulary results for "troll"

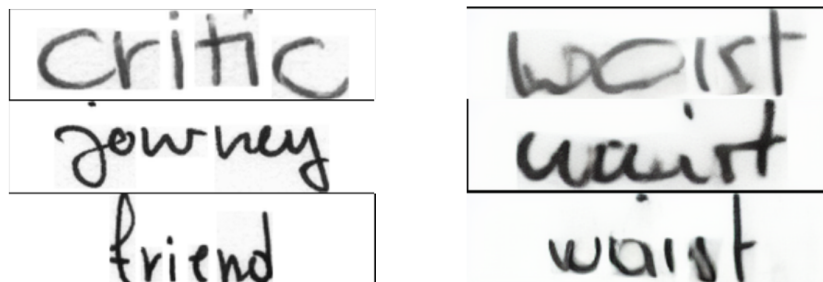


Figure 5.4.11: Out-Of-Vocabulary results for "waist"

While our model has demonstrated commendable performance in many instances, including out-of-vocabulary words, there are certain scenarios where its output is not as robust, attributable to the intricacies of Handwriting Text Generation (HTG) and inherent constraints in our methodology. Figure 5.4.12 provides visual insights into such cases, featuring examples of the words "dance" (first row), "troll" (second row), and "waist" (third row) that are not up to the quality of previously showcased results.

The broad spectrum of writing styles (339 in total) and the existing imbalance within the IAM dataset concerning these styles occasionally result in suboptimal performance for specific styles. Despite this, it's noteworthy that the model still manages to encapsulate textual information effectively, indicating a degree of resilience and capability even amid these challenges.

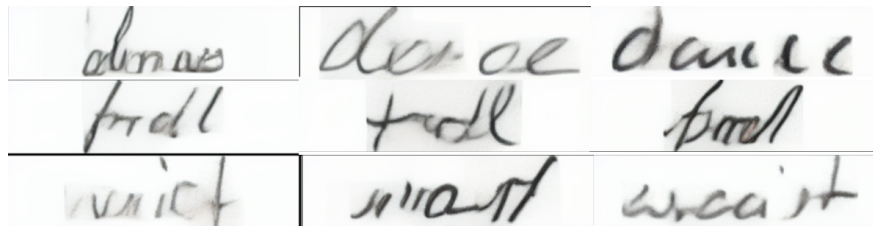


Figure 5.4.12: Out-Of-Vocabulary results of lower quality

Our concluding segment of qualitative results focuses on illustrating the style adaptation capabilities of our model. In Figure 5.4.13, we present outcomes for eight randomly selected styles applied to the in-vocabulary word "what". The results evidently highlight the model's proficiency in distinctly capturing and replicating each style. Despite each generated image representing the same word, there is a noticeable variance in appearance attributed to the diverse writing styles, underscoring the model's effectiveness in style adaptation.

It is essential to note that, akin to the previous tests with out-of-vocabulary words, we employed random degraded samples from the IAM dataset as the starting points for the image sampling process in these style adaptation evaluations.

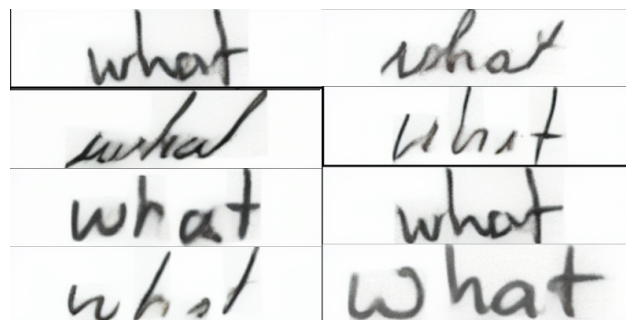


Figure 5.4.13: Results for different styles of "what"

Quantitative Results

We now turn our attention to the quantitative assessment of our model, complementing the earlier qualitative analysis. The initial focus of this evaluation is a comprehensive comparison of Fréchet Inception Distance (FID) scores to provide a numerical measure of the quality and diversity of images generated by our model. These scores will offer an objective basis for evaluating the realism and diversity of the generated handwriting images in a statistical context, establishing a foundation for further detailed analysis and insights.

