# Variational Autoencoders for Supervision, Calibration and Multimodal Learning


Thomas W. Joy

St Catherine's College

University of Oxford

A thesis submitted for the degree of

*Doctor of Philosophy*

Trinity 2022

# Acknowledgements

I would like to extend my deepest gratitude to my supervisor Phil and my co-supervisor Sid. Their consistent and unwavering support has been invaluable throughout my PhD, I would like to thank them for believing in me and providing me with unbridled opportunities to undertake research, allowing me to develop both professionally and personally. I would also like to thank Puneet, for his inspirational mentorship and guidance throughout my PhD.

I would also like to extend my gratitude to Pawan, who supervised me throughout my Masters degree and set me on a path towards a PhD. His support and patience provided me with the confidence and vision to pursue a path in research.

I would also like to thank my undergraduate tutors, Byron Byrne and David Gillespie for their academic and administrative support during my applications. Furthermore, I would also to thank the staff at St Catherines College for being the unsung heroes and heroines of the university. I would also like to particularly thank Joanna Zapisek for her continued support and reassurance throughout the process.

From a personal point of view, I wold like to thank my family for believing in me and providing essential emotional support. I would also like to thank my friends for their empathy during the difficult times and covering the Uber when I was financially insolvent. Finally, to anyone else who was part of the past four years, no matter how insignificant, I would like to say thank you for being a part of this capricious but immensely gratifying period of my life.

# Abstract

Learning representations of data has long been a desirable goal in machine learning. Constructing such representations enables downstream tasks such as classification or object detection to be preformed efficiently. Furthermore, it is desirable to have these representations be constructed in such a way so they are *interpretable*, which allows for fine grained intervention and reasoning on characteristics of the input. Other tasks may include, cross-generation between modalities, or calibrating predictions such that their confidence matches their accuracy. An effective way to learn representations is through a Variational Autoencoder (VAE), which performs variational inference on the latent variables of the observable input. In this thesis we show how the VAE, can be utilised to: incorporate label information into the learning process; learn shared-representations of multimodal data; and calibrate predictions of existing neural classifiers.

Data sources are often accompanied by additional label information, which may indicate the presence of a characteristic in the input. A question naturally arises as to whether the additional label information can be used to structure the representation such that it provides a notion of interoperability about the characteristic; such as "to what extent is the person smiling?". The first contribution of this thesis is to address the aforementioned problem and propose a method which successfully uses label information to structure the latent space. Furthermore, this allows us to perform additional tasks such as fine grained interventions; classification; and conditional generations. Moreover, we are also successfully able to handle the case when label information is missing, drastically reducing the data burden when training these models.

Rather than being presented with labels, we sometimes instead observe another unstructured observation of the same object, *e.g.* a caption of an image. In this scenario, the objective changes slightly to one where the model is able to learn shared-representations of data, allowing it to perform cross-generations between modalities. The second contribution of this theses addresses this problem. Here, learning is performed by employing *mutual supervision* between the modalities and introducing a bi-directional objective, which faithfully ensures symmetry in the model. Furthermore, by virtue of this approach, we are able to learn these

representations in situations where some of the modalities may be missing during training.

Uncertainty quantification is an important task in machine learning, with it now being well known that current deep learning models severely overestimate their confidence. The final contribution of this thesis is to address how the representations of VAEs can be used to extract reliable confidence estimates for neural-classifiers. This investigation leads to a novel approach to calibrate neural-classifiers, which is applied post-hoc to off the shelf classifiers and is very fast to train and test.

# Contents

# List of Figures

# Publications

Throughout my thesis I have compiled the following publications, all of which are published in the proceedings of major conferences, journals or workshops. I am the sole first author on Joy et al. (2021, 2022, 2023), where the development of objective functions, programming and writing was completed by myself. The additional authors served to guide and advise me on the narrative of the project and provided helpful feedback on the manuscripts. These three publications will make up the individual chapter of my thesis due to their research into VAEs. For the additional publications (Joy et al., 2019; Tonioni et al., 2019), despite the fact I contributed significantly to the development, writing and coding, I chose to omit them from the main body of this thesis due to a lack of coherence between them and the other publications.

- T. Joy, A. Desmaison, T. Ajanthan, R. Bunel, M. Salzmann, P. Kohli, P. H. Torr, and M. P. Kumar. Efficient relaxations for dense CRFs with sparse higher-order potentials. *SIAM journal on imaging sciences*, 12(1):287–318, 2019.

- A. Tonioni, O. Rahnama, T. Joy, L. D. Stefano, T. Ajanthan, and P. H. Torr. Learning to adapt for stereo. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9661–9670, 2019

- T. Joy, S. M. Schmon, P. H. S. Torr, N. Siddharth, and T. Rainforth. Capturing label characteristics in VAEs. In *International Conference on Learning Representations*, 2021

- T. Joy, Y. Shi, P. H. S. Torr, T. Rainforth, S. M. Schmon, and N. Siddharth. Learning multimodal VAEs through mutual supervision. In *International Conference on Learning Representations*, 2022

- T. Joy, F. Pinto, S. Lim, P. H. S. Torr, and P. K. Dokania. Sample-dependent adaptive temperature scaling for improved calibration. *AAAI Conference on Artificial Intelligence*, 2023

# 1 Introduction

## 1.1 Preamble

For machine learning models to be applied on complex domains such as image or language problems, they need to be able to construct representations, or features, which encapsulate the information in such a way that downstream tasks can easily be performed. Typically, these features are learned, avoiding the need to construct hand-crafted features. There are multiple approaches to learning these features, ranging from autoencoders, which reconstruct the input (Hinton and Zemel, 1994; Kingma and Welling, 2013; Devlin et al., 2018; He et al., 2021); through to discriminative approaches which learn the features by explicitly optimising a task (He et al., 2016; Simonyan and Zisserman, 2014). Given the motivation for learning features, it is natural now to ask what tasks the features are needed for.

For an artificial agent to interact in the world it firstly needs the capability to recognise and identify objects in its surrounding. Doing so is one of the most prominent tasks needed to be performed by an artificial agent, and falls under the broader objective of recognition. Within this objective there are numerous specific tasks such as classification (Simonyan and Zisserman, 2014; He et al., 2016), semantic-segmentation (Long et al., 2015) and object detection (Felzenszwalb et al.,

2010). These aforementioned tasks form an essential part of an agent's pipeline, typically they will represent the first stage, with planning and execution following subsequently. Clearly then, it is essential that these recognition tasks can be performed quickly, but also it is imperative that they are done with a reasonably high accuracy. Moreover, it is essential that the predictions also produce reliable uncertainty estimates, enabling the planning module to successfully settle on its actions whilst mitigating adverse consequences.

The applications of artificial intelligence and machine learning extend beyond the realm of an agent acting in the real world. For instance we may want systems which can generate example instances of data, such as art or maps in a video game. Here representation learning also plays a critical role, as the representation will contain enough information to reconstruct (Hinton and Zemel, 1994) or generate (Kingma and Welling, 2013) instances of data. Moreover, by taking the generative approach of the VAE (Kingma and Welling, 2013), we are also able to obtain a bound on the likelihood of an observation.

Whilst evaluating the likelihood and generating examples of data is a useful ability, often we want to specify the properties of the generation (conditional generation) or generate another instance which shares the same underlying attributes (cross-generation). Doing so requires us to learn representation whilst reconciling multiple pieces of information such as: `image`, `caption` and `label`; referred to as multi-modal learning. The ability to leverage multiple pieces of information to learn representation is not dissimilar to how human brains learn representations, which jointly embeds information across different modalities (Quiroga et al., 2009; Stein et al., 2009) enabling reasoning and understanding between them (Bauer and Johnson-Laird, 1993; Fan et al., 2015).

In this work we explore the uses of the representations learnt through a VAE, which is a flexible framework enabling the fast learning of features through variational inference. Specifically, the contributions of thesis are as follows:

- We provide a novel method for integrating label information into the learning of a VAE. Doing so enables the construction of representations which encapsulate and isolate the characteristics of the label, enabling interventions on specified characteristics. Furthermore, this model can also perform additional tasks such as conditional generation and classification, surpassing existing models in accuracy. This model is explored in Chapter 3.

- Secondly, we demonstrate how shared-representations for multiple modalities can be learned through the use of mutual-supervision and a bi-directional objective, permitting cross-generations between the modalities. Furthermore, we also demonstrate how this approach can be utilised in the case where one of the modalities may be missing during training, reducing the dependence on paired data in the training dataset. This approach is demonstrated in Chapter 4.

