TÉCNICO LISBOA

Leveraging Deep Unsupervised Models Towards Learning Robust Multimodal Representations

Afonso Cruz Bandeira Fernandes

Thesis to obtain the Master of Science Degree in

Computer Science and Engineering

Supervisors: Prof. Francisco António Chaves Saraiva de Melo
Eng. Miguel Serras Vasco

Examination Committee

Chairperson: Prof. Pedro Tiago Gonçalves Monteiro
Supervisor: Prof. Francisco António Chaves Saraiva de Melo
Member of the Committee: Prof. Manuel Fernando Cabido Peres Lopes

December 2023

Declaration
I declare that this document is an original work of my own authorship and that it fulfills all the requirements of the Code of Conduct and Good Practices of the Universidade de Lisboa.

Acknowledgments

I would like to thank my family and close friends for their help during my most difficult experiences in life, a support that helped make this project possible.

I would also like to thank my dissertation supervisors Prof. Francisco Melo and Miguel Vasco for their understanding during this learning experience and also for sharing tips and knowledge that has helped make this Thesis possible.

To each and every one of you – Thank you

Abstract

Deep learning models have been increasingly used in commercial applications, from detecting hate speech in the internet to diagnosing conditions in medical patients, affecting almost all facets of our lives. However, existing architectures may not perform well with noisy data and are vulnerable to adversarial attacks, where small perturbations in the input data can lead models to perform far worse than would be expected, which can lead to catastrophic consequences. New research on contrastive learning algorithms has powered the development of generative models and unsupervised learning techniques which focus on learning meaningful representations from datasets with multiple modalities. While in theory, these multimodal models should be more robust to single-modality perturbations, previous works have shown that this assumption does not hold in practice. In this work, we propose several new architectures that leverage multiple data modalities and principles from denoising models and odd-one-out learning to increase robustness in scenarios where one of its input modalities has been perturbed by adversarial attacks or noise. Towards evaluating the robustness of these architectures, we test several models on multimodal datasets with and without perturbations, and in scenarios where we remove one of its modalities.

Keywords

Deep Learning; Representation Learning; Unsupervised Learning; Contrastive Learning; Generative Models; Multimodal Learning; Adversarial Attacks.

Resumo

Os modelos de aprendizagem profunda têm sido cada vez mais utilizados em aplicações comerciais, desde a deteção de discursos de ódio na Internet até ao diagnóstico de doenças em pacientes médicos, tendo impacto em quase todas as facetas das nossas vidas. No entanto, as arquitecturas existentes podem não funcionar corretamente com dados com ruído e são vulneráveis a ataques adversariais, em que pequenas perturbações nos dados podem levar os modelos a ter um desempenho muito pior do que o esperado, sendo que isto pode levar a consequências catastróficas. Nova investigação sobre algoritmos de aprendizagem contrastiva tem impulsionado o desenvolvimento de modelos generativos e de técnicas de aprendizagem não supervisionada que têm o propósito de aprender representações significativas a partir de conjuntos de dados com múltiplas modalidades. Em teoria, estes modelos multimodais deveriam ser mais robustos contra perturbações numa só modalidade, mas trabalhos anteriores demonstram que isto não se verifica na prática. Neste trabalho, propomos várias novas arquitetura que usam dados multimodais, combinado com princípios de modelos treinados para remover ruído e para detetar elementos que não pertencem a um conjunto, que são é robustas em cenários no qual uma das suas modalidades foi perturbada por ataques adversariais ou ruído. Para avaliar a robustez destas arquiteturas, testamos vários modelos em conjuntos de dados multimodais com e sem perturbações, e em cenários em que removemos uma das suas modalidades.

Palavras Chave

Aprendizagem Profunda; Aprendizagem de Representações; Aprendizagem Não Supervisionada; Aprendizagem Contrastiva; Modelos Generativos; Aprendizagem Multimodal; Ataques Adversariais.

Contents

1 Introduction **1**

1.1 Research Objectives . . . 3

1.2 Contributions . . . 3

1.3 Document Outline . . . 4

2 Background Work **5**

2.1 Deep Learning . . . 5

2.1.1 Convolutional Neural Network . . . 6

2.2 Representation Learning . . . 8

2.2.1 Unsupervised Learning . . . 8

2.2.1.A Auto-Encoder . . . 9

2.2.1.B Denoising Auto-Encoder . . . 10

2.2.2 Contrastive Learning . . . 11

2.2.2.A SimCLR . . . 12

2.2.2.B Contrastive Predictive Coding . . . 13

2.3 Generative Models . . . 15

2.3.1 Variational Auto-Encoder . . . 16

2.3.2 Generative Adversarial Network . . . 18

3 Related Work **19**

3.1 Robustness of Deep Learning Models . . . 19

3.1.1 Robustness to Adversarial Attacks . . . 20

3.1.1.A White-Box Adversarial Attacks . . . 22

A – FGSM . . . 22

B – BIM . . . 23

C – Projected Gradient Descent . . . 23

D – C&W Attack . . . 24

3.1.1.B Black-box Attacks . . . 26

A – WGAN for Adversarial Sample Generation . . . 26

B – Transferability of Adversarial Attacks . . . 27
3.1.2 Defenses Against Adversarial Attacks . . . 28
3.1.2.A Defensive Distillation . . . 28
3.1.2.B Defensive Generative Models . . . 29
3.2 Multimodal Deep Learning . . . 29
3.2.1 MVAE . . . 31
3.2.2 GMC . . . 33
3.3 Robustness of Deep Multimodal Models . . . 34
3.3.1 Robustness to single-modality adversarial perturbations . . . 35
3.3.2 Defenses against single-modality adversarial attacks on multimodal models . . . 37
3.3.2.A Adversarially Robust Fusion Mechanism . . . 38
3.3.2.B MATCH . . . 39
4 Implementation and Methodology **40**
4.1 Model Architectures . . . 41
4.1.1 Models based on unimodal Auto-Encoders . . . 41
4.1.2 Models based on Multimodal Auto-Encoders . . . 42
4.1.3 Models based on the Geometric Multimodal Contrastive framework . . . 43
4.2 Datasets . . . 45
4.3 Noise and Adversarial Attacks . . . 46
5 Evaluation Experiments **47**
5.1 Experiments on the MHD dataset . . . 48
5.2 Experiments on the MNIST-SVHN dataset . . . 54
6 Conclusions and Future Work **60**
6.1 Conclusions . . . 61
6.2 Limitations and Future Work . . . 62
Bibliography **62**
7 Algorithms Hyperparameters **72**

List of Figures

1.1 Overview of the risks that DL algorithms face when being deployed in the real world. Figure taken from [1]. 2

2.1 Comparison between the traditional ML process (a), where a human must manually select and engineer useful features that are then provided to a shallow classifier, and a end-to-end approach (b), which leverages DL for a fully automated process. Figure from [2]. . . . 6

2.2 Illustration of the connectivity of an arbitrary CNN with two layers (a) and an arbitrary MLP with one layer (b), adapted from [3]. Each arrowed line represents a weight. As can be seen, even though the CNN has far less weights than the MLP, the receptive field of the g_3 unit also contains information from all input datapoints. 7

2.3 Examples of the outputs of max pooling and average pooling operations on a given input. Figure taken from [4]. 7

2.4 Illustration of several different types of DL strategies, where the red and blue circles represent datapoints of different classes and the lines between the circles represent the decision boundary between the classes. The grey circles represent unlabeled datapoints and the stripped circles represent datapoints which are initially unlabeled, but use label information at other stages of the training process. Figure from [5]. 8

2.5 Example of the results of a clustering algorithm (U-k-means) on a 6-cluster dataset with 50 noisy datapoints, after 28 iterations. Figure from [6]. 9

2.6 Overview of a general Convolutional AE architecture. Following a successful training process, the bottleneck layer, which has a significant lower dimension than the original image, should encode a meaningful latent space representation that allows the decoder NN to recreate an image similar to the original image that was given as input to the encoder NN. Figure from [7]. 10

2.7 Overview of the DAE architecture and training process. Random noise is mixed in with the original input data before being feed into the DAE and the model then has to remove the noise from its reconstructions in order to minimize the loss between its outputs and the original noise-less input data. Figure from [8]. 11

2.8 Example of the learning process using a triplet loss, where the distance between the anchor and the positive sample is minimized, while at the same time the distance between the anchor and the negative sample is maximized. Figure from [9]. 12

2.9 Overview of the SimCLR framework [10]. The contrastive loss minimizes the difference between images of the same element, while maximizing the difference between images of differences objects. Figure from [11]. 13

2.10 Overview of the CPC representation learning approach using audio as input. Figure from [12]. 14

2.11 Illustration from OpenCV [13], which shows the difference between an arbitrary generative model and an arbitrary discriminative model, trained to perform a simple binary classification task of distinguishing between images of cats and dogs. 15

2.12 Illustration of how the reparameterization trick separates the probabilistic and deterministic parts of the model, allowing for easier optimization. Figure from [14]. 16

2.13 Overview of the VAE architecture, where the latent variables $z = \mu + \sigma \odot \epsilon$. Figure from [15]. 17

2.14 Overview of the GAN architecture and training scheme, where the random input given to the generator is a sum of the latent space and random (generally Gaussian) noise. Figure from [16]. 18

3.1 Input images next to the respective relevance maps for a model trained to detect horses on images from the Pascal VOC 2007 dataset [17]. The model classifies images a) and c) as containing an horse due to the presence of a copyright watermark, while classifying images b) and d) as not containing an horse since the watermark is not present, while the images a) and b) contain a horse. Figure from [18]. 20

3.2 Example of an adversarial attack in a CV task. By adding a small amount of noise that is imperceptible to the human eye, the adversary can fool the NN into classifying a panda as a gibbon with high confidence. Figure adapted from [19]. 22

3.3 The C&W attack applied to images from the MNIST dataset, for each of the three possible distance metrics. The original images correspond to the diagonal, where the source and target classification correspond to the same digit label. Notice how the difference between each image is almost imperceptible. Figure adapted from [20]. 25

3.4 The architecture of the WGAN combined with an Inverter. In our notation, the critic $\mathcal{C}$ corresponds to the discriminator D and its parameters ω indicated in the figure corresponds to ϕ, i.e, $D_\phi = \mathcal{C}_\omega$. Figure from [21]. 27
3.5 Overview of the Defense-GAN defense mechanism. Figure from [22]. 29
3.6 The text-conditional convolutional GAN architecture that is based on the DC-GAN model, where both the generator and discriminator networks are conditioned on the text encoding $\varphi(t)$. Figure from [23]. 29
3.7 Visualization of the CLIP contrastive objective. After encoding the data, CLIP calculates a similarity matrix for the images and texts. Similarly to other contrastive methods, the objective is that the N true image-text pairs (positive pairs) score high in terms of similarity, while the other $N^2 - N$ possible combinations (negative pairs) have a low similarity. Figure from [24]. 30
3.8 Overview of several VAE-based architectures, including the MVAE. Figure taken from [25]. 31
3.9 Overview of the GMC framework instantiated with two modalities. Figure taken from [26]. 34
3.10 Overview of the robust fusion mechanism. Figure taken from [27]. 38
3.11 Overview of the MATCH detection pipeline. Figure taken from [28]. 39

4.1 Overview of the GMCWD framework instantiated with 2 modalities. Blue circles denote clean data, red circles noisy data and encodings, and green circles reconstructions and respective intermediate representations. 43
4.2 Overview of the DGMC architecture for 2 modalities. Blue circles denote clean data, red circles noisy data and encodings, green circles reconstructions and respective intermediate representations, and purple circles the encodings with the reconstructions as input. . 44
4.3 Overview of the RGMC architecture for 2 modalities. Blue circles denote clean modalities and red circles modalities under adversarial attack. 44
4.4 Distribution of digit labels in the MNIST-SVHN training dataset. 45
4.5 Distribution of digit labels in the MNIST-SVHN test dataset. 46

5.1 Adversarial examples generated by the FGSM adversarial attack with $\epsilon = 0.2$, targeting each modality. At this value of ϵ, the perturbations in many samples start to be highly visible to the human eye. 48
5.2 Comparison of the loss values of the GMC, GMCWD and DGMC architectures across the ten unsupervised training runs. The dark blue line equals the mean and shadowed blue area corresponds to the standard deviation. 48
5.3 Model accuracy in the MHD dataset on clean data with all modalities present. 49

5.4 Examples of a image of the digit zero perturbed by Gaussian noise. Standard deviation values for the Gaussian noise are: 0, 0.25, 0.5, 0.75 and 1. 50
5.5 Examples of a trajectory of the digit eight perturbed by Gaussian noise. Standard deviation values for the Gaussian noise are: 0, 0.25, 0.5, 0.75 and 1. 50
5.6 Model accuracy in the MNIST-SVHN dataset on clean data with all modalities present. . . 55
5.7 Examples of a image of the digit zero from the SVHN modality perturbed by Gaussian noise. Standard deviation values for the Gaussian noise are: 0, 0.25, 0.5, 0.75 and 1. . . 56

List of Tables

5.1 Accuracy results for models with all modalities present (None) and with one of the modalities missing on the MHD dataset. . . . 49

5.2 Accuracy results for models with Gaussian noise targeting each of the modalities on the MHD dataset. . . . 50

5.3 Accuracy results for models with the FGSM adversarial targeting each of the modalities on the MHD dataset. . . . 51

5.4 Accuracy results for models with the BIM adversarial attack targeting each of the modalities on the MHD dataset, where we set the number of steps $\mathcal{K} = 10$. . . . 52

5.5 Accuracy results for models with the PGD adversarial attack targeting each of the modalities on the MHD dataset, where we set the number of steps $\mathcal{K} = 10$. . . . 53

5.6 Comparison of accuracy of both the GMCWD and DGMC models on the MNIST-SVHN with missing modalities for different values of the weights for the reconstruction loss of each modality during model training. Hyperparameter set a) corresponds to both the MNIST and SVHN reconstruction loss weights being set to 0.5, b) to the MNIST weight being set to 0.5 and the SVHN weight being set to 0.05 and c) to the MNIST weight being set to 0.4 and the SVHN weight being set to 0.1. . . . 54

5.7 Accuracy results for models with all modalities present (None) and with one of the modalities missing on the MNIST-SVHN dataset. . . . 55

5.8 Accuracy results for models with Gaussian noise targeting each of the modalities on the MNIST-SVHN dataset . . . 56

5.9 Accuracy results for models with the FGSM adversarial targeting each of the modalities on the MNIST-SVHN dataset. . . . 57

5.10 Accuracy results for models with the BIM adversarial attack targeting each of the modalities on the MNIST-SVHN dataset, where we set the number of steps $\mathcal{K} = 10$. . . . 58

5.11 Accuracy results for models with the PGD adversarial attack targeting each of the modalities on the MNIST-SVHN dataset, where we set the number of steps $\mathcal{K} = 10$. . . . 59

5.12 Accuracy results for models with the PGD adversarial attack targeting each of the modalities on the MNIST-SVHN dataset, where we set the number of steps $\mathcal{K} = 10$, for the GMCWD and DGMC models trained with the hyperparameter sets c) and b), as indicated in 5.6. 59

7.1 Hyperparameters used to train the various models during the unsupervised learning setting. We trained all models using the SGD optimizer with momentum set to 0.9. NE is the number of epochs, BS is the batch size, LR is the learning rate, LD is the latent dimension, CD is the common dimension, λ_{img} is the image reconstruction loss weight, λ_{traj} is the trajectory reconstruction loss weight, λ_{mnist} is the MNIST reconstruction loss weight, λ_{svhn} is the SVHN reconstruction loss weight, τ is the InfoNCE temperature, λ_o is the odd-one-out network loss scale, σ is the noise standard deviation and ϵ is the FGSM adversarial attack parameter. 73

7.2 Hyperparameters used to train the various models during the supervised learning setting. We trained all models using the SGD optimizer with momentum set to 0.9. 74

Acronyms

AE	Auto-Encoder
API	Application Programming Interface
AR	Auto-Regressive
BCE	Binary Cross-Entropy
BIM	Basic Iterative Method
CCE	Categorical Cross-Entropy
CL	Contrastive Learning
CLIP	Contrastive Language-Image Pre-Training
CMDAE	Common Multimodal Denoising Auto-Encoder
CMDVAE	Common Multimodal Denoising Variational Auto-Encoder
CMVAE	Common Multimodal Variational Auto-Encoder
CNN	Convolutional Neural Network
CPC	Contrastive Predicting Coding
CV	Computer Vision
DAE	Denoising Auto-Encoder
DC-GAN	Deep Convolutional-Generative Adversarial Network
DL	Deep Learning
DGMC	Decoding Geometric Multimodal Contrastive
DNN	Deep Neural Network
ELBO	Evidence Lower Bound
FC	Fully-Connected
FGSM	Fast Gradient Sign Method
GAN	Generative Adversarial Network

GD	Gradient Descent
GMC	Geometric Multimodal Contrastive
GMCWD	Geometric Multimodal Contrastive with Decoders
KLD	Kullback-Leibler Divergence
L-BFGS	Limited-memory Broyden–Fletcher–Goldfarb–Shanno
MAE	Mean Absolute Error
MATCH	Multimodal Feature Consistency Check
MDAE	Multimodal Denoising Auto-Encoder
MHD	Multimodal Handwritten Digits
MI	Mutual Information
ML	Machine Learning
MLP	Multi-Layer Perceptron
MNIST	Modified National Institute of Standards and Technology
MSE	Mean Squared Error
MVAE	Multimodal Variational Auto-Encoder
NCE	Noise-Contrastive Estimation
NN	Neural Network
NT-Xent	Normalized Temperature-Scaled Cross Entropy
PCA	Principal Component Analysis
PGD	Projected Gradient Descent
RF	Random Forest
RGMC	Robust Geometric Multimodal Contrastive
SL	Supervised Learning
SOTA	State Of The Art
SSL	Self-Supervised Learning
SVHN	Street View House Numbers
SVM	Support Vector Machine
UL	Unsupervised Learning
VAE	Variational Auto-Encoder
VI	Variational Inference

VQ-VAE Vector Quantised-Variational Auto-Encoder

WGAN Wasserstein Generative Adversarial Network

1

Introduction

Contents

1.1 Research Objectives 3

1.2 Contributions 3

1.3 Document Outline 4

Deep Learning (DL) algorithms have seen increasing commercial uses in recent years [29], with Deep Neural Networks (DNNs) being applied in sensitive tasks such as in the medical area [30]. This increase in real-world applications of DL has led to an increase in research about the properties of these algorithms, which have traditionally been seen as black-boxes with little explainability [31], an important step in order to increase the trust the average person has about these models [30].

This research into the properties of DL has made clear that, despite their recent successes, they suffer from some limitations, such as the fact that they learn fairly discontinuous input-output mappings [32]. These properties make DL models vulnerable to noisy data [33] and to small input perturbations called adversarial examples, which can be exploited by malicious individuals to force models to make mistakes in their tasks [34]. This allows attackers to, for example, force an autonomous vehicle to crash [35] or make medical DL systems to make mistakes [1], issues that can potentially cost the lives of the people which these models were supposed to assist.

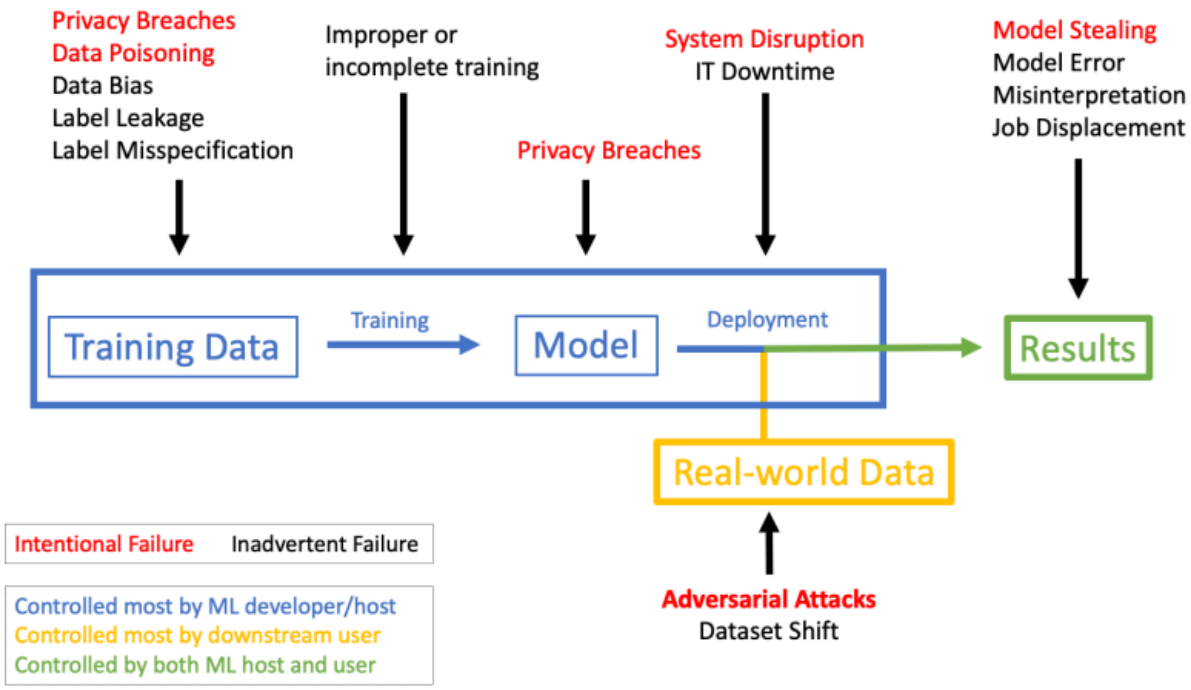

Figure 1.1: Overview of the risks that DL algorithms face when being deployed in the real world. Figure taken from [1].

A possible solution to increasing robustness of DL models against perturbed data has emerged from a distinct line of research, which focuses in developing multimodal models [36, 37]. Recently developed multimodal generative models, such as DALL-E [38, 39], and GLIDE [40], leverage Unsupervised Learning (UL) methods to learn meaningful latent representations from multiple data modalities [41].
In theory, the performance of such models should not suffer significantly when all but one of its modalities provide relevant information towards encoding meaningful representations. However, previous studies have shown that the performance of multimodal models can be overly dependent on only one of their input modalities [26, 42], and also that they do not guarantee increased robustness against both single-modality adversarial attacks [27, 28] and noisy data [43] vis-à-vis their unimodal counterparts. Nevertheless, recently proposed defense strategies [27, 28] have been shown to increase the robustness of multimodal models against adversarial perturbations by detecting which modality is under attack, while a complementary line of research introduced the Geometric Multimodal Contrastive (GMC) framework [26], which combines a two-level architecture with modality-specific encoders (along an encoder for the combination of all modalities) and a shared projection head with a multimodal contrastive cost function, that is capable of high performance with any modality missing due to the geometric alignment of the latent representations enforced by the loss function.

1.1 Research Objectives

In this dissertation, we aim to explore how multiple data modalities can be leveraged to increase robustness of deep learning models against both noisy data and adversarial attacks targeting a single modality, by combining ideas from previous works, such as:

1. The GMC framework, which maintains high performance with missing modalities;
2. Denoising architectures, where a model is trained to remove noise from its input;
3. A odd-one-out network, where a DNN is trained to identify the odd element from a set of related elements.

Taking into account these objectives, we try to tackle the following *research problem*: **Given an arbitrary number of data modalities, how can we improve the robustness of multimodal models against single-modality noise and adversarial perturbations, without decreasing their performance otherwise?**

1.2 Contributions

With regards to our research problem, the main contributions of this work are as follows:

- Introduction of the Geometric Multimodal Contrastive with Decoders (GMCWD) framework, which combines the GMC framework with an additional decoder for combination of all modalities, a deprojection head and a reconstruction loss function for each modality, where noise is injected in the input during training, encouraging more robust latent representations;
- Introduction of the Decoding Geometric Multimodal Contrastive (DGMC) architecture, an extension of the GMCWD framework, with additional modality-specific decoders and where, at test time, the model takes as input its own reconstructions of the modalities;
- Introduction of the Robust Geometric Multimodal Contrastive (RGMC) framework, which combines the GMC framework with an odd-one-out network and a Binary Cross-Entropy (BCE) loss function, where the output of the model is the product of the probability of each modality being unperturbed and its latent representation;
- Evaluation of the vulnerability of several different unimodal and multimodal models to random noise and adversarial perturbations targeting different modalities, where we demonstrate the robustness guarantees of the models we introduce during this work.

1.3 Document Outline

This dissertation is is organized as follows: chapter 2 provides a brief background on deep learning, representation learning through unsupervised learning and contrastive learning, and generative models, as well as some examples of relevant architectures. Chapter 3 focuses on previous work related to the study of robustness of deep learning models with regards to noise and adversarial attacks, the use of multimodal models in representation learning and, finally, previous analysis on the robustness of multimodal models. Chapter 4 introduces the architectures of the models developed for this work, as well as those of the baseline models, and a description of the training process. Chapter 5 presents a overview of the test scenarios and the obtained experimental results. Chapter 6 discusses some conclusions - as well as limitations - of this work and indicates possible future research.

2

Background Work

Contents

2.1 Deep Learning 5
2.2 Representation Learning 8
2.3 Generative Models 15

2.1 Deep Learning

Deep learning algorithms differ from traditional Machine Learning (ML) algorithms, like Support Vector Machines (SVMs) [44], since DL methods leverage large datasets to train models in a purely end-to-end fashion, as illustrated in Figure 2.1. This allows models such as deep Multi-Layer Perceptrons (MLPs) [3], a Neural Network (NN) comprised of multiple sequential Fully-Connected (FC) layers, to learn meaningful representations without human intervention and, as such, provide overall better generalization.

These advantages have made DL models the prime choice in many of the current State Of The Art (SOTA) representation learning algorithms. However, these algorithms result in more complex models whose outputs are very difficult to explain [31]. Such limitations also make it difficult to study the vulnerabilities of these models to certain inputs while providing little safeguard against them, since a model's

robustness does not necessarily scale with its complexity [32].

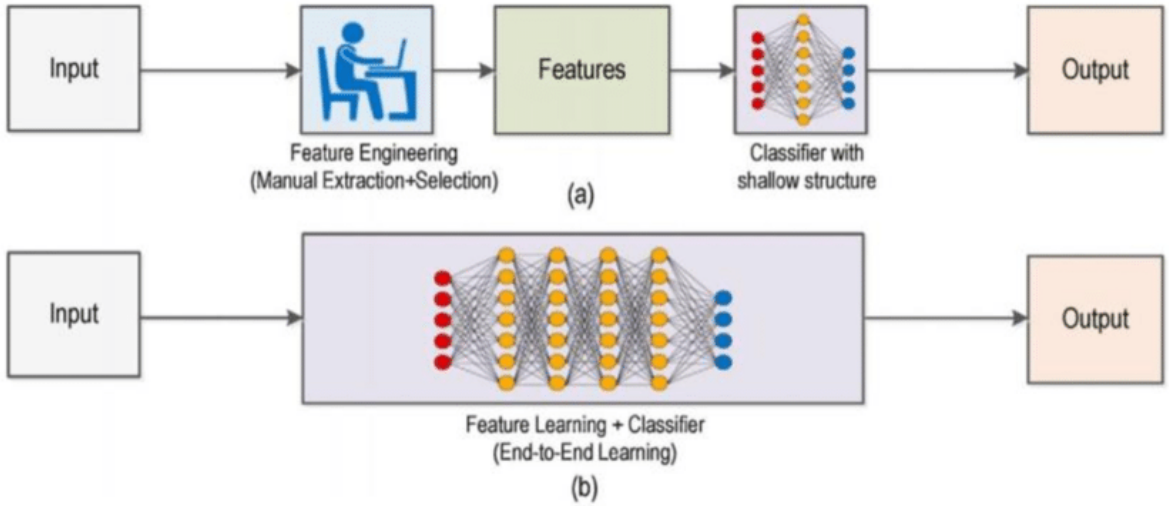

Figure 2.1: Comparison between the traditional ML process (a), where a human must manually select and engineer useful features that are then provided to a shallow classifier, and a end-to-end approach (b), which leverages DL for a fully automated process. Figure from [2].

2.1.1 Convolutional Neural Network

The convolutional neural network is a type of architecture that was inspired by the visual cortex in a cat's brain [45], and specializes in processing data with a grid-like topology. Convolutional Neural Networks (CNNs) are commonly applied to tasks related to Computer Vision (CV), since images can be thought of as 2-D grids of pixels [3].