Table 5.4 presents a comparison of the FID score of our model against three contemporary state-of-the-art methods: GANWriting [51], SmartPatch [30], and WordStylist [31]. Our model records an FID score of 27.85, showcasing its competitiveness when compared with the performance metrics of the models mentioned above.

Model	FID ↓
GANWriting [51]	29.94
SmartPatch [30]	22.55
WordStylist [31]	22.74
Ours	27.85

Table 5.4: FID score comparison for our model with state-of-the-art.

As mentioned in [31], although FID is a commonly employed metric for assessing generative models, its suitability can be questioned for tasks that don't deal with natural images akin to those in ImageNet [76], the dataset on which the network was originally trained.

Due to the aforementioned limitation of the FID metric, we extend our evaluation to assess the writing style adaptation of our model, employing the assessment framework outlined in [31]. Specifically, we will utilize a ResNet18 CNN [77] to categorize the handwriting styles of our generated samples. Given that our chosen classifier is pretrained on ImageNet [76], a preliminary fine-tuning on the IAM dataset will be conducted before proceeding with the classification. The outcomes of this analysis are detailed in table 5.5.

Test Set	Accuracy (%) ↑
GANWriting [51]	4.81
SmartPatch [30]	4.09
WordStylist [31]	70.6
Ours	26.7

Table 5.5: Style Adaptation comparison for our model with state-of-the-art.

The table reveals that our model's generated samples achieve an accuracy of 26.7%, significantly surpassing GANwriting [51] and SmartPatch [30], though falling short of the performance exhibited by WordStylist [31]. While 26.7% might initially appear modest, it is indicative of a commendable performance in emulating writing styles, especially considering the presence of 339 distinct styles and the notable imbalance in the IAM dataset concerning the representation of each specific style.

Chapter 6

Conclusion and Future Work

6.1	Conclusion	108
6.2	Future Work	108

6.1 Conclusion

In this thesis, we addressed the problem of Handwritten Text Generation using a latent morphological diffusion model. We initially presented the theoretical background in which our approach is founded in chapter 2, as well as the related work which constituted the inspiration for the architecture, the training and the functionalities of our models including cold diffusion for generalized diffusion processes [33], latent diffusion [32] and the state-of-the-art architectures for Handwritten Text Generation [31, 51, 30].

We addressed a wide variety of problems related to our proposed morphological diffusion process, which provided many limitations to our implementation. To begin with, this is one of the first, if not the first, attempt to define a non-linear diffusion process, which is considerably restrictive, since the properties of the dilation operation allowed a very specific number of timesteps depending on the image size. Additionally, the fully degraded images of both MNIST [62] and IAM [63] datasets (which were the main datasets we experimented with) tended to be an entirely "white" image, thus influencing negatively the diversity of our models' generated samples. Furthermore, contrary to other works with generalized deterministic diffusions, we provided an approach for conditional generation using deterministic degradations by training a conditional Generative Adversarial Network to transform pure noise to an image which matches the degraded images' distribution.

After experimenting in MNIST and CIFAR-10 datasets and comparing our results with the cold diffusion work [33], we performed our main experiments with the IAM dataset for the task of Handwritten Text Generation. We originally provided some optimizations regarding the sampling process of the WordStylist model [31], using the sampling methods of Denoising Diffusion Implicit Models [68] and Pseudo Numerical methods for Diffusion Models [69], resulting in a substantial decrease of the sampling time needed, as we maintained a similar quality of generated images after reducing the sampling steps from 600 to just 50 (12× speed-up). This way, we addressed one of the main limitations of this work without affecting the quality and the diversity of the generated samples.