- Finally, in Chapter 5, we address the problem of calibration in neural-classifiers, which are known to be severely miscalibrated. In this section we highlight how the representations learned by a VAE can be used to provide a basis to produce reliable confidence estimates for neural-classifier predictions.

## 1.2 Variational Autoencoder

Stochastic Variational Inference (VI) (Hoffman et al., 2013) was popularised by the introduction of the VAE (Kingma and Welling, 2013; Rezende et al., 2014) which enabled a scalable and fast method to perform inference in graphical models. They combine deep autoencoders (Hinton and Zemel, 1994) with generative latent-variable models; resulting in a model which captures the generative factors of the observation in a low dimensional representation. Unlike deep autoencoders, VAEs capture representations of data, not as distinct values corresponding to observations, but rather as *distributions* of values. Before proceeding to definitions and explanations, we will now briefly describe the motivation for learning VAEs.

**Why do we care about VI?** Often, the distribution of observations in the real world may be dependant on a set of common factors. Considering the attributes of a face, examples of these factors may be attributes such as: `pose`, `hair color`, `skin color`, `facial hair`, `sex`, `facial expression`. It is natural then, that along with modelling the distribution of observations, we should also learn a set of common factors and the causal relationship between them and their corresponding realisations in data.

Mathematically, we can represent this relationship through a graphical model, with the random variables $\mathbf{x}$ and $\mathbf{z}$ corresponding to the data observations and common factors, which are commonly referred to as latent variables; the graphical model in Figure 1.1 indicates this causal relationship. Modelling such a relationship allows us to explicitly intervene on the factors of a data sample, allowing for visual changes in the output. Moreover, we may want to structure the latent variables in such a way that we can isolate and alter selected characteristics in data space by manipulating known factors in the latent space; a feature known as *disentanglement* in the VAE literature.

Learning is achieved by introducing an approximate posterior into the learning process, and reconstructing the data sample in a similar manner to a standard autoencoder. Doing so subsequently allows us to perform inference on the data samples, permitting us to reason and draw conclusions about the nature of the data sample. However, one can not simply treat this like a standard autoencoder. Due to the stochastic nature of the latent space, learning via gradient descent can quickly become problematic due to the variance of the gradient estimator. A variety of methods have been proposed to address learning which will now be discussed.

### VAE Background

The REINFORCE (Williams, 1992) technique used to be a popular approach to obtain unbiased gradient estimates in variational inference, as it enabled the use of non-differentiable cost functions. However, in practice, this estimator still has a

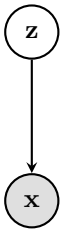



**Figure 1.1:** Graphical representation between data observations **x** and their corresponding generative factors **z**.

variance which is too high to learn effectively. To some extent, this can be alleviated through the use of control variates (Ranganath et al., 2014; Paisley et al., 2012), enabling effective learning of the parameters.

The wake-sleep algorithm (Hinton et al., 1995) is another popular approach to solve the problem of stochastic variational inference, which applied an inference model to approximate the true posterior and a generative model for the observed variable and the continuous latent variables. Learning occurred via the joint optimisation of two objectives, which failed to provide a bound on the log-evidence. A re-weighted version of the wake-sleep algorithm was introduced by Bornschein and Bengio (2014), which is shown to be beneficial to discrete latent models (Le et al., 2020).

Another approach which needs to be mentioned is that of the Denoising Autoencoder (DAE) (Vincent et al., 2010; Bengio et al., 2013b). Here, the inputs are corrupted with noise and then the autoencoder reconstructs to the clean input. Sampling is then performed through Langevin or Metropolis-Hastings MCMC; which introduces a high computational burden compared to the VAE.

Recently, the most effective approach to emerge is that of estimating the pathwise derivative, which is also termed the: the process derivative (Pflug, 2012); the general area of perturbation analysis (Glasserman and Ho, 1991); the pathwise derivative (Glasserman, 2004); and more recently as the reparameterisation trick in stochastic back-propagation (Kingma and Welling, 2013; Rezende et al., 2014). Here the stochastic computation graph is constructed and is thus amenable to back-propagation and gradient descent.

**Definition**

The VAE defines a generative model which is a joint distribution over observed data $\mathbf{x}$ and latent variables $\mathbf{z}$ as $p_\theta(\mathbf{x}, \mathbf{z}) = p(\mathbf{z})p_\theta(\mathbf{x} \mid \mathbf{z})$. Given this model, learning representations of data can be viewed as performing *inference*—learning the *posterior* distribution $p_\theta(\mathbf{z} \mid \mathbf{x})$ that constructs the distribution of latent values for a given observation. To do this, VAEs employ amortised VI (Wainwright and Jordan, 2008; Kingma and Welling, 2013) which approximates the intractable posterior $p_\theta(\mathbf{z} \mid \mathbf{x})$ as a variational approximation $q_\phi(\mathbf{z} \mid \mathbf{x})$ by predicting the parameters of the distribution; the likelihood $p_\theta(\mathbf{x} \mid \mathbf{z})$ is parameterised using a neural-network decoder. Using this variational approximation of the posterior and by maximising the log-likelihood through importance sampling, we are able to perform effective estimation of the objective which derives as the Evidence Lower Bound (ELBO) of the model

$$\log p_\theta(\mathbf{x}) = \log \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\frac{p_\theta(\mathbf{z}, \mathbf{x})}{q_\phi(\mathbf{z} \mid \mathbf{x})}\right] \geq \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log \frac{p_\theta(\mathbf{z}, \mathbf{x})}{q_\phi(\mathbf{z} \mid \mathbf{x})}\right] \equiv \mathcal{L}(\mathbf{x}; \phi, \theta). \quad (1.1)$$

Here, the observations $\mathcal{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ are random variables sampled from an unknown distribution $p_\mathcal{D}(\mathbf{x})$, which are used for training using stochastic gradient descent.

## 1.3 Variational Autoencoders for Supervision, Calibration and Multimodal Learning

Given this short summary of VAE background, we now provide a short introduction, and motivate the individual contributions of this thesis.

### 1.3.1 Capturing Label Characteristics in VAEs

Label information is often present with the corresponding data sample, such as class information or attributed information in an image. Naturally, it would prove beneficial if we can leverage this information when constructing the latent representations, as doing so enables a whole host of additional tasks to be performed. Specifically, this would enable us to perform classification and conditional generations, but also

enable us to alter characteristics in the resulting reconstructions without affecting others.

The typical approach to learning Semi-Spervised VAEs (SSVAEs), is to place the labels $\mathbf{y}$ directly into the latent space (Kingma et al., 2014; Siddharth et al., 2017), leading to $\mathbf{z} = \{\mathbf{z}_{\setminus \mathbf{c}}, \mathbf{y}\}$, with $\mathbf{z}_{\setminus \mathbf{c}}$ representing the non-labelled generative factors such as background. In situations where $\mathbf{y}$ is observed, the latent variable is supervised and variational inference is performed when it is not. Whilst this approach is able to attain a high classification accuracy, it completely fails to encapsulate the characteristics of the class into the latent space; this is primarily due to the restrictive capacity of the discrete latent variable $\mathbf{y}$. This pathology prevents fine grained interventions from being performed, such as adjusting the extent to which a class is present in the reconstruction, restricting the model to only performing binary interventions, i.e. the person is `smiling` or they are not.

Here, instead of capturing the labels directly as discrete latent variables, we directly learn to capture the characteristics indicated by the label. For example, rather than simply capturing if the person is `smiling`, we encapsulate the characteristics of the smile in a continuous space, allowing much finer control when performing interventions of conditional generations. Full details are provided in Chapter 3.

### 1.3.2 Multi-Modal learning through Mutual Supervision

Labels typically provide the minimum amount of information for a class or attribute, and often these labels have to be explicitly obtained. Instead data may often come in the form of pairs, i.e. a caption to an image or a set of matching images displaying digits. In this situation, we require models which are able to learn the relationship between these modalities and construct shared representations. Doing so enables the functionality to perform cross-generation or manipulate reconstructions using unstructured information, such as changing the color in the description of an object.

The goal of learning these joint representations is referred to as multi-modal learning, where here we restrict ourselves to the use of VAEs. Approaches in the literature (Wu

and Goodman, 2018b; Shi et al., 2019a) reconcile these multiple modalities by factorising the approximate posteriors as a product (Wu and Goodman, 2018b) or mixture (Shi et al., 2019a). However, this approach typically fails in the case where one of the modalities may be unobserved (Wu and Goodman, 2018b) or is not possible due to the construction of the model (Shi et al., 2019a). Moreover, this approach further restricts the model as it requires priors to be placed over the latent space.

