Learning Generative Models from Incomplete Data

Yao Chong Lim

CMU-CS-19-120

August 2019

School of Computer Science

Computer Science Department

Carnegie Mellon University

Pittsburgh, PA 15213

Thesis Committee:

Louis-Philippe Morency, chair

Alexander Hauptmann

Submitted in partial fulfillment of the requirements

for the Degree of Master of Science.

Keywords: machine learning, generative models, imputation, missing data, variational autoencoders

Abstract

Real-world machine learning systems must handle missing data well in order to build robust and reliable systems. One way to tackle this problem is to build a generative model that can learn from incomplete data. Such models can then be used in applications such as image restoration and missing value imputation. To achieve this, this thesis introduces a deep generative model, the Variational Auto-decoder (VAD), a variant of the stochastic gradient variational Bayes (SGVB) estimator first introduced by Kingma and Welling in 2013. To improve the robustness of the model to varying rates of missing data during training and testing, the VAD framework directly optimizes parameters of the approximate posterior of the latent variables. Compared to the common variational autoencoder (VAE) implementation of SGVB, no encoder network is used to approximate the parameters of the approximate posterior. Through empirical evaluation on six datasets, including datasets on image classification, facial landmark detection, and multimodal language, we show that the VAD framework is more robust to different rates of missing data than previous generative models for incomplete data.

Acknowledgments

I would like to first thank my advisor, Louis-Philippe Morency, for his generous advice and guidance while completing this thesis, and also over my time in the Multicomp Lab. I learned a lot from him about presenting my research effectively. I also want to thank Alexander Hauptmann for being my committee member, and for the helpful advice he gave. I am also very grateful to my mentor Amir Zadeh for his endless guidance and discussions over the three years we have worked together. Much of the work I have done here came from experiments and work we did together, and I have learnt a great deal from him. I must also thank my colleague Paul Liang for the many discussions we have had during my time at CMU, which have always been illuminating, and for his advice, which has always been helpful. I also want to thank everyone in the Multicomp Lab for creating a stimulating environment to do research in. Finally, I want to thank all of my friends and peers at CMU, who have made my experience here a memorable one.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Motivation and Introduction

Dealing with missing data is an important part of any real-world machine learning system [29]. Incomplete or unreliable data due to non-responses, faulty sensors, corrupted data, censorship, or other causes present distinct challenges to the training of such systems [10, 16]. Modern machine learning systems are largely trained on complete data only. However, ignoring incomplete data when training our models potentially throws away a lot of useful information contained in the incomplete data. Furthermore, generative models that are learned from incomplete data can be easily used to generate plausible replacements for missing values in the incomplete data [29, 12]. This can be used in applications such as image restoration or inpainting by replacing occluded areas or imperfections, or imputation of incomplete datasets. While there are existing models that model and can learn from incomplete data [29, 12], we show that these models perform poorly when trained and tested on different rates of missingness.

To tackle this problem, this thesis presents the Variational Autodecoder (VAD), a framework that learns a generative model of possibly incomplete data by optimizing latent variables to maximize the likelihood of the observed values. In contrast to related approaches which make an approximate prediction of the latent variables from observed values [12], the

VAD framework avoids the use of such an encoder, which makes the model more robust to different patterns and rates of missing data [13]. We evaluate and compare the models on a wide range of datasets in cases where missingness during train and test time are similar and different to support this claim.

In this thesis, we aim to answer the following research questions:

1. What are the limitations of the existing variational autoencoder-based and generative adversarial network-based approaches to modeling incomplete data [13, 29, 12]?
2. How sensitive are generative models for incomplete data to the pattern of missingness during training and testing?
3. How does an encoder-less parameterization of the stochastic gradient variational Bayes (SGVB) framework perform in comparison to existing encoder-based models with regards to robustness to missing data during test time despite not being explicitly trained on missing data?
4. How does the generative model learned by an encoder-less parameterization of the SGVB framework compare to existing generative models that deal with incomplete data?

1.1 Contributions

To that end, this thesis makes the following contributions:

1. We show that two existing neural approaches to generative modeling of incomplete data, *Generative Adversarial Imputation Networks* (GAIN) [29] and *Variational Autoencoders with Arbitrary Conditioning* (VAEAC) [12] are sensitive to the rates of missingness in train and test sets.

2. To address these shortcomings of existing approaches, we present an adaptation of the SGVB framework, which we call the Variational Auto-decoder (VAD), which directly optimizes the parameters of the approximate latent posterior by maximizing the likelihood of the training data.

3. We conduct empirical comparisons of VAD with both existing approaches on a wide range of datasets, varying the rate of missing data in both the training and testing datasets, and show that the VAD framework more accurately predicts missing values, even when not explicitly trained to do so.

1.2 Other Research Projects

While this thesis focuses on the contributions mentioned in the previous subsection, I also performed research on simple but strong baselines for multimodal utterance embeddings during my program [15]. Here, we extended previous work on simple but strong sentence embeddings to the multimodal utterances [1], which are trimodal sequences of acoustic, visual, and language modalities. The model assumes unimodal, bimodal, and trimodal factorizations of the utterance conditioned on the utterance embedding, and further assumes that the features follow a Gaussian distribution. The resulting model can be efficiently optimized using coordinate ascent, and displays competitive results with many neural models on downstream tasks involving multimodal embeddings, while showing speedup of at least an order of magnitude.

1.3 Thesis Outline

Chapter 2 discusses related work on deep generative models and recent attempts to learn generative models over incomplete data, as well as related work in imputation and inpainting

applications. Chapter 3 presents the general formulation of the imputation problem, as well as the VAE framework that the VAD framework is based on. Chapter 4 presents and discusses the VAD framework in detail, describes the experimental setup of datasets and experimental procedures employed to evaluate the performance of the VAD framework, and presents and analyzes the results of these experiments. Finally, Chapter 5 concludes the thesis and discusses possible directions for future work.

Chapter 2

Related Work

In this chapter, we provide an overview of generative models, with a focus on deep generative models designed to learn from incomplete data and relate them to the VAD framework. We also discuss classic approaches to data imputation, as well as application-specific approaches from inpainting.

2.1 Deep generative models

Generative models describe a distribution over the space of all possible data (for example, images of faces), which allows for sampling of datapoints, often from a random variable that represents the latent space. Deep generative models parameterize the data distribution using deep neural networks, and have obtained impressive results in image generation [3]. Two main types of deep generative models are in use today: variational autoencoders (VAEs) and generative adversarial networks (GANs).

VAEs are a variant of the stochastic gradient variational Bayes (SGVB) algorithm [13]. In its basic form, it consists of an encoder network, which approximates a posterior

distribution over the latent space given input data, and a decoder that generates the datapoint from the randomly sampled latent variables. The framework is explained in more detail in Section 3.2.

One approach to adapting the VAE framework to tackle imputation introduces arbitrary conditioning to the model [12]. The use of an encoder network to determine a distribution of latent variables from the incomplete input allows the model to compute fast imputations using a single forward pass of the trained model. However, because the model still uses an encoder, it is still susceptible to drops in performance when the training and testing data follow different patterns of missingness.

GANs contain two networks: a generator network, which learns to generate datapoints from random latent variables, and a discriminator network, which learns to differentiate between random outputs from the generator and the true datapoints in the training set [11]. As the two networks have opposing goals, the generator eventually learns to generate outputs that are similar to the true datapoints to better fool the discriminator. The generator then defines a distribution over the data, which can be sampled from.

Generative Adversarial Imputation Networks (GAIN) [29] adapt the Generative Adversarial Network (GAN) framework [11] for the problem of imputation. In GAIN, the generator additionally takes as input an incomplete datapoint, and imputes the missing values. The completed datapoints are given to the discriminator to identify the observed values and imputed values, with the help of a hint vector that provides some information about the identity of the imputed variables. While the use of a discriminator encourages the generator to generate plausible imputations, both the discriminator and generator only learn to handle datapoints with rates of missingness seen during training. This weakens the generative model's ability to generalize beyond rates of missingness seen during training. Our results in Section 4.3.2 support this observation.

2.2 Imputing missing data

One application of a generative model trained on incomplete data is to impute values for missing data. A large number of approaches exist to tackle the problem of imputing missing data. We discuss some of the common statistical approaches are discussed here.

A simple approach to handling missing data is to just ignore datapoints with missing values, known as *complete-case analysis* [10]. However, if missing values are common in the dataset, doing so greatly reduces the amount of available data, which can hurt the performance of trained models. Furthermore, biases could arise if datapoints with missing values are systematically different from datapoints with complete data.

One simple imputation method is to replace missing values with the *mean* value across observed values of the same variable [10]. However, this method can also easily bias the dataset if the missing values are different from the observed values, as in complete-case analysis. Furthermore, this method affects the summary statistics of the dataset, such as underestimating the variance of the missing variable, since values are now more concentrated about the mean [16]. Other simple methods include imputation based on logical rules or information from related observations, but this requires knowledge about the domain and dataset and must be hand-written for each variable, which does not generalize to an approach that works for any dataset. For example, if trying to obtain household income estimates, and the income of only one parent is known, a reasonable imputation of the total household income might be just the single income, or twice the single income.