Using the morphological diffusion framework described above, we intended to further reduce the computational needs for the task of Handwritten Text Generation by introducing a model which is trained for 30 sampling steps. Since, in this case, both training and sampling require 30 steps (contrary to the previous method which required 50 sampling steps but the training was conducted for 1000 steps), hence reducing not only the time and resources needed for efficient sampling but also for adequate training (250 epochs). As discussed in chapter 5, our model achieved decent results and was able to even generalize, as it was capable of generating out-of-vocabulary words. Even though its performance did not reach the level of WordStylist [31], we provided a proof-of-concept that, considering the recent advancements on generalized diffusions, selecting and adjusting a suitable diffusion process to a dataset may have a positive impact on the performance of the model. In our case, this impact was a trade-off between image quality and sampling complexity, but, since it is a newly discovered area of research, further investigation on such types of diffusion processes may lead to even better results.

6.2 Future Work

Despite demonstrating a promising performance, there are still various limitations to our approach which may force future research on the area of generalized diffusions and Handwritten Text Generation. The following bullets present some possible future areas of research to address the current limitations of our approach.

- **Diffusion Process:** During our tests, a morphological diffusion method was applied where degradation at each timestep was consistent, unaffected by the timestep's sequence. We utilized a 3×3 square structuring element to dilate image batches at each phase. This led to an intense diffusion process, compromising the model's data distribution capture efficacy. In contrast to this, conventional diffusion methods intensify the degradation with each increasing timestep. Adapting and optimizing a refined degradation schedule that suits non-linear degradation functions stands as a potential area for development.
- **Sampling Algorithm:** As evident from the study [33], the sampling algorithm employed in our research is grounded in the characteristics of linear degradation functions. Consequently, an area for

subsequent research involves enhancing this algorithm to accommodate non-linear degradation functions effectively.

- **Adding a pix2pix model to improve the quality of the results:** Following the reduced quality of our generated samples, one prospective enhancement could involve the incorporation of a pix2pix model [57]. This model could be configured to refine the output images from our diffusion model, enhancing their quality. This approach would mirror the methodology delineated in section chapter 4, with a focus not on generating initial image samples but on applying a super-resolution task to the images already produced. Given that pix2pix employs a u-net for generator, its integration would be seamless, requiring an additional step in the sampling process to substantially improve the quality of the generated images.