This type of NN gets its name from the fact that it uses a kernel to apply a convolution operation across the input data, where a convolution operation using a two-dimensional kernel K applied on a two-dimensional image I is given by:

$$(I \star K)(i,j) = \sum_m \sum_n I(m,n)K(i-m, j-n) = \sum_m \sum_n I(i-m, j-n)K(m,n), \tag{2.1}$$

where $\star$ denotes the convolution operation, which, in its most general form, is simply an operation on two functions of real-valued arguments [3]. One of the main advantages of CNNs over other types of NNs, is that it uses far less parameters to process data. One characteristic behind this benefit is that a CNN has sparse weights, only requiring weights for the kernel K, which is usually far smaller than the input data. Another important characteristic is that it has tied weights, meaning that the value of a weight applied to a given input is tied to the value of a weight applied in a different location [3, 45]. This comes from the fact that the receptive field - which is the region of the input which affects a particular NN neuron - of the units in the final layers of a CNN is larger than that of the units in the first layers. This means that, even

though a CNN has very sparse direct connections, units in the final layers are still indirectly connected to most of the input image, as shown in Figure 2.2, allowing it to have similar performance to a MLP with a far larger number of parameters.

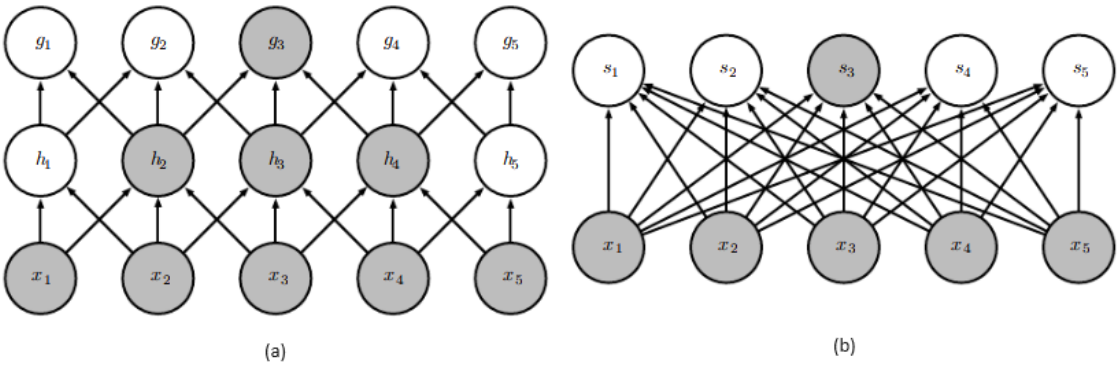

Figure 2.2: Illustration of the connectivity of an arbitrary CNN with two layers (a) and an arbitrary MLP with one layer (b), adapted from [3]. Each arrowed line represents a weight. As can be seen, even though the CNN has far less weights than the MLP, the receptive field of the g_3 unit also contains information from all input datapoints.

CNNs combine these convolutional layers with pooling layers [3, 4, 45], that are used to reduce the spatial dimensions of the input, while preserving channel information. Different methods of pooling can be used, such as max and mean pooling, which are exemplified in Figure 2.3.

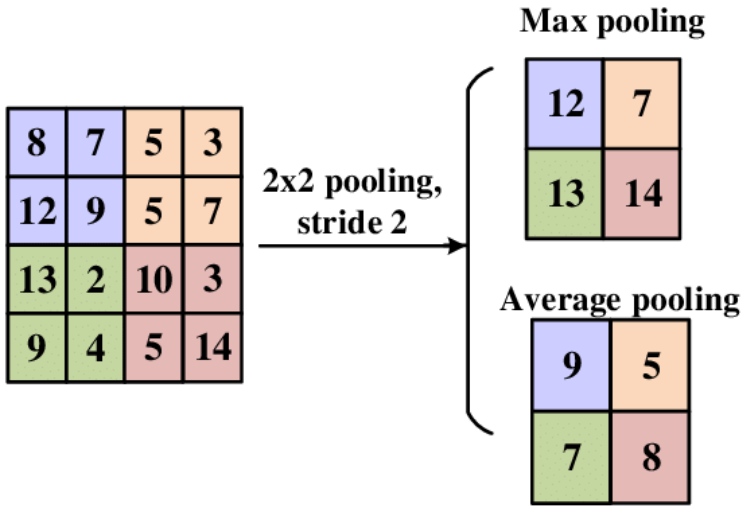

Figure 2.3: Examples of the outputs of max pooling and average pooling operations on a given input. Figure taken from [4].

2.2 Representation Learning

The effectiveness of ML models is heavily dependent on the representation of the input data. Earlier ML algorithms, such as Random Forests (RFs) [46] and SVMs [44], traditionally relied on feature engineering [47], where the original feature set is augmented with new features that are calculated based on the values of the other features. However, this is a manual time-consuming process that must be repeated for each different algorithm and dataset, since different types of models will not have the same effectiveness when provided with the same engineered features.

Representation learning (also known as feature learning) has emerged as a field that aims to tackle the limitations of traditional feature engineering, by automating the previously manual process of engineering useful features that are then feed into a downstream model, such as a classifier, or even to train a model in a end-to-end fashion. Beyond this, feature learning is also an important component of some ML paradigms, such as UL, which allows the training of models with unlabeled data, and can be leveraged to help these models achieve better generalization, through methods such as transfer learning [48].

Representation learning in ML can be divided into different categories, illustrated in Figure 2.4. Beyond

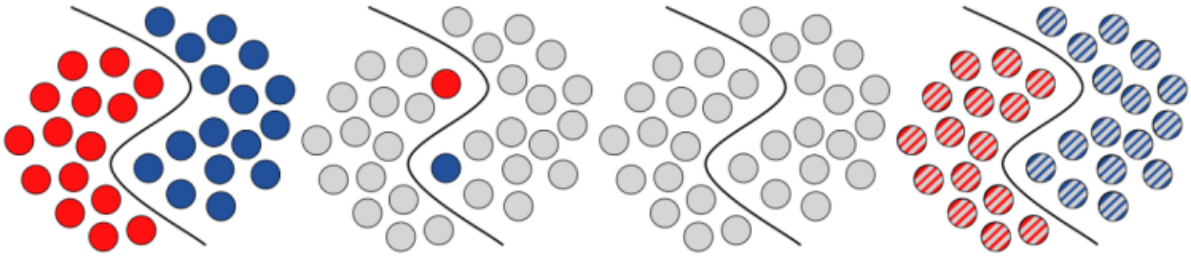

Figure 2.4: Illustration of several different types of DL strategies, where the red and blue circles represent datapoints of different classes and the lines between the circles represent the decision boundary between the classes. The grey circles represent unlabeled datapoints and the stripped circles represent datapoints which are initially unlabeled, but use label information at other stages of the training process. Figure from [5].

UL, this work also studies models that learn latent representations using Self-Supervised Learning (SSL) methods [49], which can be seen as particular case of UL which leverages a pretext task to generate pseudo-labels for a previously unlabeled dataset.

2.2.1 Unsupervised Learning

Unsupervised learning, also known as knowledge discovery, is a ML strategy that, unlike Supervised Learning (SL), which requires all data to be labelled in order to train a model, it can leverage a completely unlabeled dataset for model training. UL algorithms achieve this by analysing differences and similarities between the given data to uncover hidden patterns, which are used to learn meaningful

features of the training data [41]. UL methods have demonstrated significant success in many ML problems, where they achieve similar performance to many SL methods [41], while bypassing the need for the costly process of manual labelling of data.

Traditional UL algorithms include several clustering algorithms, such as K-Means Clustering [50] and U-k-Means Clustering [6], the latter of which is shown in Figure 2.5, along with some other types of algorithms, such as Principal Component Analysis (PCA), which emerged in multivariate statistics and is based on the calculation of the eigenvalues and eigenvectors for a given dataset [51].

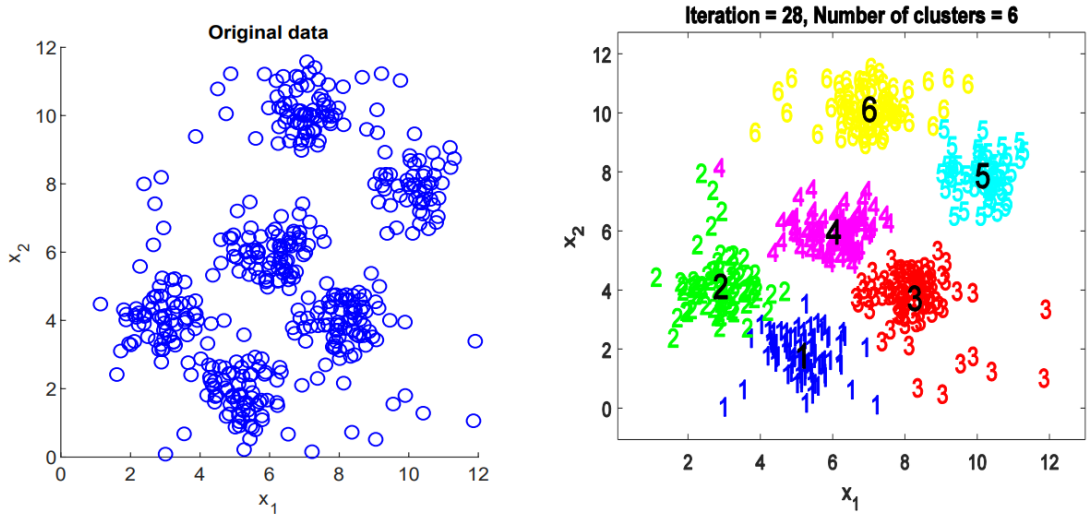

Figure 2.5: Example of the results of a clustering algorithm (U-k-means) on a 6-cluster dataset with 50 noisy datapoints, after 28 iterations. Figure from [6].

The previously mentioned achievements in DL research has resulted in the development of algorithms capable of learning meaningful representations in an unsupervised manner, such as Auto-Encoders (AEs) [52] and diffusion models [53]. Among the former family of algorithms, some of the most well-known models include the Denoising Auto-Encoder (DAE) [54] and the Variational Auto-Encoder (VAE) [55] (the latter of which we will further explore in a latter section, which focuses on Generative Models).

2.2.1.A Auto-Encoder

The typical auto-encoder is a model comprised of two NNs, where the first NN is known as the encoder, with parameters ϕ and represented as Enc_ϕ, and the second NN is known as the decoder, with parameters θ and represented as Dec_θ. When processing a input data batch, the encoder first processes the input x into a latent representation z that has a lower dimension compared to that of the input data, also known as the bottleneck. Afterwards, the decoder receives this latent representation z as its input and

outputs a reconstruction of original data $\hat{x}$. This means that, in essence, since the bottleneck layer has a lower dimension than that of the input data, the AE is forced to focus on the most important details of the input data and ignore redundant information when encoding the latent representation, which can be visualized for the case of a convolutional AE in Figure 2.6.

During the training process, the AE tries to minimize the reconstruction loss, usually the Mean Squared Error (MSE) or Mean Absolute Error (MAE), between the original data x and the reconstructed data $\hat{x}$. For the MAE loss function, this corresponds to minimizing the following equation:

$$\mathcal{L}(\theta, \phi; x_i) = \frac{1}{N}\sum_{i=1}^{N} |x_i - Dec_\theta(Enc_\phi(x_i))| = \frac{1}{N}\sum_{i=1}^{N} |x_i - Dec_\theta(z_i)| = \frac{1}{N}\sum_{i=1}^{N} |x_i - \hat{x_i}|, \tag{2.2}$$

where N is the number of data samples per batch. It is also worthy of note that, for an AE comprised purely of linear operations, the resulting latent representation z is the same as the latent representation obtained from PCA, so an AE can be seen as a generalization of the PCA algorithm [8].

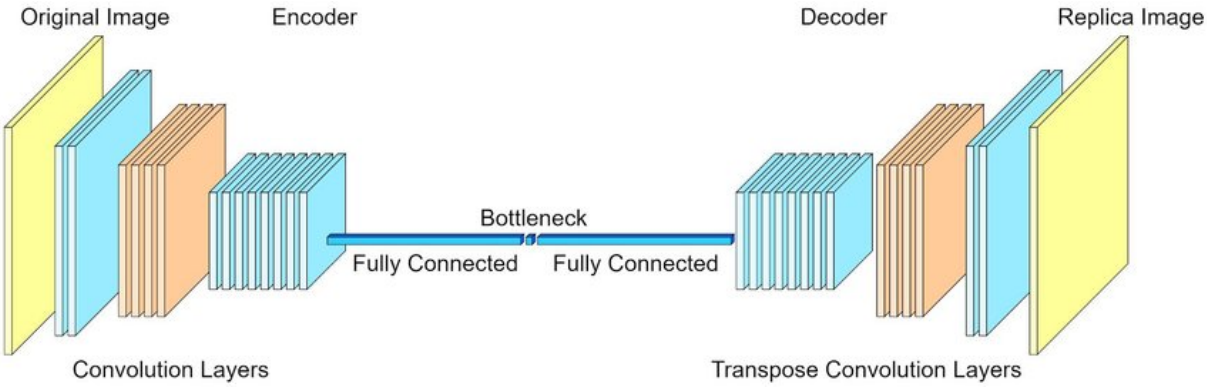

Figure 2.6: Overview of a general Convolutional AE architecture. Following a successful training process, the bottleneck layer, which has a significant lower dimension than the original image, should encode a meaningful latent space representation that allows the decoder NN to recreate an image similar to the original image that was given as input to the encoder NN. Figure from [7].

2.2.1.B Denoising Auto-Encoder

The denoising auto-encoder is an extension of the AE architecture that adds an extra step to the training process, where the input data batch x is mixed the with some form of (generally Gaussian) noise and then this corrupted data $\tilde{x}$ is feed to the encoder NN, while keeping the rest of training process exactly the same, as shown in Figure 2.7, meaning the DAE will be forced to learn to ignore noise from the input data when encoding its latent representations, so that it can successfully minimize the reconstruction loss between the original data x and the reconstructed data $\hat{x}$ given by the decoder NN. The corrupted

data $\tilde{x}$ is obtained through a distribution given by $Noise_\sigma(\tilde{x}|x)$. A general choice for $Noise$ is:

$$Noise_\sigma(\tilde{x}|x) = \mathcal{N}(x, \sigma^2\mathcal{I}), \tag{2.3}$$

where the standard deviation σ determines how much noise is mixed with the input data and is commonly set as $\sigma = \frac{1}{2}$. As such, the latent representation is given by $z = Enc_\phi(\tilde{x})$, where $\tilde{x} \sim Noise_\sigma(\tilde{x}|x)$ and, for the MSE loss function, the training process corresponds to minimizing the following equation:

$$\mathcal{L}(\theta, \phi; x_i) = \frac{1}{N}\sum_{i=1}^{N}(x_i - Dec_\theta(Enc_\phi(\tilde{x}_i)))^2. \tag{2.4}$$

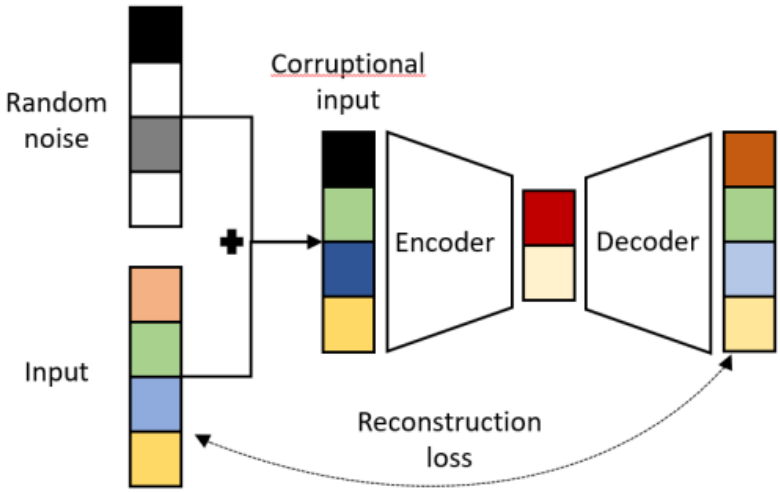

Figure 2.7: Overview of the DAE architecture and training process. Random noise is mixed in with the original input data before being feed into the DAE and the model then has to remove the noise from its reconstructions in order to minimize the loss between its outputs and the original noise-less input data. Figure from [8].

This training process helps the DAE to learn more robust latent representations, when compared to a simple AE, and also allows the model to be used for error correction.

2.2.2 Contrastive Learning

Contrastive learning is technique where a model is trained to differentiate between positive and negative pairs of datapoints by maximizing and minimizing the similarity between those of the same class and between those of different classes, respectively [10, 56]. While Contrastive Learning (CL) was initially proposed for representation learning in discriminative models [56], it quickly became a popular approach used in a wide array of UL scenarios [10, 12], in particular due to the aforementioned developments in representation learning using DL models [57].

In general, contrastive methods learn representations of the input data through the process illustrated in Figure 2.8, where samples that are similar to each other, known as positive samples, are brought closer together in the latent space, while maximizing the distance between samples that have high dissimilarity, known as negative samples.

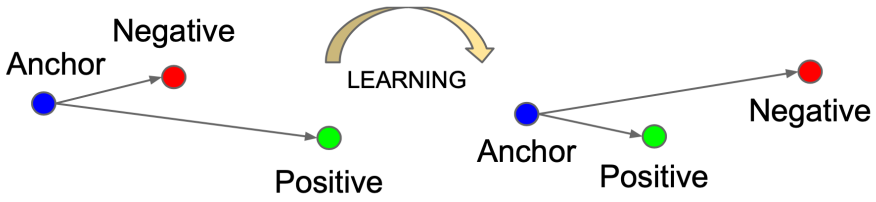

Figure 2.8: Example of the learning process using a triplet loss, where the distance between the anchor and the positive sample is minimized, while at the same time the distance between the anchor and the negative sample is maximized. Figure from [9].

2.2.2.A SimCLR

SimCLR is a CL framework for learning visual representations introduced by [10] that was shown by the authors as being capable of training a linear classifier in a self-supervised fashion such that it was able to match the performance of - or even outperform - SOTA SL models, like the ResNet-50 and AlexNet. Unlike previous CL techniques for UL scenarios, which either utilized a specialized architecture [58] or required a memory bank [59], the SimCLR framework, shown in Figure 2.9, enables a model to learn meaningful representations in a contrastive manner by leveraging three main ideas: 1) composition of multiple data augmentation operations, since UL benefits from stronger data augmentations when compared to SL; 2) introduction of a learnable nonlinear transformation between the latent representation and the contrastive loss; and 3) usage of values for batch sizes and training epochs, since CL benefits from these during the training of DL models.

The SimCLR framework is comprised of four main components:

- A stochastic data augmentation pipeline which transforms input data samples randomly, resulting in two correlated views of the same sample, which are denoted as $\tilde{x}_i$ and $\tilde{x}_j$;
- A CNN base encoder $f(\cdot)$, which encodes the augmented data samples into representation vectors $h_i = f(\tilde{x}_i)$;
- A MLP projection head $g(\cdot)$ that maps the intermediate representation vectors to the latent representations $z_i = g(h_i)$, where the contrastive loss is then applied;
- A contrastive loss function - first introduced by [57] - that, given a set $\{\tilde{x}_s\}$, which includes a positive pair of samples $\tilde{x}_i$ and $\tilde{x}_j$, seeks to identify $\tilde{x}_j$ in $\{\tilde{x}_s\}_{s \neq i}$ for a given sample $\tilde{x}_i$.

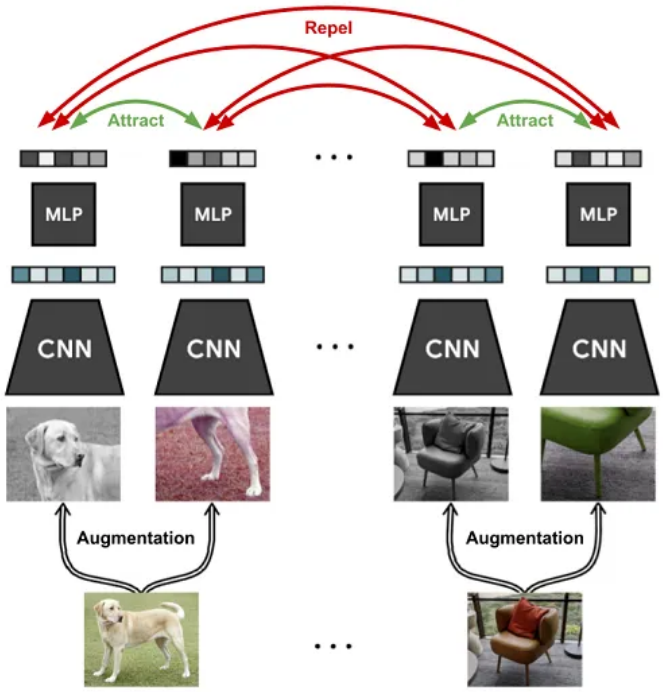

Figure 2.9: Overview of the SimCLR framework [10]. The contrastive loss minimizes the difference between images of the same element, while maximizing the difference between images of differences objects. Figure from [11].

During the training process, a mini-batch of N examples is randomly sampled and then go through the augmentation process, resulting in $2N$ data samples and, for the negative samples, the strategy described in [60] is followed, where, given a positive pair, the remaining $2(N-1)$ augmented samples in the mini-batch are treated as negative examples. Then, letting the similarity measure be the cosine similarity $\text{sim}(u,v) = \frac{u^T v}{||u||||v||}$, the loss function for a positive pair (i,j), which the authors call Normalized Temperature-Scaled Cross Entropy (NT-Xent), is given by:

$$\mathcal{L}_{i,j} = -\log \frac{\exp(\text{sim}(z_i, z_j)/\tau)}{\sum_{s=1}^{2N} \mathbb{1}_{s \neq i} \exp(\text{sim}(z_i, z_s)/\tau)}, \tag{2.5}$$

where τ is a temperature parameter and $\mathbb{1}_{s \neq i} \in \{0,1\}$ is a function that outputs 1 if and only if $s \neq i$ and 0 otherwise.

2.2.2.B Contrastive Predictive Coding

Contrastive Predicting Coding (CPC), shown in Figure 2.10, is a novel UL method introduced by [12], which seeks to learn meaningful representations from high-dimensional data and that is based on the data compression technique known as predictive coding and shares similarity with other CL approaches [10]. The motivation behind CPC is to efficiently learn representations that encode shared

information between different parts of the input, while discarding low-level or noisy information.

The authors argue that previous unimodal loss functions, such as MSE and cross-entropy, are not

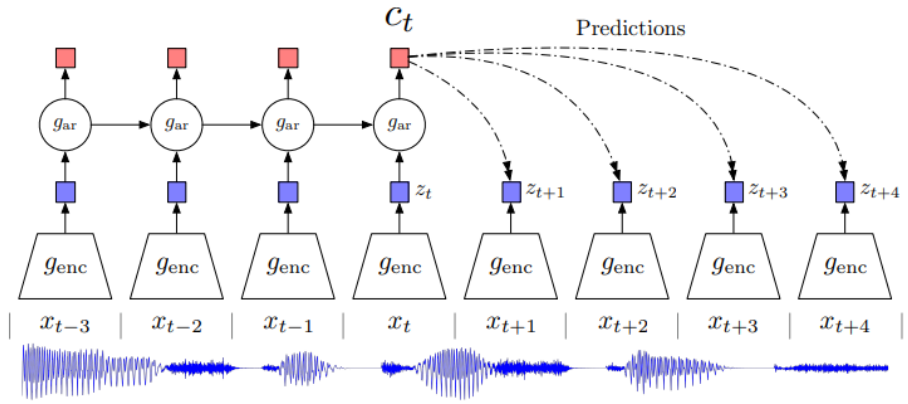

Figure 2.10: Overview of the CPC representation learning approach using audio as input. Figure from [12].

useful in predicting high-dimensional, and that more powerful generative models, while capable of reconstructing high-dimensional data with high detail, are computationally expensive and often ignore the context c by wasting capacity at modeling the complex relationships in the data x, which seems to suggest that explicitly modeling $p(x|c)$ is not an optimal choice for extracting the shared information between x and c. To deal with this issue and preserve the Mutual Information (MI) between x and c in an optimal fashion, the authors propose predicting the future information by encoding the target x and context c into compact distributed vector representations defined as:

$$\text{MI}(x;c) = \sum_{x,c} p(x,c) \log \frac{p(x|c)}{p(x)}, \tag{2.6}$$

since it is possible to extract the underlying latent variables which the inputs have in common by maximizing the MI between the encoded representations, and introduce a loss function based on the principle of Noise-Contrastive Estimation (NCE).

The CPC architecture is comprised of a non-linear Encoder $Enc\phi$ that maps the sequence of observations x_t to another sequence of latent representations $z_t = Enc\phi(x_t)$, and an Auto-Regressive (AR) model which creates a summary of all $z_{\leq t}$ in the latent space, and then outputs a latent representation of the context $c_t = AR_\theta(z_{\leq t})$. Further, it predicts future observations x_{t+k} by modeling a density ratio to preserve the MI between the future observations x_{t+k} and the context c_t using a log-bilinear model:

$$f_f(x_{t+k}, c_t) = \exp(z_{t+k}^T W_k c_t), \tag{2.7}$$

where the authors use a different W_k for every step k with a linear transformation $W_k^T c_t$ being used for the predictions.

For training the model, the authors propose jointly optimizing the Encoder and AR model with the InfoNCE loss, which, given a set $X = x_1, x_2, \ldots, x_N$ of N random samples containing one positive example from $p(x_{t+k}|c_t)$ and $N-1$ negative samples from a proposal distribution $p(x_{t+k})$, results in optimizing

$$\mathcal{L}_N = -\mathbb{E}_X[\log \frac{f_k(x_{t+k}, c_t)}{\sum_{x_j \in X} f_K(x_j, c_t)}], \tag{2.8}$$

where this equation is the Categorical Cross-Entropy (CCE) of classifying the positive sample correctly, with $\frac{f_k}{\sum_X f_k}$ being the prediction outputted by the model.

2.3 Generative Models

In ML literature, models are generally classified into one of two different types: discriminative or generative. While discriminative models explicitly model the conditional probability $p_\theta(y|x)$, with parameters θ learned from the training data, generative models use the training data to estimate the likelihood probability $p_\theta(x|y)$ and prior probability $p_\theta(y)$, i.e., they model the joint probability $p_\theta(x, y)$, such that they can then calculate the conditional probability through Bayes' theorem:

$$p_\theta(y|x) \propto p_\theta(y) \cdot p_\theta(x|y).$$

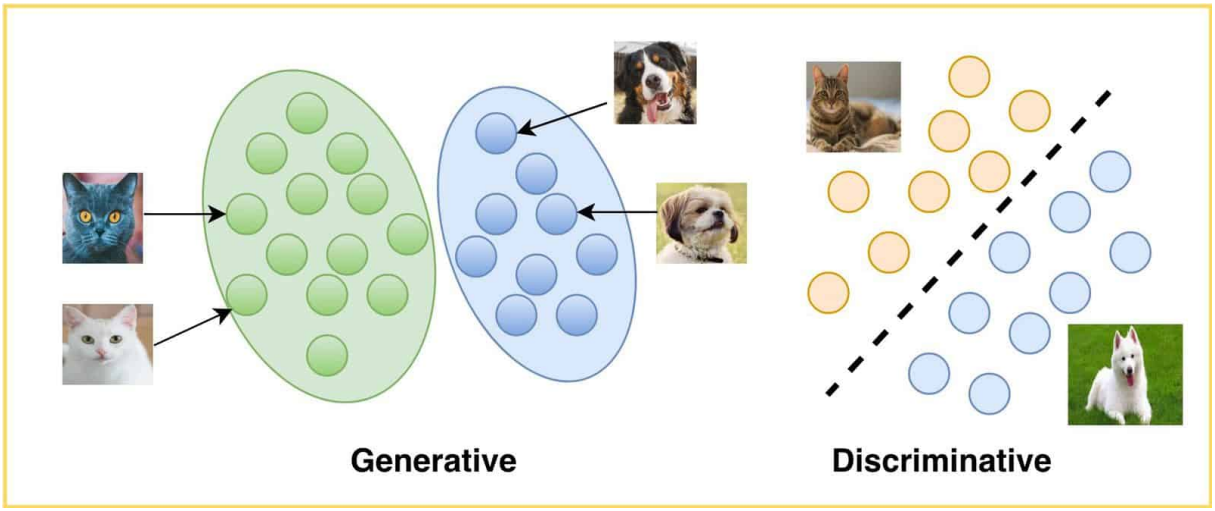

Figure 2.11: Illustration from OpenCV [13], which shows the difference between an arbitrary generative model and an arbitrary discriminative model, trained to perform a simple binary classification task of distinguishing between images of cats and dogs.