When a single variable has missing values, two simple approaches are imputing the missing value using a trained regression model, and hot-deck imputation or matching [10]. In the regression method, a regression model is trained over the fully observed variables to predict the values of the missing variable, using datapoints with complete information. The predictions for the datapoints with missing values are then used as the imputed values. In hot-deck imputation, given the datapoint for which we are trying to impute a missing value,

we find the datapoint that is the most similar and has an observed value for the missing variable, and simply use that observed value as the imputation. While hot-deck imputation is potentially a non-parametric imputation method depending on the similarity metric used, its effectiveness depends on the presence of complete datapoints that are similar enough to the incomplete datapoint to give a good imputation [9].

In general, these approaches for imputing a single variable do not scale well to imputing multiple missing variables [24]. If multiple variables are missing, regression models have to be trained for each missing variable, and the pool of candidate datapoints for hot-deck imputation shrinks.

To impute multiple missing variables, regression-based methods are often used. One simple method is to simply fit a multivariate regression model, where the predicted values are the variables with missing values. Another method is iterative regression imputation, which involves progressively repeatedly imputing each missing value individually, using the newly imputed values, until convergence is reached [25].

Most of the approaches listed above produce a single imputed value per missing value, which ignores the inherent uncertainty in trying to predict these values. Multiple imputation attempts to encode this uncertainty by simply running the previous methods multiple times to obtain multiple imputed values for each missing value, giving multiple *completed* datasets (datasets where the missing values have been imputed). The desired tasks are then performed on all completed datasets, and uncertainty estimates of the final results can be obtained from the results over each completed dataset.

To impute multiple missing variables, most modern statistical methods use *iterative imputation*, which involves progressively repeatedly imputing each missing value individually, using the newly imputed values, until convergence is reached. One general approach is *multivariate imputation by chained equations* (MICE), which involves sampling imputed values from conditional distributions, conditioned on the remaining variables [5]. While the

model has seen much practical success, particularly in medical applications (van Buuren and Groothuis-Oudshoorn [4] give a large list of applications of MICE in various fields), the conditional distributions learned in the process are not theoretically guaranteed to be consistent with an overall joint distribution [2]. Furthermore, the success of the method depends on the choice of parameterization of the conditional distributions, which may require expert fine-tuning based on relationships between the variables [18], while the VAD framework described in this thesis does not require this. Another related approach is missForest, which also performs iterative imputation, but instead uses a random forest to predict missing values [24].

2.3 Inpainting

Another application of generative models learned from incomplete data is in image restoration or inpainting. Here, the input is an image with missing pixels, denoted using a binary mask. The goal is then to reconstruct the missing pixels as best as possible. We discuss some inpainting-specific approaches below.

One approach is the deep image prior [26] finds the best network parameters that minimizes reconstruction error, while keeping the latent variables fixed. This means that during testing, gradient descent must still be conducted to find the set of network parameters that best reconstruct this image, like in the VAD framework. One drawback of this method is that since the model is non-trainable, shared information between datapoints cannot be used to improve the model's performance. Another drawback is that the method does not scale well, since different network parameters must be found for each image. In contrast, in the VAD framework, the generator network parameters are shared across all datapoints, and the latent variables that minimize reconstruction loss for each datapoint are learned.

Most autoencoders that use convolutional neural networks in their encoders and

decoders use fully-connected layers to obtain a common latent representation for the encoder and decoder, to allow information to propagate across the whole image. Context encoders modify this by replacing the fully-connected layers with a channel-wise fully-connected layer, which propagates information spatially but not across channels, and a 1×1 convolution, which propagates information across channels but not spatially [20]. This greatly reduces the number of parameters in the model, allowing for larger latent representations that seem to improve performance. The model also uses adversarial loss during training to tackle the phenomenon that L2 and L1 loss functions commonly used in image reconstruction tend to lead to blurry solutions, giving them a small gain in performance. However, without data augmentation, the model is still dependent on the quality of the encoder and its ability to handle general patterns of missingness when only trained on certain patterns [29].

Chapter 3

Technical Background

In this chapter, we provide a background on related work in data imputation and variational auto-encoders, and provide a mathematical formulation of the problem.

3.1 Data imputation

Our task is to model the distribution of some d-dimensional random variable $x \sim p(x)$, where $x \in \mathbb{R}^d$. Unfortunately, x is not fully observable. To model the distribution of missing values, we introduce a random variable $\alpha \sim p(\alpha); \alpha \in \{0,1\}^d$ that is a binary mask indicating whether each dimension of x is observed. If $\alpha_i = 1$, then x_i is observed, otherwise x_i is missing. The final observed datapoints $\hat{x}$ are then assumed to be generated by first generating the ground-truth x and mask α, and then removing the missing values from x to generate $\hat{x}$:

$$\hat{x} = x \odot \alpha + * \odot (1 - \alpha) \tag{3.1}$$

Here, $*$ can be any value, such as zero, or values drawn from a uniform random distribution.

Hence, the dataset consists of both the observed values as well as the corresponding masks $\mathcal{D} = \{\hat{x}^{(i)}, \alpha^{(i)}\}_{i=1}^N$. We do not have access to the ground-truth $X = \{x_1\}_{i=1}^N$ since it is

strictly unknown.

We assume that the identities of the missing values are given to us via the mask α. This is consistent with many real-world applications involving missing data, such as where study participants withhold data, or where readings are not available due to faulty sensors or data corruption.

Patterns of missingness in data can be categorized into three categories: 1) *missing completely at random* (MCAR), where missingness probabilities are the same for all variables, 2) *missing at random* (MAR), where missingness depends only on the observed variables, and 3) *missing not at random* (MNAR), where missingness depends on both observed and unobserved variables (including the missing values themselves) [10]. In this thesis, we primarily experiment with MCAR for simplicity, as generating data missing at random requires knowledge of the dataset and interactions between the variables. However, we present some results on other patterns of missingness (see Figure 4.8).

3.2 Stochastic gradient variational Bayes

The variational auto-decoder framework is largely based on the stochastic gradient variational Bayes (SGVB) framework proposed by Kingma and Welling [13]. We provide a review of the relevant background before describing the VAD formulation.

Consider a continuous random variable $\mathbf{x} \in \mathbb{R}^d$, and a dataset containing N i.i.d. samples $\mathbf{X} = \{\mathbf{x}_1, ..., \mathbf{x}_N\}$, where $x_i \sim p(x)$. We assume that each sample is generated from a corresponding latent variable z_i that follows some prior distribution $p(z)$, and that this generation is parameterized using some parameters θ. This is illustrated in the plate diagram in Figure 3.1. To estimate θ during training, we optimize θ to maximize the (log-)likelihood

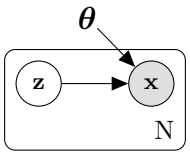

Figure 3.1: Latent variable diagram for the problem in plate notation. $\mathbf{z}_i \in \mathbb{R}^k$ are the latent variables for each corresponding datapoint $\mathbf{x}_i \in \mathbb{R}^d$, and the distribution of $\mathbf{x}$ is parameterized using $\mathbf{z}$ and some parameters $\boldsymbol{\theta}$, e.g. using a feedforward neural network.

of the dataset:

$$\log p(\mathbf{X}) = \sum_{i=1}^{n} \log p(\mathbf{x}_i) = \sum_{i=1}^{n} \log \left(\int p(\mathbf{z}_i) p_\theta(\mathbf{x}_i \mid \mathbf{z}_i) \, d\mathbf{z}_i \right)$$

However, in general, integrating over the latent variable $\mathbf{z}$ is intractable. Hence, most approaches approximate the prior $p(z)$ using some approximate distribution $q(z)$ to make training and inference tractable. In the common variational auto-encoder (VAE) formulation, this approximation is obtained using a recognition/encoder network that predicts the parameters of $q(\mathbf{z})$ from $\mathbf{x}_i$. Hence the encoder parameterizes a posterior distribution $q_\phi(\mathbf{z} \mid \mathbf{x}_i)$. During training, the parameters of the encoder network are then optimized as well.