Appendix A

Bibliography

- [1] Goodfellow, I., Bengio, Y., and Courville, A. *Deep Learning*. MIT Press, 2016.
- [2] Bengio, Y., Simard, P., and Frasconi, P. “Learning long-term dependencies with gradient descent is difficult”. In: *IEEE transactions on neural networks* 5.2 (1994), pp. 157–166.
- [3] Hochreiter, S. and Schmidhuber, J. “Long short-term memory”. In: *Neural computation* 9.8 (1997), pp. 1735–1780.
- [4] Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. “Learning phrase representations using RNN encoder-decoder for statistical machine translation”. In: *arXiv preprint arXiv:1406.1078* (2014).
- [5] Kingma, D. P. and Welling, M. “Auto-encoding variational bayes”. In: *arXiv preprint arXiv:1312.6114* (2013).
- [6] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. “Generative adversarial nets”. In: *Advances in neural information processing systems* 27 (2014).
- [7] Matheron, G. *Random Sets and Integral Geometry*. Wiley, 1975.
- [8] Serra, J. *Image analysis and mathematical morphology*. 1982.
- [9] Kimori, Y., Hikino, K., Nishimura, M., and Mano, S. “Quantifying morphological features of actin cytoskeletal filaments in plant cells based on mathematical morphology”. In: *Journal of theoretical biology* 389 (2016), pp. 123–131.
- [10] Miri, M. S. and Mahloojifar, A. “Retinal image analysis using curvelet transform and multistructure elements morphology by reconstruction”. In: *IEEE Transactions on Biomedical Engineering* 58.5 (2010), pp. 1183–1192.
- [11] Masci, J., Angulo, J., and Schmidhuber, J. “A learning framework for morphological operators using counter-harmonic mean”. In: *Mathematical Morphology and Its Applications to Signal and Image Processing: 11th International Symposium, ISMM 2013, Uppsala, Sweden, May 27-29, 2013. Proceedings 11*. Springer. 2013, pp. 329–340.
- [12] Borra, S. R., Reddy, G. J., and Reddy, E. S. “Classification of fingerprint images with the aid of morphological operation and AGNN classifier”. In: *Applied computing and informatics* 14.2 (2018), pp. 166–176.
- [13] Van Droogenbroeck, M. and Talbot, H. “Fast computation of morphological operations with arbitrary structuring elements”. In: *Pattern recognition letters* 17.14 (1996), pp. 1451–1460.
- [14] Meyer, F. and Beucher, S. “Morphological segmentation”. In: *Journal of visual communication and image representation* 1.1 (1990), pp. 21–46.
- [15] Kalshetti, P., Bundele, M., Rahangdale, P., Jangra, D., Chattopadhyay, C., Harit, G., and Elhence, A. “An interactive medical image segmentation framework using iterative refinement”. In: *Computers in biology and medicine* 83 (2017), pp. 22–33.
- [16] Yu-Qian, Z., Wei-Hua, G., Zhen-Cheng, C., Jing-Tian, T., and Ling-Yun, L. “Medical images edge detection based on mathematical morphology”. In: *2005 IEEE engineering in medicine and biology 27th annual conference*. IEEE. 2006, pp. 6492–6495.