In the context of a classification task, exemplified in Figure 2.11, we can think of discriminative models as directly mapping the inputs x to the class labels y, i.e., they learn a function f such that $f(x) = \hat{y}$ and $y = \hat{y}$, whereas generative models first perform an intermediate step of modeling the joint

probability $p_\theta(x, y)$ using Bayes' theorem and then pick the most likely class label $\hat{y}$. By estimating the prior distribution $p_\theta(y)$, generative models attempt to generate samples $\hat{x} \sim p_\theta(\hat{x})$ that come from the same distribution as that of the training data $x \sim p_{data}(x)$. Some of the best known generative models are the VAE [55] (from the variational family of models) and the Generative Adversarial Network (GAN) [61] (from the adversarial family of models).

We must also stress that, while this distinction is the most common in ML literature, it is not the only one, as other authors further divide models into other different types, e.g., descriptive models [62]. However, for simplicity's sake, this work uses the generative and discriminative model categorization.

2.3.1 Variational Auto-Encoder

Based on the Variational Inference (VI) method and the AE model architecture [8], the variational auto-encoder was first introduced by [55]. The VAE architecture, shown in Figure 2.13, consists of a (Gaussian) probabilistic encoder NN $q_\phi(z|x)$ (which approximates the intractable true posterior $p_\theta(z|x)$) and a (Gaussian/Bernoulli) decoder $p_\theta(x|z)$, where the parameters θ and ϕ are jointly optimized using gradient-based methods. The variational approximate posterior is a multivariate Gaussian with a diagonal covariance structure

$$\log q_\phi(z|x_i) = \log \mathcal{N}(z; \mu_i, \sigma_i^2 \mathcal{I}), \tag{2.9}$$

where the mean (μ_i) and standard deviation (σ_i) are outputs of the MLP encoder and the prior $p_\theta(z) = \mathcal{N}(z; 0, \mathcal{I})$ is a centered isotropic multivariate Gaussian. We can sample the posterior $z_{i,v} \sim q_\phi(z|x_i)$ by using the reparameterization trick $z_i = g_\phi(x_i, \epsilon) = \mu_i + \sigma_i \odot \epsilon$, which separates the deterministic and probabilistic elements of the model, illustrated in Figure 2.12, where $\epsilon \sim \mathcal{N}(0, \mathcal{I})$ (with $\odot$ being the Hadamard product).

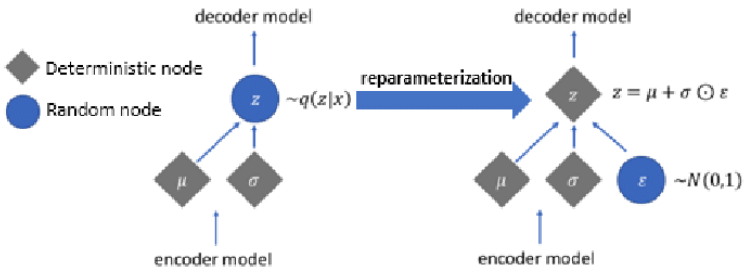

Figure 2.12: Illustration of how the reparameterization trick separates the probabilistic and deterministic parts of the model, allowing for easier optimization. Figure from [14].

The marginal likelihood, which is composed of a summation over the marginal likelihoods of each

individual datapoint $\log p_\theta(x_1, \ldots, x_N) = \sum_{i=1}^{N} \log p_\theta(x_i)$, can be rewritten as:

$$\log p_\theta(x_i) = \text{KLD}(q_\phi(z|x_i)||p_\theta(z|x_i)) + \mathcal{L}(\theta, \phi; x_i), \tag{2.10}$$

where the right-hand side is the Kullback-Leibler Divergence (KLD) [63] of the approximate from the true posterior. Since this KLD is non-negative, the second right-hand side term, called the variational lower bound on the marginal likelihood of the i^{th} datapoint, can be written as:

$$\mathcal{L}(\theta, \phi; x_i) = \mathbb{E}_{q_\phi(z|x_i)}[-\log q_\phi(z|x) + \log p_\theta(x, z)]. \tag{2.11}$$

As such, the objective function corresponds to maximizing the Evidence Lower Bound (ELBO), which is given by the following equation:

$$\mathcal{L}_{ELBO}(\theta, \phi; x_i) = -\beta \cdot \text{KLD}(q_\theta(z|x_i)||p_\phi(z)) + \mathbb{E}_{q_\phi(z|x_i)}[\log p_\theta(x_i|z)], \tag{2.12}$$

where β is a scalar parameter and $\mathbb{E}_{q_\phi(z|x_i)}[\log p_\theta(x_i|z)]$ is the expected reconstruction error, used a the loss function in most AE architectures [52]. In practice, during training the model minimizes the negative ELBO. If we assume both probability distributions $p_\theta(z)$ and $q_\phi(z|x)$ are Gaussian, the KLD can be computed and differentiated without estimation, with the resulting loss function for the VAE model, for each datapoint x_i, being given by the following equation:

$$\mathcal{L}_{ELBO}(\theta, \phi; x_i) \simeq \beta \sum_{j=1}^{J} (1 + \log(\sigma_{i,j}^2) - \mu_{i,j}^2 - \sigma_{i,j}^2) + \frac{1}{L} \sum_{l=1}^{L} \log p_\theta(x_i|z_{i,l}), \tag{2.13}$$

where β is parameter that defines the weight of the KLD in the ELBO loss, L can be typically set as 1 for a large enough mini-batch size, $z_{i,l} = \mu_i + \sigma_i \odot \epsilon_l$ and $\epsilon_l \sim \mathcal{N}(0, \mathcal{I})$.

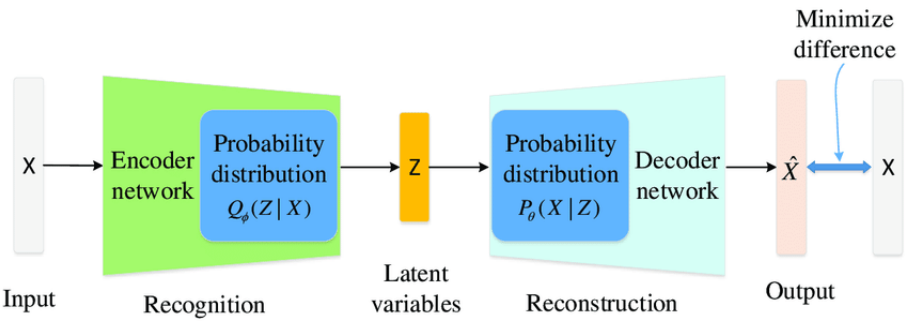

Figure 2.13: Overview of the VAE architecture, where the latent variables $z = \mu + \sigma \odot \epsilon$. Figure from [15].

2.3.2 Generative Adversarial Network

The generative adversarial network is a generative model introduced by [61] that consists of two NNs (a generator G, with parameters θ, and a discriminator D, with parameters ϕ) that play a two-player mini-max game with the value function $V(G_\theta, D_\phi)$. The optimization of a GAN, shown in Figure 2.14, corresponds to training D_ϕ, such that the probability $D_\phi(x)$ correctly assigns labels to examples from the training distribution and those that come from $G_\theta(z)$, and $G_\theta(z)$ to minimize $\log(1 - D_\phi(G_\theta(z)))$:

$$\min_G \max_D V(D, G) = \mathbb{E}_{x \sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z \sim p_z(z)}[log(1 - D(G(z)))], \tag{2.14}$$

where, in practice, the game is implemented in an iterative manner, where we optimize D_ϕ by k steps for every step we optimize G_θ, in order to avoid over-fitting and to improve computational efficiency.

Minimizing $\log(1 - D_\phi(G_\theta(z)))$ may lead to insufficient gradients for G_θ to learn to generate good samples, since the samples generated in the first epochs are rather poor and D_ϕ will be able to distinguish these samples from the training data with high confidence. To address this, the authors propose training G_θ to maximize $\log(D_\phi(G_\theta(z)))$ instead, since this objective function provides stronger gradients in earlier training epochs, while keeping the same fixed point of the dynamics of G_θ and D_ϕ.

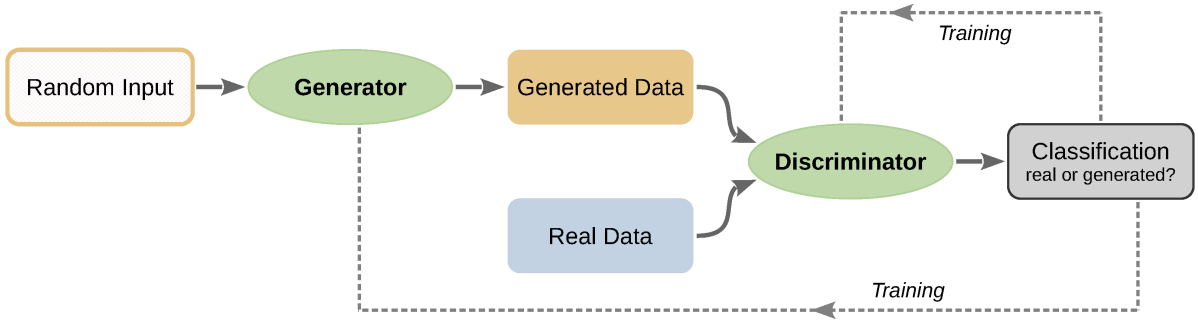

Figure 2.14: Overview of the GAN architecture and training scheme, where the random input given to the generator is a sum of the latent space and random (generally Gaussian) noise. Figure from [16].

The GAN model architecture offers some benefits when compared to other generative models, e.g., most images generated by a GAN are of a higher quality and are less noisy compared to those generated by a VAE [64]. However, they present several significant drawbacks, in particular with regards to their training process, as GANs are known to be rather unstable and suffer from mode collapse, as well as a general lack of convergence, meaning hyperparameter tuning for GANs is a very difficult task compared to most other generative models [65].

3

Related Work

Contents

3.1 Robustness of Deep Learning Models 19
3.2 Multimodal Deep Learning 29
3.3 Robustness of Deep Multimodal Models 34

3.1 Robustness of Deep Learning Models

An important characteristic of most ML models is that its performance is highly correlated with the quality of the input data [47]. Despite the success of DL models in a wide array of tasks without requiring extensive feature engineering, there are still several open challenges that limit their use in commercial applications. In particular, the vulnerability of these models to adversarial examples (also known as adversarial samples) [20, 32], which are input samples with small amounts of noise that are imperceptible to humans, but can significantly degrade model performance [32], and the fact that real world data can be far more noisy than the data in datasets used to train most of DL models [66] and deep models, such as CNNs, may not be able to properly handle this noisy data [67]. Models may even perform well according to their metrics on the training dataset, while in fact they have learned representations that do not

correlate well with the intended objective, as shown in Figure 3.1, which reinforces the importance of performing a comprehensive testing of the robustness of DL models before deploying them in the real world.

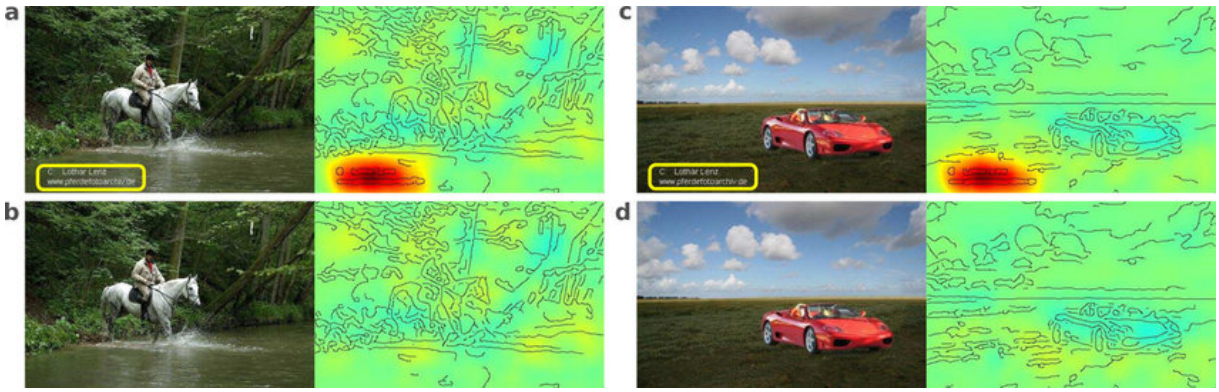

Figure 3.1: Input images next to the respective relevance maps for a model trained to detect horses on images from the Pascal VOC 2007 dataset [17]. The model classifies images a) and c) as containing an horse due to the presence of a copyright watermark, while classifying images b) and d) as not containing an horse since the watermark is not present, while the images a) and b) contain a horse. Figure from [18].

While ML algorithms such as SVMs and RFs have also been shown to have their performance degraded by noisy data [68, 69], and to be vulnerable to adversarial attacks [70, 71], several challenges are particular to DL. For traditional ML algorithms, an extensive feature engineering pipeline can help models cope with noisy data [72, 73] and, while this would also be theoretically possible for DL models, it would negate one of the biggest advantages of the DL paradigm, and most datasets used to train DL models are also far larger than those used to train traditional ML models, making it more difficult to accurately evaluate the quality of the data present in these datasets [66]. Also, since adversarial examples are usually imperceptible to humans, DL models suffer from a lack of explainability [74] and even algorithms developed to improve explainability in DL models have been shown to be vulnerable to adversarial examples [75], it can be much harder to understand what a model is actually learning, and consequentially to identify the vulnerabilities present in deeper models.

3.1.1 Robustness to Adversarial Attacks

According to [19, 76], adversarial attacks can be summarized as follows: given an ML model which maps an input to a discrete label set $f : \mathbb{R}^n \rightarrow \{y_1, \ldots, y_L\}$, that has an associated continuous loss function $\mathcal{L}$, and an input sample x, free of perturbations, we assume that the model accurately classifies the input sample, i.e., $f(x) = y$. Then, it is possible to build an adversarial sample $\tilde{x}$ that is indistinguishable from x by humans, but is incorrectly classified by the model, i.e., $f(\tilde{x}) \neq y$, by finding a small perturbation η such that $\tilde{x} = x + \eta$, as exemplified in Figure 3.2. Also, like it is indicated in [20], all adversarial attacks require the use of distance metrics to quantify similarity between the generated adversarial samples and

the original input samples, of which three commonly used distance metrics exist, all are l_p norms, where the $l_p = ||\cdot||_p$ distance between x and $\tilde{x}$ can be defined as

$$||x - \tilde{x}||_p = (\sum_{i=1}^{N} |x_i \tilde{x}_i|^p)^{\frac{1}{p}}, \tag{3.1}$$

for the l_0 and l_2 distances, and the other commonly used distance, l_∞, can be defined as

$$||x - \tilde{x}||_\infty = \max(|x_1 - \tilde{x}_1|, \ldots, |x_N - \tilde{x}_N|). \tag{3.2}$$

The existence of blind spots in DNNs that make generating adversarial samples possible was first discovered in [32], where, given an input image x, the authors used a box-constrained Limited-memory Broyden–Fletcher–Goldfarb–Shanno (L-BFGS) algorithm [77] to find the minimal $\epsilon > 0$ for which η satisfies $f(x + \eta) = y_{\text{target}}$, where y_{target} is a target label, i.e., they minimized the following equation:

$$\epsilon|\eta| + \mathcal{L}(x + \eta, y_{\text{target}}) \quad s.t. \quad x + \eta \in [0, 1]^n. \tag{3.3}$$

The experiments performed in this initial work [32] showed that several different architectures were vulnerable to adversarial attacks - including a MLP trained on the Modified National Institute of Standards and Technology (MNIST) dataset, AlexNet and a DNN with one billion parameters trained in a totally unsupervised manner on a dataset with ten million images taken from Youtube videos - and also that a significant amount of the adversarial samples generated for a given model and dataset generalized to other models, with different architectures and hyperparameters, and across models trained on disjoint datasets of the same original dataset, i.e., an adversarial sample generated for a model trained on half of an arbitrary dataset, would also cause misclassifications on a model trained on the other half of the same dataset. Further research has shown that increasing the capacity and expressiveness of the model does not necessarily improve its robustness to adversarial attacks [19].

Most work focuses on studying these attacks in the context of CV tasks [19], where small pixel-level perturbations, indistinguishable to the human eye, can result in the misclassification of image labels. More recent research focuses on learning about the existence of these adversarial examples in other domains, such as in NLP and Graph ML [34]. Approaches used by adversarial attacks to generate adversarial samples can be broadly classified into one of two scenarios: white-box or black-box. Some authors define a third category, known as gray-box adversarial attacks [78], but, for simplicity's sake and since in this work we do not focus on these types of attacks, we define all attacks where the adversary does not possess all information about a model as a black-box adversarial attack.

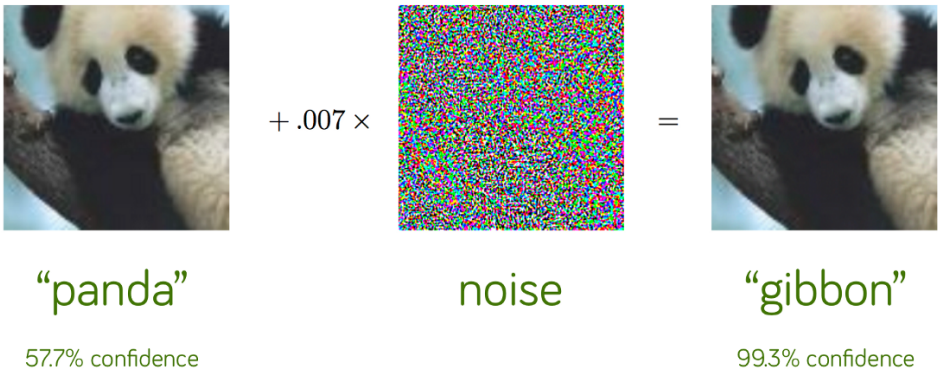

Figure 3.2: Example of an adversarial attack in a CV task. By adding a small amount of noise that is imperceptible to the human eye, the adversary can fool the NN into classifying a panda as a gibbon with high confidence. Figure adapted from [19].

Adversarial attacks can also be classified into different categories based on their objective: untargeted adversarial attacks [79, 80], where the objective is to make a model to predict any label that is not the correct one, targeted adversarial attacks [79, 80], where the objective is to force the model output a specific class label chosen by the adversary. Some authors also define other classes of adversarial attacks, e.g., universal adversarial attacks [80], where the objective is to calculate a special imperceptible noise that can be mixed with any sample of a given dataset in order to force a model to predict the wrong class. However, we will only focus on untargeted and targeted adversarial attacks in this work.

3.1.1.A White-Box Adversarial Attacks

In white-box adversarial attacks, the adversary can access all the information about the target model, such as the training dataset, the hyperparameters, the model's gradients and its parameters [79]. Some of the most well known white-box adversarial attacks include: the fast gradient sign method, the basic iterative method, the C&W attack and projected gradient descent.

A – FGSM

One of the very first adversarial attacks, dubbed the Fast Gradient Sign Method (FGSM), was introduced by [19] and consists of finding a optimal max-norm constrained perturbation η, i.e., finding a

perturbation η which maximizes the cost function $\mathcal{L}_\theta(x, y)$ used to train the model, such that,

$$\eta = \arg\max_{\eta} \mathcal{L}_\theta(x + \eta, y) \quad s.t. \quad ||\eta||_\infty \leq \epsilon, \tag{3.4}$$

$$\eta = \arg\max_{\eta} \mathcal{L}_\theta(x, y) + \nabla_x \mathcal{L}_\theta(x, y)^T \eta \quad s.t. \quad ||\eta||_\infty \leq \epsilon, \tag{3.5}$$

$$\eta = \epsilon \cdot \text{sign}(\nabla_x \mathcal{L}_\theta(x, y)) \tag{3.6}$$

where sign$(\cdot)$ extracts the sign of a real number, $||\eta||_\infty = \max_i |\eta_i|$ and ϵ is a scalar that represents how many of the input image pixels the attack is going to alter. In practice, we want to minimize the cost function, which can be easily done by negating the the sign of the of the gradient, i.e.,

$$\eta = -\epsilon \cdot \text{sign}(\nabla_x \mathcal{L}_\theta(x, y)), \tag{3.7}$$

where this method will then generate adversarial samples $\tilde{x} = x + \epsilon \cdot \text{sign}(\nabla_x \mathcal{L}_\theta(x, y))$. Although this algorithm is fairly simple, it is able to find adversarial examples on a multitude of models, which, as the authors argue, is a result of the linearity of these models. While the proposed algorithm uses the l_∞ distance metric, FGSM can be generalized to use the l_0 or l_2 distances metrics [81].

B – BIM

The Basic Iterative Method (BIM), introduced by [76], is an adversarial attack that extends FGSM by first initializing the algorithm with a clean input, i.e., $\tilde{x}_0 = x$, and then iteratively applying FGSM multiple times with a small step size, while clipping the values of the intermediate results after each step to ensure that they are in the ϵ-neighbourhood of the original image, such that

$$\text{Clip}_{x,\epsilon}\{x'\}(h, w, c) = \min\{1.0, x(h, w, c) + \epsilon, \max\{0, x(h, w, c) - \epsilon, x'(h, w, c)\}\}, \tag{3.8}$$

where $x(h, w, c)$ is the value of the channel c of the image x at the (h, w) coordinates. Then, the equation for the adversarial samples generated by BIM is given by:

$$\tilde{x}_{K+1} = \text{Clip}_{x,\epsilon}\{\tilde{x}_K + \alpha \cdot \text{sign}(\nabla_x \mathcal{L}_\theta(\tilde{x}, y))\}, \tag{3.9}$$

where α is a parameter that defines how much the value of each pixel is changed at each step and K is the number of iterations of the algorithm.

C – Projected Gradient Descent

A more recently proposed adversarial attack [82] views the issue of finding adversarial samples as

a saddle point problem, given by the following equation:

$$\min_{\rho} \rho(\theta), \quad \text{where} \quad \rho(\theta) = \mathbb{E}_{(x,y)\sim\mathcal{D}}[\max_{\eta\in S} \mathcal{L}(\theta; x+\eta, y)], \tag{3.10}$$

where $\mathcal{D}$ is the underlying data distribution over pairs of examples $x \in \mathbb{E}^d$ and corresponding labels y, $S \subseteq \mathbb{R}^d$ is a set of possible adversarial perturbations and $\mathbb{E}_{(x,y)\sim\mathcal{D}}$ is the population risk, i.e., the vulnerability of the model to adversarial perturbations.
The resulting attack is a multi-step Projected Gradient Descent (PGD) on the negative loss function that finds perturbed samples such that

$$x_{k+1} = \prod_{x+S} (x_k + \epsilon \cdot \text{sign}(\nabla_x \mathcal{L}(\theta; x, y))). \tag{3.11}$$

D – C&W Attack

Another proposed white-box attack, called the C&W attack, was introduced by [20] to test the effectiveness of a previously developed adversarial defense, called defensive distillation [83], which was previously shown to reduce the success rate of several adversarial attacks from 95% down to 5%. C&W proved remarkably capable of generating adversarial samples that bypass existing adversarial defenses, as experiments targeting models which employed defensive distillation demonstrated that the C&W attack still achieved near 100% success rate. This attack consists of solving the following optimization problem

$$\begin{aligned} \text{minimize} \quad & \text{dist}(x, x+\eta) + \epsilon \cdot \mathcal{L}_{cw}(x+\eta), \\ & s.t. \quad x+\eta \in [0,1]^n, \end{aligned} \tag{3.12}$$

where dist$(\cdot)$ is some distance metric and $\epsilon > 0$ is a chosen constant.
In practice, this corresponds to three different attacks, depending on whether the l_2 norm, the l_0 norm or the l_∞ norm was chosen as the distance metric dist$(\cdot)$. However, the C&W attack is non-differentiable when using the l_0 norm and not fully differentiable when using the l_∞ norm, making it unsuitable to optimize via Gradient Descent (GD). As such, when we mention this attack in this work, we will be referring to the C&W attack using the l_2 norm as the distance metric. As such, using the l_2 norm as our distance metric, and given $\tilde{x} = x + \eta$, the optimization problem then becomes

$$\begin{aligned} \text{minimize} \quad & ||\tilde{x} - x||_2^2 + \epsilon \cdot \mathcal{L}_{cw}(x+\eta), \\ & s.t. \quad x+\eta \in [0,1]^n, . \end{aligned} \tag{3.13}$$

Given this, the perturbations will be of the form

$$\eta = \arg\min_{\eta} ||\tilde{x} - x||_2^2 + \epsilon \cdot \mathcal{L}_{cw}(x + \eta) \quad s.t. \quad (x + \eta) \in [0, 1]^n. \tag{3.14}$$

To ensure that the adversarial perturbation yields a valid image, η must be constrained, in what is commonly known in literature as a box constraint, similarly to the original algorithm used to find the first adversarial examples [32], where the box constraint for the C&W attack is given by

$$0 \leq x_i + \eta_i \leq 1, \quad \forall i. \tag{3.15}$$

The authors propose three different ways to ensure this constraint:

- PGD, a strategy similar to the one used in [82], which performs one step of GD and then clips all the coordinates to be within the box, but can perform poorly in GD approaches with a complicated update step, such as GD with momentum;

- Clipped GD which, instead of clipping x_i at each iteration, incorporates the clipping into the objective function, i.e., $f(x+\eta) = f(\min(\max(x+\eta, 0), 1))$, similarly to BIM [76], which, while solving the main issue with PGD, introduces the possibility that the algorithm may get stuck in a saddle point;

- Change of variables, which introduces a new variable v and, instead of optimizing over the previously defined variable η, we optimize over v, setting $\eta_i = \frac{1}{2}(\tanh(v_i) + 1) - x_i$, and, since $-1 \leq \tanh(v_i) \leq 1$, we have that $0 \leq x_i + \eta_i \leq 1$, so the solution will be valid.

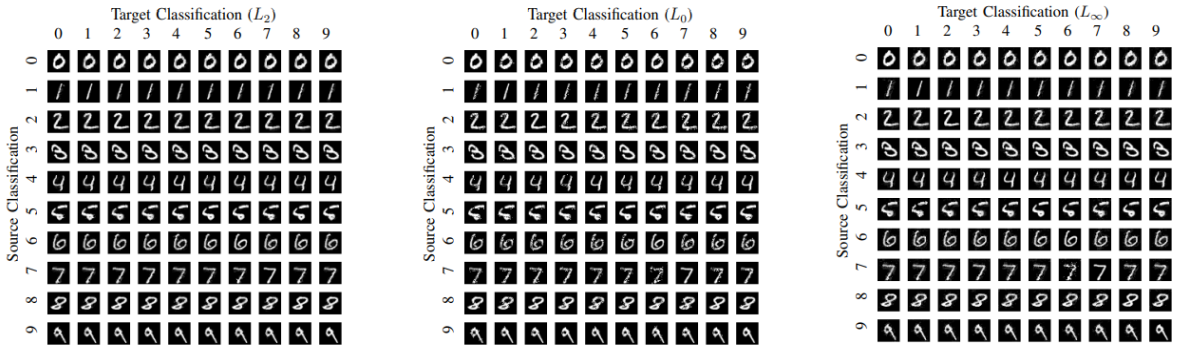

Figure 3.3: The C&W attack applied to images from the MNIST dataset, for each of the three possible distance metrics. The original images correspond to the diagonal, where the source and target classification correspond to the same digit label. Notice how the difference between each image is almost imperceptible. Figure adapted from [20].

Combining the change of variables approach with the attack using the l_2 norm as the distance metric,

and given an input x, the problem corresponds to searching for a v that solves the following equation:

$$\text{minimize} \quad ||\frac{1}{2}(\tanh(v)+1) - x||_2^2 + \epsilon \cdot \mathcal{L}_{cw}(\frac{1}{2}(\tanh(v)+1), \tag{3.16}$$

where $\mathcal{L}_{cw}$ is an objective function indicated by the authors as the best function for this problem, and is defined as

$$\mathcal{L}_{cw}(x') = \max(\max\{Z(x')_k : k \neq y_{\text{target}}\} - Z(x')_{\text{target}}, -u), \tag{3.17}$$

where u is a parameter that controls the confidence with which the misclassification occurs.

3.1.1.B Black-box Attacks

In contrast to white-box approaches to generating adversarial examples, black-box adversarial attacks are more similar to real world scenarios, since we assume the adversary does not have access to the complete information about a model. Usually, we assume the adversary will only have access to a public Application Programming Interface (API) which he can query and obtain an output from the model [79]. However, black-box scenarios are rather varied, e.g., a model can be closed-source and the adversary may not know its specific architecture, its parameters and hyperparameters, but he may know and have access to the dataset on which the model was trained on or even know about the general architecture on which the model was based on, since such information is usually public.