Rather than directly reconciling the posteriors through various factorisation, we instead learn shared representations by implicitly regularising the posteriors against each other through the Kullback–Leibler divergence (KL). This approach alleviates the need to factorise the posteriors and place a prior over the latent variables, allowing more flexibility in the model. Furthermore, this approach naturally extends to the case when one of the modalities may be missing during training. Details for this approach are given in Chapter 4.

### 1.3.3 Sample-dependent Temperature Scaling for Improved Calibration

It has recently become known that neural-classifier are miscalibrated, in that their confidence does not match their accuracy; typically this miscalibration manifests as overconfidence, where the confidence is higher than the expected accuracy (Guo et al., 2017). As a short example, if a classifier is 80% confident then we expect it to be 80% accurate.

To address the miscalibration problem, Guo et al. (2017) propose a simple method which scales the logits by a single scalar $T$. This values is obtained through cross-validation, by minimising the Expected Calibration Error (ECE), a quantity which represents the difference between expected accuracy and confidence. Whilst on average this approach obtains networks which are better calibrated, it fails to respect the fact that some samples will contribute to the ECE in a way which makes them over-confident and some will make the ECE underconfident.

Instead, rather than applying the same temperature value to all samples in the dataset, we instead predict the temperature on a *per-datapoint* basis. That is, we learn a temperature prediction network, which given an input $\mathbf{x}$, will predict the temperature. This gives the model the flexibility to respect the fact that not all data-points contribute equally to the ECE, unlike vanilla temperate scaling which uses the same value for *all* data-points. Assuming a temperature prediction network $T(\mathbf{x})$, the resulting predictive probability distribution is give by

$$p(y|\mathbf{x}) = \frac{\exp(\frac{f(\Phi(\mathbf{x}))}{T(\mathbf{x})})}{\sum_k \exp(\frac{f(\Phi(\mathbf{x}))_k}{T(\mathbf{x})})}. \tag{1.2}$$

This approach allows the network to assign lower confidences to samples it *should* be uncertain about and higher confidences to samples it should be confident about. Full details of this method are given in Chapter 5.

# 2

# Literature Review and Background

Here we provide an extended literature review to the contributions of this thesis. Specifically, the first three sections contain related work and background information to the first three chapters respectively; with Section 2.4 containing work which is relevant, but tangential to the contributions of this thesis.

## 2.1 Capturing Label Characteristics in Variational Autoencoders (VAEs)

This section can be viewed as an introduction to Chapter 3, and provides the relevant background information and notation needed.

### 2.1.1 Semi-Supervised VAE Background

Semi-Spervised VAEs (SSVAEs) (Kingma et al., 2014; Maaløe et al., 2016; Siddharth et al., 2017; Joy et al., 2021) consider the setting where a subset of data $\mathcal{S} \subset \mathcal{D}$ is assumed to also have corresponding *labels* $\mathbf{y}$. Denoting the (unlabelled) data as $\mathcal{U} = \mathcal{D} \backslash \mathcal{S}$, the log-marginal likelihood is decomposed as

$$\log p\left(\mathcal{D}\right) = \sum\nolimits_{(\mathbf{x},\mathbf{y})\in\mathcal{S}} \log p_\theta(\mathbf{x},\mathbf{y}) + \sum\nolimits_{\mathbf{x}\in\mathcal{U}} \log p_\theta(\mathbf{x}),$$

where the individual log-likelihoods are lower bounded by their Evidence Lower Bound (ELBO)s. Typically, the labels $\mathbf{y}$ are treated as a latent variables and

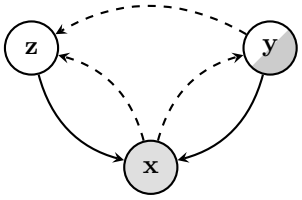



**Figure 2.1:** Graphical representation of the M2 model. Solid and dashed line indicates generative model and inference model respectively.

marginalised over whenever the label is not provided. In more detail, most approaches split the latent space in the following way $\mathbf{z} = \{\mathbf{z}_\mathbf{y}, \mathbf{y}\}$, such that each dimension of $\mathbf{y}$ explicitly represents a predicted value of a label, which is only known for the labelled datapoints. For the unlabelled datapoints, $\mathbf{y}$ has to be inferred or imputed. This approach stems from the motivation that most practitioners aim to perform classification, in the case where training is semi-supervised. Despite this, this approach is often used for learning representation where labels are sometimes present during training; allowing users to perform tasks such as manipulations according to a change in label, and also generate examples conditioned on a specified label. In this work we focus our attention on how this approach is unsuitable for the latter task, and introduce a novel method which outlines how labels can be used to improve the quality of representations in SSVAEs. We now provide an in depth explanation of the relevant models for SSVAEs.

**Related Models**

**M2** SSVAEs aim to incorporate labels value pairs $\{\mathbf{x}, \mathbf{y}\} \in \mathcal{S}$ which are only available for a subset of the data $\mathcal{S} \subset \mathcal{D}$. The answer to the question of how to introduce supervision through $\mathbf{y}$ is not immediately obvious. As previously eluded to, a naïve approach is to introduce an inductive bias by setting a portion of the latent space to be discrete (Kingma et al., 2014). These latent variables can then be directly supervised with the labels $\mathbf{y}$ when they are available and performing variational inference when they are not. Specifically, this produces the graphical model in Figure 2.1, where the approximate posterior factorizes as

$q(\mathbf{z}, \mathbf{y}|\mathbf{x}) = q(\mathbf{z}|\mathbf{x}, \mathbf{y})q(\mathbf{y}|\mathbf{x})$. For the supervised case, the ELBO derives as:

$$\log p_\theta(\mathbf{x}, \mathbf{y}) \geq \mathbb{E}q_\phi(\mathbf{z}|\mathbf{x}, \mathbf{y}) \log \frac{p_\theta(\mathbf{x}|\mathbf{y}, \mathbf{z})p(\mathbf{y})p(\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x}, \mathbf{y})}. \tag{2.1}$$

If the label is unobserved, then the label has to be imputed and marginalized:

$$\log p_\theta(\mathbf{x}) \geq \sum_{\mathbf{y}} \mathbb{E}q_\phi(\mathbf{z}, \mathbf{y}|\mathbf{x}) \log \frac{p_\theta(\mathbf{x}|\mathbf{y}, \mathbf{z})p_\theta(\mathbf{y})p(\mathbf{z})}{q_\phi(\mathbf{z}, \mathbf{y}|\mathbf{x})}. \tag{2.2}$$

The objective can then be formed using the above bounds

$$\mathcal{J} = \sum_{\{\mathbf{x},\mathbf{y}\}\in\mathcal{S}} \log p(\mathbf{x}, \mathbf{y}) + \sum_{\mathbf{x}\in\mathcal{U}} \log p(\mathbf{x}).$$

As we will now outline, this method introduces serious pathologies into the training of the model. This approach is known as the M2 model (Kingma et al., 2014), and is a cornerstone for semi-supervised learning in VAEs.

The first issue is this approach leads to a complete failure in its ability to achieve the two main goals of conditional generation and classification. For the supervised case when $\mathbf{x}$ and $\mathbf{y}$ are both observed, the mapping between them is detached, thus preventing any gradient updates to the parameters of the distribution $q(\mathbf{y}|\mathbf{x})$, consequently the method fails to perform classification or conditional generation. To amend this, the authors introduce a weighted classifier to the objective:

$$\mathcal{J}^\alpha = \mathcal{J} + \alpha \cdot \mathbb{E}_{p_\mathcal{S}(\mathbf{x},\mathbf{y})}[-\log q(\mathbf{y}|\mathbf{x})]$$

where $\alpha$ has to be tuned manually, with the assumed challenges implicit in hyper-parameter optimisation. This issue is further compounded as we have to decide if it is best to optimise $\alpha$ to improve the log-likelihood of the data or the classification accuracy? Furthermore, different choices of $\alpha$ lead to potentially more serious pathologies. Namely, whilst the balancing $\alpha$ may just seem like making a somewhat trivial trade-off between a variational term and a classifier, it is actually explicitly controlling the flow information into $\mathbf{z}$ or $\mathbf{y}$. If $\alpha$ is too low, then more information flows into $\mathbf{z}$. Conversely, when $\alpha$ is large, a poor representation is learned, as the model places more of the emphasis on classification and not on reconstruction or regularisation of the latent space.