- [17] Mellouli, D., Hamdani, T. M., Sanchez-Medina, J. J., Ayed, M. B., and Alimi, A. M. “Morphological convolutional neural network architecture for digit recognition”. In: *IEEE transactions on neural networks and learning systems* 30.9 (2019), pp. 2876–2885.
- [18] Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., and Ganguli, S. “Deep unsupervised learning using nonequilibrium thermodynamics”. In: (2015), pp. 2256–2265.
- [19] Vincent, P., Larochelle, H., Bengio, Y., and Manzagol, P.-A. “Extracting and composing robust features with denoising autoencoders”. In: (2008), pp. 1096–1103.
- [20] Ho, J., Jain, A., and Abbeel, P. “Denoising diffusion probabilistic models”. In: *Advances in neural information processing systems* 33 (2020), pp. 6840–6851.
- [21] Nichol, A. Q. and Dhariwal, P. “Improved denoising diffusion probabilistic models”. In: (2021), pp. 8162–8171.
- [22] Plamondon, R. and Srihari, S. N. “Online and off-line handwriting recognition: a comprehensive survey”. In: *IEEE Transactions on pattern analysis and machine intelligence* 22.1 (2000), pp. 63–84.
- [23] Graves, A. “Generating sequences with recurrent neural networks”. In: *arXiv preprint arXiv:1308.0850* (2013).
- [24] Chung, J., Kastner, K., Dinh, L., Goel, K., Courville, A. C., and Bengio, Y. “A recurrent latent variable model for sequential data”. In: *Advances in neural information processing systems* 28 (2015).
- [25] Aksan, E., Pece, F., and Hilliges, O. “Deepwriting: Making digital ink editable via deep generative modeling”. In: *Proceedings of the 2018 CHI conference on human factors in computing systems*. 2018, pp. 1–14.
- [26] Aksan, E. and Hilliges, O. “STCN: Stochastic temporal convolutional networks”. In: *arXiv preprint arXiv:1902.06568* (2019).
- [27] Vinciarelli, A. “A survey on off-line cursive word recognition”. In: *Pattern recognition* 35.7 (2002), pp. 1433–1446.
- [28] Fogel, S., Averbuch-Elor, H., Cohen, S., Mazor, S., and Litman, R. “Scrabblegan: Semi-supervised varying length handwritten text generation”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 4324–4333.
- [29] Davis, B., Tensmeyer, C., Price, B., Wigington, C., Morse, B., and Jain, R. “Text and style conditioned gan for generation of offline handwriting lines. arXiv: 200900678”. In: (2020).
- [30] Mattick, A., Mayr, M., Seuret, M., Maier, A., and Christlein, V. “Smartpatch: Improving handwritten word imitation with patch discriminators”. In: *International Conference on Document Analysis and Recognition*. Springer. 2021, pp. 268–283.
- [31] Nikolaidou, K., Retsinas, G., Christlein, V., Seuret, M., Sfikas, G., Smith, E. B., Mokayed, H., and Liwicki, M. “WordStylist: Styled Verbatim Handwritten Text Generation with Latent Diffusion Models”. In: *arXiv preprint arXiv:2303.16576* (2023).
- [32] Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. “High-resolution image synthesis with latent diffusion models”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, pp. 10684–10695.
- [33] Bansal, A., Borgnia, E., Chu, H.-M., Li, J. S., Kazemi, H., Huang, F., Goldblum, M., Geiping, J., and Goldstein, T. “Cold diffusion: Inverting arbitrary image transforms without noise”. In: *arXiv preprint arXiv:2208.09392* (2022).
- [34] Romano, Y., Elad, M., and Milanfar, P. “The little engine that could: Regularization by denoising (RED)”. In: *SIAM Journal on Imaging Sciences* 10.4 (2017), pp. 1804–1844.
- [35] Metzler, C., Mousavi, A., and Baraniuk, R. “Learned D-AMP: Principled neural network based compressive image recovery”. In: *Advances in neural information processing systems* 30 (2017).
- [36] Whang, J., Delbracio, M., Talebi, H., Saharia, C., Dimakis, A. G., and Milanfar, P. “Deblurring via stochastic refinement”. In: (2022), pp. 16293–16303.
- [37] Kawar, B., Vaksman, G., and Elad, M. “Stochastic image denoising by sampling from the posterior distribution”. In: (2021), pp. 1866–1875.
- [38] Saharia, C., Ho, J., Chan, W., Salimans, T., Fleet, D. J., and Norouzi, M. “Image super-resolution via iterative refinement”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45.4 (2022), pp. 4713–4726.
- [39] Ramesh, A., Pavlov, M., Goh, G., Gray, S., Voss, C., Radford, A., Chen, M., and Sutskever, I. “Zero-shot text-to-image generation”. In: *International Conference on Machine Learning*. PMLR. 2021, pp. 8821–8831.