More concretely, the training objective is to maximize the log-likelihood of the training dataset:

$$\max \log p(\mathbf{X}) = \max \sum_{i=1}^{n} \log p(\mathbf{x}_i)$$

With our encoder network that parameterizes the conditional distribution $q_\phi(\mathbf{z} \mid \mathbf{x}_i)$, we can

rewrite the log-likelihood for a single datapoint:

$$\begin{aligned}
\log p(\mathbf{x}_i) &= \mathrm{KL}\Big[q_\phi(\mathbf{z} \mid \mathbf{x}_i) \,\|\, p_\theta(\mathbf{z} \mid \mathbf{x}_i)\Big] + \mathop{\mathbb{E}}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x}_i)} \Big[\log p_\theta(\mathbf{x}, \mathbf{z}) - \log q_\phi(\mathbf{z} \mid \mathbf{x})\Big] \\
&= \int q_\phi(\mathbf{z} \mid \mathbf{x}_i) \left(\log \frac{q_\phi(\mathbf{z} \mid \mathbf{x}_i)}{p_\theta(\mathbf{z} \mid \mathbf{x}_i)} + \log \frac{p_\theta(\mathbf{x}_i, \mathbf{z})}{q_\phi(\mathbf{z} \mid \mathbf{x}_i)}\right) d\mathbf{z} \\
&= \int q_\phi(\mathbf{z} \mid \mathbf{x}_i) \log \frac{p_\theta(\mathbf{x}_i, \mathbf{z})}{p_\theta(\mathbf{z} \mid \mathbf{x}_i)} d\mathbf{z} \\
&= \int q_\phi(\mathbf{z} \mid \mathbf{x}_i) \log p(\mathbf{x}_i)\, d\mathbf{z} \\
&= \log p(\mathbf{x}_i) \qquad\qquad \text{(since } p(\mathbf{x}_i) \text{ does not depend on } q_\phi(\mathbf{z} \mid \mathbf{x}_i)\text{)}
\end{aligned}$$

Since the KL divergence is non-negative, the second term above (which we will call the loss $\mathcal{L}(\theta, \phi, \mathbf{x}_i)$) is a lower bound for the log-likelihood:

$$\begin{aligned}
\mathcal{L}(\theta, \phi, \mathbf{x}_i) &= \mathop{\mathbb{E}}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x}_i)} \Big[\log p_\theta(\mathbf{x}, \mathbf{z}) - \log q_\phi(\mathbf{z} \mid \mathbf{x})\Big] \\
&= \log p(\mathbf{x}_i) - \mathrm{KL}\Big[q_\phi(\mathbf{z} \mid \mathbf{x}_i) \,\|\, p_\theta(\mathbf{z} \mid \mathbf{x}_i)\Big] \\
&\leq \log p(\mathbf{x}_i)
\end{aligned}$$

Under this VAE formulation, we can then indirectly maximize the dataset log-likelihood by maximizing the lower bound $\mathcal{L}(\theta, \phi, \mathbf{x}_i)$ (called the evidence lower bound [ELBO]):

$$\begin{aligned}
\mathcal{L}(\theta, \phi, \mathbf{x}_i) &= \mathop{\mathbb{E}}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x}_i)} \Big[\log p_\theta(\mathbf{x}_i, \mathbf{z}) - \log q_\phi(\mathbf{z} \mid \mathbf{x}_i)\Big] \\
&= \int q_\phi(\mathbf{z} \mid \mathbf{x}_i) \log \frac{p_\theta(\mathbf{x}_i, \mathbf{z})}{q_\phi(\mathbf{z} \mid \mathbf{x}_i)} d\mathbf{z} \\
&= \int q_\phi(\mathbf{z} \mid \mathbf{x}_i) \log \frac{p_\theta(\mathbf{z})}{q_\phi(\mathbf{z} \mid \mathbf{x}_i)} d\mathbf{z} + \int q_\phi(\mathbf{z} \mid \mathbf{x}_i) \log p_\theta(\mathbf{x}_i \mid \mathbf{z}) d\mathbf{z} \\
&= -\mathrm{KL}\Big[q_\phi(\mathbf{z} \mid \mathbf{x}_i) \| p_\theta(\mathbf{z})\Big] + \mathop{\mathbb{E}}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x}_i)} [\log p_\theta(\mathbf{x}_i \mid \mathbf{z})] \qquad (3.2)
\end{aligned}$$

However, this formulation requires the full dataset to be observed during training and testing. Recent work has proposed methods to handle missing values in VAEs. Conditional VAEs [23] can be adapted to handle missing values, albeit only in cases where the pattern of missingness is always the same, while VAEs with arbitary conditioning (VAEAC) [12] extend CVAEs to deal with arbitary and varying patterns of missingness.

Chapter 4

The variational auto-decoder

This chapter describes the formulation of the variational auto-decoder (VAD) framework and provides algorithms for optimization and inference of the model. We also present experimental comparisons of the VAD with VAE on a range of datasets and discuss the results.

4.1 Formulation

In the VAD formulation, we directly optimize the parameters of the approximate distribution q. In our experiments, we draw q from the family of multivariate Gaussian distributions. However, any distribution where random variables drawn from the distribution can be expressed as a deterministic function of an auxiliary noise variable can also be used, as this gives a differentiable expectation of the ELBO, which can be used to optimize the parameters of the decoder network and the parameters of q. This is known as the reparameterization trick in the literature [13].

As before, we maximize the ELBO, a lower bound on the log-likelihood of the training

data:

$$\mathcal{L}(\theta, \phi, \mathbf{x}_i) = -\mathrm{KL}\Big[q_\phi(\mathbf{z} \mid \mathbf{x}_i) \| p_\theta(\mathbf{z})\Big] + \mathop{\mathbb{E}}_{\mathbf{z} \sim q_\phi(\mathbf{z}|\mathbf{x}_i)}[\log p_\theta(\mathbf{x}_i \mid \mathbf{z})] \leq \log p(\mathbf{x}_i) \tag{4.1}$$

However, unlike the VAE formulation, the approximate posterior $q(\mathbf{z} \mid \mathbf{x}_i)$ is not parameterized by an encoder network. Instead, the posterior is parameterized by *learnable* parameters that are optimized during both training and testing, removing the direct dependence of $\mathbf{z}$ on $\mathbf{x}$. Each datapoint $\mathbf{x}_i$ has a corresponding set of parameters ϕ_i for the posterior $q(\mathbf{z}_i)$. For example, if $q_{\phi_i}(\mathbf{z}_i)$ is defined as a multivariate diagonal Gaussian, then the learnable parameters of q are the mean and covariance of the distribution:

$$q_\phi(\mathbf{z} \mid \mathbf{x}_i) = q_\phi(\mathbf{z}_i) \propto \frac{1}{\sigma} \exp\left[-\frac{(x-\mu)^2}{2\sigma^2}\right]$$

where $\phi_i = \{\mu, \sigma\}$.

So the ELBO in 4.1 now becomes:

$$\mathcal{L}(\theta, \phi, \mathbf{x}_i) = -\mathrm{KL}\Big[q_\phi(\mathbf{z}) \| p_\theta(\mathbf{z})\Big] + \mathop{\mathbb{E}}_{\mathbf{z} \sim q_\phi(\mathbf{z})}[\log p_\theta(\mathbf{x}_i \mid \mathbf{z})] \leq \log p(\mathbf{x}_i) \tag{4.2}$$

$$\mathcal{L}(\theta, \phi, \mathbf{x}_i) = -\mathrm{KL}\Big[q_{\phi_i}(\mathbf{z}_i) \| p_\theta(\mathbf{z}_i)\Big] + \mathop{\mathbb{E}}_{\mathbf{z} \sim q_{\phi_i}(\mathbf{z_i})}[\log p_\theta(\mathbf{x}_i \mid \mathbf{z})] \leq \log p(\mathbf{x}_i) \tag{4.3}$$

For a minibatch consisting of datapoints $\mathbf{x}_1, ..., \mathbf{x}_n$, we maximize the sum of the ELBO for each datapoint:

$$\sum_{i=1}^{n} \mathcal{L}(\theta, \phi_i, \mathbf{x}_i) = \sum_{i=1}^{n} \left(-\mathrm{KL}\Big[q_{\phi_i}(\mathbf{z}_i) \| p_\theta(\mathbf{z})\Big] + \mathop{\mathbb{E}}_{\mathbf{z} \sim q_{\phi_i}(\mathbf{z_i})}[\log p_\theta(\mathbf{x}_i \mid \mathbf{z})]\right)$$

The training and inference procedures for the VAD framework are largely the same, except that during testing, the parameters of the decoder network are not updated. The algorithm is summarized in Algorithm 1 for the case of a multivariate Gaussian as the approximate posterior. During training, we randomly initialize the decoder parameters $\theta^{(0)}$, and the parameters of the approximate distribution q. For example, for a multivariate Gaussian, we randomly initialize separate means $\mu_i^{(0)}$ and covariances $\Sigma_i^{(0)}$ for each datapoint x_i. Then, at epoch t, we sample from the approximate posterior q using the appropriate

Algorithm 1 Training (and inference) process for the VAD models with multivariate Gaussian distribution as approximate posterior.

$\mathcal{F} : \{\theta^{(0)}\} \leftarrow$ Initialization ▷ Only for training

$q : \{\mu_i^{(0)}, \Sigma_i^{(0)}\} \leftarrow$ Initialization

repeat:

  $[z] \sim q(z; \mu_i^{(t)}, \Sigma_i^{(t)})$ ▷ Sampling approximate posterior

  $[p(x|z; \theta^{(t)})] = \mathcal{N}(\mathcal{F}([z], \theta^{(t)}); x_i, \Lambda_i)$

  $\{\theta, \mu_i, \Sigma_i\}^{(t+1)} \leftarrow \underset{\theta, \mu_i, \Sigma_i}{\text{grad\_step}}\Big(\mathcal{L}\big(\theta, \phi, \mathbf{x}_i\big)\Big)$ ▷ No grad_step w.r.t θ during inference

  $t \leftarrow t + 1$

until maximization convergence on $\mathcal{L}$

parameters $\mu_i^{(t)}$ and $\Sigma_i^{(t)}$ and obtain a Monte Carlo estimate of the distribution over the input using the decoder network. We find that using a single Monte Carlo sample as the estimate is sufficient to obtain reasonable results. Gradient descent with respect to the network parameters (only during training time) and parameters of the approximate posterior is used to optimize the likelihood of the observed values. This repeats until convergence.