Another serious issue with the M2 model is its failure to properly disentangled the representations for the classes and for the non-class information). Typically, the capacity of a single binary variable is not large enough to represent the nuanced and detailed generative factors of a class. Consequently, this class information is entangled between the continuous and discrete latent space, thus preventing accurate interventions between classes.

Furthermore, as $\mathbf{y}$ is discrete, there exists a ground truth for the aggregate posterior $q(\mathbf{y})$, which during training the samples are taken from, however, at test time, the samples are drawn from $p(\mathbf{y})$. The variations between these two manifest as a train-test mismatch, thus affecting the efficacy of the generative model at test time.

Perhaps a more obvious flaw is in the design choice of using a discrete latent space. Representations are intended to encode the generative factors of data, with the generative model learning the causal relationship between the two. Typically, one would expect the representation to encapsulate the variation in a class as there are often a plethora of ways a class may look. Treating the latent space as discrete values thus only permits binary interventions on classes, which is suitable for multi-class problems like MNIST, however it decimates the potentially smooth transition between two classes e.g. transition from `smile` to not `smile`.

**Arbitrary Dependency Structure** The use of the M2 model described above is only feasible due to the dependency structure $q(\mathbf{z}, \mathbf{y}|\mathbf{x}) = q(\mathbf{z}|\mathbf{x}, \mathbf{y})q(\mathbf{y}|\mathbf{x})$. This factorisation can be restrictive, to combat this Siddharth et al. (2017) introduced an importance sampling method to permit arbitrary conditional dependency structure; enabling disentanglement between the class and additional information.

**Auxiliary Variables** To improve the bound on the variational bound, Maaløe et al. (2016) introduce an auxiliary variable into the M2 model Figure 2.2, with the following factorization $q(\mathbf{a}, \mathbf{z}|\mathbf{x}) = q(\mathbf{z}|\mathbf{a}, \mathbf{x})q(\mathbf{a}|\mathbf{x})$. Doing so increases the flexibility of the variational distribution, and leads to higher classification accuracies.

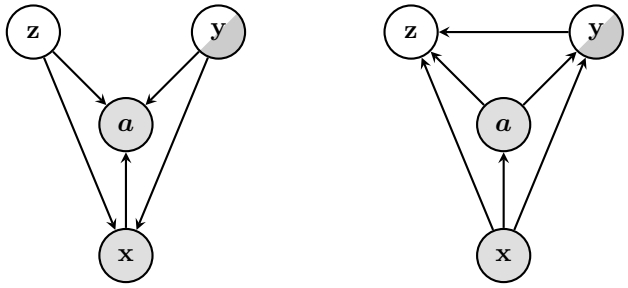



**Figure 2.2:** Left: Generative model, Right: Inference model for ADGM.

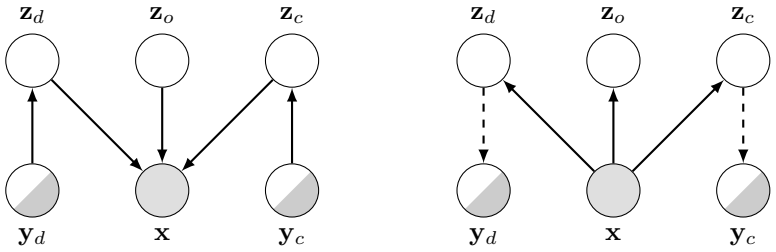



**Figure 2.3:** Left: Generative model for DIVA, Right: Inference model where dashed line indicates auxiliary classifier.

**Domain Invariant Variational Autoencoder** Tangentially to the M2 style models outlined above, Ilse et al. (2019) introduced the DIVA model with primary focus of obtaining a generalized classifier across different domains. This approach is somewhat different to the previous models, as it does not place the labels directly in the latent space. Instead, it chooses to introduce a classifier on a subset of the latent space reserved for *labeled* generative factors $\mathbf{z}_c$ and a classifier reserved for the generative factors of the *domain*. The remaining factors are captured in a subsection of the latent reserved for the domain $\mathbf{z}_d$ and the other factors $\mathbf{z}_o$. The motivation is that by encapsulating the domain in $\mathbf{z}_d$ and the other factors in $\mathbf{z}_o$, then only class information will be present in $\mathbf{z}_c$, thus permitting a classifier which can be applied to different domains.

Learning is performed by introducing three separate encoders $q(\mathbf{z}_d|\mathbf{x}), q(\mathbf{z}_o|\mathbf{x})$ and $q(\mathbf{z}_c|\mathbf{x})$, two conditional priors $p(\mathbf{z}_d|d)$ and $p(\mathbf{z}_c|\mathbf{y}_c)$, and the generative model

$p(\mathbf{x}|\mathbf{z}_d, \mathbf{z}_o, \mathbf{z}_c)$. For the supervised case, the objective is given as:

$$\begin{aligned}\mathcal{L}_s(\mathbf{x}, \mathbf{y}) = &\mathbb{E}_{q(\mathbf{z}_d|\mathbf{x})q(\mathbf{z}_o|\mathbf{x})q(\mathbf{z}_c|\mathbf{x})} \log p(\mathbf{x}|\mathbf{z}_d, \mathbf{z}_o, \mathbf{z}_c) \\ &- \beta KL(q(\mathbf{z}_d|\mathbf{x})||p(\mathbf{z}_d|d)) - \beta KL(q(\mathbf{z}_o|\mathbf{x})||p(\mathbf{z}_o) \\ &- \beta KL(q(\mathbf{z}_c|\mathbf{x})||p(\mathbf{z}_c|\mathbf{y}_c)),\end{aligned}$$

where $\beta$ is a tunable hyper-parameter. As with M2 style models, the above formulation does not encourage any learning or disentanglement associated with the labels $\mathbf{y}_d$ or $\mathbf{y}_c$. Consequently, the objective is amended through the addition of two ad-hoc classifiers

$$\mathcal{F}_{DIVA}(\mathbf{x}, \mathbf{y}) = \mathcal{L}_s(\mathbf{x}, \mathbf{y}) + \alpha_d \mathbb{E}_{q(\mathbf{z}_d|\mathbf{x})}[\log q(d|\mathbf{z}_d)] + \alpha_c \mathbb{E}_{q(\mathbf{z}_c|\mathbf{x})}[\log q(\mathbf{y}_c|\mathbf{z}_c)],$$

where $\alpha_d$ and $\alpha_c$ are hyper-parameters. As eluded to by the authors, the objective can be seen as domain invariant classifier which is regularised by a variational term $\mathcal{L}_s(\mathbf{x}, \mathbf{y})$. To deal with the unsupervised case, the labels are imputed following Louizos et al. (2015), allowing DIVA to be applied in the situation where the labels are missing.

$$\begin{aligned}\mathcal{L}_u(\mathbf{x}) = &\mathbb{E}_{q(\mathbf{z}_d|\mathbf{x})q(\mathbf{z}_o|\mathbf{x})q(\mathbf{z}_c|\mathbf{x})} \log p(\mathbf{x}|\mathbf{z}_d, \mathbf{z}_o, \mathbf{z}_c) \\ &- \beta KL(q(\mathbf{z}_d|\mathbf{x})||p(\mathbf{z}_d|d)) - \beta KL(q(\mathbf{z}_o|\mathbf{x})||p(\mathbf{z}_o) \\ &+ \beta \mathbb{E}_{q(\mathbf{z}_c|\mathbf{x})q(\mathbf{y}_c|\mathbf{z}_c)}[\log p(\mathbf{z}_c|\mathbf{y}_c) - \log q(\mathbf{z}_c|\mathbf{x})] \\ &+ \mathbb{E}_{q(\mathbf{z}_c|\mathbf{x})q(\mathbf{y}_c|\mathbf{z}_c)}[\log p(\mathbf{y}_c) - \log q(\mathbf{y}_c|\mathbf{z}_c)].\end{aligned}$$

Again this formulation requires the addition of a classifier for the domain. The final semi-supervised objective is given as

$$\mathcal{F}_{SS-DIVA} = \sum_{\{\mathbf{x},\mathbf{y}\}\in\mathcal{S}} \mathcal{F}_{DIVA}(\mathbf{x}, \mathbf{y}) + \sum_{\mathbf{x}\in\mathcal{U}} \mathcal{L}_u(\mathbf{x}) + \alpha_d \mathbb{E}_{q(\mathbf{z}_d|\mathbf{x})}[\log q(d|\mathbf{z}_d)].$$

It is worth drawing to attention the motivations of the aforementioned methods, where the primary goal is to obtain a classifier trained through semi-supervision. We argue that this goal completely disregards one of the principal features of VAEs; namely, learning meaningful representations which are amenable to interventions,

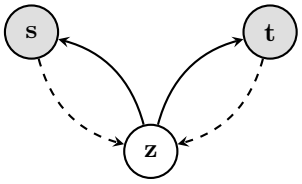



**Figure 2.4:** A general multimodal VAE, where the latent $\mathbf{z}$ encapsulates information from both $\mathbf{s}$ and $\mathbf{t}$.

but also the ability to obtain a bound on the likelihood. To some extent DIVA learns to disentangle representations, however only to a limited degree, for instance in a multi-label setting it is unable to construct isolated representations for individual labels.