-
- [40] Razavi, A., Van den Oord, A., and Vinyals, O. “Generating diverse high-fidelity images with vq-vae-2”. In: *Advances in neural information processing systems* 32 (2019).
- [41] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. “Generative adversarial networks”. In: *Communications of the ACM* 63.11 (2020), pp. 139–144.
- [42] Brock, A., Donahue, J., and Simonyan, K. “Large scale GAN training for high fidelity natural image synthesis”. In: *arXiv preprint arXiv:1809.11096* (2018).
- [43] Karras, T., Laine, S., and Aila, T. “A style-based generator architecture for generative adversarial networks”. In: (2019), pp. 4401–4410.
- [44] Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. “Score-based generative modeling through stochastic differential equations”. In: *arXiv preprint arXiv:2011.13456* (2020).
- [45] Chen, N., Zhang, Y., Zen, H., Weiss, R. J., Norouzi, M., and Chan, W. “Wavegrad: Estimating gradients for waveform generation”. In: *arXiv preprint arXiv:2009.00713* (2020).
- [46] Kingma, D., Salimans, T., Poole, B., and Ho, J. “Variational diffusion models”. In: *Advances in neural information processing systems* 34 (2021), pp. 21696–21707.
- [47] Kong, Z., Ping, W., Huang, J., Zhao, K., and Catanzaro, B. “Diffwave: A versatile diffusion model for audio synthesis”. In: *arXiv preprint arXiv:2009.09761* (2020).
- [48] Dai, B. and Wipf, D. “Diagnosing and enhancing VAE models”. In: *arXiv preprint arXiv:1903.05789* (2019).
- [49] Esser, P., Rombach, R., and Ommer, B. “Taming transformers for high-resolution image synthesis”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2021, pp. 12873–12883.
- [50] Van Den Oord, A., Vinyals, O., et al. “Neural discrete representation learning”. In: *Advances in neural information processing systems* 30 (2017).
- [51] Kang, L., Riba, P., Wang, Y., Rusinol, M., Fornés, A., and Villegas, M. “GANwriting: content-conditioned generation of styled handwritten word images”. In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIII 16*. Springer. 2020, pp. 273–289.
- [52] Huang, X., Liu, M.-Y., Belongie, S., and Kautz, J. “Multimodal unsupervised image-to-image translation”. In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, pp. 172–189.
- [53] Huang, X. and Belongie, S. “Arbitrary style transfer in real-time with adaptive instance normalization”. In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 1501–1510.
- [54] Simonyan, K. and Zisserman, A. “Very deep convolutional networks for large-scale image recognition”. In: *arXiv preprint arXiv:1409.1556* (2014).
- [55] Ioffe, S. and Szegedy, C. “Batch normalization: Accelerating deep network training by reducing internal covariate shift”. In: *International conference on machine learning*. pmlr. 2015, pp. 448–456.
- [56] Chung, J., Gulcehre, C., Cho, K., and Bengio, Y. “Empirical evaluation of gated recurrent neural networks on sequence modeling”. In: *arXiv preprint arXiv:1412.3555* (2014).
- [57] Isola, P., Zhu, J.-Y., Zhou, T., and Efros, A. A. “Image-to-image translation with conditional adversarial networks”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 1125–1134.
- [58] Michael, J., Labahn, R., Grüning, T., and Zöllner, J. “Evaluating sequence-to-sequence models for handwritten text recognition”. In: *2019 International Conference on Document Analysis and Recognition (ICDAR)*. IEEE. 2019, pp. 1286–1293.
- [59] Dhariwal, P. and Nichol, A. “Diffusion models beat gans on image synthesis”. In: *Advances in neural information processing systems* 34 (2021), pp. 8780–8794.
- [60] Ronneberger, O., Fischer, P., and Brox, T. “U-net: Convolutional networks for biomedical image segmentation”. In: *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5–9, 2015, Proceedings, Part III 18*. Springer. 2015, pp. 234–241.
- [61] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. “Attention is all you need”. In: *Advances in neural information processing systems* 30 (2017).
- [62] LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. “Gradient-based learning applied to document recognition”. In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324.
-