One drawback of this algorithm is the need for gradient descent optimization during test time. However, since the model takes no input to obtain the approximate posterior, it is robust to unseen missingness patterns during train and test time.

4.2 Experiments

Empirical evaluation of the VAD framework was conducted on a wide range of datasets. Results of the VAD model were compared to some of baseline models mentioned in Section 2. Below, we describe these baseline models in further detail, and describe the datasets and experiments used.

4.2.1 Baseline models

We compare the VAD with four alternative approaches: a *variational autoencoder* (VAE), the *Generative Adversarial Imputation Network* (GAIN) proposed by Yoon et al. [29], and the MICE and MissForest algorithms.

The VAE implementation is a simplified version of the model presented in Ivanov et al. [12]. The posterior distribution $p(\mathbf{z} \mid \mathbf{x})$ is parameterized by an encoder network, which in the experiments here is a 2-layer feedforward neural network. The prior is a standard multivariate diagonal Gaussian, and the decoder network that parameterizes $p(\mathbf{x} \mid \mathbf{z})$ is a 2-layer feedforward neural network. During training, the model receives incomplete data with missing values zeroed out, and the model is trained to maximize the ELBO (Equation 4.1), where the likelihood of the datapoint is only calculated using the observed values, to mimic real settings where missing features may not be available during training.

The GAIN framework adapts *generative adversarial networks* (GANs) for imputation [11], and consists of two models. The first is a *generator*, which takes the partial data, a mask indicating the locations of missing values, and a random noise variable, and generates an imputation for the datapoints. The second model is a discriminator which takes an imputation obtained from the generator, and attempts to predict which values were imputed by the generator, and which were originally observed. To theoretically guarantee that the discriminator can learn to distinguish between imputed and observed values, the discriminator

is additionally provided with a *hint* vector that is a random portion of the mask that indicates the location of missing values, to provide the discriminator with some information on the location of observed and imputed values [29].

In addition, some experiments were conducted with *multivariate imputation by chained equations* (MICE) and missForest where feasible. Both use iterative imputation methods, which involve progressively repeatedly imputing each missing value using newly imputed values, until convergence is reached. We use the R implementation of MICE by van Buuren [4], which randomly initializes starting imputations for the missing values, and repeatedly learns conditional distributions of each missing variable in terms of all other variables. We learn these conditional distributions [4] using linear regressions. missForest is a similar approach to MICE, but uses random forests to predict missing values instead. We use the `missingpy` Python package in our experiments, based on the original missForest implementation by Stekhoven [24].

4.2.2 Missingness patterns

As noted in Section 3.1, we primarily ran experiments where the generated missingness was completely at random (MCAR). This was done for simplicity, as generating data missing at random requires knowledge of the dataset and interactions between the variables. However, we did run some experiments on other patterns of missingness (see Figure 4.8). We also note that the VAD framework makes no assumptions about the missingness pattern of the data.

4.2.3 Datasets

Following previous literature, we first evaluated the VAD model on two small datasets from the UCI repository. Further experiments were conducted on three types of larger, more complex datasets: image datasets, facial landmark datasets, as well as multimodal datasets

Subset	Dataset	# of samples	# of variables
UCI	Breast [8]	569	30
	Spam [8]	4601	57
Images	MNIST [14]	70000	784
	Fashion-MNIST [28]	70000	784
	CelebA [17]	202599	12288
Facial Landmarks	Menpo [31]	12208	168
Multimodal	CMU-MOSEI [30]	20001	409

Table 4.1: Size and number of variables of each datasets used in our experiments, described in Section 4.2.3.

which contain information from heterogeneous sources of data. More details about the datasets follow below. A summary of the number of samples and number of variables in each dataset is given in Table 4.1.

UCI datasets

Initial experiments were run on two datasets from the UCI Machine Learning Repository: the Breast Cancer Diagnosis (Wisconsin) Data Set, and Spambase Data Set [8]. These are small datasets used in previous literature to empirically evaluate imputation approaches [29, 12]. The Breast Cancer dataset contains real-valued features for each cell nucleus from samples of a breast mass. These features include the radius, texture, perimeter, and area of each nucleus. The Spambase dataset contains features extracted from a collection of emails, such as the frequency of the word "money", or the length of the longest sequence of capital letters. All features are again continuous.

Images

The first large-scale dataset we experimented with was the MNIST dataset of handwritten digits [14]. This is a common baseline for many machine learning and computer vision models. The dataset consists of 28×28 8-bit grayscale images of handwritten single digits, split into a train/test set of 50000 and 10000 images.

Experiments were also conducted on the slightly more challenging Fashion-MNIST dataset [28]. Like MNIST, the dataset contains 28×28 8-bit grayscale images, but of 10 different types of clothing items instead of handwritten digits. This makes modelling the data more challenging since more variability is present in each datapoint, such as the pattern on the clothing, their shape, size, and the presence of text or special features on the clothing.

Thirdly, we conducted experiments on the Large-Scale CelebFaces Attributes (CelebA) dataset [17]. The dataset contains over 200000 color images of celebrity faces in a large range of lightings, backgrounds, and facial expressions. For the experiments conducted, we resized the images from (original size) to 64×64 images, with bicubic interpolation.

For all three datasets, we introduced missingness completely at random at rates from 10% to 90%. Missing values were zeroed out. For CelebA, we additionally experimented with missingness at test time as a single missing block from the center, following work from Pathak et al. [20].

Facial Landmarks

The facial landmark detection task involves detecting the locations of landmarks from 2D images of faces. These landmarks are predefined locations on the outline of the face, as well as outlines of facial features such as eyes, lips, and nose. We run experiments on the Menpo Facial Landmark dataset [31], containing $13,391$ sets of facial landmarks obtained from real facial images of various subjects, poses and expressions. As such, there is a wide variety in

the range of possible sets of landmarks, and modelling them is a complex problem. Each set of landmarks contains 84 points, regardless of self-occlusions in the original image due to factors such as a sideways pose, or obstruction by hair.

Multimodal datasets

We experimented with datasets involving multimodal language. The CMU Multimodal Sentiment and Emotion Intensity (CMU-MOSEI) [30] dataset is an in-the-wild dataset of utterances from Youtube monologue videos, and is used for multimodal sentiment and emotion recognition tasks. The dataset contains $23,500$ utterances (single sentences) from Youtube monologues of three modalities: text, visual, and acoustic. For the text modality, the dataset contains GloVe word embeddings [21]. For the visual modality, the dataset contains facial action units, facial landmarks, and head pose information. For the acoustic modality, the dataset uses the COVAREP feature set of both high-level and low-level descriptors such as pitch tracking and glottal source parameters [7].

4.2.4 Experiments conducted

To answer the research questions set out in Chapter 1, three different experiments were conducted:

Experiment 1: To evaluate the VAD framework's ability to learn a generative model from incomplete data and compare it to other generative models, we first train and test the models on data with similar rates of missingness during train and test time, and compare the quality of the imputed values in terms of their error on the missing values.

Experiment 2: To evaluate whether existing generative models are sensitive to the missingness patterns observed during training time, we evaluated GAIN, VAE, and VAD models on data with different rates of missingness during train and test time.

Experiment 3: Finally, to qualitatively evaluate the quality of the generative model learnt, we generate completed samples from partial data on a few image datasets and observe the results.

4.2.5 Evaluation metric

On all datasets, we report the mean squared error (MSE) obtained on the test set over only the missing values, as the datasets we use contain only continuous variables. We ignore the mean squared error over observed values since they are already known and can be used in place of the predicted values of the models, if any.

4.3 Results and Discussion

Below, we present the results for each experiment described in Section 4.2.4.

4.3.1 Experiment 1: Similar rates of missingness between train and test

A summary of the results for Experiment is shown in Figure 4.1. For each dataset, we plot the MSE obtained on only the unobserved values of the test set. The VAD model largely outperformed other approaches across all datasets at all rates of missingness in reconstructing the missing values. However, VAD was outperformed by MissForest on the UCI datasets (Figures 4.1b and 4.1a). This might have been due to the small sizes of both datasets, causing the VAD model to overfit. We also note that the GAIN and VAE models worsen more quickly than VAD as the rate of missingness increases, leading to a greater gap in performance. This suggests that the encoder in the VAE is unable to accurately learn an approximate posterior