A – WGAN for Adversarial Sample Generation

One approach for black-box adversarial attacks based on generative models uses a GAN-like framework [21], shown in Figure 3.4, to generate what the authors dub natural adversarial examples. This approach uses the Wasserstein Generative Adversarial Network (WGAN) [84] combined with an Inverter, as shown in Figure 3.4, with parameters ϕ, Inv_ϕ (with z' being the output of the Inverter given an input x, i.e., $z' = Inv_\phi(x)$). The authors use the WGAN due to the fact that the original GAN objective function is hard to optimize and the authors of the WGAN refined the objective function by incorporating the Wasserstein-1 distance as follows:

$$\min_\theta \max_\phi \mathbb{E}_{x\sim p_x(x)}[D_\phi(x)] - \mathbb{E}_{z\sim p_z(z)}[D_\phi(G_\theta(z))]. \tag{3.18}$$

The training objective will then consist of minimizing the reconstruction error of x and the divergence between the sampled z and $\mathcal{I}_\gamma(G_\theta)$, such that:

$$\min_\gamma \mathbb{E}_{x\sim p_x(x)}||G_\theta(\mathcal{I}_\gamma(x)) - x|| + \epsilon \cdot \mathbb{E}_{z\sim p_z(z)}[\mathcal{L}(z, \mathcal{I}_\gamma(G_\theta(z)))], \tag{3.19}$$

where ϵ is a scalar weight. This setting produces natural adversarial examples $x^\star$ as follows:

$$x^\star = G_\theta(z^\star) \quad \text{where} \quad z^\star = \arg\min_{\tilde{z}} ||\tilde{z} - \mathcal{I}_\gamma|| \quad s.t. \quad f(G_\theta(\tilde{z})) \neq f(x), \tag{3.20}$$

where, instead of x, the framework perturbs the representation $z' = Inv_\gamma(x)$, and then uses the generator to test whether a perturbation $\tilde{z}$ fools the classifier by querying f with a perturbed sample $\tilde{x} = G_\theta(\tilde{z})$.

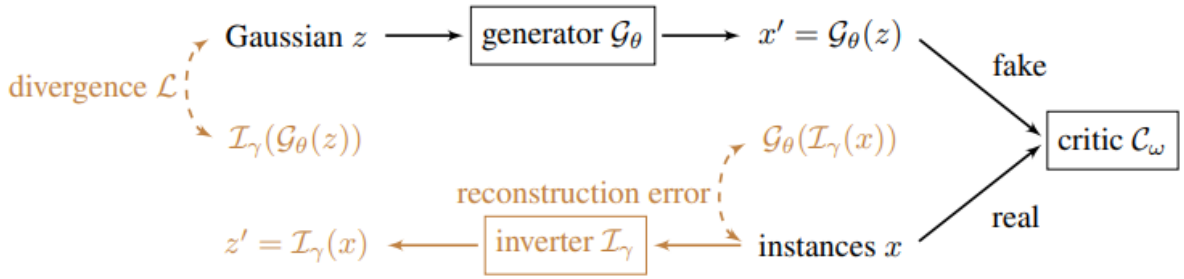

Figure 3.4: The architecture of the WGAN combined with an Inverter. In our notation, the critic $\mathcal{C}$ corresponds to the discriminator D and its parameters ω indicated in the figure corresponds to ϕ, i.e, $D_\phi = \mathcal{C}_\omega$. Figure from [21].

B – Transferability of Adversarial Attacks

Another type of black-box attack is to exploit adversarial transfer to transfer adversarial examples generated based on surrogate models [85]. The effectiveness of the this attack on a given target model is proportional to the similarity between the surrogate model that was originally used to generate the adversarial examples and model that is being targeted.

In general, black-box attacks are less effective at fooling a target model than white-box attacks, and are more easily stopped by existing adversarial defenses. However, all previously mentioned white-box adversarial attacks, FGSM [19], BIM [76] and the C&W attack [20], have been shown to create adversarial samples that successfully force misclassifications in models that have different architectures to the surrogate model which was used to generate the adversarial samples, although with different degrees of success.

Further research into the transferability of adversarial attacks [86, 87] identified important characteristics to take into account when attempting to transfer adversarial samples generated for a surrogate model to another target model.

In [87], the authors demonstrated that, if a surrogate model is trained to perform the same task as the target model, we can use the target model as an oracle to label a synthetic training dataset for the surrogate model and achieve very high attack success, as empirical data shows that adversarial samples generated using SVMs [44] where able to force misclassifications on commercial ML systems from Amazon (with 96.19% success rate) and Google (with 88.94% success rate) using only 800 queries on

the commercial models, indicating that ML models are generally vulnerable to transferred adversarial attacks, regardless of their structure or architecture.
Furthermore, it was showed in [86] that nonlinear models are more robust than linear models to transferred adversarial attacks, and also model complexity of the surrogate model is correlated with transferability of adversarial attacks, meaning that matching the complexity of the target model is important, regardless of its architecture.

3.1.2 Defenses Against Adversarial Attacks

One of the first adversarial defenses was based on the idea that a DNN could achieve more robustness by training on a mix of adversarial and clean data [19, 32], a strategy which was later called adversarial training. This approach, despite still being vulnerable to iterative attacks and often reducing the performance of the model given clean data, is able to provide some improved robustness guarantees. Another drawback of this strategy is that, depending the adversarial attack chosen to generate the adversarial samples used during the adversarial training, the training process of the model can be significantly longer, particularly in the case of iterative adversarial attacks such as BIM.

3.1.2.A Defensive Distillation

Defensive distillation [83] is a adversarial defense method based on the training procedure of distillation [88], initially designed to train a neural network by transferring knowledge across neural networks. In this approach, instead of transferring knowledge between neural networks, the authors' propose using the extracted knowledge as a way to improve the neural network's robustness to adversarial examples. Towards this purpose, they introduce a metric ρ_{adv} to classify the robustness of a neural network F,

$$\rho_{adv}(F) = \mathbb{E}_{\mu}[\sigma_{adv}(X, F)] \tag{3.21}$$

where the inputs X are drawn from the distribution μ that the neural network is attempting to model and $\sigma_{adv}(X, F)$ is the minimum perturbation required to misclassify each sample x in each of the other classes, given by:

$$\sigma_{adv}(X, F) = \arg\min_{\eta, X}\{||\eta X|| : F(X + \eta X) \neq F(X)\}. \tag{3.22}$$

This process of distillation protects DNNs from adversarial examples by reducing the gradients used in adversarial example generation, thereby increasing the number of features that need to be modified for a adversarial attack to successfully fool the model. Defensive distillation can only be employed in DNNs that produce an energy-based probability distribution and is also vulnerable to more complex adversarial attacks, such as C&W.

3.1.2.B Defensive Generative Models

One promising area of research is the use of generative models as a defense mechanism against adversarial examples [22, 89]. An example of such a mechanism is the Defense-GAN [22]. This approach uses the WGAN variant of the GAN model, which has a more stable training process due to using the objective function present in Equation 3.18, to model the distribution of unperturbed images and then, at test time, finds a close output to a given image which doesn't contain the adversarial changes by minimizing the reconstruction error, i.e.,

$$||G(z) - x||_2^2, \tag{3.23}$$

and subsequently feeds this output into the downstream classifier. This approach, provides a model agnostic defense strategy that is able to improve robustness against both white-box and black-box adversarial attacks, at the cost of increased computations.

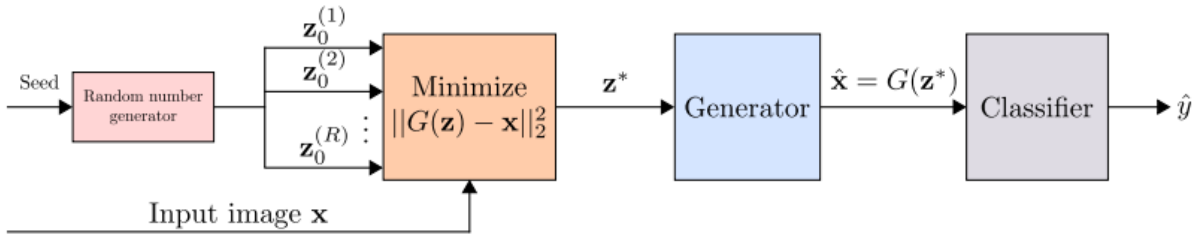

Figure 3.5: Overview of the Defense-GAN defense mechanism. Figure from [22].

3.2 Multimodal Deep Learning

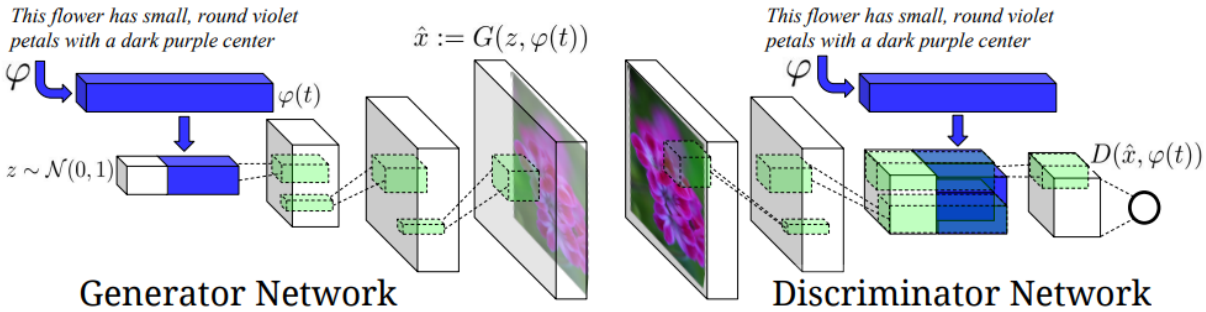

Figure 3.6: The text-conditional convolutional GAN architecture that is based on the DC-GAN model, where both the generator and discriminator networks are conditioned on the text encoding $\varphi(t)$. Figure from [23].

Most previous DL research focused on specialized architectures that process a specific type of data, be it text data in NLP [90], image data in CV [91], graph data [92], audio data [93] or even LiDAR point clouds [94], while initial work on multimodal deep learning focused mostly on generating image captions.

However, with the advent of GANs as a model capable of generating high fidelity images [61], combined with the increased capabilities of word-based LSTM image retrieval models [95], a new idea came to be developed: to use a model based on the Deep Convolutional-Generative Adversarial Network (DC-GAN) [96] architecture, which had CNNs as the generator and discriminator, conditioned on text features encoded by a hybrid character-level convolutional-RNN, to generate images from text [23], shown in Figure 3.6. This new framework lead to further development of several GAN-based text-to-image generators, such as the StackGAN [24], an architecture where several generator-discriminator are stacked to improve the results at each stage, and the AttnGAN [24], which combines the idea of stacking used in the StackGAN with the attention mechanism introduced by [97].

Beyond GAN-based architectures, models such as DALL-E, which is based on the Vector Quantised-

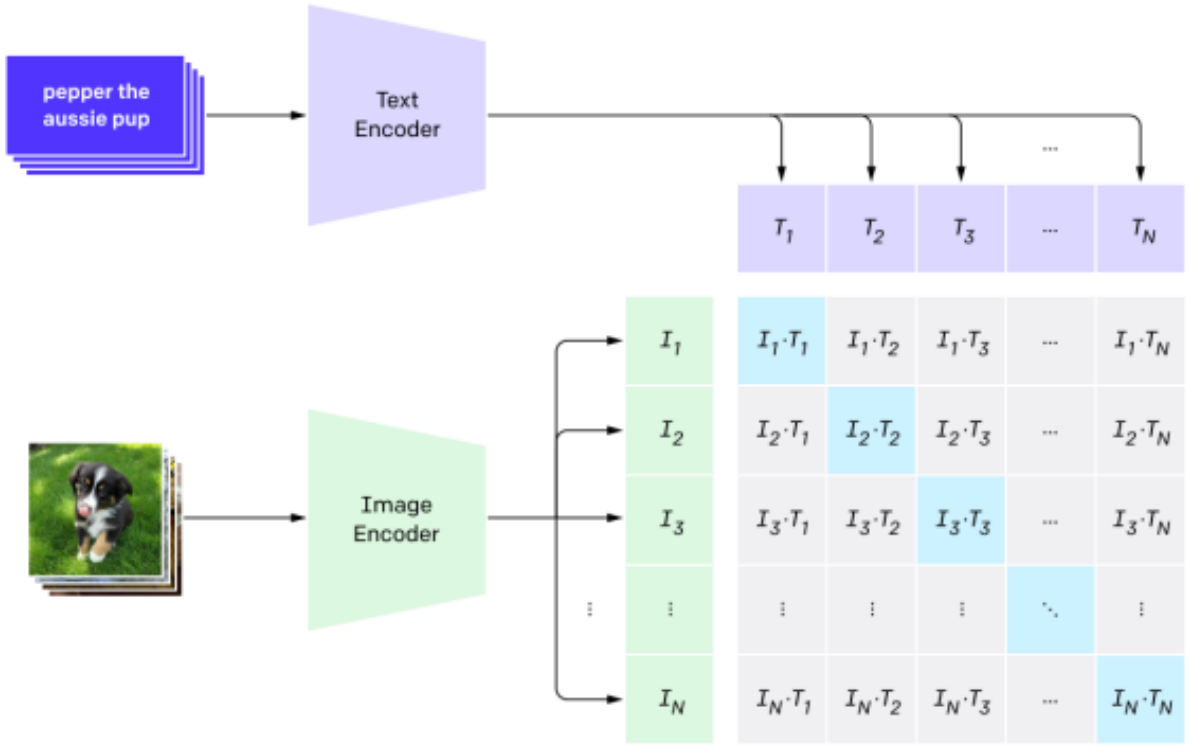

Figure 3.7: Visualization of the CLIP contrastive objective. After encoding the data, CLIP calculates a similarity matrix for the images and texts. Similarly to other contrastive methods, the objective is that the N true image-text pairs (positive pairs) score high in terms of similarity, while the other $N^2 - N$ possible combinations (negative pairs) have a low similarity. Figure from [24].

Variational Auto-Encoder (VQ-VAE) [98] - a VAE-based architecture that differs from the base VAE in that the encoder outputs discrete, instead of continuous, codes, and has a learnable, instead of fixed, prior distribution - and the transformer [97] architectures, that has significant zero-shot capabilities and is able to generate a wide array of images, and GLIDE [40], which is based on the transformer [97] and diffusion [99] architectures, that traded some diversity in image generation for increased fidelity, were the first architectures to attract significant commercial attention, as pre-trained versions of these

models brought high quality text-to-image generation to individuals outside of the DL research arena. These developments not only generated a massive increase of interest in multimodal learning research, but were also the source of some of the first multimodal representation learning algorithms to learn from large amounts of data in an efficient manner, such as Contrastive Language-Image Pre-Training (CLIP) [38], which the authors show is able to efficiently learn meaningful representations from a dataset comprised of 400 million (image, text) pairs using natural language supervision, and unCLIP [39], a hierarchical successor to the CLIP algorithm, which is comprised of two main stages: 1) a prior that, when given a text caption, generates a CLIP image embedding, and a decoder that generates an image, which is conditioned on the previously generated embedding.

3.2.1 MVAE

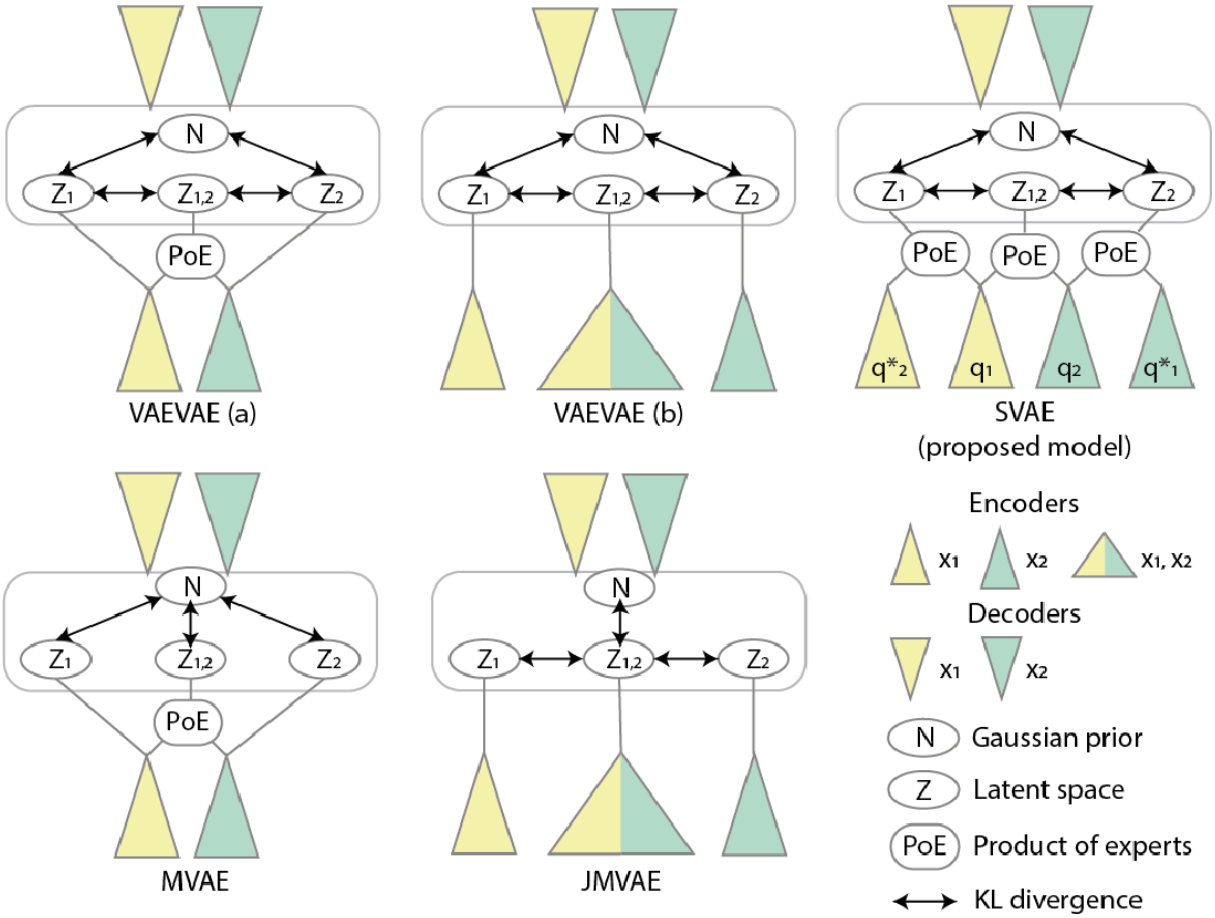

Figure 3.8: Overview of several VAE-based architectures, including the MVAE. Figure taken from [25].

The Multimodal Variational Auto-Encoder (MVAE) architecture, introduced by [100], is an extension of the VAE architecture designed to learn a joint distribution of multiple data modalities under weak supervision. Unlike previous architectures that extend the VAE model to multimodal scenarios by explicitly

learning the joint distribution, which do not efficiently generalize to datasets with more than two input modalities due to requiring an inference network for each combination of modalities [36], the MVAE architecture only requires one inference network for each input modality and combines the latent representations from all modalities by using a product-of-experts approach [101], meaning it can efficiently scale to datasets with several input modalities.

This is achieved by assuming that the M input modalities, $\{x_1, \ldots, x_M\}$, are conditionally independent given the common latent representation z, i.e., they assume that the generative model can be factorized as $p_\theta(x_1, \ldots, x_M, z) = p(z) \cdot p_\theta(x_1|z) \cdots p_\theta(x_M|z)$. Given this factorization, unobserved modalities can be ignored when evaluating the marginal likelihood. As such, if we write a datapoint X as a collection of the modalities present, i.e., $X = \{x_m : m^{th} \text{ modality present}\}$, the $\mathcal{L}_{ELBO}$ loss can be formalized as follows

$$\mathcal{L}_{ELBO}(X) = \mathbb{E}_{q_\theta(z|X)}[\sum_{x_m} \log p_\theta(x_m|z)] - \beta \cdot \mathsf{KLD}[q_\phi(z|X), p(z)]. \tag{3.24}$$

As the authors indicate, a relationship among the joint-modality and single-modality posteriors follows the conditional independence assumption:

$$\begin{aligned} p(z|x_1, \ldots, x_M) &= \frac{p(x_1, \ldots, x_M|z) \cdot p(z)}{p(x_1, \ldots, x_M)} \\ &= \frac{p(z)}{p(x_1, \ldots, x_M)} \prod_{m=1}^{M} p(x_m|z) \\ &= \frac{p(z)}{p(x_1, \ldots, x_M)} \prod_{m=1}^{M} \frac{p(z|x_m) \cdot p(x_m)}{p(z)} \\ &= \frac{\prod_{m=1}^{M} p(z|x_m)}{\prod_{m=1}^{M-1} p(z)} \cdot \frac{\prod_{m=1}^{M} p(x_m)}{p(x_m, \ldots, x_M)} \\ &\propto \frac{\prod_{m=1}^{M} p(z|x_m)}{\prod_{m=1}^{M-1} p(z)}. \end{aligned} \tag{3.25}$$

Then, assuming that the true posteriors for each individual factor $p(z|x_m)$ are contained in the family of its variational counterpart, $q(z|x_m)$, Equation 3.25 indicates that the correct $q(z|x_1, \ldots, x_M)$ is a product and quotient of experts: $\frac{\prod_{m=1}^{M} q(z|x_m)}{\prod_{m=1}^{M-1} p(z)}$. However, by approximating

$$p(z|x_m) \approx q(z|x_m) \equiv \bar{q}(z|x_m) \cdot p(z), \tag{3.26}$$

where $\bar{q}(z|x_m)$ is the underlying inference network, the quotient term can thus be avoided:

$$p(z|x_1, \ldots, x_M) \propto \frac{\prod_{m=1}^{M} p(z|x_M)}{\prod_{m=1}^{M-1} p(z)} \approx \frac{\prod_{m=1}^{M} \bar{q}(z|x_m) \cdot p(z)}{\prod_{m=1}^{M-1} p(z)} = p(z) \cdot \prod_{m=1}^{M} \bar{q}(z|x_m). \tag{3.27}$$

Thus, as shown in Equation 3.27, the joint-posterior distribution can be approximated by using a prod-

uct of experts and a prior expert $p(z)$, and this derivation can be extended to any possible subset of modalities, yielding the following equation for the MVAE model

$$q(z|X) \propto p(z) \cdot \prod_{x_m \in X} \hat{q}(z|x_m), \tag{3.28}$$

which has an analytical solution when $p(z)$ and $\hat{q}(z|x_m)$ are both Gaussian distributions.

3.2.2 GMC

The GMC framework, proposed by [26], is able to learn representations that are robust to missing modality information. This framework, as shown in Figure 3.9, consists of three main components:

- Collection of NN base encoders $f(\cdot) = \{f_{1:M}(\cdot)\} \bigcup \{f_1(\cdot), \ldots, f_M(\cdot)\}$, with $f_{1:M}$ taking the complete $x_{1:M}$ observation and f_m each modality specific observation x_m, and output intermediate d-dimensional representations $\{h_{1:M}, h_1, \ldots, h_M\} \in \mathbb{R}^d$;

- A NN shared projection head $g(\cdot)$ that maps the previously outputted intermediate representations $\{h_{1:M}, h_1, \ldots, h_M\} \in \mathbb{R}^d$ to the latent representations $\{z_{1:M}, z_1, \ldots, z_M\} \in \mathbb{R}^s$, where we will apply the contrastive term;

- The multimodal contrastive normalized NT-Xent ($\mathcal{L}_{GMC}$), based on the NT-Xent loss function [10], that encourages the geometric alignment of z_m and $z_{1:M}$.

This model uses geometric alignment through CL so that it can learn representations that are robust to missing modality information in several scenarios, such as in the context of UL. The $\mathcal{L}_{GMC}$ contrastive loss is key to this process of geometrically aligning z_m with the modalities $z_{1:M}$. Letting $B = \{z_{i,1:M}\}_{i=1}^{B} \subset h(f(X))$ be a mini-batch of B complete representations, sim$(\cdot)$ the cosine similarity and $\tau \in]0, \infty[$ the temperature hyper-parameter, we define,

$$s_{p,q}(z_i, z_j) = \exp(\text{sim}(z_{i,p}, z_{j,q})/\tau), \tag{3.29}$$

as the similarity between the representations $z_{i,p}$ and $z_{j,q}$, corresponding to the i-th and j-th samples of the mini-batch B, respectively. Furthermore, for a given modality m, we define positive pairs as $(z_{i,m}, z_{i,M:1})$ and $(z_{i,M:1}, z_{i,m})$ for $i = 1, \ldots, B$ and treat the remaining pairs in negative ones, denoting,

$$\Omega_{p,q}(z_i) = \sum_{i \neq j} s_{p,p}(z_i, z_j) + \sum_{j} s_{p,q}(z_i, z_j), \tag{3.30}$$

as the sum of similarities among the negative pairs corresponding to the $(z_{i,p}, z_{i,q})$ positive pair, and we define the contrastive loss for the previous positive pairs for modality m as the sum

$$\omega_m(z_i) = -\log \frac{s_{m,1:M}(z_i, z_i)}{\Omega_{m,1:M}(z_i)} - \log \frac{s_{1:M,m}(z_i, z_i)}{\Omega_{1:M,m}(z_i)}. \tag{3.31}$$

We then sum over each modality and mini-batch to obtain the total loss,

$$\mathcal{L}_{GMC}(B) = \sum_{m=1}^{M} \sum_{i=1}^{B} \omega_m(z_i). \tag{3.32}$$

Multimodal models robust to missing modalities such as GMC are a promising first step towards developing multimodal models that are robust to single-modality adversarial attacks, since, in theory, as long as the model has a single modality that is not targeted by the attack, it should not suffer from a relevant decrease in task performance as a result of adversarial attacks.

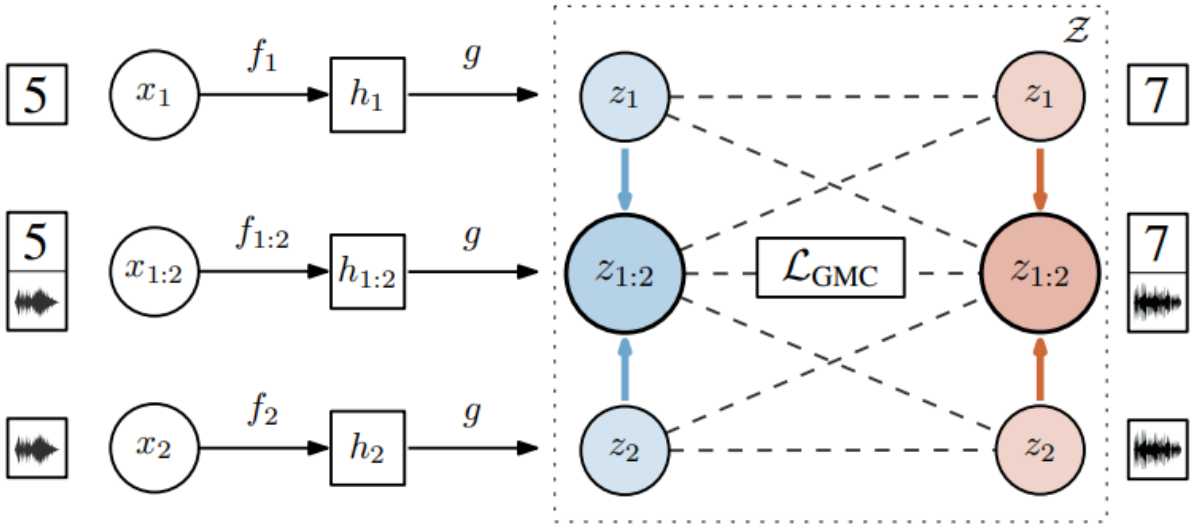

Figure 3.9: Overview of the GMC framework instantiated with two modalities. Figure taken from [26].

3.3 Robustness of Deep Multimodal Models

With the increasing introduction of multimodal models into several real world tasks with great societal consequences, such as detecting hate speech on social media and evaluating biomedical data, it has become increasingly important to analyse the robustness of these kinds of models to both noisy data and adversarial perturbations. Evaluating the robustness of these models is made more difficult due to the existence of a fusion mechanism to merge the information from the different modalities [102], which can range from a simple sum or mean of the latent representations to more complex fusion mechanisms

using product of experts or CL.

Recent results show that multimodal adversarial attacks are far more efficient at decreasing the performance of multimodal models than adversarial attacks targeting only a single-modality [102–104]. More interestingly, some results also seem to suggest that merging information between different modalities (e.g., audio and video) may even decrease the robustness of the models under multimodal adversarial attacks, rather than strengthen them [103]. Several defense strategies have been proposed to deal with attacks that target several modalities at once [103] or that target the fusion mechanism directly [102].