## 2.2 Learning Multimodal VAEs through Mutual Supervision

This section can be viewed as an introduction to Chapter 4, and provides the relevant background information and notation needed.

### 2.2.1 Background: Multimodal VAEs

In the real world, data sources manifest themselves not as single modalities, but instead as multi-modal streams. For instance, video usually contains information through visual and audible mediums, but may also contain associated auxiliary information such as a title. Multi-modal VAEs aim to capture these joint modalities and form shared representations, resulting in the use of fewer samples and training and a model with an improved understanding of the world. The ability to capture these shared representations is essential for a number of downstream tasks, such as cross-generations but also in prediction tasks which require reasoning about the modalities in a joint manner such as the Hateful Memes challenge (Kiela et al., 2020).

The general graphical model for a multimodal VAE is given in Figure 2.4, which highlights how the latent space is a shared representation for both modalities. Performing variational inference in such a model requires us to approximate the posterior distribution $q(\mathbf{z}|\mathbf{s},\mathbf{t})$ whilst also evaluating the generative model $p(\mathbf{z},\mathbf{s},\mathbf{t}) =$

$p(\mathbf{z})p(\mathbf{s}|\mathbf{z})p(\mathbf{t}|\mathbf{z})$. Assuming the above is possible, the ELBO is given as

$$p(\mathbf{z}, \mathbf{s}, \mathbf{t}) \geq \mathbb{E}_{q(\mathbf{z}|\mathbf{s},\mathbf{t})} \log \frac{p(\mathbf{z}, \mathbf{s}, \mathbf{t})}{q(\mathbf{z}|\mathbf{s}, \mathbf{t})}. \tag{2.3}$$

One of the first attempts to model joint representations was the JMVAE (Suzuki et al., 2016), which used a joint encoder $q(\mathbf{z}|\mathbf{x}_1, \mathbf{x}_2)$, trained in conjunction with two uni-modal encoders $q(\mathbf{z}|\mathbf{x}_1)$ and $q(\mathbf{z}|\mathbf{x}_2)$ which aim to minimise the KL between themselves and $q(\mathbf{z}|\mathbf{x}_1, \mathbf{x}_2)$. Similarly Vedantam et al. (2017), also explicitly define multi-modal and uni-modal inference networks, but encourage convergence using a two step training regieme. The same approach was also taken by Tsai et al. (2019), but propose a strategy to infer the missing modalities using the modalities which are observed.

Taking a different approach, the Multi-modal Variational Autoencoder (MVAE) (Wu and Goodman, 2018a) modelled the joint posterior through a Product-of-Expert (PoE) of the marginals $q(\mathbf{z}_i|\mathbf{x}_i)$. Through a sub-sampled training regime, this approach handled the case when modalities without the need for additional inference networks. This was extended by Shi et al. (2019b) with the Multi-modal Mixture-of-expert Variational Autoencoder (MMVAE) which used a Mixture-of-Expert (MoE). More in depth explanations of these models and their limitations are given below.

**MVAE (Wu and Goodman, 2018b)**

To address approximating the posterior, the authors assume the modalities are conditionally independent $\mathbf{s}|\mathbf{z} \perp\!\!\!\perp \mathbf{t}|\mathbf{z}$, resulting in the following *true* posterior

$$p(\mathbf{z}|\mathbf{s}, \mathbf{t}) = \frac{p(\mathbf{s}, \mathbf{t}|\mathbf{z})p(\mathbf{z})}{p(\mathbf{s}, \mathbf{t})} = \frac{p(\mathbf{s}|\mathbf{z})p(\mathbf{t}|\mathbf{z})p(\mathbf{z})}{p(\mathbf{s}, \mathbf{t})} \propto \frac{p(\mathbf{z}|\mathbf{t})p(\mathbf{z}|\mathbf{s})}{p(\mathbf{z})} \tag{2.4}$$

which the authors refer to as the MVAE-Q (Quotient). To prevent numerical instabilities in the quotient, the authors further approximate the posterior as $p(\mathbf{z}|\mathbf{s}) \approx p(\mathbf{z})q(\mathbf{z}|\mathbf{s})$, leading to the following approximate posterior

$$p(\mathbf{z}|\mathbf{s}, \mathbf{t}) = p(\mathbf{z})q(\mathbf{z}|\mathbf{s})q(\mathbf{z}|\mathbf{t}). \tag{2.5}$$

One major issue with this approach is due to the fact that each expert retains the power to reduce the density of a region which the others provide a high density for. Furthermore, the authors note that training fails when all of the modalities are present during training and subsequently have to introduce a sub-sampling regime to address this deficiency.

**MMVAE (Shi et al., 2019a)**

Rather than taking the approach of the product of experts, Shi et al. (2019b) made use of a MoE, which approximates the posterior as $q(\mathbf{z}|\mathbf{s},\mathbf{t}) \approx 0.5 \cdot q(\mathbf{z}|\mathbf{s}) + 0.5 \cdot q(\mathbf{z}|\mathbf{t})$. This above formulation addresses the two major deficiencies of MVAE, and is also amenable to using models with tighter bounds such as the Importance Weighted Autoencoder (IWAE) (Burda et al., 2015). However, one of the major drawbacks of MMVAE, is its inability to handle the case where one of the modalities may be missing during training, hence placing a requirement that all of the data must be paired.

## 2.3 Sample-dependent Temperature Scaling for Improved Calibration

This section can be viewed as an introduction to Chapter 5, and provides the relevant background information and notation needed.

### 2.3.1 Background

For neural-classifiers to be used effectively in real world scenarios it is essential that they produce appropriate confidences with their predictions. This is especially important for safety critical systems, such as pedestrian detection in self driving cars; or in diagnosing patients with life threatening diseases. Clearly a failure in any one of these case could potentially lead to a serious loss of life and inflict significant damage on societies perception of machine learning. Fortunately, neural-classifiers can achieve remarkably high accuracy (Simonyan and Zisserman, 2014; He et al., 2016; Szegedy et al., 2016), making them appropriate for such systems. However, despite this high accuracy, they are also prone to suffer from *miscalibration*; where their confidence does not match their accuracy. Achieving calibrated predictions is

hence a desirable goal; in this work we show how VAEs can be employed to improve the calibration of neural networks.

### 2.3.2 Definitions

Here we consider the input and labels to be the random variables $\mathbf{x} \in \mathcal{X}$ and $y \in \{1, \ldots, K\}$ where $K$ is the number of possible classes. Let the output probabilities for each class be $\boldsymbol{p} = p(y|\mathbf{x}) = \sigma(f(\Phi(\mathbf{x})))$, with $\sigma(\mathbf{s}) = \frac{\exp(\mathbf{s})}{\sum_k \exp(s_k)}$ as the softmax function and $\theta = \{\mathbf{W}, \phi\}$ where $\mathbf{W}$ and $\phi$ are the parameters of the last layer and feature extractor of the neural-classifiers respectfully. The predicted label is then simply $\hat{y} = \arg\max_k p_k$. If the neural-classifier is calibrated, then given 100 samples which have an output probability $p$, we would expect the classifier to correctly classify $p \cdot 100$ of them. Formally, we can quantify how calibrated a classifier is by defining the Expected Calibration Error (ECE) as

$$\mathbb{E}_p[|\mathbb{P}(\hat{y} = y|p = \tilde{p}) - \tilde{p}|] \tag{2.6}$$

where $\tilde{p}$ is the resulting accuracy for the model. Clearly evaluating such a metric is computational infeasible, consequently an approximation needs to be formulated.