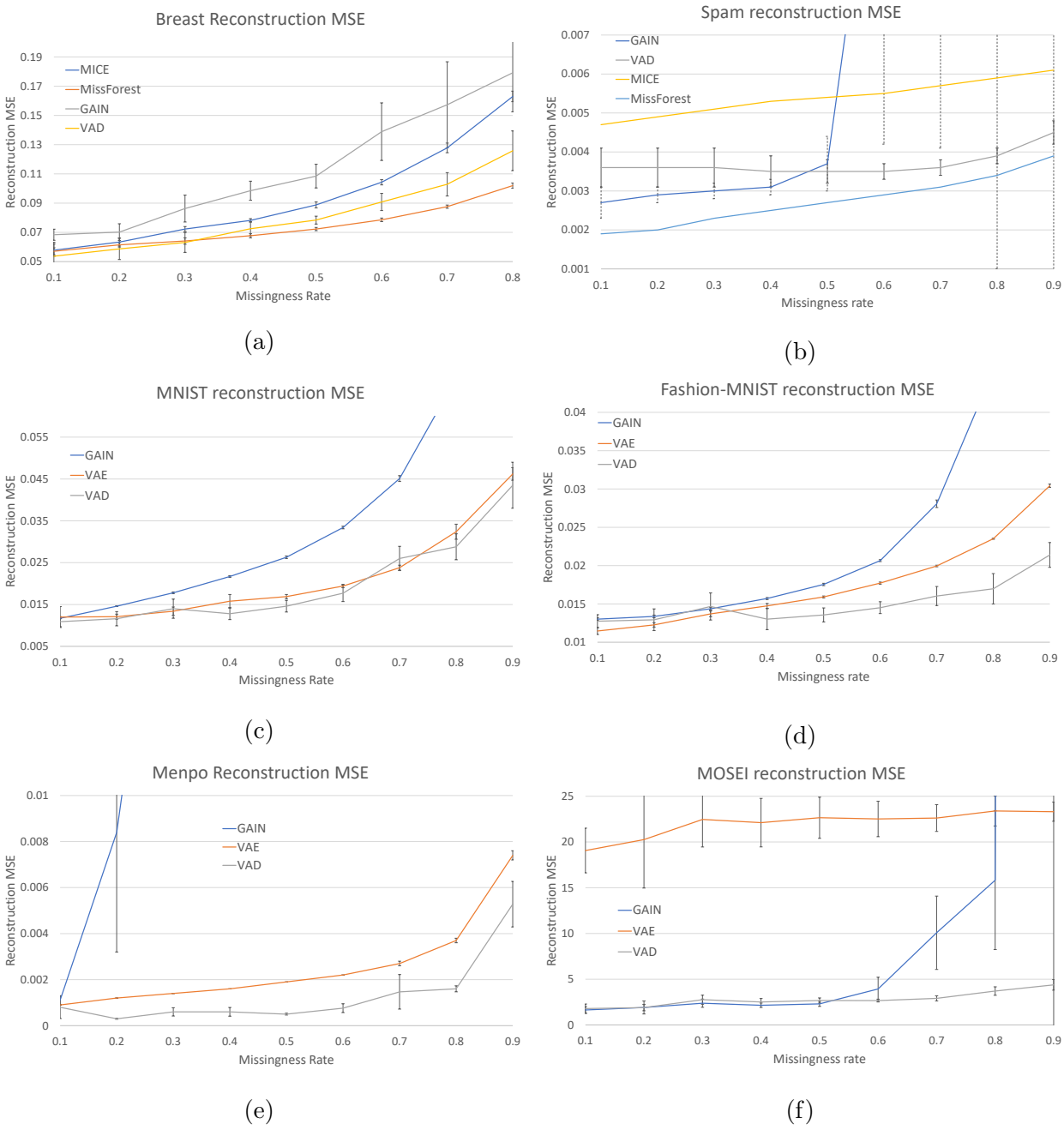

Figure 4.1: MSE values on unobserved values in the test set for Experiment 1 on 6 datasets described in Section 4.2.3: the Breast Cancer (a) and Spambase (b) UCI datasets, the MNIST (c) and Fashion-MNIST (d) image datasets, the Menpo facial landmarks dataset (e), and the MOSEI multimodal language dataset (f). VAD frequently outperforms other models in reconstructing missing values across all rates of missingness.

as the rate of missingness increases, and the generator in GAIN is unable to learn a good imputation model.

4.3.2 Experiment 2: Missingness rate sensitivity

To further investigate the limitations of the VAE model with incomplete data, we trained a denoising VAE [27], which, given an input with missing values, tries to predict the complete version. The training objective is then to minimize the reconstruction loss between the predicted output and the complete datapoint. We first experimented with having a fixed ratio of missing values during training. For example, some models would be trained only on data where 50% of the training data is missing. Figure 4.2 shows the mean-squared error on the missing values on the Fashion-MNIST test set with various rates of missing values, using models trained using different rates of missing values. The results show that when trained on only a single rate of missing values, the model does not perform well when given test data with different rates of missing values, showing that it is unable to generalize the imputation procedure for different rates of missing values. This aligns with the results obtained in Section 4.3 in Scenario 2, where the performance of the VAE trained only on complete data dramatically worsened on the test set as the rate of missingness increased.

We also trained GAIN and VAD for comparison. The results are shown in Figure 4.3 for GAIN, and Figure 4.4 for VAD. Similar to VAE, GAIN is sensitive to differences in the rate of missingness between train and test time. GAIN's performance degrades if the rate of missingness during test time is higher than during train time, like VAE, but does not worsen as much if the rate during test time is lower than during train time. This suggests that the generator in GAIN mainly learns to impute a specific rate of missingness and cannot generalize well to different rates of missing values.

On the other hand, VAD remains consistent regardless of the rate of missingness during train and test time, greatly outperforming GAIN and VAE. This is likely due to the

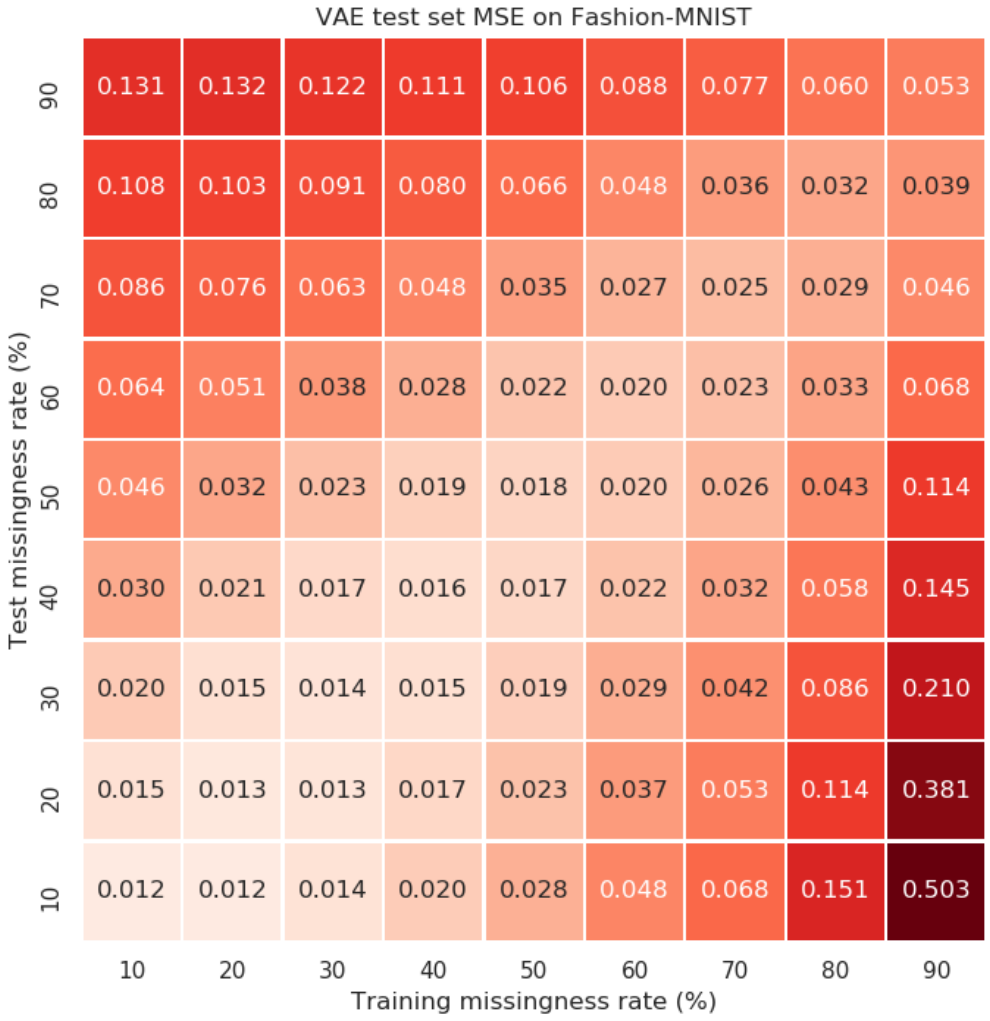

Figure 4.2: MSE loss on the Fashion-MNIST test set of VAE trained on training data with a single rate of missing data (x-axis) and tested on different rates of missing data during testing (y-axis). Models trained on only a single rate of missing data are unable to handle drastically different rates of missing data well, as shown by the increased error at the off-diagonal squares of the grid.