In this dissertation project, we will use the information obtained from multiple modalities to defend an model in scenarios in which he suffers perturbations to just a single modality, in line with other recent works [27, 28], without focusing on the fusion mechanism of the model. This follows our intuition that an adversary cannot easily perform attacks targeting multiple modalities or the fusion mechanism itself in most real world scenarios.

3.3.1 Robustness to single-modality adversarial perturbations

Although multimodal models combine input from several modalities, they are not inherently more robust to single-modality adversarial attacks than their unimodal counterparts. In particular, previous work [27] has shown that a multimodal model (e.g., with $m = 3$ modalities) under a single-modality attack may perform equivalently to a unimodal model which is subjected to the same adversarial attack.

Further studies demonstrated that a multimodal model was not equally vulnerable to adversarial perturbations in different modalities [28, 104, 105]. For example, on testing the robustness of multimodal models, which combined both image and text modalities, to single-modality perturbations, it was shown by [104] that they were far more vulnerable to image-only adversarial attacks than to text-only adversarial attacks. The same result was also reproduced by [105], where the perturbations on the video modality seemed to have more impact on the performance of a multimodal model than similar perturbations on the text modality. This is an interesting result that seems to require further study, as there are several possible hypothesis for this occurrence, and finding the causal link that leads to such a significant difference in vulnerability between modalities can have powerful implications in how to defend these models from single-modality attacks. We find four plausible hypothesis as possible explanations for this conundrum:

1. The inherent difference in complexity in the modalities and the amount of information they contribute to the latent representation;
2. The type of fusion mechanism used by the model to combine information from different modalities;
3. The impact of the data the models were trained on;
4. Current adversarial attacks are more effective against some modalities than others.

If hypothesis 1 was to hold, we would expect that these models would not be able to perform cross-modal transfer learning in the context of challenging tasks as they have been proven to do in [106], much less perform object recognition on images with zero-shot cross-modal transfer from training on text corpora only [107], since these results seem to contradict the idea that somehow some modalities are so much more complex than others, that their perturbation more significantly impact a model's performance. More specifically, the fact that text-only training data enables a model to compete with image-trained models on traditional Computer Vision tasks such as object detection [107], suggest that the greater vulnerability of multimodal models to image-only (when compared to text-only) single-modality adversarial attacks seen in [104], is not necessary to corroborate hypothesis 1.

A comparison of the mixture of experts approach used in [37] and the product of experts approach used in [100] to obtain the joint distribution of the different modalities, appears to give some weight to hypothesis 2, as the comparison of the results of the 2 approaches show how the PoE factorization of the joint variational posterior can lead to the model learning undesirable latent representations across modalities, as this method, which can suffer from the undue influence of overconfident experts due to miscalibrated precisions, will produce a joint distribution with low density across modalities, even if just one of the marginal posteriors has low density, while the MoE approach is more sensitive to information across all modalities [37], which shows that the methods used to learn the multimodal latent representation indeed effect the importance a model gives to a single modality and, as such, can also also result in that model being significantly more vulnerable to single-modality attacks in the modality the model attributes more importance to. Results from [27] also show that different methods of modality information fusion (e.g., concatenation and mean fusion) lead to different levels of vulnerabilities, however, the results in [104] show that different types of multimodal fusion fail under adversarial inputs at similar rates, which contradicts hypothesis 2, and implies that, even if it does play a role, a model's fusion mechanism is not the only factor at play with regards to a multimodal model robustness to single-modality adversarial perturbations.

For hypothesis 3, to the best of our knowledge, no studies have been performed to test the impact of the datasets used to train multimodal models on their robustness to single-modality adversarial perturbations. However, following the results from [108], which shows that the relationship between a unimodal model robustness and the dataset (and amount of data) on which it was trained is rather complex (i.e., it changes from dataset to dataset and more data doesn't necessarily mean more robustness), we expect that the relationship between a multimodal generative model's robustness and the dataset on which it is trained on will be at least as complex, likely even more so, due to components such as fusion mechanisms that are present in these models, but not in their unimodal counterparts.

Hypothesis 4 suffers from a similar issue as hypothesis 3, where not enough studies have been performed such that we could effectively confirm or deny this hypothesis. However, in [104], the authors

introduce 2 new kinds of text-based attacks they claim are similar to 2 of the types of image-based adversarial attacks they use during their experiments, suggesting that even under adversarial attacks of similar 'strength', the models will still not be equally vulnerable to adversarial perturbations to different modalities.

A summary of some of the most important conclusions are that, the difference in complexity of the modalities does seem to play a role in the predictions of some multimodal models, such as the product of experts approach proposed in [100], while other approaches, like the mixture of experts approach proposed in [37], appear able to deal with such difference in complexity, suggesting that, while hypothesis 1 may play a role, hypothesis 2 appears to be stronger, as some approaches to fusing the multiple modalities appear to produce multimodal models that can adequately handle the difference in complexity between modalities. Also, from the results in [104], hypothesis 4 appears not to play a significant role in this regard, and, while hypothesis 3 does intuitively seem to play a role in the robustness of the model to perturbations to each different modality, not enough work has been conducted to accurately measure the impacts of this compared to other concurrent hypotheses. The above results, although not conclusive, can provide an important window into possible avenues for how to use multimodal generative models' inherent qualities in order to provide an adequate defense from single-modality adversarial perturbations.

3.3.2 Defenses against single-modality adversarial attacks on multimodal models

The complex nature of multimodal models, when compared to their unimodal counterparts, makes studying their adversarial robustness a challenging problem and naively extending existing adversarial defenses from the unimodal to the multimodal scenario may not improve their performance and robustness. For example, studies have shown that to adversarial attacks in both unimodal [109] and multimodal models [105], the use of a pre-trained model can lead to an improvement in robustness. However, adversarial training, a strategy commonly used to increase the robustness of unimodal models [19], is an inefficient strategy for multimodal models due to the increased performance loss with clean data and the inability to scale to models with larger number of parameters [27]. However, since a multimodal model is able to use information provided by different modalities to represent the same scene, this can be leveraged to protect these models against single-modality adversarial attacks, an adversarial defense strategy not possible on their unimodal counterparts. In particular, to use the lack of consistency between modalities to detect an adversarial perturbation on one of the input modalities [27, 28].

3.3.2.A Adversarially Robust Fusion Mechanism

In the approach introduced by [27], the authors were able to leverage this idea by developing an adversarially robust fusion mechanism, shown in Figure 3.10. The proposed mechanism uses an odd-one-out network, which is based on the principle of odd-one-out learning [110, 111], to map a vector of features z to a vector of size $r+1$, where r_m is the probability of the z_m feature modality having been perturbed, and the last entry indicates the probability that none of the modalities where perturbed. Moreover, they propose a robust fusion layer consisting of the ensemble of $r+1$ feature fusion operations (i.e., $e_1, \ldots, e_{r+1}$), where only the last operation e_{r+1} fuses features from all modalities,

$$e_m(z) = \text{NN}(\oplus z_j)\ \forall j \in [r] \setminus \{m\}, \quad e_{r+1}(z) = \text{NN}(\oplus z_m)\ \forall m \in [r], \tag{3.33}$$

where NN is a shallow neural network and $\oplus$ is the concatenation operation. This way, if the feature z_m is not consistent with remaining ones, e_m (which excludes information from the m^{th} modality) will be given more weight than the other fusion operations $z_{out} = \sum_{m=1}^{M+1} e_i(z)o(z)_m$, due to the output of the odd-one-out network o. The robust fusion model is then formed by equipping the original fusion network f with these two additions (where f_{layer} is the original model with robust fusion layer and each g_m is a feature extractor for that modality):

$$f_{\text{robust}} := f_{\text{layer}}(g_1(x_1), ..., g_r(x_r)); o(\{g_m(x_m)\}_{m\in[r]}). \tag{3.34}$$

This model is trained with self-supervised training for the odd-one-out network o and adversarial training for the robust fusion network $\hat{f}$, to be able to defend itself against adversarial perturbations, while freezing the parameters of the feature extractors $g_1, ..., g_r$, to allow the model to maintain the same performance on clean data.

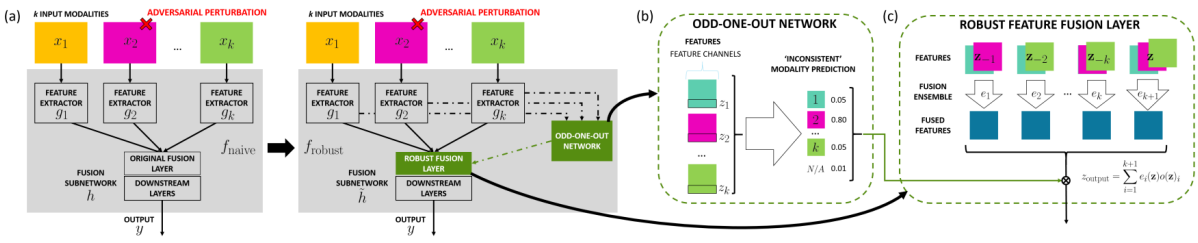

Figure 3.10: Overview of the robust fusion mechanism. Figure taken from [27].

3.3.2.B MATCH

A different approach inspired by this idea, introduced by [28], and denoted Multimodal Feature Consistency Check (MATCH), illustrated in Figure 3.11, first pre-trains two models for each modality separately, using only clean data, and then applying a projection to the output of their last FC layer (the extracted features) to bring the two feature sets into the same feature space:

$$\min_{\theta,\theta_2} MSE(p_{\theta_1}(f_1(x_1)) - p_{\theta_2}(f_2(x_2))), \tag{3.35}$$

where x_m is the m^{th} input modality, f_m and p_{θ_m} are the feature extractor and projector of x_m, respectively. Then, a consistency check model is trained only on clean data by minimizing the l_2 norm between the projected features of the two modalities. This information is combined into a system that can detect an adversarial example if the consistency level between two modalities is greater than a given threshold δ:

$$||p_{\theta_1}(f_1(x_1)) - p_{\theta_2}(f_2(x_2)))||_2^2 > \delta. \tag{3.36}$$

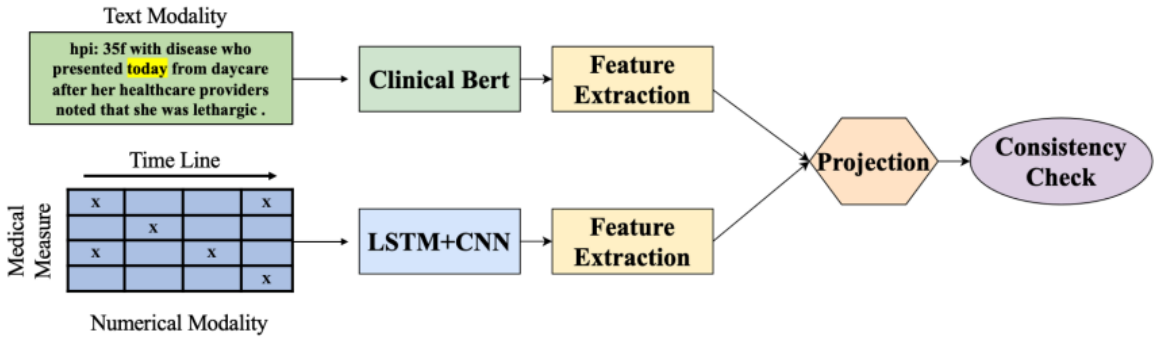

Figure 3.11: Overview of the MATCH detection pipeline. Figure taken from [28].

4

Implementation and Methodology

Contents

4.1 Model Architectures 41
4.2 Datasets 45
4.3 Noise and Adversarial Attacks 46

In order to evaluate the robustness of the representations learned by DL algorithms, we first trained several different DL architectures on a dataset in an unsupervised manner. In a second stage, we froze the unsupervised models weights, and trained a downstream classifier, which received the latent representations generated by the previously trained models as input, to perform a SL task. Finally, we tested the performance of the combined unsupervised model and downstream classifier in several different scenarios: 1) when of the modalities is removed from the input, 2) when one of the modalities is perturbed with random Gaussian noise and 3) when one of the modalities is targeted by an adversarial attack. We repeated this process for different datasets.

We only use the models with two different input modalities for each dataset, although every model can be easily extended to allow for an arbitrary number of modalities, and the equations will reflect this. If an equation does not account for an arbitrary number of modalities, we will indicate that after the equation. In terms of notation, we define $x_{i,m}$ as the i^{th} data sample of the m^{th} modality, z_m as the

latent representation of the m^{th} modality, N is the number of samples and M is the number of modalities. Also, in general, a $\sim$ above a given variable indicates a perturbed version of that variable, e.g., $\tilde{x}$ is a perturbed version of the sample x, and a $\hat{}$ above a given variable corresponds to a reconstruction of the original variable, e.g., $\hat{x}$ is a reconstruction of the original sample x.

For AE-based architectures, we define ϕ_m as the parameters of the encoder Enc_{ϕ_m} for the m^{th} modality and θ_m as the parameters of the decoder Dec_{θ_m} of the m^{th} modality. Since in the cases of the naive extensions of unimodal architectures to multimodal scenarios the encoder and decoder are the same for all modalities, we omit the m notation, i.e., we write ϕ and θ for the parameters, as well as Enc_ϕ and Dec_θ for the NNs.

For the GMC-based architectures, we define the base NN encoders as $f(\cdot) = \{f_{1:M}\} \bigcup \{f_1(\cdot), \ldots, f_M(\cdot)\}$ with parameters ϕ_m, i.e., $f\phi_m := f_m$, the NN shared projection head with parameters γ as $g_\gamma(\cdot)$, the base NN decoders as $\overline{f}(\cdot) = \{\overline{f}_{1:M}(\cdot)\} \bigcup \{\overline{f}_1(\cdot), \ldots, \overline{f}_M(\cdot)\}$ with parameters θ, i.e., $\overline{f}\theta_m := \overline{f}_m$, and the shared re-projection head with parameters ψ as $\overline{g}_\psi(\cdot)$. Also, we define o has an odd-one-out network, where $o(z)_m$ is the probability that the m^{th} modality is perturbed and $o(z)_{M+1}$ is the probability that none of the modalities has been perturbed.

4.1 Model Architectures

For our model architectures, we implemented several different unimodal and multimodal models to evaluate in this work. For the unimodal models, we first concatenate the several input modalities, and then feed them into the models.

4.1.1 Models based on unimodal Auto-Encoders

The first models we implemented were a naive extension of unimodal AEs to the multimodal scenario. Since in these architectures, the input modalities are all concatenated before being feed into the models, their structure is largely identical to their original counterparts. The only difference is that the reconstruction loss is extended to account for the multiple data modalities. We implemented two naive extensions of the unimodal AE architecture, described in 2.2.1.A, to deal with multimodal data:

- A model based on the DAE architecture, previously described in 2.2.1.B;
- An model based on the VAE architecture, previously described in 2.3.1.

For the naive multimodal DAE implementation, the new loss function is given by:

$$\mathcal{L}_{DAE}(x_{i,m}) = \frac{1}{N} \sum_{m=1}^{M} \sum_{i=1}^{N} \lambda_m (Dec_\theta(Enc_\phi(\tilde{x}_{i,m})) - x_{i,m})^2 = \frac{1}{N} \sum_{m=1}^{M} \sum_{i=1}^{N} \lambda_m \cdot (\hat{x}_{i,m} - x_{i,m})^2, \quad (4.1)$$

where λ_m is a scalar parameter that defines the weight of the reconstruction error of the m^{th} modality on the total loss.
In the case of the naive multimodal VAE implementation, taking the evidence as being the reconstruction loss, the new loss function is given by:

$$\mathcal{L}_{VAE}(x_{i,m}) = \beta \sum_{i=1}^{N}(1 + \log(\sigma_i)^2 - \mu_i^2 - \sigma_i^2) + \frac{1}{N}\sum_{m=1}^{M}\sum_{i=1}^{N}\lambda_m \cdot (\hat{x}_{i,m} - x_{i,m})^2. \tag{4.2}$$

4.1.2 Models based on Multimodal Auto-Encoders

After the naive extension of unimodal AE models to multimodal scenarios, we also implemented several multimodal AEs, which use different networks to encode and decode each modality, as well as a mechanism to fuse the various latent representations $z = \{z_1, \ldots, z_M\}$. We implemented five different multimodal AEs:

- A model based on the MVAE architecture, previously described in 3.2.1;
- The Multimodal Denoising Auto-Encoder (MDAE), an extension of the DAE architecture with an encoder Enc_{ϕ_m} and decoder Dec_{θ_m} for each data modality, and which fuses the latent representations into a shared latent representation through a simple mean, i.e., $z_{1:M} = \frac{1}{M}\sum_m^M z_m$;
- The Common Multimodal Denoising Auto-Encoder (CMDAE), another extension of the DAE architecture, that is similar to the MDAE, which has an additional encoder $Enc_{\phi_{1:M}}$, used to process the sum of the latent representations, i.e., $z_{1:M} = Enc_{\phi_{1:M}}(\sum_m^M z_m)$, and decoder $Dec_{\theta_{1:M}}$;
- The Common Multimodal Variational Auto-Encoder (CMVAE), an extension of the VAE architecture that is similar to the CMDAE;
- The Common Multimodal Denoising Variational Auto-Encoder (CMDVAE), which has the same architecture has the CMVAE model, but that is also trained to remove noise from inputs, similar to a DAE.

The loss of the MDAE model can be defined as:

$$\mathcal{L}_{MDAE}(x_{i,m}) = \frac{1}{N}\sum_{m=1}^{M}\sum_{i=1}^{N}\lambda_m(Dec_{\theta_m}(Enc_{\phi_m}(\tilde{x}_{i,m})) - x_{i,m})^2 = \frac{1}{N}\sum_{m=1}^{M}\sum_{i=1}^{N}\lambda_m \cdot (\hat{x}_{i,m} - x_{i,m})^2. \tag{4.3}$$

In turn, the loss of the CMDAE model can be defined as:

$$\mathcal{L}_{CMDAE}(x_{i,m}) = \frac{1}{N}\sum_{m=1}^{M}\sum_{i=1}^{N}\lambda_m(Dec_{\theta_m}(Dec_{\theta_{1:M}}(z_{1:M})) - x_{i,m})^2 = \frac{1}{N}\sum_{m=1}^{M}\sum_{i=1}^{N}\lambda_m \cdot (\hat{x}_{i,m} - x_{i,m})^2. \tag{4.4}$$

The CMVAE and the CMDVAE loss function, which is similar to the CMDAE reconstruction loss with an additional term for the KLD, can be defined as:

$$\mathcal{L}_{CMVAE}(x_{i,m}) = \beta \sum_{i=1}^{N}(1 + \log(\sigma_i)^2 - \mu_i^2 - \sigma_i^2) + \frac{1}{N}\sum_{m=1}^{M}\sum_{i=1}^{N}\lambda_m \cdot (\hat{x}_{i,m} - x_{i,m})^2. \tag{4.5}$$

4.1.3 Models based on the Geometric Multimodal Contrastive framework

Beyond the use of architectures based on AEs, we also implemented several models based on the GMC representation learning framework. We implemented four different versions of the GMC framework:

- A model based on the GMC framework, previously described in 3.2.2;
- The GMCWD, illustrated in Figure 4.1, a extension of the GMC framework which has an additional base NN joint decoder $\overline{f}_{\theta_{1:M}}$ and a shared re-projection head $\overline{g}_{\psi}$, where Gaussian noise is added to one of the input modalities;
- The DGMC, illustrated in Figure 4.2, a extension of GMCWD which has an additional base NN decoder for each modality, i.e., $\overline{f}_{\theta} = \{\overline{f}_{\theta_{1:M}}\} \bigcup \{\overline{f}_{\theta_1}, \ldots, \overline{f}_{\theta_M}\}$, and where, after the training process, the latent representations are obtained from the reconstructed samples, i.e., $z_m = g_\gamma(f_{\phi_m}(\overline{f}_{\theta_m}(\overline{g}_\psi(g_\gamma(f_{\phi_m}(\tilde{x}))))))$;
- The RGMC, illustrated in Figure 4.3, which is an extension of the GMC framework with an additional odd-one-out network o, similar to the one discussed in [110], and where the output of the model will be given by the product of the odd-one-out network predictions and the latent representations, i.e., $z_{\text{out}} = z_{1:M} \cdot o(z)_{M+1} + \sum_m^M z_m \cdot (\sum_j^M o(z)_{j \neq m})$.

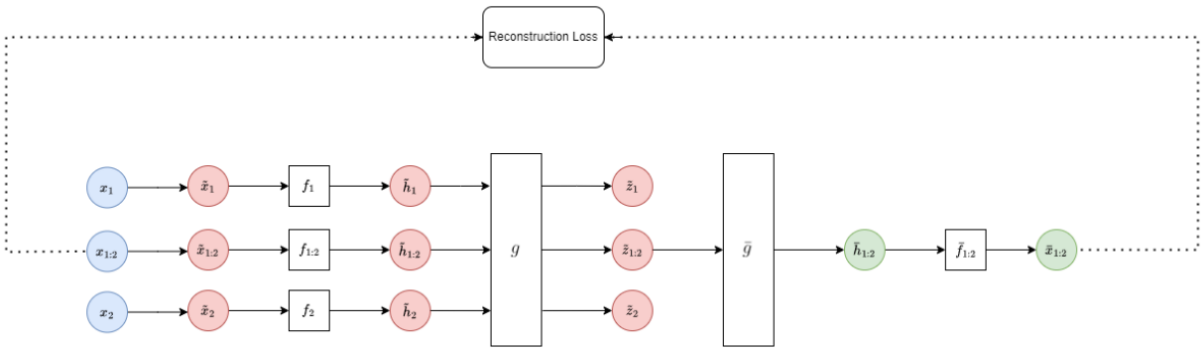

Figure 4.1: Overview of the GMCWD framework instantiated with 2 modalities. Blue circles denote clean data, red circles noisy data and encodings, and green circles reconstructions and respective intermediate representations.

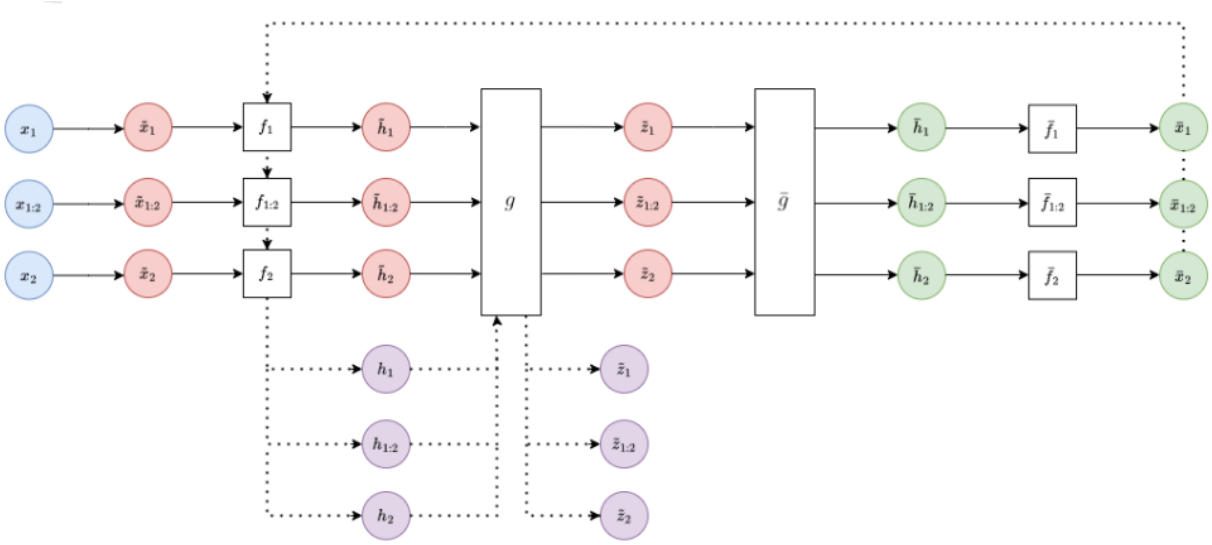

Figure 4.2: Overview of the DGMC architecture for 2 modalities. Blue circles denote clean data, red circles noisy data and encodings, green circles reconstructions and respective intermediate representations, and purple circles the encodings with the reconstructions as input.

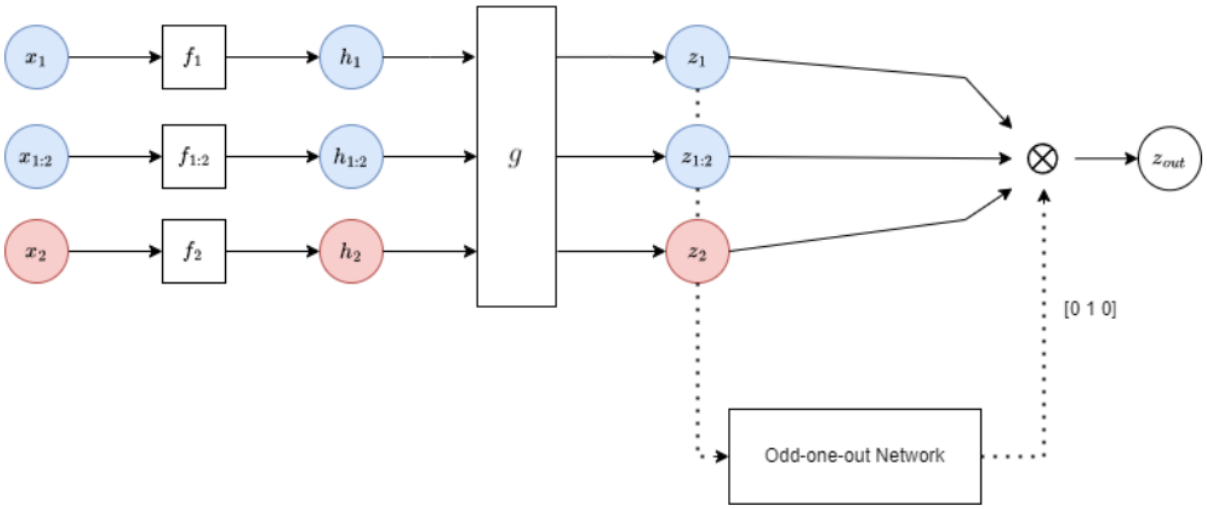

Figure 4.3: Overview of the RGMC architecture for 2 modalities. Blue circles denote clean modalities and red circles modalities under adversarial attack.

Even though GMCWD and DGMC have different architectures, they share the same loss function, which is defined as:

$$\mathcal{L}_{GMCWD}(x_{i,m}) = \mathcal{L}_{DGMC}(x_{i,m}) = \sum_{m=1}^{M}\sum_{i=1}^{N}\omega_m(z_{i,m}) + \frac{1}{N}\sum_{m=1}^{M}\sum_{i=1}^{N}\lambda_m(\hat{x}_{i,m} - x_{i,m})^2. \tag{4.6}$$

Combining a contrastive loss with a reconstruction loss has been previously used in single-modality settings [112], where it showed remarkable results in unsupervised learning of robust latent representa-

tions. Also, by adding Gaussian noise to the input, minimizing the reconstruction loss trains the GMCWD and DGMC architectures to remove perturbations from their input, resulting in more robust models.
For the RGMC model, we simplify the loss of the odd-one-out network to a supervised BCE loss function, where this network is trained to predict which modality was perturbed by a adversarial example produced by FGSM, and the total loss is given by:

$$\mathcal{L}_{RGMC}(x_{i,m}, y) = \sum_{m=1}^{M}\sum_{i=1}^{N}\omega_m(z_{i,m}) + \sum_{i=1}^{N}\lambda_o[y_i \cdot \log \text{sigmoid}(x_i) + (1-y_i)\cdot\log(1-\text{sigmoid}(x_i))]. \quad (4.7)$$

4.2 Datasets

For the training and testing of the various models, we utilized two multimodal datasets:

- The Multimodal Handwritten Digits (MHD) dataset, which combines image, trajectory, sound and label data modalities for handwritten digits, where the training set has 50000 - and the test set 10000 - equally distributed samples, from which we used the image and trajectory modalities;
- The MNIST-SVHN dataset [42], a dataset of images of digits which is comprised of the MNIST dataset and the Street View House Numbers (SVHN) dataset as the two different modalities, where the training set has 56068 samples and the test set x samples, with the sample distributions shown in Figure 4.4 and in Figure 4.5, respectively.