- [63] Marti, U.-V. and Bunke, H. “The IAM-database: an English sentence database for offline handwriting recognition”. In: *International Journal on Document Analysis and Recognition* 5 (2002), pp. 39–46.
- [64] Krizhevsky, A., Hinton, G., et al. “Learning multiple layers of features from tiny images”. In: (2009).
- [65] Rombach, R., Esser, P., and Ommer, B. “Network-to-network translation with conditional invertible neural networks”. In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 2784–2797.
- [66] Yan, W., Zhang, Y., Abbeel, P., and Srinivas, A. “Videogpt: Video generation using vq-vae and transformers”. In: *arXiv preprint arXiv:2104.10157* (2021).
- [67] Yu, J., Li, X., Koh, J. Y., Zhang, H., Pang, R., Qin, J., Ku, A., Xu, Y., Baldrige, J., and Wu, Y. “Vector-quantized image modeling with improved vqgan”. In: *arXiv preprint arXiv:2110.04627* (2021).
- [68] Song, J., Meng, C., and Ermon, S. “Denosing diffusion implicit models”. In: *arXiv preprint arXiv:2010.02502* (2020).
- [69] Liu, L., Ren, Y., Lin, Z., and Zhao, Z. “Pseudo numerical methods for diffusion models on manifolds”. In: *arXiv preprint arXiv:2202.09778* (2022).
- [70] Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. “Gans trained by a two time-scale update rule converge to a local nash equilibrium”. In: *Advances in neural information processing systems* 30 (2017).
- [71] Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., and Chen, X. “Improved techniques for training gans”. In: *Advances in neural information processing systems* 29 (2016).
- [72] Wang, Z., Bovik, A. C., Sheikh, H. R., and Simoncelli, E. P. “Image quality assessment: from error visibility to structural similarity”. In: *IEEE transactions on image processing* 13.4 (2004), pp. 600–612.
- [73] Kingma, D. P. and Ba, J. “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980* (2014).
- [74] Loshchilov, I. and Hutter, F. “Decoupled weight decay regularization”. In: *arXiv preprint arXiv:1711.05101* (2017).
- [75] Mirza, M. and Osindero, S. “Conditional generative adversarial nets”. In: *arXiv preprint arXiv:1411.1784* (2014).
- [76] Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. “Imagenet: A large-scale hierarchical image database”. In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee. 2009, pp. 248–255.
- [77] He, K., Zhang, X., Ren, S., and Sun, J. “Deep residual learning for image recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.
- [78] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. “Dropout: a simple way to prevent neural networks from overfitting”. In: *The journal of machine learning research* 15.1 (2014), pp. 1929–1958.
- [79] Krizhevsky, A., Sutskever, I., and Hinton, G. E. “Imagenet classification with deep convolutional neural networks”. In: *Advances in neural information processing systems* 25 (2012).
- [80] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. “Going deeper with convolutions”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 1–9.
- [81] Schuster, M. and Paliwal, K. K. “Bidirectional recurrent neural networks”. In: *IEEE transactions on Signal Processing* 45.11 (1997), pp. 2673–2681.
- [82] Rumelhart, D. E., Hinton, G. E., and Williams, R. J. “Learning representations by back-propagating errors”. In: *nature* 323.6088 (1986), pp. 533–536.
- [83] Duchi, J., Hazan, E., and Singer, Y. “Adaptive subgradient methods for online learning and stochastic optimization.” In: *Journal of machine learning research* 12.7 (2011).
- [84] Hinton, G., Srivastava, N., and Swersky, K. “Neural networks for machine learning lecture 6a overview of mini-batch gradient descent”. In: *Cited on* 14.8 (2012), p. 2.
- [85] Elfving, S., Uchibe, E., and Doya, K. “Sigmoid-weighted linear units for neural network function approximation in reinforcement learning”. In: *Neural networks* 107 (2018), pp. 3–11.
- [86] Hendrycks, D. and Gimpel, K. “Gaussian error linear units (gelus)”. In: *arXiv preprint arXiv:1606.08415* (2016).
- [87] Zhao, S., Song, J., and Ermon, S. “Infovae: Balancing learning and inference in variational autoencoders”. In: *Proceedings of the aaai conference on artificial intelligence*. Vol. 33. 01. 2019, pp. 5885–5892.