**Evaluating ECE** Due to the finite nature of the test-set we need to empirically estimate the accuracy for a given confidence. To do this, each prediction is grouped into $M$ bins of equal size. Let $\mathcal{B}_m$ be the set of indices for the samples who's confidence predictions falls in the range $I_m = (\frac{m-1}{M}, \frac{m}{M}]$. The accuracy for the samples in $\mathcal{B}_m$ is then

$$\text{acc}(\mathcal{B}_m) = \frac{1}{|\mathcal{B}_m|} \sum_{i \in \mathcal{B}_m} \mathbb{1}(\hat{y}_i = y_i). \tag{2.7}$$

In a similar fashion, the confidence for each bin is given as

$$\text{conf}(\mathcal{B}_m) = \frac{1}{|\mathcal{B}_m|} \sum_{i \in \mathcal{B}_m} p_i. \tag{2.8}$$

From this, the ECE can be empirically evaluated as follows

$$\text{ECE} = \sum_m \frac{|\mathcal{B}_m|}{N} |\,\text{acc}(\mathcal{B}_m) - \text{conf}(\mathcal{B}_m)| \tag{2.9}$$

with $N$ being the number of samples in the dataset. This provides a quantifiable metric for evaluating the calibration of neural-classifiers and is often the default metric to refer to when comparing methods.

**Maximum ECE** Another key metric which gives an indication of how well calibrated neural-classifiers are is the maximum calibration error

$$\max_{\tilde{p}\in[0,1]}[|\mathbb{P}(\hat{y}=y|p=\tilde{p})-\tilde{p}|] \tag{2.10}$$

which can be empirically estimated by taking the maximum calibration error given by each bin

$$\text{MCE}=\max_{m}|\,\text{acc}(\mathcal{B}_m)-\text{conf}(\mathcal{B}_m)|. \tag{2.11}$$

This metric is often used as an alternative to, or in conjunction with ECE.

**Reliability Plots** Reliability plots gives us a visual representation of how calibrated a model is. A perfectly calibrated model would yield accuracies from (2.7) which were equivalent in value to the bin which they were assigned to. Displaying these values on a bar chart then, will give an indication of how well calibrated the model is; with a calibrated model displaying values which show an equivalence between the $x$ and $y$ axis. We display an example reliability plot in Figure 2.5, which was obtained from the predicted probabilities for a ResNet-50 classifier using CIFAR10 as inputs. A perfectly calibrated model would be represented by a straight line indicating that confidence is equivalent to accuracy for all values—pink bars. Here however, we see that the accuracy for given confidence values is actually much lower than their confidence values, indicating that this model is overconfident and is indeed miscalibrated.

Given the definitions and a high level overview of the problem set up, we now highlight and outline the import pieces of background work which are relevant to Chapter 5. An additional literature review is included in Section 5.4.

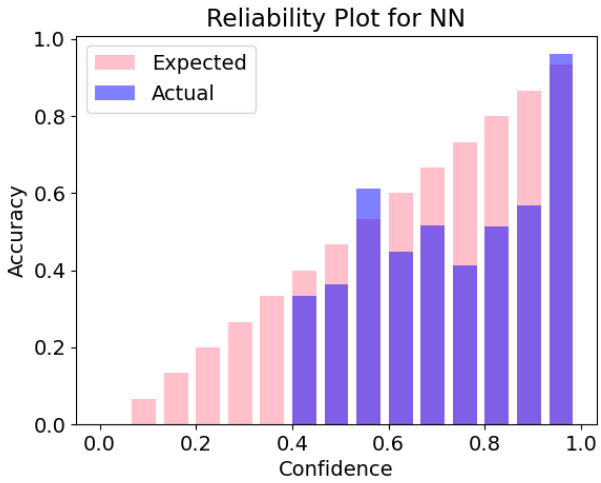



**Figure 2.5:** Reliability Plot for classification of CIFAR10 on ResNet-50.

### 2.3.3 Related Work

**Temperature Scaling** Temperature scaling is a post-hoc method which improves the calibration of neural-classifiers (Guo et al., 2017). The main premise is to artificially compensate for the miscalibrated confidences by introducing a scalar factor $T$ into the softmax function

$$\sigma(\mathbf{s}, T) = \frac{\exp(\frac{\mathbf{s}}{T})}{\sum_k \exp(\frac{s_k}{T})}. \tag{2.12}$$

Adjusting $T$ has a significant impact on the resulting softmax distribution, a low $T$ makes the distribution much "peakier", whereas a high $T$ flattens the distribution. An example of the softmax distribution with different values of $T$ is given in Figure 2.6.

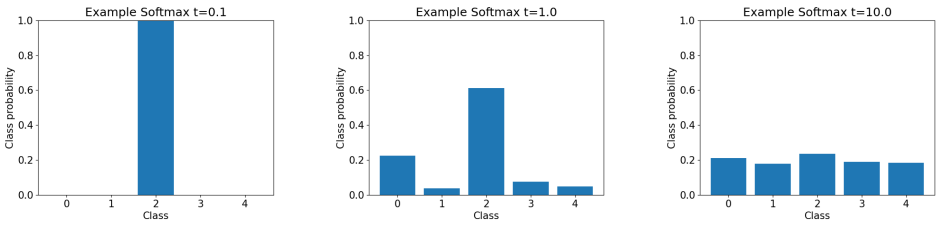



**Figure 2.6:** Example plots of Softmax distribution with different temperature values for fixed logits. Left to right: $T = 0.1, T = 1.0$ and $T = 10.0$.

Typically, to obtain $T$, practitioners are required to perform a cross-validation step using grid search, using the ECE or the Negative Log Likelihood (NLL) as an

optimisation metric. One of the main advantages of temperature scaling is that is can be applied post-hoc to almost any existing training algorithm, hence its popularity within the calibration literature.

**Deep-Ensembles** Deep-ensembles of neural-classifiers have been used to improve various metrics such as classification (Dietterich, 2000), but there advantages also extend to calibration (Lakshminarayanan et al., 2016). The deep-ensemble model is very simple to train and test. Given $A$ neural-classifiers, the model averages the predicted probabilities resulting in the following predictive distribution

$$p(y|\mathbf{x}) = \frac{1}{A}\sum_{a=1}^{A} p_{\theta_a}(y|\mathbf{x};\theta_a), \tag{2.13}$$

where the probabilities can thus be used to calculate the ECE or produce reliability plots.

**Mixup** Another popular, but simple, approach is *mixup* (Zhang et al., 2017). This work trains neural-networks using a convex combination of image and label pairs, which are given as

$$\tilde{\mathbf{x}} = \alpha\mathbf{x}_i + (1-\alpha)\mathbf{x}_j \tag{2.14}$$

$$\tilde{\mathbf{y}} = \alpha\mathbf{y}_i + (1-\alpha)\mathbf{y}_j \tag{2.15}$$

where $i$ and $j$ are random sample indices drawn from the dataset and $\mathbf{y}$ is the one-hot vectorised representation of $y$. The main premise behind mixup, is to incorporate the knowledge that linear interpolations of the input should lead to linear interpolations of the label.

## 2.4 Additional VAE Literature

Here we provide an additional literature review of related work to theVAE. Consequently, this section should be viewed as an additional literature review containing information on work which is relevant but tangential to the contributions of this thesis.

### 2.4.1 In Search of Tighter Lower Bounds

The VAE objective forms an ELBO on log marginal of the data; consequently, there a numerous pieces of work which aim to tighten the bound.

### Importance Weighted Encoders

The tightness of the lower bound can be increased by taking multiple samples from the approximate posterior distribution. This idea was introduced in IWAE (Burda et al., 2015), leading to the following objective

$$\log p(\mathbf{x}) \geq \mathbb{E}_{\mathbf{z}_1,\ldots,\mathbf{z}_k \sim q_\phi(\mathbf{z}|\mathbf{x})} \log \left[ \frac{1}{K} \sum_k \frac{p(\mathbf{x}, \mathbf{z}_k)}{q_\phi(\mathbf{z}_k|\mathbf{x})} \right], \tag{2.16}$$

for $K$ samples. Which leads to the importance weighted gradient update

$$\nabla_{\theta,\phi} \mathbb{E}_{\mathbf{z}_i,\ldots,\mathbf{z}_k \sim q_\phi(\mathbf{z}|\mathbf{x})} \left[ \frac{p(\mathbf{x}, \mathbf{z}_k)}{q_\phi(\mathbf{z}_k|\mathbf{x})} \right] = \frac{1}{K} \sum_k \tilde{w} \nabla_{\theta,\phi} \log \frac{p(\mathbf{x}, g(\epsilon_k, \mathbf{x}))}{q_\phi(g(\epsilon_k, \mathbf{x})|\mathbf{x})}. \tag{2.17}$$

where $\mathbf{z}_k = g(\epsilon_k, \mathbf{x})$ is the reparameterisation trick and $\tilde{w_k} = \frac{w_k}{\frac{1}{K}\sum_k w_k}$ are the importance weights with $w_k = \frac{p(\mathbf{z}_k, \mathbf{x})}{q_\phi(\mathbf{z}_k|\mathbf{x})}$; which decomposes as an importance weighted average of VAE gradients.