All modalities of both datasets where normalized, such that each datapoint $x_{i,m}$ becomes:

$$x_{i,m} = \frac{x_{i,m} - \min_j(x_{j,m})}{\max_j(x_{j,m}) - \min_j(x_{j,m})}. \quad (4.8)$$

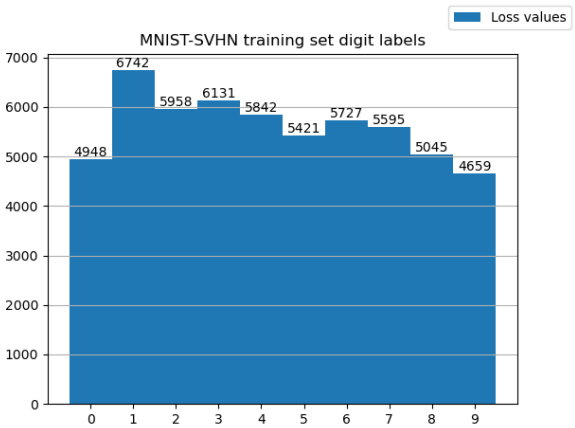

Figure 4.4: Distribution of digit labels in the MNIST-SVHN training dataset.

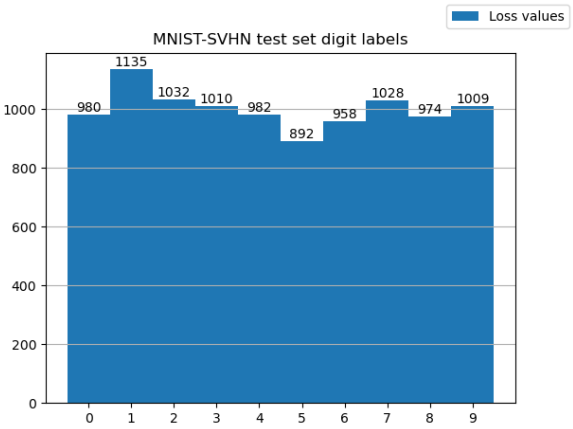

Figure 4.5: Distribution of digit labels in the MNIST-SVHN test dataset.

4.3 Noise and Adversarial Attacks

In order to test the robustness of the several architectures we injected multiple types of adversarial attacks and noise into the input data at test time. We used four different types of perturbations:

- Gaussian noise $\eta \sim \mathcal{N}(\mu, \sigma^2)$, where we set the mean as $\mu = 0$, and the standard deviation σ is a parameter that defines how much the noise affects the perturbed sample $\tilde{x} = x + \eta$;
- A – FGSM adversarial attack;
- B – BIM adversarial attack;
- C – PGD adversarial attack.

5

Evaluation Experiments

Contents

5.1 Experiments on the MHD dataset 48

5.2 Experiments on the MNIST-SVHN dataset 54

For the experiments, we first train each model in an unsupervised manner on each dataset and then evaluate the performance across the several models by training a downstream classifier which receives as input the latent representations encoded by the respective model. For each architecture, we consider ten runs with different seeds, and then calculate the average and standard deviation of the accuracy at test time.

To evaluate the robustness of each architecture, we compare the metrics of each model on purely clean data, with missing modalities, with Gaussian noise added to each modality with multiple standard deviation values and for each different type of adversarial attack.

We also show the result of the noise perturbation applied to the different modalities in each dataset, where the image modality on the MHD dataset corresponds to the MNIST modality on the MNIST-SVHN dataset.

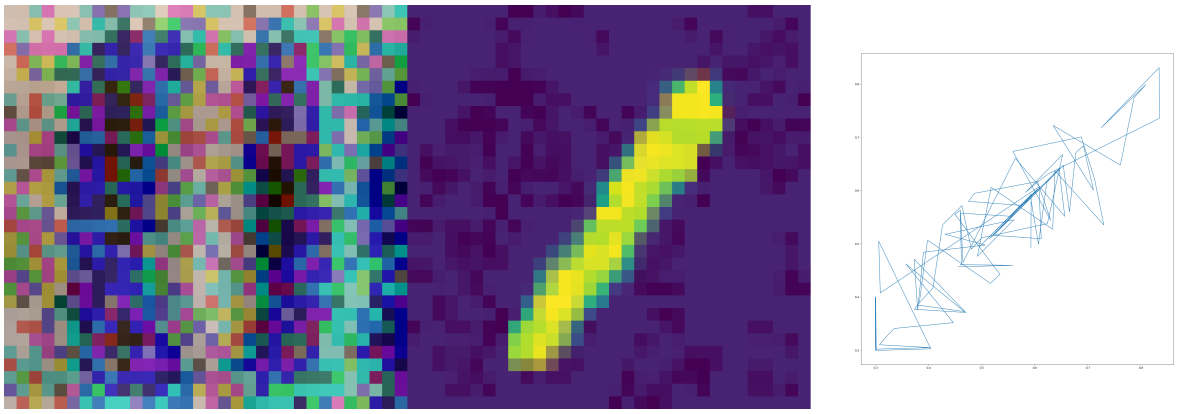

Figure 5.1: Adversarial examples generated by the FGSM adversarial attack with $\epsilon = 0.2$, targeting each modality. At this value of ϵ, the perturbations in many samples start to be highly visible to the human eye.

5.1 Experiments on the MHD dataset

During our training runs, we noticed that the GMCWD and DGMC models converged faster and, unlike the GMC and RGMC models, never failed to converge during training. These results, illustrated in Figure 5.2, suggest that adding the reconstruction loss can benefit the GMC framework not only in adding robustness against noise, but also in regularizing model training.

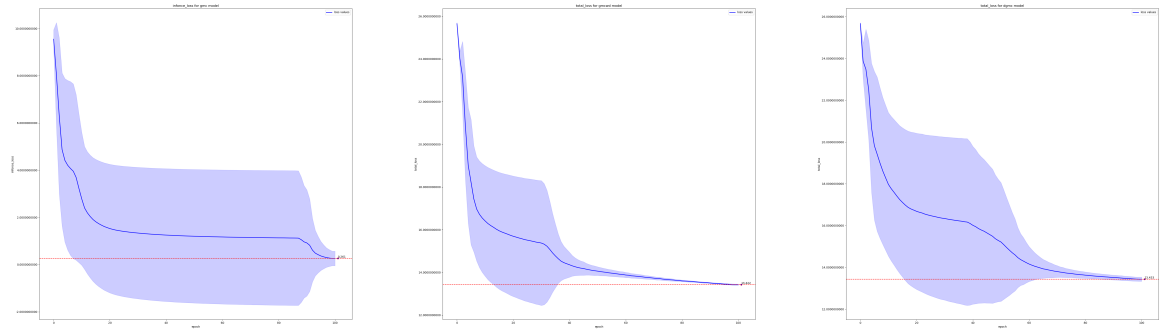

Figure 5.2: Comparison of the loss values of the GMC, GMCWD and DGMC architectures across the ten unsupervised training runs. The dark blue line equals the mean and shadowed blue area corresponds to the standard deviation.

After training the base models in an unsupervised manner and the downstream classifiers in the supervised classification task, we then performed an initial evaluation of the combined models by testing them in the test set. We obtained high classification accuracy in all models, as can be seen in Figure 5.3, a good result since any deviations from that accuracy we obtain from later results should be due to input perturbations and not due to the model's lack of expressiveness.

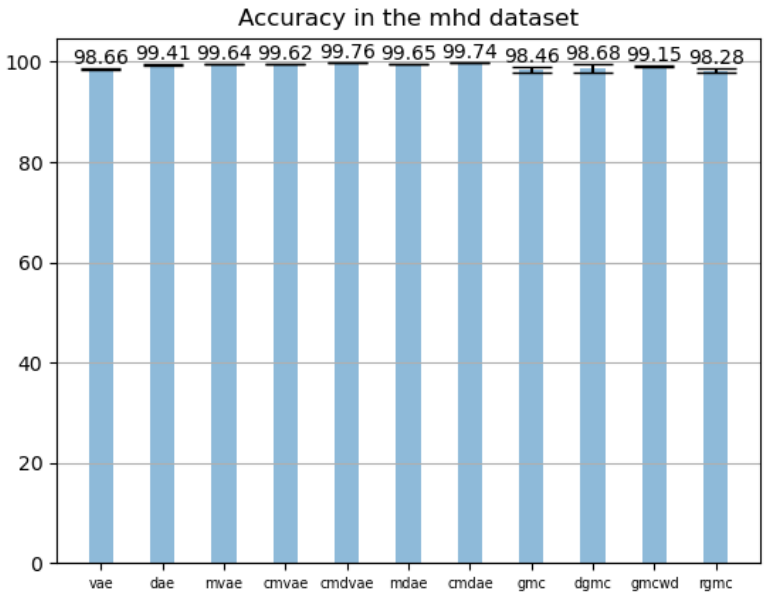

Figure 5.3: Model accuracy in the MHD dataset on clean data with all modalities present.

In our second evaluation experiment, we tested the models with one of the modalities missing, where the results can be seen in Table 5.1. While the unimodal models struggled with any of the modalities missing, the multimodal AEs generally maintained a good accuracy with the trajectory modality missing, but performed rather poorly when the image modality was missing, suggesting their output is largely dependent on one of the modalities. The GMC-based models all performed well with either modality missing, as expected.

	Excluded Modality		
	None	Image	Trajectory
DAE	99.41 ± 0.04	13.45 ± 3.78	27.30 ± 5.57
VAE	98.66 ± 0.14	10.37 ± 1.14	28.82 ± 8.58
MDAE	99.65 ± 0.04	38.61 ± 5.24	98.00 ± 0.49
MVAE	99.64 ± 0.05	13.25 ± 3.36	65.15 ± 14.60
CMDAE	99.74 ± 0.03	36.97 ± 4.16	98.24 ± 0.45
CMDVAE	99.76 ± 0.03	29.07 ± 2.00	98.65 ± 0.56
CMVAE	99.62 ± 0.03	21.08 ± 5.25	85.34 ± 9.49
GMC	98.44 ± 0.52	96.11 ± 1.27	95.62 ± 1.31
GMCWD	99.15 ± 0.07	85.68 ± 1.25	91.15 ± 1.16
DGMC	98.68 ± 0.75	80.69 ± 7.04	89.43 ± 3.87
RGMC	98.28 ± 0.57	95.31 ± 2.24	96.03 ± 1.70

Table 5.1: Accuracy results for models with all modalities present (None) and with one of the modalities missing on the MHD dataset.

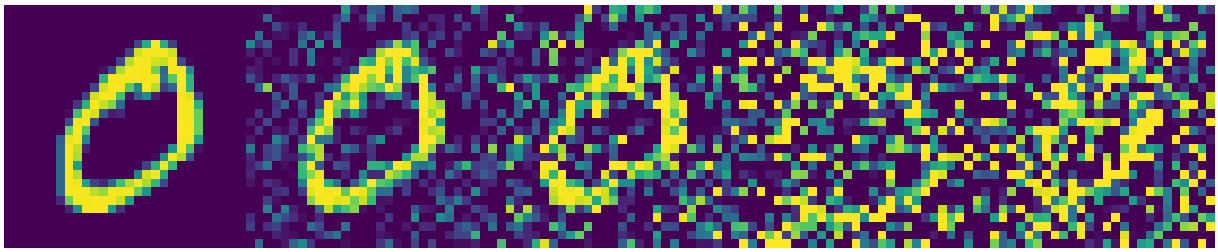

Figure 5.4: Examples of a image of the digit zero perturbed by Gaussian noise. Standard deviation values for the Gaussian noise are: 0, 0.25, 0.5, 0.75 and 1.

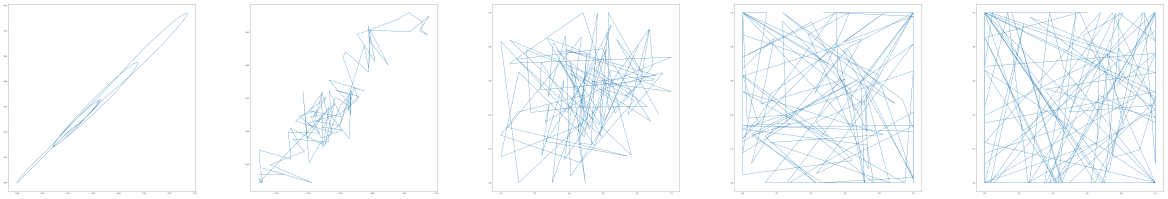

Figure 5.5: Examples of a trajectory of the digit eight perturbed by Gaussian noise. Standard deviation values for the Gaussian noise are: 0, 0.25, 0.5, 0.75 and 1.

		Gaussian Noise			
Std (σ)		0.25	0.5	0.75	1
Image	DAE	98.79 ± 0.09	89.93 ± 0.96	71.21 ± 1.79	56.24 ± 1.88
	VAE	85.78 ± 1.96	52.35 ± 2.76	33.98 ± 2.68	25.93 ± 2.13
	MDAE	99.10 ± 0.13	90.50 ± 1.41	65.18 ± 2.64	44.48 ± 3.22
	MVAE	91.62 ± 2.84	63.89 ± 6.65	43.16 ± 6.97	32.37 ± 6.09
	CMDAE	99.44 ± 0.09	95.82 ± 0.51	78.95 ± 1.96	59.02 ± 3.46
	CMDVAE	**99.50±0.05**	96.07 ± 0.36	78.31 ± 1.79	57.55 ± 2.90
	CMVAE	82.64 ± 6.97	34.37 ± 11.14	20.76 ± 5.58	16.77 ± 4.00
	GMC	95.61 ± 0.76	85.35 ± 1.31	77.75 ± 2.12	73.78 ± 2.54
	GMCWD	98.70 ± 0.13	**96.64±0.32**	93.05 ± 0.70	89.72 ± 1.06
	DGMC	98.01 ± 0.93	94.56 ± 2.69	88.46 ± 5.63	82.63 ± 8.06
	RGMC	95.21 ± 2.40	95.06 ± 2.83	**94.99±3.01**	**94.98±3.05**
Trajectory	DAE	99.32 ± 0.05	98.90 ± 0.06	98.04 ± 0.17	97.09 ± 0.32
	VAE	98.37 ± 0.19	97.48 ± 0.27	96.41 ± 0.43	95.44 ± 0.53
	MDAE	99.58 ± 0.07	99.27 ± 0.08	98.67 ± 0.18	98.04 ± 0.27
	MVAE	99.56 ± 0.07	99.17 ± 0.21	98.48 ± 0.57	97.81 ± 0.78
	CMDAE	99.66 ± 0.04	99.48 ± 0.07	99.02 ± 0.11	98.55 ± 0.17
	CMDVAE	**99.74±0.04**	**99.63±0.08**	**99.38±0.15**	**99.13±0.23**
	CMVAE	99.50 ± 0.04	98.94 ± 0.14	98.00 ± 0.32	97.17 ± 0.52
	GMC	79.93 ± 2.00	57.70 ± 3.22	45.38 ± 3.04	38.91 ± 2.79
	GMCWD	92.89 ± 0.51	77.31 ± 1.30	64.92 ± 1.74	57.12 ± 1.92
	DGMC	94.73 ± 0.93	84.56 ± 4.71	75.83 ± 7.63	70.03 ± 9.43
	RGMC	80.78 ± 4.66	79.56 ± 4.34	81.06 ± 3.57	82.29 ± 3.47

Table 5.2: Accuracy results for models with Gaussian noise targeting each of the modalities on the MHD dataset.

Our third experiment involved injecting Gaussian noise while altering the σ parameter, to test each

model's robustness on noisy data, where the results can be seen in Table 5.2. We show examples of the effect of the noise on the image modality in Figure 5.4 and on the trajectory modality in Figure 5.5. As expected from the previous results, the AE-based models did not struggle with noisy trajectories, but were extremely vulnerable to noisy images, specially the VAE-based models, again suggesting that their output is dependent on the image features. In contrast, the GMC-based models were all relatively robust to noisy image data, but were in general less robust than the multimodal DAEs against noisy trajectories, with the base GMC model performing the worst compared with all models, including the unimodal AEs, suggesting that the geometrical alignment of modalities may make the GMC framework less robust to noise that targets the less complex modality. All GMC extensions were more robust to noise than the base GMC model, with the RGMC model outperforming all other models with very noisy images and maintaining an accuracy near 80% even with high trajectory noise, while both the GMCWD and the DGMC models were able to outperform all other models with noisy images, they fared less well with noisy trajectories, although the DGMC was still able to average around 70% accuracy.

FGSM Adversarial Attack						
Epsilon (ϵ)		0.015	0.035	0.075	0.13	0.2
Image	DAE	98.86 ± 0.06	97.39 ± 0.13	87.35 ± 0.35	50.47 ± 1.52	11.79 ± 0.89
	VAE	96.81 ± 0.25	89.73 ± 0.68	51.05 ± 1.45	11.78 ± 0.85	1.66 ± 0.34
	MDAE	99.38 ± 0.07	98.77 ± 0.09	95.74 ± 0.25	83.27 ± 0.62	55.50 ± 1.93
	MVAE	98.70 ± 0.11	92.91 ± 0.63	50.44 ± 1.90	7.55 ± 1.88	0.56 ± 0.33
	CMDAE	99.50 ± 0.05	98.43 ± 0.07	96.15 ± 0.24	84.92 ± 0.77	57.13 ± 1.51
	CMDVAE	**99.60±0.04**	**99.11±0.06**	**96.69±0.16**	85.31 ± 0.31	54.55 ± 1.55
	CMVAE	98.47 ± 0.10	88.50 ± 1.54	33.98 ± 2.75	5.14 ± 1.00	1.38 ± 0.35
	GMC	96.80 ± 0.90	91.81 ± 1.44	71.25 ± 2.15	41.28 ± 2.04	20.63 ± 2.30
	GMCWD	98.44 ± 0.15	96.61 ± 0.25	87.73 ± 0.79	60.17 ± 1.78	25.18 ± 1.95
	DGMC	97.64 ± 1.22	95.09 ± 2.31	82.94 ± 6.85	53.20 ± 9.83	20.40 ± 6.00
	RGMC	94.68 ± 1.46	91.73 ± 2.70	90.31 ± 4.78	**89.25±7.21**	**88.09±0.22**
Trajectory	DAE	99.10 ± 0.04	98.47 ± 0.10	95.30 ± 0.29	83.05 ± 1.33	59.06 ± 1.56
	VAE	97.95 ± 0.22	96.34 ± 0.38	89.34 ± 1.22	68.77 ± 2.58	37.12 ± 3.20
	MDAE	99.42 ± 0.03	98.88 ± 0.08	96.19 ± 0.43	85.49 ± 1.38	65.96 ± 2.80
	MVAE	99.47 ± 0.06	99.01 ± 0.13	94.99 ± 1.56	78.23 ± 5.74	49.57 ± 7.34
	CMDAE	99.59 ± 0.05	99.20 ± 0.06	97.46 ± 0.24	89.65 ± 0.89	71.30 ± 1.84
	CMDVAE	**99.65±0.04**	**99.46±0.06**	**98.45±0.23**	**93.79±1.14**	79.17 ± 2.90
	CMVAE	99.40 ± 0.06	98.71 ± 0.18	93.43 ± 1.11	75.84 ± 2.74	50.81 ± 4.24
	GMC	84.02 ± 1.86	48.99 ± 3.35	19.22 ± 2.67	9.40 ± 2.32	6.11 ± 1.66
	GMCWD	94.87 ± 0.39	74.38 ± 1.50	22.98 ± 3.69	3.69 ± 3.03	1.29 ± 2.03
	DGMC	95.34 ± 0.45	81.90 ± 5.38	42.41 ± 22.73	21.75 ± 25.62	15.33 ± 20.95
	RGMC	84.40 ± 2.03	74.25 ± 4.69	85.60 ± 7.15	93.25 ± 6.15	**94.65±5.42**

Table 5.3: Accuracy results for models with the FGSM adversarial targeting each of the modalities on the MHD dataset.

In our first experiment with adversarial attacks targeting each modality, we tested each model with adversarial perturbations generated with the FGSM algorithm with varying ϵ values, the results of which can be seen in Table 5.3. As we can see, all unimodal models and most multimodal VAEs performed

poorly with adversarial samples in both modalities. However, unlike previous results, most GMC-based models performed worse than multimodal DAEs and, even though the GMCWD and DGMC models were more robust than the base GMC model, which performed even worse than the DAE when $\epsilon = 0.13$, they performed rather poorly with high values of ϵ. The RGMC model maintained an accuracy above 74% in all scenarios and was able to average around 90% accuracy with high ϵ values, which was above the accuracy it obtained when the $\epsilon \leq 0.075$, suggesting that adversarial perturbations with ϵ values near or above 0.1, the value with which the RGMC framework was trained on, may be easier for the odd-one-out network to detect.

BIM Adversarial Attack							
Epsilon (ϵ)		0.1		0.2		0.3	
Alpha (α)		0.02	0.04	0.06	0.08	0.1	0.12
Image	DAE	98.60 ± 0.09	96.69 ± 0.11	92.11 ± 0.24	83.07 ± 0.66	68.27 ± 1.45	50.34 ± 1.53
	VAE	95.57 ± 0.31	85.34 ± 0.87	62.48 ± 1.73	35.74 ± 1.25	16.93 ± 0.85	6.77 ± 0.56
	MDAE	99.24 ± 0.06	98.52 ± 0.10	97.16 ± 0.20	94.57 ± 0.26	90.17 ± 0.42	83.44 ± 0.72
	MVAE	97.92 ± 0.16	88.50 ± 0.88	62.80 ± 2.07	30.72 ± 2.70	11.42 ± 2.17	3.51 ± 1.35
	CMDAE	99.39 ± 0.03	98.71 ± 0.07	97.46 ± 0.13	95.07 ± 0.40	91.10 ± 0.67	85.03 ± 0.88
	CMDVAE	99.51±0.04	98.93±0.07	97.80±0.08	95.75±0.17	91.98±0.20	85.87 ± 0.22
	CMVAE	97.22 ± 0.29	80.47 ± 2.61	47.24 ± 3.65	20.00 ± 2.38	7.54 ± 1.47	7.54 ± 1.47
	GMC	95.86 ± 1.00	88.95 ± 1.49	76.31 ± 1.89	60.92 ± 1.76	46.11 ± 1.82	46.11 ± 1.82
	GMCWD	98.07 ± 0.14	95.92 ± 0.28	91.75 ± 0.49	84.54 ± 1.08	73.89 ± 1.57	73.89 ± 1.57
	DGMC	97.14 ± 1.43	93.99 ± 2.83	87.93 ± 5.53	78.59 ± 8.54	66.17 ± 10.65	66.17 ± 10.65
	RGMC	92.91 ± 1.91	90.74 ± 3.55	90.41 ± 5.55	90.29 ± 7.48	90.38 ± 9.49	98.98±11.26
Trajectory	DAE	98.97 ± 0.06	98.20 ± 0.09	96.79 ± 0.16	94.11 ± 0.33	89.68 ± 0.76	83.40 ± 1.19
	VAE	97.61 ± 0.25	95.71 ± 0.48	92.46 ± 0.83	86.97 ± 1.54	79.01 ± 2.13	68.97 ± 2.76
	MDAE	99.31 ± 0.05	98.71 ± 0.07	97.52 ± 0.25	95.20 ± 0.50	91.40 ± 0.92	86.08 ± 1.38
	MVAE	99.38 ± 0.08	98.77 ± 0.19	97.09 ± 0.84	93.20 ± 2.35	86.63 ± 4.46	86.63 ± 4.46
	CMDAE	99.50 ± 0.04	99.07 ± 0.09	98.28 ± 0.15	96.83 ± 0.33	94.35 ± 0.53	90.37 ± 0.89
	CMDVAE	99.61±0.03	99.36±0.08	98.92±0.17	98.18±0.01	96.79±0.55	94.52±1.05
	CMVAE	99.28 ± 0.07	98.39 ± 0.24	96.20 ± 0.72	91.43 ± 1.60	84.31 ± 2.22	84.31 ± 2.22
	GMC	69.98 ± 4.03	30.53 ± 3.33	14.64 ± 2.05	10.69 ± 2.81	10.63 ± 3.21	10.63 ± 3.21
	GMCWD	91.35 ± 0.69	62.37 ± 1.91	30.08 ± 3.04	11.50 ± 4.06	4.66 ± 4.29	4.66 ± 4.29
	DGMC	92.87 ± 0.91	74.03 ± 8.59	49.39 ± 19.69	32.30 ± 25.80	24.00 ± 26.71	24.00 ± 26.71
	RGMC	83.19 ± 1.70	73.96 ± 3.71	79.35 ± 6.01	86.79 ± 6.09	91.12 ± 5.68	93.18 ± 5.47

Table 5.4: Accuracy results for models with the BIM adversarial attack targeting each of the modalities on the MHD dataset, where we set the number of steps $\mathcal{K} = 10$.

For our second experiment with adversarial examples, we used the BIM adversarial attack with different combinations of the ϵ and α parameters, while setting the number of steps to 10, where the results can be seen in Table 5.4. As expected, since BIM is an iterative version of FGSM, most results were similar to the ones in the previous experiment.

PGD Adversarial Attack							
Epsilon (ϵ)		0.1		0.2		0.3	
Alpha (α)		0.01	0.02	0.04	0.06	0.08	0.1
Image	DAE	94.89 ± 0.14	94.86 ± 0.13	68.27 ± 1.46	68.20 ± 1.45	4.66 ± 0.54	4.64 ± 0.53
	VAE	75.56 ± 1.56	75.36 ± 1.55	15.52 ± 0.89	15.44 ± 0.87	0.03 ± 0.01	0.03 ± 0.01
	MDAE	97.98 ± 0.15	97.97 ± 0.15	90.10 ± 0.41	90.08 ± 0.42	36.05 ± 1.23	35.93 ± 1.20
	MVAE	78.08 ± 1.38	77.82 ± 1.41	10.77 ± 2.13	10.72 ± 2.14	0.03 ± 0.02	0.03 ± 0.01
	CMDAE	98.23 ± 0.09	98.22 ± 0.09	91.09 ± 0.67	91.07 ± 0.68	39.49 ± 1.81	39.38 ± 1.78
	CMDVAE	**98.44±0.07**	**98.43±0.07**	**91.96±0.21**	**91.94±0.21**	36.96 ± 1.94	36.85 ± 1.93
	CMVAE	64.76 ± 3.63	64.39 ± 3.64	6.94 ± 1.35	6.92 ± 1.36	0.08 ± 0.07	0.08 ± 0.07
	GMC	82.70 ± 1.58	82.57 ± 1.57	40.62 ± 1.58	41.49 ± 1.43	3.61 ± 0.84	4.24 ± 0.78
	GMCWD	94.15 ± 0.36	94.13 ± 0.36	73.95 ± 1.58	73.91 ± 1.58	15.60 ± 1.69	15.56 ± 1.69
	DGMC	91.43 ± 4.02	91.40 ± 4.04	66.08 ± 10.80	66.00 ± 10.81	10.17 ± 4.53	10.15 ± 4.46
	RGMC	86.30 ± 5.68	85.78 ± 7.68	84.76 ± 14.29	86.14 ± 15.97	**84.57±18.61**	**85.32±19.69**
Trajectory	DAE	97.62 ± 0.12	97.61 ± 0.12	89.74 ± 0.78	89.72 ± 0.77	49.81 ± 1.46	49.47 ± 1.47
	VAE	94.32 ± 0.58	94.29 ± 0.59	79.14 ± 2.10	79.07 ± 2.12	26.05 ± 2.78	25.70 ± 2.76
	MDAE	98.26 ± 0.12	98.26 ± 0.11	91.48 ± 0.91	91.45 ± 0.90	58.67 ± 2.98	58.40 ± 3.00
	MVAE	98.13 ± 0.42	98.12 ± 0.43	86.74 ± 4.38	86.68 ± 4.42	35.80 ± 8.57	35.54 ± 8.60
	CMDAE	98.72 ± 0.11	98.71 ± 0.12	94.38 ± 0.52	94.37 ± 0.52	63.70 ± 2.23	63.46 ± 2.12
	CMDVAE	**99.18±0.10**	**99.18±0.10**	**96.60±0.55**	**96.80±0.55**	**74.08±3.43**	73.99 ± 3.43
	CMVAE	97.55 ± 0.44	97.54 ± 0.44	84.41 ± 2.18	84.36 ± 2.19	40.15 ± 4.15	39.85 ± 4.15
	GMC	13.23 ± 3.71	11.03 ± 3.73	0.55 ± 0.27	0.64 ± 0.34	0.04 ± 0.03	0.07 ± 0.04
	GMCWD	45.58 ± 2.44	44.31 ± 2.57	2.36 ± 0.65	2.34 ± 0.77	0.12 ± 0.0	0.15 ± 0.13
	DGMC	60.29 ± 12.00	59.23 ± 12.19	16.55 ± 19.19	17.36 ± 20.27	14.10 ± 8.15	14.93 ± 8.50
	RGMC	37.63 ± 4.75	31.88 ± 4.11	37.74 ± 6.51	51.04 ± 9.61	69.29 ± 14.21	**80.43±15.27**

Table 5.5: Accuracy results for models with the PGD adversarial attack targeting each of the modalities on the MHD dataset, where we set the number of steps $\mathcal{K} = 10$.