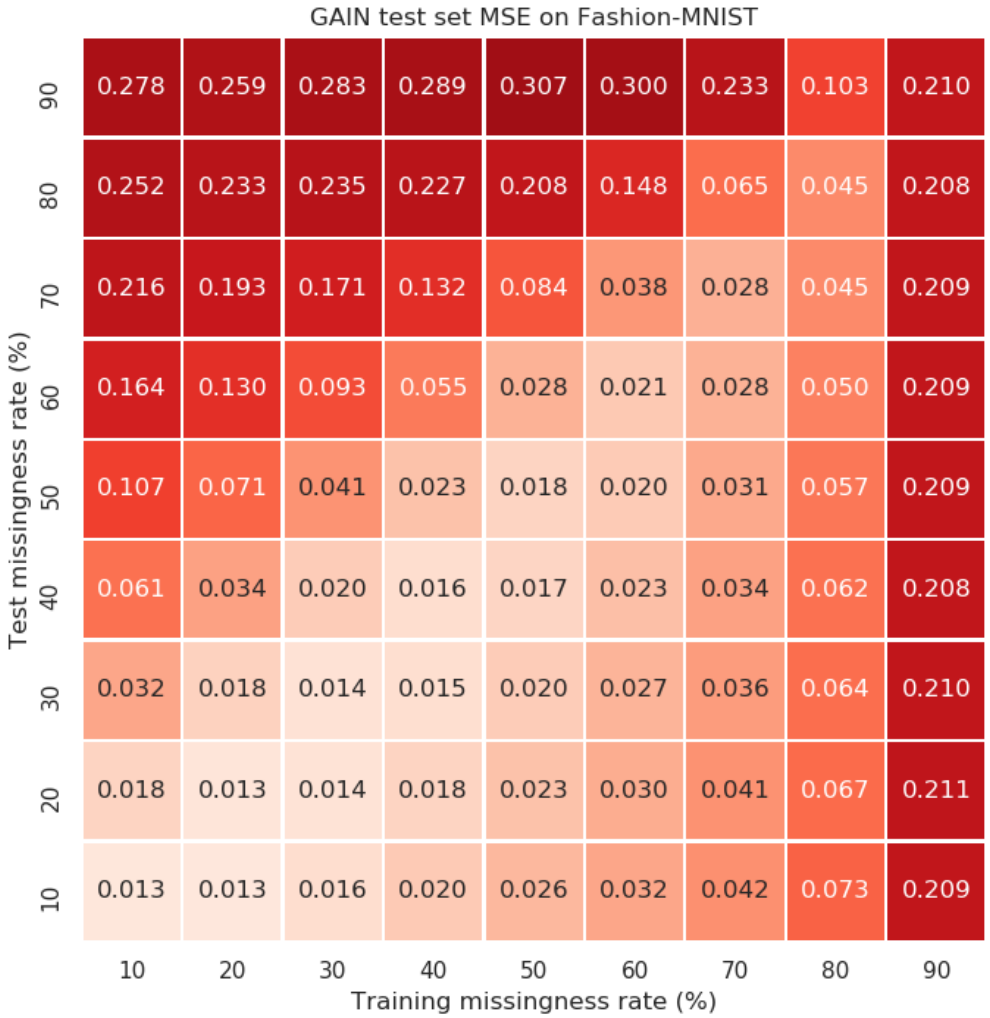

Figure 4.3: MSE loss on the Fashion-MNIST test set of GAIN trained on training data with a single rate of missing data (x-axis) and tested on different rates of missing data during testing (y-axis). Models trained on only a single rate of missing data are unable to handle drastically different rates of missing data well, as shown by the increased error at the off-diagonal squares of the grid. Compared to VAE, GAIN is slightly better at imputing missing values when the rate of missingness is lower during test time than at train time.

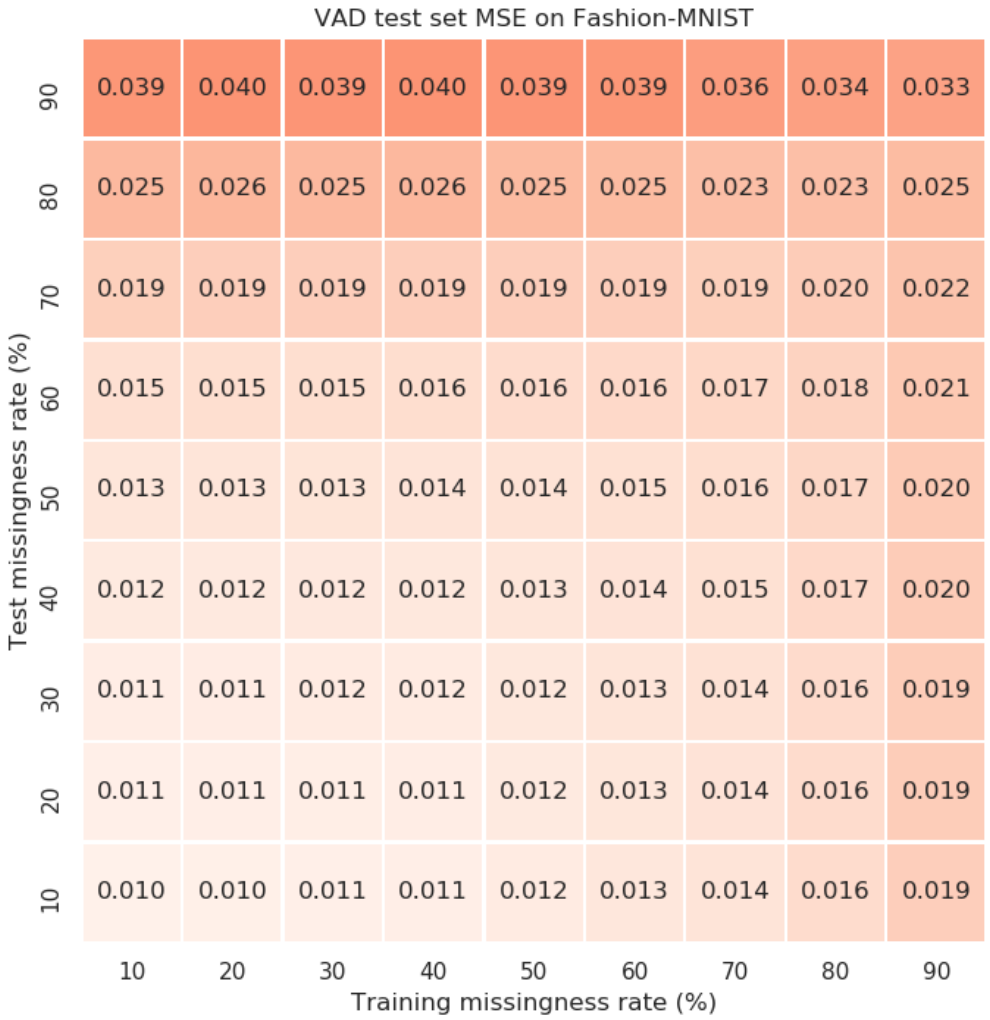

Figure 4.4: MSE loss on the Fashion-MNIST test set of VAD trained on training data with a single rate of missing data (x-axis) and tested on different rates of missing data during testing (y-axis). Unlike GAIN and VAE, VAD remains robust to various rates of missingness regardless of the rate of missingness seen during training.

lack of an encoder that creates a dependence between the parameters of the approximate posterior and the input, so the model is trained to be robust to different rates of missingness.

4.3.3 Experiment 3: Samples from image datasets

In addition to the numerical results presented above, we also inspect some of the imputations generated by the GAIN, VAE, and VAD models on the image datasets. This is equivalent to the task of inpainting, since inpainting involves filling in the values of missing pixels. We inspect imputations on both the MNIST and Fashion-MNIST datasets. On both datasets, pixels were randomly dropped and used as inputs for the models during test time.

We additionally inspect imputations on the CelebA dataset under two scenarios: first, where central block is missing, and secondly, where a region of the image is missing.

MNIST

Sample inpaintings on the MNIST dataset are shown in Figure 4.5. When trained on complete data, the VAD is still able to reconstruct the original image quite well even when only given partial data (Figure 4.5a). GAIN performs slightly worse than VAD visually, while the VAE only reconstructs the original input. This suggests that the VAD is more robust to increasing rates of missing values compared to the other two models, although this may also be partly due to the training objective of the VAE when given complete data during train time, which then trains it to reconstruct the input, and not impute the missing pixels.

When training is done on data with a similar missingness rate as test time (in Figure 4.5b), both VAE and VAD seem to perform equally well with GAIN performing slightly worse. All the models have poor performance in both models when 90% of the image is missing, likely due to the small amount of observed data given to the models. The relatively equal performance is likely due to the simplicity of the MNIST dataset.