-
- [88] Nogueira, K., Chanussot, J., Dalla Mura, M., and Dos Santos, J. A. “An introduction to deep morphological networks”. In: *IEEE Access* 9 (2021), pp. 114308–114324.
- [89] Wu, Y. and He, K. “Group normalization”. In: (2018), pp. 3–19.
- [90] Perez, E., Strub, F., De Vries, H., Dumoulin, V., and Courville, A. “Film: Visual reasoning with a general conditioning layer”. In: 32.1 (2018).
- [91] Anderson, B. D. “Reverse-time diffusion equation models”. In: *Stochastic Processes and their Applications* 12.3 (1982), pp. 313–326.
- [92] Chen, R. T., Rubanova, Y., Bettencourt, J., and Duvenaud, D. K. “Neural ordinary differential equations”. In: *Advances in neural information processing systems* 31 (2018).
- [93] Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., and Chen, M. “Hierarchical text-conditional image generation with clip latents”. In: *arXiv preprint arXiv:2204.06125* (2022).
- [94] Kadkhodaie, Z. and Simoncelli, E. “Stochastic solutions for linear inverse problems using the prior implicit in a denoiser”. In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 13242–13254.
- [95] Liu, Z., Luo, P., Wang, X., and Tang, X. “Deep learning face attributes in the wild”. In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 3730–3738.
- [96] Zhang, R., Isola, P., Efros, A. A., Shechtman, E., and Wang, O. “The unreasonable effectiveness of deep features as a perceptual metric”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 586–595.
- [97] Dosovitskiy, A. and Brox, T. “Generating images with perceptual similarity metrics based on deep networks”. In: *Advances in neural information processing systems* 29 (2016).
- [98] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. “Bert: Pre-training of deep bidirectional transformers for language understanding”. In: *arXiv preprint arXiv:1810.04805* (2018).
- [99] Ho, J. and Salimans, T. “Classifier-free diffusion guidance”. In: *arXiv preprint arXiv:2207.12598* (2022).
- [100] Suvorov, R., Logacheva, E., Mashikhin, A., Remizova, A., Ashukha, A., Silvestrov, A., Kong, N., Goka, H., Park, K., and Lempitsky, V. “Resolution-robust large mask inpainting with fourier convolutions”. In: *Proceedings of the IEEE/CVF winter conference on applications of computer vision*. 2022, pp. 2149–2159.
- [101] Chi, L., Jiang, B., and Mu, Y. “Fast fourier convolution”. In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 4479–4488.
- [102] Zhao, S., Cui, J., Sheng, Y., Dong, Y., Liang, X., Chang, E. I., and Xu, Y. “Large scale image completion via co-modulated generative adversarial networks”. In: *arXiv preprint arXiv:2103.10428* (2021).
- [103] Ma, Y., Liu, X., Bai, S., Wang, L., Liu, A., Tao, D., and Hancock, E. R. “Regionwise generative adversarial image inpainting for large missing areas”. In: *IEEE transactions on cybernetics* (2022).
- [104] Yu, J., Lin, Z., Yang, J., Shen, X., Lu, X., and Huang, T. S. “Free-form image inpainting with gated convolution”. In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2019, pp. 4471–4480.
- [105] Nazeri, K., Ng, E., Joseph, T., Qureshi, F. Z., and Ebrahimi, M. “Edgeconnect: Generative image inpainting with adversarial edge learning”. In: *arXiv preprint arXiv:1901.00212* (2019).
- [106] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. “Rethinking the inception architecture for computer vision”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 2818–2826.
- [107] Mohamed, S. and Lakshminarayanan, B. “Learning in implicit generative models”. In: *arXiv preprint arXiv:1610.03483* (2016).
- [108] Sauer, T. *Numerical analysis*. Addison-Wesley Publishing Company, 2011.
- [109] Fréchet, M. “Sur la distance de deux lois de probabilité”. In: *Annales de l’ISUP*. Vol. 6. 3. 1957, pp. 183–198.
- [110] Vaserstein, L. N. “Markov processes over denumerable products of spaces, describing large systems of automata”. In: *Problemy Peredachi Informatsii* 5.3 (1969), pp. 64–72.
- [111] Wang, Z. *Rate-scalable foveated image and video communications*. The University of Texas at Austin, 2001.
- [112] Wang, Z. and Bovik, A. C. “A universal image quality index”. In: *IEEE signal processing letters* 9.3 (2002), pp. 81–84.
-

- [113] Liu, Z., Mao, H., Wu, C.-Y., Feichtenhofer, C., Darrell, T., and Xie, S. “A convnet for the 2020s”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, pp. 11976–11986.