For our last experiment in the MHD dataset, we used the PGD adversarial attack with different combinations of the ϵ and α parameters, while setting the number of steps to 10, where the results can be seen in Table 5.5. Unlike previous results, the RGMC model showed a rather poor performance with adversarial perturbations targeting the trajectory modality with low values of ϵ and α, suggesting that training the RGMC with either varying adversarial parameter values or with more complex adversarial attacks, such as PGD, might help increase the RGMC model's robustness to adversarial perturbations. Also, the base GMC model performed worse than the unimodal DAE in every single test with the PGD adversarial attack, and the multimodal models based on the VAE also demonstrated higher vulnerability to adversarial perturbations than the unimodal DAE in multiple adversarial settings, which seems to confirm the results from previous works that imply that multimodal models may sometimes perform worse under adversarial settings than unimodal models [27, 103].

The full results of the experiments performed on the MHD dataset appear to indicate that base GMC model is rather vulnerable to both noisy data and adversarial samples, sometimes performing worse than the unimodal DAE model, and that most VAE-based models lack robustness in all scenarios. Also, training models to remove noise from the input seems to increase robustness across all models against both noisy data and adversarial perturbations, although the GMC-based models still showed a partic-

ular vulnerability to perturbed trajectories, implying that the geometrical alignment of modalities may introduce a vulnerability to perturbations in the less complex modality. It is also shown that, although Finally, the RGMC model, while showing equal or better robustness compared to all other models across almost all scenarios, performed poorly in some tests with adversarial samples generated with the PGD adversarial attack on the trajectory modality, suggesting that some possible alterations to the RGMC architecture or training scheme may be needed to increase its robustness.

5.2 Experiments on the MNIST-SVHN dataset

In a slight deviation from the methodology used for the MHD experiments, we performed an initial test of the GMCWD and DGMC models with different hyperparameter values for the weight of the reconstruction losses of each modality, since some values for these hyperparameters resulted in the very high values for the reconstruction loss and it dominated the training process. We tested both models with each modality missing for all hyperparameter sets.

Model	Hyperparameter Set	Excluded Modality	
		MNIST	SVHN
GMCWD	a)	54.16 ± 6.49	90.84 ± 0.12
	b)	56.44 ± 7.30	92.25 ± 0.52
	c)	56.66±5.73	92.28±0.35
DGMC	a)	56.57 ± 5.68	89.89 ± 0.49
	b)	49.52 ± 6.17	89.92 ± 1.03
	c)	73.72±.17	90.84±0.40

Table 5.6: Comparison of accuracy of both the GMCWD and DGMC models on the MNIST-SVHN with missing modalities for different values of the weights for the reconstruction loss of each modality during model training. Hyperparameter set a) corresponds to both the MNIST and SVHN reconstruction loss weights being set to 0.5, b) to the MNIST weight being set to 0.5 and the SVHN weight being set to 0.05 and c) to the MNIST weight being set to 0.4 and the SVHN weight being set to 0.1.

After performing this initial experiment, we defined the hyperparameter values as being those of set c), since it resulted in the best accuracy, as can be seen in Table 5.6, and also made the reconstruction loss values similar to the contrastive loss values.

Afterwards, we mostly followed the same methodology we used for the experiments performed in the MHD dataset, where first we performed an evaluation of all models on the test set with clean data and all modalities present. Although results were more varied, we still obtained classification accuracy above 90% in all models, as can be seen in Figure 5.6, which allows us to perform the experiments with input perturbations, as all models possess enough expressiveness to represent the data with high accuracy.

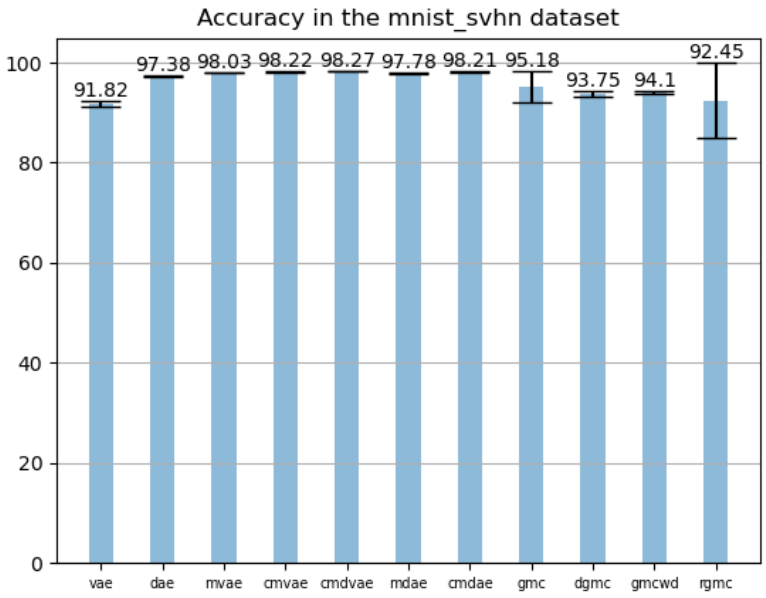

Figure 5.6: Model accuracy in the MNIST-SVHN dataset on clean data with all modalities present.

We then tested all models with one of the modalities missing, where the results can be seen in Table 5.7. The results were rather similar to the ones obtained in experiments performed on the MHD dataset, where, in this case, the multimodal AEs seem to be mostly dependent on the SVHN modality of the dataset. However, the GMC-based models appear to be more vulnerable to removing the least complex modality when compared to the results obtained in the MHD dataset, since even the base GMC model only achieved around 74% accuracy when removing the MNIST modality, while it achieved around 96% accuracy when removing the trajectory modality.

	Excluded Modality		
	None	MNIST	SVHN
DAE	97.38 ± 0.12	9.85 ± 0.69	28.97 ± 7.31
VAE	91.82 ± 0.62	10.98 ± 0.48	11.79 ± 2.33
MDAE	97.78 ± 0.11	25.02 ± 2.48	97.72 ± 0.13
MVAE	98.03 ± 0.13	12.63 ± 3.22	74.65 ± 17.70
CMDAE	98.21 ± 0.07	27.46 ± 2.32	97.94 ± 0.11
CMDVAE	**98.27±0.04**	30.50 ± 1.91	**98.00±0.10**
CMVAE	98.22 ± 0.09	16.96 ± 1.94	97.77 ± 0.18
GMC	95.18 ± 3.23	**73.71±8.20**	95.89 ± 1.90
GMCWD	94.10 ± 0.25	58.66 ± 5.73	92.28 ± 0.35
DGMC	93.75 ± 0.46	73.72 ± 8.17	90.84 ± 0.40
RGMC	92.45 ± 7.51	68.42 ± 16.83	93.65 ± 6.28

Table 5.7: Accuracy results for models with all modalities present (None) and with one of the modalities missing on the MNIST-SVHN dataset.

Our next experiment involved injecting Gaussian noise while altering the σ parameter, to test each model's robustness on noisy data. We show examples of the effect of the noise on the MNIST modality, which corresponds to the image modality in the MHD dataset, in Figure 5.4 and on the SVHN modality in Figure 5.7.

Figure 5.7: Examples of a image of the digit zero from the SVHN modality perturbed by Gaussian noise. Standard deviation values for the Gaussian noise are: 0, 0.25, 0.5, 0.75 and 1.

Gaussian Noise					
Std (σ)		0.25	0.5	0.75	1
MNIST	DAE	98.79 ± 0.09	89.93 ± 0.96	71.21 ± 1.79	56.24 ± 1.88
	VAE	85.78 ± 1.96	52.35 ± 2.76	33.98 ± 2.68	25.93 ± 2.13
	MDAE	99.10 ± 0.13	90.50 ± 1.41	65.18 ± 2.64	44.48 ± 3.22
	MVAE	91.62 ± 2.84	63.89 ± 6.65	43.16 ± 6.97	32.37 ± 6.09
	CMDAE	**99.44±0.09**	95.82 ± 0.51	78.95 ± 1.96	59.02 ± 3.46
	CMDVAE	97.94 ± 0.11	95.65 ± 0.42	84.96 ± 1.18	67.75 ± 1.19
	CMVAE	82.64 ± 6.97	34.37 ± 11.14	20.76 ± 5.58	16.77 ± 4.00
	GMC	89.17 ± 2.67	73.80 ± 6.74	63.28 ± 7.28	57.96 ± 7.27
	GMCWD	93.39 ± 0.33	85.94 ± 0.84	70.65 ± 2.02	58.00 ± 2.76
	DGMC	92.87 ± 0.51	**85.97±1.31**	**70.95±3.21**	58.45 ± 4.06
	RGMC	71.04 ± 11.44	69.61 ± 14.08	69.06 ± 15.30	**68.82±15.86**
SVHN	DAE	99.32 ± 0.05	98.90 ± 0.06	98.04 ± 0.17	97.09 ± 0.32
	VAE	98.37 ± 0.19	97.48 ± 0.27	96.41 ± 0.43	95.44 ± 0.53
	MDAE	99.58 ± 0.07	99.27 ± 0.08	98.67 ± 0.18	98.04 ± 0.27
	MVAE	99.56 ± 0.07	99.17 ± 0.21	98.48 ± 0.57	97.81 ± 0.78
	CMDAE	**99.66±0.04**	**99.48±0.07**	**99.02±0.11**	**98.55±0.17**
	CMDVAE	98.28 ± 0.06	98.24 ± 0.08	98.17 ± 0.06	98.15 ± 0.08
	CMVAE	99.50 ± 0.04	98.94 ± 0.14	98.00 ± 0.32	97.17 ± 0.52
	GMC	94.06 ± 3.48	91.84 ± 3.84	88.40 ± 4.24	86.09 ± 4.50
	GMCWD	93.83 ± 0.25	93.00 ± 0.24	91.97 ± 0.32	91.11 ± 0.42
	DGMC	93.57 ± 0.40	93.03 ± 0.27	91.26 ± 0.28	91.71 ± 0.34
	RGMC	92.65 ± 7.04	70.86 ± 14.31	68.06 ± 15.30	68.82 ± 15.86

Table 5.8: Accuracy results for models with Gaussian noise targeting each of the modalities on the MNIST-SVHN dataset

The results, which can be seen in Table 5.8, show that, although the extensions of the GMC framework provide more robustness than the base model, the RGMC model appears to be slightly more vulnerable to noisy data on both modalities, in some cases it even performs worse than the base GMC model, supporting the idea that further improvements can be made to increase the robustness of RGMC.

FGSM Adversarial Attack						
Epsilon (ϵ)		0.015	0.035	0.075	0.13	0.2
MNIST	DAE	95.96 ± 0.08	93.10 ± 0.19	82.03 ± 0.36	52.89 ± 1.01	17.52 ± 0.77
	VAE	88.16 ± 0.79	81.23 ± 1.12	59.77 ± 0.78	27.49 ± 0.69	9.12 ± 0.53
	MDAE	96.92 ± 0.19	95.42 ± 0.24	90.71 ± 0.58	78.91 ± 1.34	55.97 ± 2.45
	MVAE	95.77 ± 0.19	89.36 ± 0.42	56.87 ± 1.20	13.43 ± 1.52	1.45 ± 0.46
	CMDAE	97.48 ± 0.11	96.20 ± 0.16	**91.88±0.42**	**80.66±0.76**	57.84 ± 1.32
	CMDVAE	**97.53±0.13**	**96.21±0.15**	91.71 ± 0.42	79.74 ± 1.03	55.38 ± 1.95
	CMVAE	94.02 ± 0.28	75.93 ± 1.97	24.21 ± 2.23	3.86 ± 0.55	0.76 ± 0.23
	GMC	92.79 ± 3.66	88.07 ± 3.98	71.17 ± 4.33	42.64 ± 9.79	20.40 ± 11.46
	GMCWD	92.00 ± 0.46	88.15 ± 1.13	75.16 ± 4.08	46.46 ± 9.30	15.74 ± 7.23
	DGMC	91.20 ± 0.51	86.63 ± 0.54	72.05 ± 0.84	40.24 ± 3.02	10.59 ± 1.96
	RGMC	88.37 ± 8.49	77.74 ± 10.17	69.15 ± 13.94	66.67 ± 16.96	**65.92±18.64**
SVHN	DAE	96.43 ± 0.08	94.59 ± 0.15	88.93 ± 0.30	75.55 ± 0.78	53.89 ± 1.00
	VAE	85.75 ± 1.01	72.71 ± 1.35	39.97 ± 1.22	11.58 ± 0.91	3.39 ± 0.41
	MDAE	96.85 ± 0.18	95.09 ± 0.33	88.78 ± 0.87	73.24 ± 1.99	51.22 ± 2.51
	MVAE	97.23 ± 0.09	**95.83±0.30**	**91.44±1.33**	82.93 ± 3.25	72.67 ± 4.86
	CMDAE	97.30 ± 0.15	95.61 ± 0.26	89.27 ± 0.58	74.05 ± 1.29	54.07 ± 1.90
	CMDVAE	**97.40±0.12**	95.64 ± 0.20	89.65 ± 0.29	75.43 ± 1.06	57.06 ± 1.86
	CMVAE	97.38 ± 0.11	95.74 ± 0.14	90.08 ± 0.67	76.27 ± 2.01	53.58 ± 3.23
	GMC	79.30 ± 5.48	43.12 ± 7.74	13.98 ± 11.03	11.38 ± 10.78	14.24 ± 10.78
	GMCWD	88.45 ± 0.90	77.93 ± 2.08	62.74 ± 3.64	55.10 ± 9.76	51.16 ± 11.94
	DGMC	89.42 ± 0.37	81.67 ± 1.57	67.28 ± 6.35	56.26 ± 11.51	50.73 ± 14.37
	RGMC	76.62 ± 10.66	50.60 ± 13.22	73.02 ± 24.59	**86.61±19.12**	**88.75±19.28**

Table 5.9: Accuracy results for models with the FGSM adversarial targeting each of the modalities on the MNIST-SVHN dataset.

In our first experiment with adversarial attacks targeting each modality on the MNIST-SVHN dataset, we tested each model with adversarial perturbations generated with the FGSM algorithm with varying ϵ values, the results of which can be seen in Table 5.9. The RGMC model shows against a high level of robustness against adversarial perturbations with high values of ϵ and its extremely high standard deviation in the accuracy values when compared to the results obtained in the MHD dataset seem to suggest that a single seed result might be responsible for the decreased robustness of the RGMC model. This might be due to the fact that in the RGMC framework we train both the GMC model and the odd-one-out network simultaneously, making it harder for the model to successfully converge, suggesting that initializing the model's weights, which facilitates model convergence, or training the model in two separate phases, as is performed in [27], might improve the robustness of the RGMC architecture. Another important observation is that adjusting the weights of the reconstruction losses of the GMCWD and DGMC architectures can lead to different levels of robustness against perturbations targeting different modalities, making it important for future work to explore hyperparameter optimization strategies to increase the robustness of these architectures.

For our second experiment with adversarial examples in the MNIST-SVHN dataset, we used the BIM algorithm with different combinations of the ϵ and α parameters, while setting the number of steps to

BIM Adversarial Attack							
Epsilon (ϵ)		0.1		0.2		0.3	
Alpha (α)		0.02	0.04	0.06	0.08	0.1	0.12
MNIST	DAE	95.32 ± 0.10	91.98 ± 0.19	86.41 ± 0.30	77.94 ± 0.40	65.61 ± 0.61	51.45 ± 0.97
	VAE	86.60 ± 0.93	78.35 ± 1.07	66.35 ± 1.06	51.29 ± 0.80	35.62 ± 0.55	23.45 ± 0.76
	MDAE	96.57 ± 0.18	94.91 ± 0.28	92.52 ± 0.49	89.17 ± 0.68	84.55 ± 1.02	78.31 ± 1.36
	MVAE	94.56 ± 0.22	85.66 ± 0.57	67.07 ± 1.25	41.56 ± 1.80	19.60 ± 1.51	7.61 ± 0.92
	CMDAE	97.18 ± 0.13	95.79±0.20	93.64±0.46	90.57±0.46	86.41±0.62	80.82±0.84
	CMDVAE	97.25±0.15	95.71 ± 0.15	90.30 ± 0.52	85.93 ± 0.72	79.99 ± 1.06	79.99 ± 1.06
	CMVAE	90.59 ± 0.64	63.20 ± 3.21	29.24 ± 2.95	11.43 ± 1.41	4.14 ± 0.61	1.56 ± 0.41
	GMC	91.77 ± 3.77	86.10 ± 4.04	76.81 ± 4.07	64.32 ± 5.62	50.82 ± 8.70	38.64 ± 11.37
	GMCWD	91.12 ± 0.56	86.73 ± 1.41	80.17 ± 2.96	71.13 ± 5.1	59.34 ± 7.58	45.92 ± 9.25
	DGMC	90.17 ± 0.49	84.96 ± 0.54	77.71 ± 0.57	67.34 ± 1.24	54.08 ± 2.19	39.39 ± 2.83
	RGMC	83.08 ± 8.90	75.31 ± 11.44	71.82 ± 13.85	69.70 ± 15.96	68.35 ± 17.58	67.59 ± 18.84
SVHN	DAE	96.00 ± 0.09	93.98 ± 0.17	91.12 ± 0.29	87.04 ± 0.43	81.76 ± 0.66	75.37 ± 0.74
	VAE	82.68 ± 1.17	66.89 ± 1.37	47.26 ± 1.33	28.25 ± 1.07	15.22 ± 1.04	8.91 ± 0.84
	MDAE	96.43 ± 0.21	94.34 ± 0.42	90.95 ± 0.72	85.64 ± 1.26	78.03 ± 2.15	68.27 ± 2.23
	MVAE	96.90 ± 0.14	95.25±0.42	92.80±1.15	89.07±2.41	83.72 ± 4.25	77.46 ± 6.50
	CMDAE	96.93 ± 0.17	94.86 ± 0.30	91.49 ± 0.46	86.56 ± 0.88	79.79 ± 1.34	71.71 ± 2.21
	CMDVAE	97.00 ± 0.15	94.91 ± 0.24	91.81 ± 0.24	87.49 ± 0.46	81.68 ± 0.95	75.33 ± 1.25
	CMVAE	97.02±0.14	95.09 ± 0.17	92.35 ± 0.42	88.35 ± 0.88	83.17 ± 1.37	77.07 ± 2.08
	GMC	66.81 ± 5.33	31.09 ± 12.24	24.52 ± 11.45	28.38 ± 8.36	32.76 ± 6.18	35.48 ± 5.19
	GMCWD	85.73 ± 1.22	77.30 ± 1.78	72.10 ± 2.45	68.12 ± 4.09	64.95 ± 5.9	62.77 ± 6.99
	DGMC	87.48 ± 0.66	80.83 ± 2.35	76.39 ± 3.80	72.91 ± 5.24	69.86 ± 6.51	67.64 ± 7.50
	RGMC	67.38 ± 10.12	51.94 ± 21.19	70.11 ± 21.99	82.05 ± 19.46	86.44±18.74	87.78±19.29

Table 5.10: Accuracy results for models with the BIM adversarial attack targeting each of the modalities on the MNIST-SVHN dataset, where we set the number of steps $\mathcal{K} = 10$.

10, where the results can be seen in Table 5.10. As expected, most results were similar to the ones in the previous experiment. In particular, the RGMC model's accuracy have high standard deviations, further supporting the hypothesis that a single seed result is responsible for the higher vulnerability seen in the RGMC model. Remarkably, the MVAE model was able to achieve the highest average accuracy of all models in some tests in the SVHN modality. While the MVAE fared rather poorly when the MNIST modality was perturbed, these results suggest that a multimodal VAE which trades the product of experts approach, that has been shown to suffer from overconfident experts which make the model highly dependent on a single modality [42], might be able to achieve significant robustness gains.

For our last experiment in the MNIST-SVHN dataset, we used the PGD algorithm with different combinations of the ϵ and α parameters, while setting the number of steps to 10. The results, which can be seen in Table 5.11, seem to support most assumptions made in previous experiments. Since PGD has been shown to generate the strongest adversarial examples, and the results of the GMCWD and DGMC models show low robustness, we performed another experiment where we compared the robustness of these models when trained with hyperparameter sets c) and b) indicated in Table 5.6. The results, which can be seen in Table 5.12, show that these models offer significantly more robustness when trained with hyperparameter set b), further proving the importance of hyperparameter optimization.

PGD Adversarial Attack							
Epsilon (ϵ)		0.1		0.2		0.3	
Alpha (α)		0.01	0.02	0.04	0.06	0.08	0.1
MNIST	DAE	89.55 ± 0.26	89.51 ± 0.29	65.53 ± 0.61	65.48 ± 0.61	7.24 ± 0.42	7.20 ± 0.42
	VAE	72.83 ± 1.09	72.71 ± 1.07	34.95 ± 0.65	34.85 ± 0.63	3.39 ± 0.40	3.39 ± 0.38
	MDAE	93.83 ± 0.37	93.82 ± 0.37	84.46 ± 0.97	84.45 ± 0.96	38.58 ± 2.40	38.37 ± 2.42
	MVAE	77.67 ± 0.85	77.54 ± 0.86	17.64 ± 1.47	17.76 ± 1.48	0.03 ± 0.01	0.03 ± 0.01
	CMDAE	94.82±0.23	94.80±0.24	86.31±0.60	86.31±0.61	42.50 ± 1.42	42.45 ± 1.42
	CMDVAE	94.73 ± 0.20	94.71 ± 0.20	85.81 ± 0.74	85.80 ± 0.74	40.77 ± 1.92	40.72 ± 1.91
	CMVAE	43.48 ± 4.03	43.09 ± 3.97	2.62 ± 0.57	2.74 ± 0.59	0.01 ± 0.0	0.01 ± 0.0
	GMC	82.01 ± 4.05	81.89 ± 4.09	49.62 ± 8.20	49.60 ± 8.18	6.73 ± 5.91	7.22 ± 6.42
	GMCWD	65.50 ± 6.60	59.95 ± 7.40	8.41 ± 4.54	8.03 ± 4.37	0.21 ± 0.19	0.20 ± 0.18
	DGMC	61.10 ± 2.02	54.79 ± 2.12	4.78 ± 0.86	4.53 ± 0.78	0.05 ± 0.02	0.04 ± 0.02
	RGMC	69.01 ± 16.66	68.29 ± 17.74	65.19 ± 22.28	65.51 ± 22.60	64.83±23.30	64.99±23.39
SVHN	DAE	92.67 ± 0.20	92.67 ± 0.21	80.73 ± 0.84	80.70 ± 0.83	34.67 ± 1.76	34.44 ± 1.70
	VAE	57.95 ± 1.49	57.25 ± 1.49	14.31 ± 1.03	14.10 ± 1.03	0.52 ± 0.19	0.50 ± 0.18
	MDAE	92.74 ± 0.53	92.66 ± 0.55	76.22 ± 2.48	76.09 ± 2.54	24.13 ± 4.2	23.93 ± 1.24
	MVAE	93.69 ± 0.79	93.55 ± 0.84	78.63 ± 5.46	78.72 ± 5.45	31.68 ± 10.06	32.29 ± 9.93
	CMDAE	93.23 ± 0.36	93.12 ± 0.36	75.91 ± 1.73	75.93 ± 1.76	24.40 ± 2.79	24.48 ± 2.28
	CMDVAE	93.32 ± 0.18	93.22 ± 0.18	77.36 ± 0.87	77.39 ± 0.90	29.61 ± 1.47	29.82 ± 1.51
	CMVAE	93.74±0.34	93.70±0.34	81.18±1.65	81.21±1.62	32.97 ± 3.78	33.09 ± 3.77
	GMC	12.33 ± 6.33	11.05 ± 5.99	0.74 ± 1.06	1.27 ± 1.89	0.45 ± 0.34	0.94 ± 0.60
	GMCWD	40.60 ± 5.28	23.72 ± 4.23	6.65 ± 3.95	8.86 ± 4.78	6.06 ± 4.38	6.96 ± 5.06
	DGMC	52.79 ± 7.19	35.16 ± 11.66	12.04 ± 14.88	13.55 ± 15.60	8.93 ± 13.00	9.61 ± 13.41
	RGMC	10.65 ± 17.04	6.12 ± 11.27	5.87 ± 7.25	25.35 ± 32.20	43.67±35.48	52.02±36.11

Table 5.11: Accuracy results for models with the PGD adversarial attack targeting each of the modalities on the MNIST-SVHN dataset, where we set the number of steps $\mathcal{K} = 10$.

PGD Adversarial Attack							
Epsilon (ϵ)		0.1		0.2		0.3	
Alpha (α)		0.01	0.02	0.04	0.06	0.08	0.1
MNIST	GMCWD c)	65.50 ± 6.60	59.95 ± 7.40	8.41 ± 4.54	8.03 ± 4.37	0.21 ± 0.19	0.20 ± 0.18
	GMCWD b)	83.86±1.75	83.80±1.77	59.67±6.68	59.58±6.72	7.62±3.70	7.55±3.67
	DGMC c)	61.10 ± 2.02	54.79 ± 2.12	4.78 ± 0.86	4.53 ± 0.78	0.05 ± 0.02	0.04 ± 0.02
	DGMC b)	82.09 ± 0.60	82.04 ± 0.61	53.84 ± 3.21	53.72 ± 3.22	3.42 ± 0.88	3.37 ± 0.87
SVHN	GMCWD c)	40.60 ± 5.28	23.72 ± 4.23	6.65 ± 3.95	8.86 ± 4.78	6.06 ± 4.38	6.96 ± 5.06
	GMCWD b)	58.88 ± 5.18	57.82 ± 5.37	29.09 ± 6.91	36.16 ± 7.46	15.80 ± 6.38	19.51 ± 7.16
	DGMC c)	52.79 ± 7.19	35.16 ± 11.66	12.04 ± 14.88	13.55 ± 15.60	8.93 ± 13.00	9.61 ± 13.41
	DGMC b)	83.28±3.25	83.23±3.53	68.72±14.14	70.68±13.99	49.93±25.52	51.81±25.39

Table 5.12: Accuracy results for models with the PGD adversarial attack targeting each of the modalities on the MNIST-SVHN dataset, where we set the number of steps $\mathcal{K} = 10$, for the GMCWD and DGMC models trained with the hyperparameter sets c) and b), as indicated in 5.6.

6

Conclusions and Future Work

Contents

6.1 Conclusions 61
6.2 Limitations and Future Work 62

In this work, we explored the use of unsupervised generative models and representation learning methods in order to train models that are robust to missing modalities, noisy data and adversarial perturbations. We introduced several new models based on pre-existing architectures, such as the CMDVAE, a Multimodal Denoising Variational Auto-Encoder which uses a common encoder to encode the multiple modality latent representations into a single latent representation, the GMCWD, which combines the GMC framework with a joint decoder and a shared re-projection head, where the model is trained to remove perturbations in the inputs by minimizing the reconstructions loss, the DGMC, which is an extension of the GMCWD architecture that has an additional decoder for each modality, where the model's latent representations are the encoding of the respective reconstructions and the RGMC, which uses an odd-out-network to detect which (if any) input modality has suffered perturbations.

We then test the several architectures in two different datasets under multiple scenarios: 1) on clean data; 2) with missing modalities; 3) with noise added to the input, varying how much noise in mixed with the input; 4) with adversarial attacks targeting one of the modalities.

6.1 Conclusions

With the performed experiments, we where able to show that the addition of a reconstruction loss, not only helps the GMC framework learn representations that are more robust to input perturbations, but also regularizes its training, decreasing the deviation between the performance of models trained with different seeds, which assists model converge, as can be seen in Figure 5.2. However, one significant drawback of these architectures is that they are less robust to missing modalities when compared to the standard GMC framework, as can be seen in Tables 5.1 and 5.7.

We were also able to show that the performance of most of the generative models and AE-based architectures suffer from being over-dependent on one of their modalities. While this was previously known to occur in the MVAE architecture [42], we showed that this result also extends to several other AE-based architectures, as can be seen in Tables 5.1 and 5.7. An interesting side-effect of this is that, due to the geometrical alignment of the modalities enforced by the contrastive loss, the GMC-based architectures were more vulnerable to perturbations to the modality which the AEs architectures were not dependent on, an effect that was more clearly visible in the base GMC architecture.