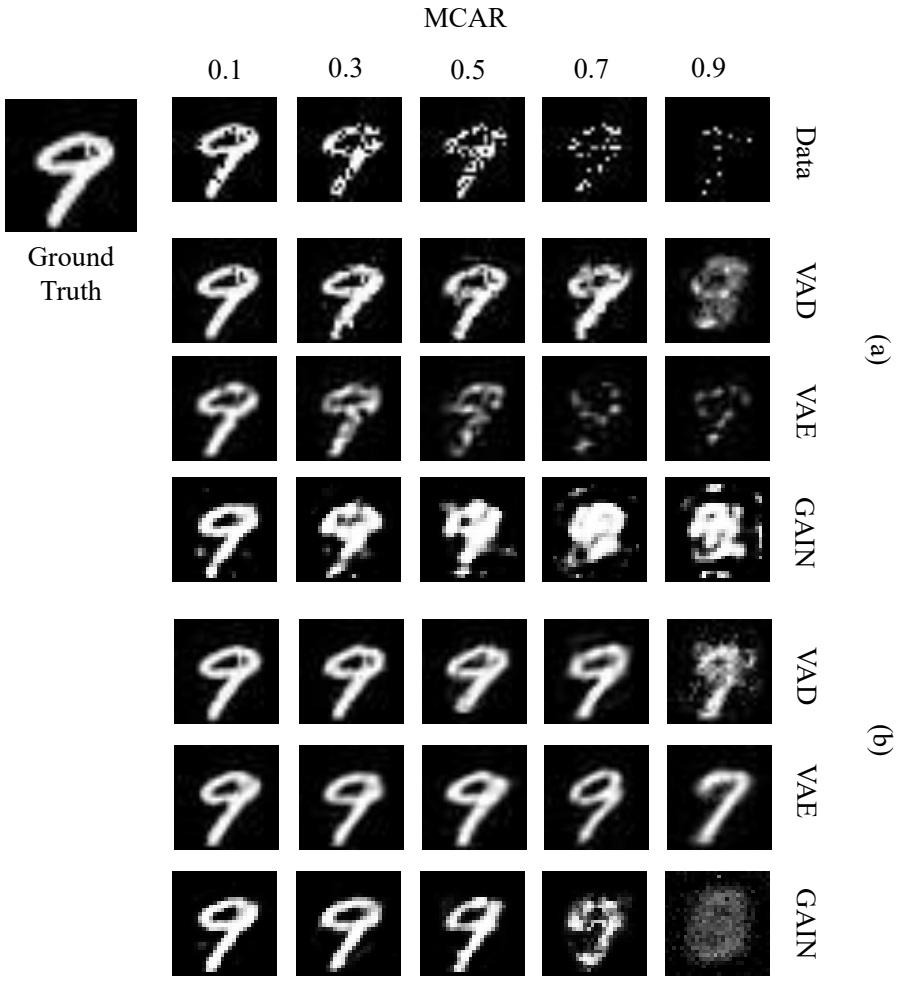

Figure 4.5: Visualization of inpainting for experiments on a MNIST datapoint. The top row (Data) shows the incomplete data given to the models. Ground Truth is the complete 28×28 8-bit grayscale image. For (a), training is done on complete data and testing is done on incomplete data. For (b) training and testing is done on incomplete data with similar missing rate. In both cases, VAD sees to outperform both VAE and GAIN visually, particularly in (a), although all models start to fail at 90% missingness.

Fashion-MNIST

On Fashion-MNIST, we additionally experimented with random 4×4 blocks missing during test time. Visualizations of some results are shown in Figure 4.6. When training is done on complete data (in Figure 4.6a), the VAE model performs poorly on test data with missing values, and ends up only producing a blurred version of the input and does not impute the missing pixels. In contrast, the VAD model is able to reconstruct the ground truth image relatively well, even when the rate of missing data is high, only producing visible failure when the missingness rate $r = 0.9$. This suggests that the VAD framework is indeed more robust to missingness introduced during test time when trained on complete data, as the encoder in the VAE model is not able to obtain a good approximate posterior when presented with missing values during test time. Like in the MNIST results, GAIN performs slightly worse than VAD but better than VAE, up to about 50% missingness.

When training is done on data with a similar missingness rate as test time (in Figure 4.6b), the VAE model produces better inpaintings than in a), but visually still seems to perform worse than VAD. In particular, in Figure 4.6b) the VAE model is able to reconstruct the shape of the original image, but seems unable to recreate textures, while VAD is able to. This again suggests that the encoder in VAE is unable to learn a good approximation of the posterior when given missing data, even if it is trained on incomplete data. GAIN performs similarly to both VAE and VAD on MCAR missingness, but performs poorly when blocks are missing.

CelebA

To investigate the generative utility of the VAD framework, we generated multiple inpaintings on the CelebA test set. Figure 4.7 shows some inpainted samples, obtained by randomly initializing the latent variables multiple times and fitting the latent variables to the incomplete image. The images show that the VAD model is able to produce a range of plausible inpaintings

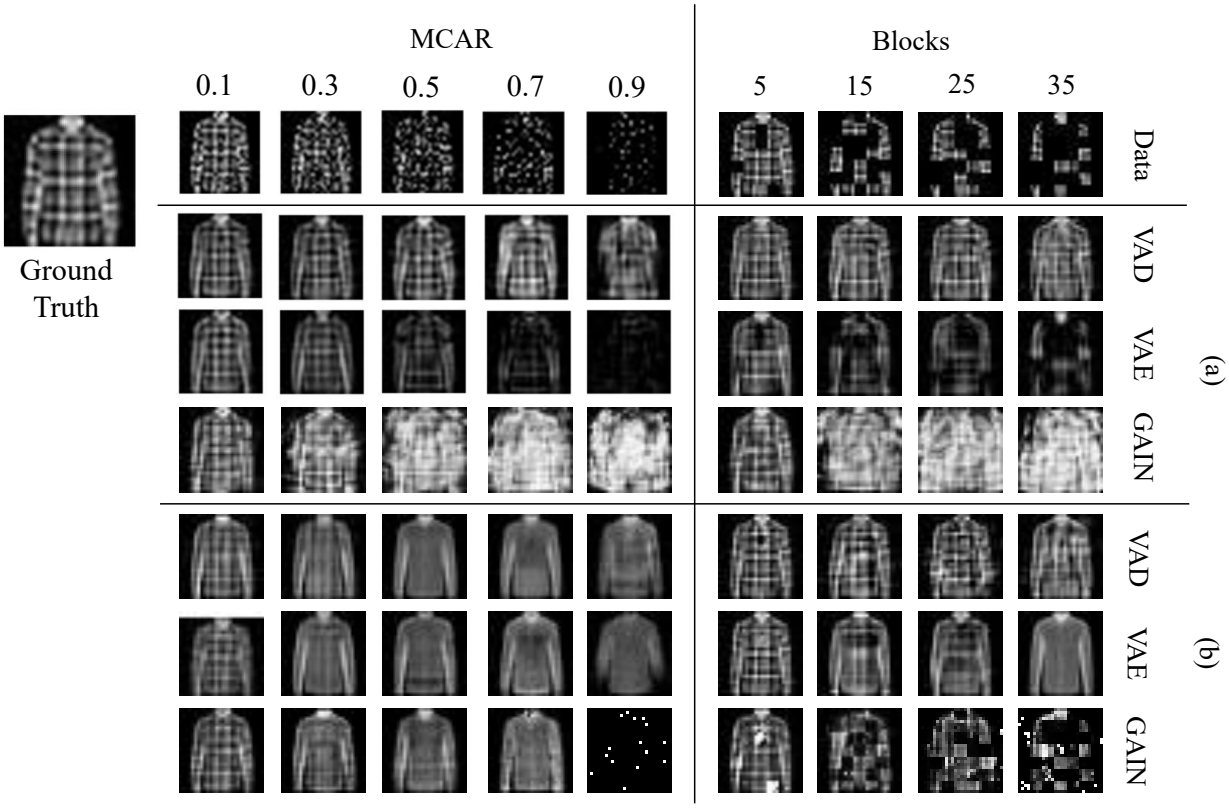

Figure 4.6: Visualization of inpainting for experiments on a Fashion-MNIST datapoint. The top row (Data) shows the incomplete data given as input to the models. Ground-truth is the complete 28×28 8-bit grayscale image. For (a), training is done on ground-truth data and testing is done on incomplete data. For (b) training and testing is done on similar missing rate. In both cases, VAD seems to outperform both VAE and GAIN visually, especially in (a), where the models are trained and tested on different rates of missingness.

for the same image, although the blockiness of the images could be attributed to the use of the DCGAN architecture as the generator network, which has been known to produce checkerboard-like artifacts in generated images [19]. These results suggest that the VAD model is indeed learning a distribution over the data space and is able to generate a variety of images conditioned on the observed values.

We also test the model on an instance of missing not at random (MNAR) missingness, by removing a region of the face, such as part of the hair, or face. Some inpaintings are shown in Figure 4.8. The images show that VAD is able to generate relatively plausible inpaintings, showing that it can handle other patterns of missingness as well.

Figure 4.7: Samples generated by the VAD model on the CelebA dataset, when inpainting a central block. The first column is the ground truth, the second is the input given to the VAD model, and the remaining columns are sample inpaintings produced by the model. The VAD model is able to produce a range of plausible inpaintings for the same image.

Figure 4.8: Samples generated by the VAD model on the CelebA dataset, when inpainting a region, as a case of MNAR missingness. The first column is the ground truth, the second is the input given to the VAD model, and the last column is a sample inpainting produced by the model. The VAD model is able to produce plausible inpaintings for each image.

Chapter 5

Conclusion and Future Work

This thesis presented the variational autodecoder (VAD) framework, a generative model based on the variational autoencoder that tackles the problem of modeling posterior distributions from incomplete data by directly optimizing the parameters of the approximate posterior during both train and test time. By avoiding an encoder that approximates the posterior using the (incomplete) input, we obtain a generative model that is more robust to various patterns and rates of missingness during both training and testing without the need for data augmentation or domain-specific tricks. This provides a simple framework that can be adapted for different domains and extended to many applications of imputation such as inpainting.