Due to the tighter lower bound, IWAE achieves a higher log-likelihood and increases the number of active dimensions used in the latent space. These factors make it a popular approach in hierachical VAEs Kingma et al. (2016); Sønderby et al. (2016) and multimodal VAEs Shi et al. (2019a).

### More Expressive priors

One of the primary reasons for choosing the prior to be an isotropic Gaussian with unit variance and zero mean, is the ease of use and the emergence of an analytical solution for the $KL$ term. However, the use of this prior restricts the approximate posterior's ability to model the true posterior. Ideally, we would like to choose a prior which matches the aggregate posterior $\mathbb{E}_{p_\mathcal{D}(\mathbf{x})} p_\theta(\mathbf{z} \mid \mathbf{x})$, tightening the lower bound.

There have been numerous approaches to improve the quality of the prior. The work by Tomczak and Welling (2018b) learns the prior by approximating the aggregate posterior using a mixture model. In Makhzani et al. (2015), they took an adversarial approach to fit the aggregate posterior to the prior. Separately, normalising flows (Kingma et al., 2016) provide a flexible distribution to learn the prior in Chen et al. (2016); Huang et al. (2017).

**Flow Posteriors**

Improving the expressiveness of the posterior improves its ability to match and model the true posterior, thus tightening the bound. A common way to improve the expressivity is to use flow priors (Rezende and Mohamed, 2015), that allow more flexibility to model the true posterior than the standard isotropic Gaussian. Since Rezende and Mohamed (2015), numerous works have used flow based posteriors to improve the quality of the model (Kingma et al., 2016; Tomczak and Welling, 2018b; Berg et al., 2018; Huang et al., 2018; Grathwohl et al., 2018; Durkan et al., 2019).

**Auxiliary Variables**

The bound on the likelihood can also be improved through the introduction of auxiliary variables, which introduces a dependence on an auxiliary variable $\alpha$ in the inference process

$$q(\mathbf{z}|\mathbf{x}) = \int q(\mathbf{z}|\mathbf{x}, \alpha) q(\alpha|\mathbf{x}) d\alpha. \tag{2.18}$$

The introduction of $\alpha$ is designed to capture the dependency between the latent variables of $\mathbf{z}$. This approach has been employed in ADGM (Maaløe et al., 2016) and Ranganath et al. (2016). The graphical model for Maaløe et al. (2016) can be seen in Figure 2.2.

### 2.4.2 Improving Generative Quality

Another core feature beyond obtaining a bound on the likelihood is the ability for a VAE to sample from the learned distribution. Unlike the Generative Adversarial Network (GAN) (Goodfellow et al., 2014), the VAE tends to have a low generation fidelity; in this section we highlight how with some modifications, the VAE can produce high fidelity samples.

**Hierarchical**

The expressivity of both the posterior and the prior can be improved significantly through the introduction of a hierarchy in the latent space. Here, the VAE has

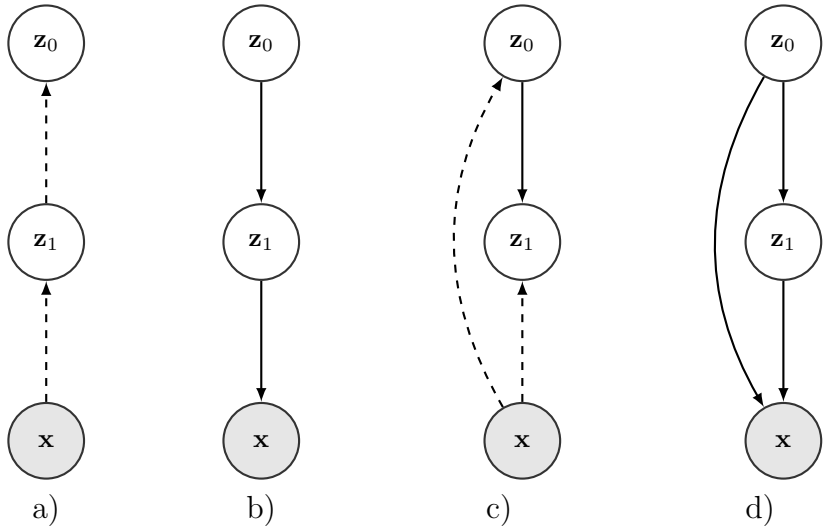



**Figure 2.7:** a) Inference model for hierarchical VAE; b) Generative model for VAE; c) Inference model for a hierarchical VAE *with bi-directional inference*; and d) Generative model for a hierarchical VAE *with bi-directional inference*.

multiple layers of latent variables as outlined in a) and b) in Figure 2.7. This leads to the following lower bound on the likelihood

$$\log p(\mathbf{x}) \geq \mathbb{E}_{q(\mathbf{z}_L|\mathbf{x}),\ldots,q(\mathbf{z}_0|\mathbf{z}_1)}\left[\log \frac{p(\mathbf{x}|\mathbf{z}_L)p(\mathbf{x}_0)\prod_{l=1}^{l=L} p(\mathbf{z}_l|\mathbf{z}_{l-1})}{q(\mathbf{z}_L|\mathbf{x})\prod_{l=0}^{l=L-1} q(\mathbf{z}_l|\mathbf{z}_{l+1})}\right]. \tag{2.19}$$

One issue with this approach is the that the inference model does not match the dependency structure of the generative model. In the inference model, the latent variables are sampled from the bottom up—$\mathbf{z}_1$ is sampled before $\mathbf{z}_0$. Whereas in the generative model the latent variables are sampled in the opposing order. This leads to potential issues during training and which were addressed in Sønderby et al. (2016), where they introduced a bi-directional approach, which samples latent variables in the correct order. The graphical model for this approach can be seen in c) and d) of Figure 2.7. The use of a hierarchy of latent variables and bi-directional inference has been shown to dramatically improve the fidelity of the generated images (Child, 2021).

**Vector Quantized**

Recently, the *Vector Quantized* VAE (VQVAE) (Van Den Oord et al., 2017), which differs from a typical VAE in that it has a discrete latent space rather than a continuous one, demonstrated how high fidelity samples can be obtained.

The motivation for using a discrete latent space is based on the fact that many application can be represented by discrete variables, e.g. language, and the authors claim that images can be also represented by language; additional applications include reasoning and planning. The VQVAE works by matching the predicted features from the encoder to a set of learn-able codes in a codebook. The code which is closest to the feature is then used as the input to the decoder. During the backward pass, a straight through estimator is used to alleviate the issue of selecting the closest code. This approach is able to provide high-quality generations whilst also compressing the input significantly.

**Expressive Decoders**

In a standard VAE the output dimensions factorise over the entire output space, meaning that each individual variable (often a pixel) is conditionally independent. In reality, this assumption over-simplifies the task and in fact you would often expect a dependency between pixels. One way to alleviate this is to use approaches like PixelCNN (Van den Oord et al., 2016) in the decoder (Chen et al., 2016; Gulrajani et al., 2016), providing much more flexibility in the resulting distribution. One issue with this approach is that the auto-regressive decoder tends to ignore the latent code (Bowman et al., 2015). The work of Chen et al. (2016) goes some way to limit the auto-regressive component in an effort to force information through the latent space.

# 3

# Rethinking Semi-Supervised Learning in VAEs

# Abstract

We present a principled approach to incorporating labels into the Variational Autoencoders (VAEs) objective, that captures the rich characteristic information associated with those labels. While prior work has typically conflated these by learning latent variables that directly correspond to label values, we argue this is contrary to the intended effect of supervision in VAEs—capturing rich label characteristics with the latents. For example, we may want to capture the characteristics of a face that make it look young, rather than just the age of the person. To this end, we develop the *Characteristic Capturing* VAE (CCVAE), a novel VAE model and concomitant variational objective which captures label characteristics explicitly in the latent space, eschewing direct correspondences between label values and latents. Through judicious structuring of mappings between such *characteristic latents* and labels, we show that the CCVAE can effectively learn meaningful representations of the characteristics of interest across a variety of supervision schemes. In particular, we show that the CCVAE allows for more effective and more general interventions to be performed, such as smooth traversals within the characteristics for a given label, diverse conditional generation, and transferring characteristics across datapoints.