When we compare the Multimodal AE models, it can be seen that the Multimodal VAE-based architectures consistently under-performed with perturbed data and missing modalities when compared to multimodal DAE-based architectures, and they were also sometimes not even able to match the performance of multimodal DAE-based architecture on clean data. However, the fact that the CMDVAE architecture consistently achieves one of the highest accuracy's across all AE-based models implies that it is the training process, not the architecture, that is behind the increased robustness seen in the denoising models.

Also, as expected, the unimodal AE-based architectures performed the worse when one of the modalities is missing. However, some other results were rather interesting, in particular, the fact that the unimodal VAE was able to better match the unimodal DAE when one of the modalities was missing, in some cases even outperforming the DAE, unlike what occurred with their Multimodal counterparts.

Another very interesting result is that, while the GMC-based models either matched or outperformed the multimodal DAEs when subjected to Gaussian noise and with missing modalities, the multimodal DAEs significantly outperformed the GMC-based models when adversarial perturbations are added to the input, even when the adversarial perturbations targeted the modality which the MDAEs where most dependent on. However, hyperparameter optimization may result in much higher robustness in the GMCWD and DGMC models, as shown in Table 5.12, so these models may outperform MDAEs when trained with the correct hyperparameters.

In general, the RGMC architecture seems to provide the highest robustness guarantees, although the results also show that this model can be further improved in a way that provides higher robustness in a wider variety of adversarial settings.

6.2 Limitations and Future Work

Three of the most obvious limitations of this work are related to the datasets and the perturbations used. First, we trained and evaluated the robustness of ML models only in relatively small datasets, while previous research has shown that training on larger datasets changes the dynamics of the robustness of the model [113]. Second, datasets such as MNIST are not particularly illustrative of real-world data and, while we considered different kinds of perturbations injected into the input, synthetic noise may not have the same characteristics as the noise present in real-world data [114]. Third, we only used two modalities and more modalities may increase the robustness of the multimodal models.

Given these limitations, it is of further interest to explore how the various presented architectures perform under other types of noise, such as speckle noise, as well as with adversarial attacks in more settings, such as targeted attacks. Also, since little research has been conducted into how non-adversarial real-world perturbations affect a model's robustness [115], future research should test the robustness of multimodal DL models in real-world scenarios, or in more complex tasks like Reinforcement Learning.

Another important limitation of this work is that hyperparameter tuning may change a model's robustness, particular in more complex models, like GMCWD and DGMC. It is therefore of interest to explore hyperparameter optimization strategies as a way to increase the robustness of these models.

A possible line of future research is to extend the GMC-based architectures introduced in this work to learn in a multi-stage process, such that the base model would be initially trained only on clean data and would afterwards be trained with different types of perturbations, in order to maintain the same performance as the base GMC architecture on clean data and with missing modalities, while providing the robustness benefits of extended models. It would also be interesting to explore different variants of the RGMC architecture, where instead of multiplying the latent representations by the output of the odd-one-out network, we take $\arg\max o_m$ as being the modality under perturbation and drop that modality or use the joint encoding if the odd-one-out network determines that no modality is under attack, and also another possible variant using weight initialization in order to facilitate model convergence.

Finally, seeing as the architectures considered in this work that are not based on the GMC framework seem to suffer from an extreme dependence on one of the modalities, future experiments should also take into account other architectures that explicitly attempt to solve this issue, such as the MMVAE architecture introduced by [42], which uses a mixture-of-experts approach as a way to deal with the problem of over-confident experts.

Bibliography

[1] S. G. Finlayson, H. W. Chung, I. S. Kohane, and A. L. Beam, "Adversarial attacks against medical deep learning systems," 2019.

[2] A. J. del Real, F. Dorado, and J. Durán, "Energy demand forecasting using deep learning: Applications for the french grid," *Energies*, vol. 13, no. 9, 2020.

[3] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[4] H. Yingge, I. Ali, and K.-Y. Lee, "Deep neural networks on chip - a survey," in *2020 IEEE International Conference on Big Data and Smart Computing (BigComp)*, 2020, pp. 589–592.

[5] L. Schmarje, M. Santarossa, S.-M. Schroder, and R. Koch, "A survey on semi-, self- and unsupervised learning for image classification," *IEEE Access*, vol. 9, pp. 82 146–82 168, 2021.

[6] K. Sinaga and M.-S. Yang, "Unsupervised k-means clustering algorithm," *IEEE Access*, vol. PP, pp. 1–1, 04 2020.

[7] L. Hollenstein, L. Lichtensteiger, T. Stadelmann, M. Amirian, L. Budde, J. Meierhofer, R. Füchslin, and T. Friedli, *Unsupervised Learning and Simulation for Complexity Management in Business Operations*, 06 2019, pp. 313–331.

[8] D. Bank, N. Koenigstein, and R. Giryes, "Autoencoders," 2020.

[9] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, jun 2015.

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

[11] T. Chen and G. Hinton, "Advancing self-supervised and semi-supervised learning with simclr."

[12] A. van den Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," 2019.

[13] OpenCV, "Generative and discriminative models," https://learnopencv.com/generative-and-discriminative-models/, accessed: 2023-10-2023.

[14] Z. Chen, W. M. Soliman, A. Nazir, and M. Shorfuzzaman, "Variational autoencoders and wasserstein generative adversarial networks for improving the anti-money laundering process," *IEEE Access*, vol. 9, pp. 83 762–83 785, 2021.

[15] Y. Yang, K. Zheng, C. Wu, and Y. Yang, "Improving the classification effectiveness of intrusion detection by using improved conditional variational autoencoder and deep neural network," *Sensors*, vol. 19, p. 2528, 06 2019.

[16] M. Abedi, L. Hempel, S. Sadeghi, and T. Kirsten, "Gan-based approaches for generating structured data in the medical domain," *Applied Sciences*, vol. 12, no. 14, 2022.

[17] Z. Dong, K. Xu, Y. Yang, H. Bao, W. Xu, and R. W. H. Lau, "Location-aware single image reflection removal," 2021.

[18] S. Lapuschkin, S. Wäldchen, A. Binder, G. Montavon, W. Samek, and K.-R. Müller, "Unmasking clever hans predictors and assessing what machines really learn," *Nature Communications*, vol. 10, 03 2019.

[19] I. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *International Conference on Learning Representations*, 2015.

[20] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," 2016.

[21] Z. Zhao, D. Dua, and S. Singh, "Generating natural adversarial examples," 2017.

[22] P. Samangouei, M. Kabkab, and R. Chellappa, "Defense-gan: Protecting classifiers against adversarial attacks using generative models," in *International Conference on Learning Representations*, 2018.

[23] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee, "Generative adversarial text to image synthesis," 2016.

[24] C. Akkus, L. Chu, V. Djakovic, S. Jauch-Walser, P. Koch, G. Loss, C. Marquardt, M. Moldovan, N. Sauter, M. Schneider, R. Schulte, K. Urbanczyk, J. Goschenhofer, C. Heumann, R. Hvingelby, D. Schalk, and M. Aßenmacher, "Multimodal deep learning," 2023.

[25] S. Kutuzova, O. Krause, D. McCloskey, M. Nielsen, and C. Igel, "Multimodal variational autoencoders for semi-supervised learning: In defense of product-of-experts," 2021.

[26] P. Poklukar, M. Vasco, H. Yin, F. S. Melo, A. Paiva, and D. Kragic, "Geometric multimodal contrastive representation learning," *arXiv preprint arXiv:2202.03390*, 2022.

[27] K. Yang, W.-Y. Lin, M. Barman, F. Condessa, and Z. Kolter, "Defending multimodal fusion models against single-source adversaries," 2022.

[28] W. Wang, Y. Park, T. Lee, I. Molloy, P. Tang, and L. Xiong, "Utilizing multimodal feature consistency to detect adversarial examples on clinical summaries," in *Proceedings of the 3rd Clinical Natural Language Processing Workshop*. Online: Association for Computational Linguistics, Nov. 2020, pp. 259–268.

[29] I. H. Sarker, "Deep learning: A comprehensive overview on techniques, taxonomy, applications and research directions," *SN Computer Science*, vol. 2, no. 6, p. 420, Aug. 2021.

[30] N. Hallowell, S. Badger, A. Sauerbrei, C. Nellåker, and A. Kerasidou, ""i don't think people are ready to trust these algorithms at face value": trust and the use of machine learning algorithms in the diagnosis of rare disease," *BMC Medical Ethics*, vol. 23, no. 1, p. 112, Nov. 2022.

[31] S. Tsimenidis, "Limitations of deep neural networks: a discussion of g. marcus' critical appraisal of deep learning," 2020.

[32] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. J. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," in *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2014.

[33] M. Moradi, K. Blagec, and M. Samwald, "Deep learning models are not robust against noise in clinical text," 2021.

[34] H. Xu, Y. Ma, H. Liu, D. Deb, H. Liu, J. Tang, and A. K. Jain, "Adversarial attacks and defenses in images, graphs and text: A review," 2019.

[35] Y. Cao, C. Xiao, B. Cyr, Y. Zhou, W. Park, S. Rampazzi, Q. A. Chen, K. Fu, and Z. M. Mao, "Adversarial sensor attack on LiDAR-based perception in autonomous driving," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*. ACM, nov 2019.

[36] M. Suzuki, K. Nakayama, and Y. Matsuo, "Joint multimodal learning with deep generative models," *arXiv preprint arXiv:1611.01891*, 2016.

[37] Y. Shi, S. N, B. Paige, and P. Torr, "Variational mixture-of-experts autoencoders for multi-modal deep generative models," in *Advances in Neural Information Processing Systems*, H. Wallach,

H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds., vol. 32. Curran Associates, Inc., 2019.

[38] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, "Learning transferable visual models from natural language supervision," 2021.

[39] A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, and M. Chen, "Hierarchical text-conditional image generation with clip latents," 2022.

[40] A. Nichol, P. Dhariwal, A. Ramesh, P. Shyam, P. Mishkin, B. McGrew, I. Sutskever, and M. Chen, "Glide: Towards photorealistic image generation and editing with text-guided diffusion models," 2022.

[41] S. Naeem, A. Ali, S. Anam, and M. Ahmed, "An unsupervised machine learning algorithms: Comprehensive review," *IJCDS Journal*, vol. 13, pp. 911–921, 04 2023.

[42] Y. Shi, N. Siddharth, B. Paige, and P. H. S. Torr, "Variational mixture-of-experts autoencoders for multi-modal deep generative models," 2019.

[43] H. Mao, B. Zhang, H. Xu, Z. Yuan, and Y. Liu, "Robust-msa: Understanding the impact of modality noise on multimodal sentiment analysis," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, no. 13, pp. 16 458–16 460, Sep. 2023.

[44] M. Hearst, S. Dumais, E. Osuna, J. Platt, and B. Scholkopf, "Support vector machines," *IEEE Intelligent Systems and their Applications*, vol. 13, no. 4, pp. 18–28, 1998.

[45] L. Alzubaidi, J. Zhang, A. J. Humaidi, A. Al-Dujaili, Y. Duan, O. Al-Shamma, J. Santamaría, M. A. Fadhel, M. Al-Amidie, and L. Farhan, "Review of deep learning: concepts, cnn architectures, challenges, applications, future directions," *Journal of Big Data*, vol. 8, no. 1, p. 53, Mar 2021.

[46] G. Louppe, "Understanding random forests: From theory to practice," 2015.

[47] J. Heaton, "An empirical analysis of feature engineering for predictive modeling," in *SoutheastCon 2016*. IEEE, mar 2016.

[48] M. Rostami, H. He, M. Chen, and D. Roth, *Transfer Learning via Representation Learning*. Cham: Springer International Publishing, 2023, pp. 233–257.

[49] L. Ericsson, H. Gouk, C. C. Loy, and T. M. Hospedales, "Self-supervised representation learning: Introduction, advances, and challenges," *IEEE Signal Processing Magazine*, vol. 39, no. 3, pp. 42–62, may 2022.

[50] M. Ahmed, R. Seraj, and S. M. S. Islam, “The k-means algorithm: A comprehensive survey and performance evaluation,” *Electronics*, vol. 9, no. 8, 2020.

[51] A. Maćkiewicz and W. Ratajczak, “Principal components analysis (pca),” *Computers Geosciences*, vol. 19, no. 3, pp. 303–342, 1993.

[52] D. Bank, N. Koenigstein, and R. Giryes, “Autoencoders,” 2021.

[53] P. Dhariwal and A. Nichol, “Diffusion models beat gans on image synthesis,” 2021.

[54] E. Marchi, F. Vesperini, F. Eyben, S. Squartini, and B. Schuller, “A novel approach for automatic acoustic novelty detection using a denoising autoencoder with bidirectional lstm neural networks,” in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 1996–2000.

[55] D. P. Kingma and M. Welling, “Auto-encoding variational bayes,” in *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2014.

[56] R. Hadsell, S. Chopra, and Y. LeCun, “Dimensionality reduction by learning an invariant mapping,” in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’06)*, vol. 2, 2006, pp. 1735–1742.

[57] K. Sohn, “Improved deep metric learning with multi-class n-pair loss objective,” in *Proceedings of the 30th International Conference on Neural Information Processing Systems*, ser. NIPS’16. Red Hook, NY, USA: Curran Associates Inc., 2016, p. 1857–1865.

[58] P. Bachman, R. D. Hjelm, and W. Buchwalter, “Learning representations by maximizing mutual information across views,” 2019.

[59] I. Misra and L. van der Maaten, “Self-supervised learning of pretext-invariant representations,” 2019.

[60] T. Chen, Y. Sun, Y. Shi, and L. Hong, “On sampling strategies for neural network-based collaborative filtering,” 2017.

[61] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” 2014.

[62] Y. N. Wu, R. Gao, T. Han, and S.-C. Zhu, “A tale of three probabilistic families: Discriminative, descriptive and generative models,” 2018.

[63] J. Shlens, “Notes on kullback-leibler divergence and likelihood,” 2014.

[64] S. Mishra, “An introduction to vae-gans,” Oct 2021.

[65] D. Saxena and J. Cao, “Generative adversarial networks (gans): Challenges, solutions, and future directions,” *ACM Comput. Surv.*, vol. 54, no. 3, may 2021.

[66] H. Song, M. Kim, D. Park, Y. Shin, and J.-G. Lee, “Learning from noisy labels with deep neural networks: A survey,” 2022.

[67] J. Yim and K.-A. Sohn, “Enhancing the performance of convolutional neural networks on quality degraded datasets,” in *2017 International Conference on Digital Image Computing: Techniques and Applications (DICTA)*, 2017, pp. 1–8.

[68] H.-X. Li, J.-L. Yang, G. Zhang, and B. Fan, “Probabilistic support vector machines for classification of noise affected data,” *Information Sciences*, vol. 221, pp. 60–71, 2013.

[69] I. Reis, D. Baron, and S. Shahaf, “Probabilistic random forest: A machine learning algorithm for noisy data sets,” *The Astronomical Journal*, vol. 157, no. 1, p. 16, dec 2018.

[70] W. Li, X. Liu, A. Yan, and J. Yang, “Kernel-based adversarial attacks and defenses on support vector classification,” *Digital Communications and Networks*, vol. 8, no. 4, pp. 492–497, 2022.

[71] C. Zhang, H. Zhang, and C.-J. Hsieh, “An efficient adversarial attack for tree ensembles,” 2020.

[72] T. França, A. M. B. Braga, and H. V. H. Ayala, “Feature engineering to cope with noisy data in sparse identification,” *Expert Systems with Applications*, vol. 188, p. 115995, 2022.

[73] S. Pau, A. Perniciano, B. Pes, and D. Rubattu, “An evaluation of feature selection robustness on class noisy data,” *Information*, vol. 14, no. 8, 2023.

[74] G. Ras, N. Xie, M. van Gerven, and D. Doran, “Explainable deep learning: A field guide for the uninitiated,” 2021.

[75] H. Baniecki and P. Biecek, “Adversarial attacks and defenses in explainable artificial intelligence: A survey,” 2023.

[76] A. Kurakin, I. Goodfellow, and S. Bengio, “Adversarial examples in the physical world,” 2017.

[77] Y. Xiao, Z. Wei, and Z. Wang, “A limited memory bfgs-type method for large-scale unconstrained optimization,” *Computers Mathematics with Applications*, vol. 56, no. 4, pp. 1001–1009, 2008.

[78] R. Lapid and M. Sipper, “I see dead people: Gray-box adversarial attack on image-to-text models,” 2023.

[79] B. U. h. Sheikh and A. Zafar, “Untargeted white-box adversarial attack to break into deep learning based covid-19 monitoring face mask detection system,” *Multimedia Tools and Applications*, May 2023.

[80] P. Rathore, A. Basak, S. H. Nistala, and V. Runkana, “Untargeted, targeted and universal adversarial attacks and defenses on time series,” in *2020 International Joint Conference on Neural Networks (IJCNN)*. IEEE, jul 2020.

[81] C. Agarwal, B. Dong, D. Schonfeld, and A. Hoogs, “An explainable adversarial robustness metric for deep learning neural networks,” 2018.

[82] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, “Towards deep learning models resistant to adversarial attacks,” 2019.

[83] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami, “Distillation as a defense to adversarial perturbations against deep neural networks,” in *2016 IEEE Symposium on Security and Privacy (SP)*, 2016, pp. 582–597.

[84] M. Arjovsky, S. Chintala, and L. Bottou, “Wasserstein generative adversarial networks,” 2017.

[85] L. E. Richards, A. T. Nguyen, R. Capps, S. Forsyth, C. Matuszek, and E. Raff, “Adversarial transfer attacks with unknown data and class overlap,” *CoRR*, vol. abs/2109.11125, 2021.

[86] A. Demontis, M. Melis, M. Pintor, M. Jagielski, B. Biggio, A. Oprea, C. Nita-Rotaru, and F. Roli, “Why do adversarial attacks transfer? explaining transferability of evasion and poisoning attacks,” 2019.

[87] N. Papernot, P. McDaniel, and I. Goodfellow, “Transferability in machine learning: from phenomena to black-box attacks using adversarial samples,” 2016.

[88] G. E. Hinton, O. Vinyals, and J. Dean, “Distilling the knowledge in a neural network,” *CoRR*, vol. abs/1503.02531, 2015.

[89] D. Meng and H. Chen, “Magnet: a two-pronged defense against adversarial examples,” 2017.

[90] D. W. Otter, J. R. Medina, and J. K. Kalita, “A survey of the usages of deep learning in natural language processing,” 2019.

[91] C. Chen, M.-Y. Liu, O. Tuzel, and J. Xiao, “R-cnn for small object detection,” in *Computer Vision – ACCV 2016*, S.-H. Lai, V. Lepetit, K. Nishino, and Y. Sato, Eds. Cham: Springer International Publishing, 2017, pp. 214–230.

[92] Z. Zhang, P. Cui, and W. Zhu, “Deep learning on graphs: A survey,” 2020.

[93] A. Huang and R. Wu, “Deep learning for music,” 2016.

[94] Y. Li, L. Ma, Z. Zhong, F. Liu, D. Cao, J. Li, and M. A. Chapman, “Deep learning for lidar point clouds in autonomous driving: A review,” 2020.

[95] S. Reed, Z. Akata, H. Lee, and B. Schiele. Los Alamitos, CA, USA: IEEE Computer Society, jun 2016, pp. 49–58.

[96] A. Radford, L. Metz, and S. Chintala, “Unsupervised representation learning with deep convolutional generative adversarial networks,” 2016.

[97] V. et al., “Attention is all you need,” in *Advances in Neural Information Processing Systems*, vol. 30. Curran Associates, Inc., 2017.

[98] A. van den Oord, O. Vinyals, and K. Kavukcuoglu, “Neural discrete representation learning,” 2018.

[99] J. Ho, A. Jain, and P. Abbeel, “Denoising diffusion probabilistic models,” 2020.

[100] M. Wu and N. Goodman, “Multimodal generative models for scalable weakly-supervised learning,” ser. NIPS’18. Red Hook, NY, USA: Curran Associates Inc., 2018, p. 5580–5590.

[101] G. E. Hinton, “Training products of experts by minimizing contrastive divergence,” *Neural Comput.*, vol. 14, no. 8, p. 1771–1800, aug 2002.

[102] N. Vishwamitra, H. Hu, Z. Zhao, L. Cheng, and F. Luo, “Understanding and measuring robustness of multimodal learning,” 2021.

[103] Y. Tian and C. Xu, “Can audio-visual integration strengthen robustness under multimodal attacks?” 2021.

[104] I. Evtimov, R. Howes, B. Dolhansky, H. Firooz, and C. C. Ferrer, “Adversarial evaluation of multimodal models under realistic gray box assumption,” 2020.

[105] M. C. Schiappa, S. Vyas, H. Palangi, Y. S. Rawat, and V. Vineet, “Robustness analysis of video-language models against visual and language perturbations,” 2022.

[106] M. Vasco, H. Yin, F. S. Melo, and A. Paiva, “Leveraging hierarchy in multimodal generative models for effective cross-modality inference,” *Neural Netw.*, vol. 146, no. C, p. 238–255, feb 2022.

[107] R. Socher, M. Ganjoo, H. Sridhar, O. Bastani, C. D. Manning, and A. Y. Ng, “Zero-shot learning through cross-modal transfer,” 2013.

[108] S. Lei, H. Zhang, K. Wang, and Z. Su, “How training data affect the accuracy and robustness of neural networks for image classification,” 2019.

[109] D. Hendrycks, K. Lee, and M. Mazeika, “Using pre-training can improve model robustness and uncertainty,” *Proceedings of the International Conference on Machine Learning*, 2019.

[110] S. Mohammadi, A. Uhrenholt, and B. Jensen, “Odd-one-out representation learning,” 12 2020.

[111] B. Fernando, H. Bilen, E. Gavves, and S. Gould, “Self-supervised video representation learning with odd-one-out networks,” 2017.

[112] J. Li, C. Xiong, and S. C. Hoi, “Learning from noisy data with robust representation learning,” in *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021, pp. 9465–9474.

[113] Q. Xie, M.-T. Luong, E. Hovy, and Q. V. Le, “Self-training with noisy student improves imagenet classification,” 2020.

[114] T. Wu, X. Ding, M. Tang, H. Zhang, B. Qin, and T. Liu, “Noisywikihow: A benchmark for learning with real-world noisy labels in natural language processing,” 2023.

[115] N. Drenkow, N. Sani, I. Shpitser, and M. Unberath, “A systematic review of robustness in deep learning for computer vision: Mind the gap?” 2022.

7

Algorithms Hyperparameters

Unsupervised Learning Hyperparameters															
Hyperparams		Seeds	NE	BS	LR	LD	CD	λ_{img}	λ_{traj}	λ_{mnist}	λ_{svhn}	τ	λ_o	σ	ϵ
MHD	DAE	$\{0, \ldots, 9\}$	100	64	10^{-3}	64	-	0.5	0.5	-	-	-	-	0.5	-
	VAE	$\{0, \ldots, 9\}$	100	64	10^{-3}	64	-	0.5	0.5	-	-	-	-	-	-
	MDAE	$\{0, \ldots, 9\}$	100	64	10^{-3}	64	-	0.5	0.5	-	-	-	-	0.5	-
	MVAE	$\{0, \ldots, 9\}$	100	64	10^{-3}	64	-	0.5	0.5	-	-	-	-	0.5	-
	CMDAE	$\{0, \ldots, 9\}$	100	64	10^{-3}	64	-	0.5	0.5	-	-	-	-	0.5	-
	CMDVAE	$\{0, \ldots, 9\}$	100	64	10^{-3}	64	-	0.5	0.5	-	-	-	-	0.5	-
	CMVAE	$\{0, \ldots, 9\}$	100	64	10^{-3}	64	-	0.5	0.5	-	-	-	-	-	-
	GMC	$\{0, \ldots, 9\}$	100	64	10^{-3}	64	64	-	-	-	-	0.1	-	-	-
	GMCWD	$\{0, \ldots, 9\}$	100	64	10^{-3}	64	64	0.5	0.5	-	-	0.1	-	0.5	-
	DGMC	$\{0, \ldots, 9\}$	100	64	10^{-3}	64	64	0.5	0.5	-	-	0.1	-	0.5	-
	RGMC	$\{38, \ldots, 47\}$	100	64	10^{-3}	64	64	-	-	-	-	0.1	4.0	-	0.1
MNIST-SVHN	DAE	$\{0, \ldots, 9\}$	100	64	10^{-3}	64	-	-	-	0.5	0.5	-	-	0.5	-
	VAE	$\{0, \ldots, 9\}$	100	64	10^{-3}	64	-	-	-	0.5	0.5	-	-	-	-
	MDAE	$\{0, \ldots, 9\}$	100	64	10^{-3}	64	-	-	-	0.5	0.5	-	-	0.5	-
	MVAE	$\{0, \ldots, 9\}$	100	64	10^{-3}	64	-	-	-	0.5	0.5	-	-	-	-
	CMDAE	$\{0, \ldots, 9\}$	100	64	10^{-3}	64	-	-	-	0.5	0.5	-	-	0.5	-
	CMDVAE	$\{0, \ldots, 9\}$	100	64	10^{-3}	64	-	-	-	0.5	0.5	-	-	0.5	-
	CMVAE	$\{0, \ldots, 9\}$	100	64	10^{-3}	64	-	-	-	0.5	0.5	-	-	-	-
	GMC	$\{0, \ldots, 9\}$	100	64	10^{-3}	64	64	-	-	-	-	0.1	-	-	-
	GMCWD	$\{0, \ldots, 9\}$	100	64	10^{-3}	64	64	-	-	0.4	0.1	0.1	-	0.5	-
	DGMC	$\{0, \ldots, 9\}$	100	64	10^{-3}	64	64	-	-	0.4	0.1	0.1	-	0.5	-
	RGMC	$\{0, \ldots, 9\}$	100	64	10^{-3}	64	64	-	-	-	-	0.1	4.0	-	0.1

Table 7.1: Hyperparameters used to train the various models during the unsupervised learning setting. We trained all models using the SGD optimizer with momentum set to 0.9. NE is the number of epochs, BS is the batch size, LR is the learning rate, LD is the latent dimension, CD is the common dimension, λ_{img} is the image reconstruction loss weight, λ_{traj} is the trajectory reconstruction loss weight, λ_{mnist} is the MNIST reconstruction loss weight, λ_{svhn} is the SVHN reconstruction loss weight, τ is the InfoNCE temperature, λ_o is the odd-one-out network loss scale, σ is the noise standard deviation and ϵ is the FGSM adversarial attack parameter.

Supervised Learning Hyperparameters							
Hyperparams		Seeds	# Epochs	Batch Size	Learning Rate	Latent Dim	Common Dim
MHD	DAE	$\{0, \ldots, 9\}$	50	64	10^{-3}	64	-
	VAE	$\{0, \ldots, 9\}$	50	64	10^{-3}	64	-
	MDAE	$\{0, \ldots, 9\}$	50	64	10^{-3}	64	-
	MVAE	$\{0, \ldots, 9\}$	50	64	10^{-3}	64	-
	CMDAE	$\{0, \ldots, 9\}$	50	64	10^{-3}	64	-
	CMDVAE	$\{0, \ldots, 9\}$	50	64	10^{-3}	64	-
	CMVAE	$\{0, \ldots, 9\}$	50	64	10^{-3}	64	-
	GMC	$\{0, \ldots, 9\}$	50	64	10^{-3}	64	64
	GMCWD	$\{0, \ldots, 9\}$	50	64	10^{-3}	64	64
	DGMC	$\{0, \ldots, 9\}$	50	64	10^{-3}	64	64
	RGMC	$\{38, \ldots, 47\}$	50	64	10^{-3}	64	64
MNIST-SVHN	DAE	$\{0, \ldots, 9\}$	50	64	10^{-3}	64	-
	VAE	$\{0, \ldots, 9\}$	50	64	10^{-3}	64	-
	MDAE	$\{0, \ldots, 9\}$	50	64	10^{-3}	64	-
	MVAE	$\{0, \ldots, 9\}$	50	64	10^{-3}	64	-
	CMDAE	$\{0, \ldots, 9\}$	50	64	10^{-3}	64	-
	CMDVAE	$\{0, \ldots, 9\}$	50	64	10^{-3}	64	-
	CMVAE	$\{0, \ldots, 9\}$	50	64	10^{-3}	64	-
	GMC	$\{0, \ldots, 9\}$	50	64	10^{-3}	64	64
	GMCWD	$\{0, \ldots, 9\}$	50	64	10^{-3}	64	64
	DGMC	$\{0, \ldots, 9\}$	50	64	10^{-3}	64	64
	RGMC	$\{0, \ldots, 9\}$	50	64	10^{-3}	64	64

Table 7.2: Hyperparameters used to train the various models during the supervised learning setting. We trained all models using the SGD optimizer with momentum set to 0.9.