Experiments on a diverse range of datasets show superior performance to GAIN and VAE models in the presence of missing data, in situations where either similar or different rates of missingness were seen during training or testing. VAD particularly outperforms both models when the rates of missingness between train and test time are different, as the encoder in VAE and generator in GAIN only learn to impute rates of missingness seen during training, while the lack of such an encoder in VAD gives a generative model that is robust to unseen rates of missingness.

5.1 Future Work

Possible areas of further study include:

- *Increased inference speed*: One major drawback of the VAD framework is the need for gradient descent to find the optimal latent posterior distribution for the given (partial) input, even during test time. Exploring methods to improve this inference speed, especially during test time, would improve the practicality of using the framework in real systems. One possible solution could be to ensure that the decoder is invertible by using normalizing flows in the decoder network [22], and adapting it for the case of partial inputs.

- *Detecting missing/noisy values*: Currently, the VAD framework assumes that the location of the missing values is known, through the provided mask. However, in some cases, such a mask might not be available, such as in the case of noisy data. For example, a sensor that is faulty but is still reporting values would appear in the dataset, but would ideally be ignored by any predictive system trained on it [6]. Since VAD models are optimized by maximizing the likelihood of the datapoints, noisy values could potentially be detected by comparing the final output of the VAD model, and the actual values in the dataset, to find consistent discrepancies that could indicate a faulty/unreliable feature.

Bibliography

[1] Sanjeev Arora, Yingyu Liang, and Tengyu Ma. "A Simple but Tough-to-Beat Baseline for Sentence Embeddings". In: *International Conference on Learning Representations* (2017).

[2] Melissa J Azur et al. "Multiple imputation by chained equations: what is it and how does it work?" eng. In: *International journal of methods in psychiatric research* 20.1 (Mar. 2011), pp. 40–49. ISSN: 1557-0657. DOI: 10.1002/mpr.329. URL: https://www.ncbi.nlm.nih.gov/pubmed/21499542.

[3] Andrew Brock, Jeff Donahue, and Karen Simonyan. "Large Scale GAN Training for High Fidelity Natural Image Synthesis". In: *ArXiv* abs/1809.11096 (2018).

[4] Stef van Buuren and Karin Groothuis-Oudshoorn. "mice: Multivariate Imputation by Chained Equations in R". In: *Journal of Statistical Software, Articles* 45.3 (2011), pp. 1–67. ISSN: 1548-7660. DOI: 10.18637/jss.v045.i03. URL: https://www.jstatsoft.org/v045/i03.

[5] Stef van Buuren and Karin Oudshoorn. *Flexible multivariate imputation by MICE*. English. Tech. rep. Leiden: Netherlands Organization for Applied Scientific Research (TNO), Oct. 1999.

[6] Raghavendra Chalapathy and Sanjay Chawla. *Deep Learning for Anomaly Detection: A Survey*. 2019.

[7] Gilles Degottex et al. "COVAREP - A collaborative voice analysis repository for speech technologies". In: *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Florence, Italy, 2014. URL: https://covarep.github.io/covarep/pdfs/COVAREP_ICASSP14.pdf.

[8] Dheeru Dua and Casey Graff. *UCI Machine Learning Repository*. University of California, Irvine, School of Information and Computer Sciences, 2017. URL: http://archive.ics.uci.edu/ml.

[9] Craig Enders. *Applied Missing Data Analysis*. English. 1st ed. Methodology in the Social Sciences. The Guilford Press, Apr. 2010. ISBN: 978-1-60623-639-0.

[10] Andrew Gelman and Jennifer Hill. "Data Analysis Using Regression and Multilevel/Hierarchical Models". In: *Data Analysis Using Regression and Multilevel/Hierarchical Models*. Cambridge University Press, Dec. 2006, p. 648. ISBN: 978-0-521-68689-1.

[11] Ian Goodfellow et al. "Generative Adversarial Nets". In: *Advances in Neural Information Processing Systems 27*. Ed. by Z. Ghahramani et al. Curran Associates, Inc., 2014, pp. 2672–2680. URL: http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf.

[12] Oleg Ivanov, Michael Figurnov, and Dmitry Vetrov. "Variational Autoencoder with Arbitrary Conditioning". In: *International Conference on Learning Representations*. 2019. URL: https://openreview.net/forum?id=SyxtJh0qYm.

[13] Diederik P Kingma and Max Welling. "Auto-encoding variational bayes". In: *arXiv preprint arXiv:1312.6114* (2013).

[14] Y. LeCun et al. "Gradient-Based Learning Applied to Document Recognition". In: *Proceedings of the IEEE* 86.11 (Nov. 1998), pp. 2278–2324.

[15] Paul Pu Liang et al. "Strong and Simple Baselines for Multimodal Utterance Embeddings". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational

Linguistics, June 2019, pp. 2599–2609. DOI: 10.18653/v1/N19-1267. URL: https://www.aclweb.org/anthology/N19-1267.

[16] Roderick Little and Donald Rubin. *Statistical Analysis with Missing Data*. English. 2nd ed. Wiley Series in Probability and Statistics. Wiley-Interscience, Sept. 2002. ISBN: 978-0-471-18386-0.

[17] Ziwei Liu et al. "Deep Learning Face Attributes in the Wild". In: *Proceedings of International Conference on Computer Vision (ICCV)*. Dec. 2015.

[18] Tim P Morris, Ian R White, and Patrick Royston. "Tuning multiple imputation by predictive mean matching and local residual draws". eng. In: *BMC medical research methodology* 14 (June 2014), pp. 75–75. ISSN: 1471-2288. DOI: 10.1186/1471-2288-14-75. URL: https://www.ncbi.nlm.nih.gov/pubmed/24903709.

[19] Augustus Odena, Vincent Dumoulin, and Chris Olah. "Deconvolution and Checkerboard Artifacts". In: *Distill* (2016). DOI: 10.23915/distill.00003. URL: http://distill.pub/2016/deconv-checkerboard.

[20] Deepak Pathak et al. "Context Encoders: Feature Learning by Inpainting". In: *CVPR*. 2016.

[21] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. "GloVe: Global Vectors for Word Representation". In: *Empirical Methods in Natural Language Processing (EMNLP)*. 2014, pp. 1532–1543. URL: http://www.aclweb.org/anthology/D14-1162.

[22] Danilo Jimenez Rezende and Shakir Mohamed. "Variational Inference with Normalizing Flows". In: *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*. ICML'15. event-place: Lille, France. JMLR.org, 2015, pp. 1530–1538. URL: http://dl.acm.org/citation.cfm?id=3045118.3045281.

[23] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. "Learning Structured Output Representation using Deep Conditional Generative Models". In: *Advances in Neural Information Processing Systems 28*. Ed. by C. Cortes et al. Curran Associates, Inc., 2015, pp. 3483–

3491. URL: http://papers.nips.cc/paper/5775-learning-structured-output-representation-using-deep-conditional-generative-models.pdf.

[24] Daniel J. Stekhoven and Peter Bühlmann. "MissForest—non-parametric missing value imputation for mixed-type data". In: *Bioinformatics* 28.1 (Oct. 2011), pp. 112–118. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btr597. URL: https://doi.org/10.1093/bioinformatics/btr597 (visited on 07/14/2019).

[25] Matthias Templ, Alexander Kowarik, and Peter Filzmoser. "Iterative stepwise regression imputation using standard and robust methods". In: *Computational Statistics & Data Analysis* 55.10 (2011), pp. 2793–2806. ISSN: 0167-9473. DOI: https://doi.org/10.1016/j.csda.2011.04.012. URL: http://www.sciencedirect.com/science/article/pii/S0167947311001411.

[26] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. "Deep Image Prior". In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2018.

[27] Pascal Vincent et al. "Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion". In: *J. Mach. Learn. Res.* 11 (Dec. 2010), pp. 3371–3408. ISSN: 1532-4435. URL: http://dl.acm.org/citation.cfm?id=1756006.1953039.

[28] Han Xiao, Kashif Rasul, and Roland Vollgraf. *Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms*. arXiv: cs.LG/1708.07747. Aug. 2017.

[29] Jinsung Yoon, James Jordon, and Mihaela van der Schaar. "GAIN: Missing Data Imputation using Generative Adversarial Nets". In: *Proceedings of the 35th International Conference on Machine Learning*. Ed. by Jennifer Dy and Andreas Krause. Vol. 80. Proceedings of Machine Learning Research. Stockholmsmässan, Stockholm Sweden: PMLR, July 2018, pp. 5689–5698. URL: http://proceedings.mlr.press/v80/yoon18a.html.

[30] AmirAli Bagher Zadeh et al. "Multimodal language analysis in the wild: CMU-MOSEI dataset and interpretable dynamic fusion graph". In: *Proceedings of the 56th Annual*

Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Vol. 1. 2018, pp. 2236–2246.

[31] S. Zafeiriou et al. "The Menpo Facial Landmark Localisation Challenge: A Step Towards the Solution". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. July 2017, pp. 2116–2125. DOI: 10.1109/CVPRW.2017.263.