## 3.1 Introduction

Learning the characteristic factors of perceptual observations has long been desired for effective machine intelligence (Brooks, 1991; Bengio et al., 2013a; Hinton and Salakhutdinov, 2006; Tenenbaum, 1998). In particular, the ability to learn *meaningful* factors—capturing human-understandable characteristics from data—has been of interest from the perspective of human-like learning (Tenenbaum and Freeman, 2000; Lake et al., 2015) and improving decision making and generalization across tasks (Bengio et al., 2013a; Tenenbaum and Freeman, 2000).

At its heart, learning meaningful representations of data allows one to not only make predictions, but critically also to *manipulate* factors of a datapoint. For example, we might want to manipulate the age of a person in an image. Such manipulations allow for the expression of causal effects between the meaning of factors and their corresponding realizations in the data. They can be categorized into conditional generation—the ability to construct whole exemplar data instances with characteristics dictated by constraining relevant factors—and intervention—the ability to manipulate just particular factors for a given data point, and subsequently affect only the associated characteristics.

A particularly flexible framework within which to explore the learning of meaningful representations are VAEs, a class of deep generative models where representations of data are captured in the underlying latent variables. A variety of methods have been proposed for inducing meaningful factors in this framework (Kim and Mnih, 2018; Mathieu et al., 2019; Mao et al., 2019; Kingma et al., 2014; Siddharth et al., 2017; Vedantam et al., 2018b), and it has been argued that the most effective generally exploit available labels to (partially) supervise the training process (Locatello et al., 2019). Such approaches aim to associate certain factors of the representation (or equivalently factors of the generative model) with the labels, such that the former encapsulate the latter—providing a mechanism for manipulation via targeted adjustments of relevant factors.

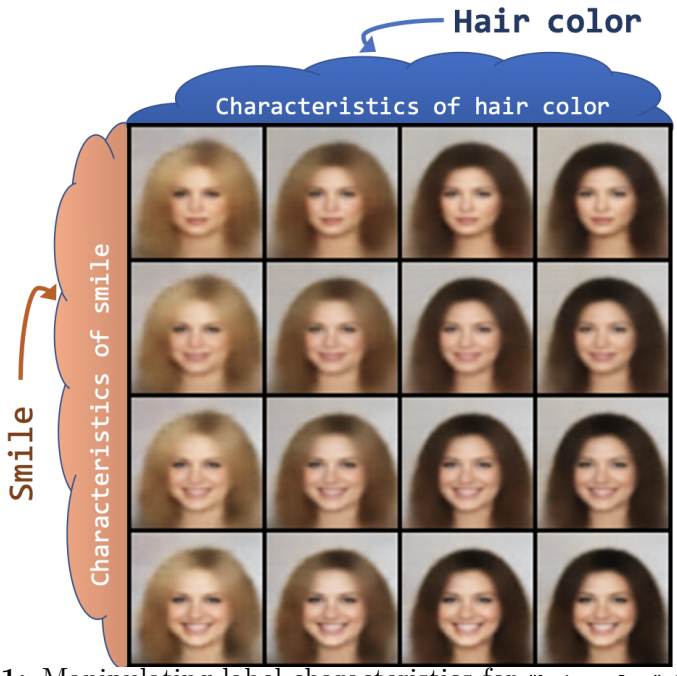



**Figure 3.1:** Manipulating label characteristics for `"hair color"` and `"smile"`.

Prior approaches have looked to achieve this by directly associating certain latent variables with labels (Kingma et al., 2014; Siddharth et al., 2017; Maaløe et al., 2016). Originally motivated by the desiderata of semi–supervised classification, each label is given a corresponding latent variable of the same type (e.g. categorical), whose value is fixed to that of the label when the label is observed and imputed by the encoder when it is not.

Though natural, we argue that this assumption is not just unnecessary but actively harmful from a representation-learning perspective, particularly in the context of performing manipulations. To allow manipulations, we want to learn latent factors that capture the characteristic information *associated* with a label, which is typically much richer than just the label value itself. For example, there are various visual characteristics of people's faces associated with the label "young," but simply knowing the label is insufficient to reconstruct these characteristics for any particular instance. Learning a meaningful representation that captures these characteristics, and *isolates* them from others, requires encoding more than just the label value itself, as illustrated in Figure 3.1.

The key idea of our work is to use labels to help capture and isolate this related characteristic information in a VAE's representation. We do this by exploiting the interplay between the labels and inputs to capture more information than the labels alone convey; information that will be lost (or at least entangled) if we directly encode the label itself. Specifically, we introduce the CCVAE framework, which employs a novel VAE formulation which captures label characteristics explicitly in the latent space. For each label, we introduce a set of *characteristic latents* that are induced into capturing the characteristic information associated with that label. By coupling this with a principled variational objective and carefully structuring the characteristic-latent and label variables , we show that CCVAEs successfully capture meaningful representations, enabling better performance on manipulation tasks, while matching previous approaches for prediction tasks. In particular, they permit certain manipulation tasks that cannot be performed with conventional approaches, such as manipulating characteristics *without* changing the labels themselves and producing *multiple* distinct samples consistent with the desired intervention. We summarize our contributions as follows:

i) showing how labels can be used to capture and isolate rich *characteristic* information;
ii) formulating CCVAEs, a novel model class and objective for supervised and semi-supervised learning in VAEs that allows this information to be captured effectively;
iii) demonstrating CCVAEs' ability to successfully learn meaningful representations in practice.

## 3.2 Background

VAEs (Kingma and Welling, 2013; Rezende et al., 2014) are a powerful and flexible class of model that combine the unsupervised representation-learning capabilities of deep autoencoders (Hinton and Zemel, 1994) with generative latent-variable models—a popular tool to capture factored low-dimensional representations of higher-dimensional observations. In contrast to deep autoencoders, generative

models capture representations of data not as distinct values corresponding to observations, but rather as *distributions* of values. A generative model defines a joint distribution over observed data $\mathbf{x}$ and latent variables $\mathbf{z}$ as $p_\theta(\mathbf{x}, \mathbf{z}) = p(\mathbf{z})p_\theta(\mathbf{x} \mid \mathbf{z})$. Given a model, learning representations of data can be viewed as performing *inference*—learning the *posterior* distribution $p_\theta(\mathbf{z} \mid \mathbf{x})$ that constructs the distribution of latent values for a given observation.

VAEs employ amortized Variational Inference (VI) (Wainwright and Jordan, 2008; Kingma and Welling, 2013) using the encoder and decoder of an autoencoder to transform this setup by i) taking the model likelihood $p_\theta(\mathbf{x} \mid \mathbf{z})$ to be parameterized by a neural network using the *decoder*, and ii) constructing an amortized variational approximation $q_\phi(\mathbf{z} \mid \mathbf{x})$ to the (intractable) posterior $p_\theta(\mathbf{z} \mid \mathbf{x})$ using the *encoder*. The variational approximation of the posterior enables effective estimation of the objective—maximizing the marginal likelihood—through importance sampling. The objective is obtained through invoking Jensen's inequality to derive the Evidence Lower Bound (ELBO) of the model which is given as:

$$\log p_\theta(\mathbf{x}) = \log \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\frac{p_\theta(\mathbf{z}, \mathbf{x})}{q_\phi(\mathbf{z} \mid \mathbf{x})}\right] \geq \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log \frac{p_\theta(\mathbf{z}, \mathbf{x})}{q_\phi(\mathbf{z} \mid \mathbf{x})}\right] \equiv \mathcal{L}(\mathbf{x}; \phi, \theta). \quad (3.1)$$

Given observations $\mathcal{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ taken to be realizations of random variables generated from an unknown distribution $p_\mathcal{D}(\mathbf{x})$, the overall objective is $\frac{1}{N}\sum_n \mathcal{L}(\mathbf{x}_n; \theta, \phi)$. Hierarchical VAEs (Sønderby et al., 2016) impose a hierarchy of latent variables improving the flexibility of the approximate posterior, however we do not consider these models in this work.

Semi-Spervised VAEs (SSVAEs) (Kingma et al., 2014; Maaløe et al., 2016; Siddharth et al., 2017) consider the setting where a subset of data $\mathcal{S} \subset \mathcal{D}$ is assumed to also have corresponding *labels* $\mathbf{y}$. Denoting the (unlabeled) data as $\mathcal{U} = \mathcal{D} \backslash \mathcal{S}$, the log-marginal likelihood is decomposed as

$$\log p\left(\mathcal{D}\right) = \sum\nolimits_{(\mathbf{x},\mathbf{y})\in\mathcal{S}} \log p_\theta(\mathbf{x}, \mathbf{y}) + \sum\nolimits_{\mathbf{x}\in\mathcal{U}} \log p_\theta(\mathbf{x}),$$

where the individual log-likelihoods are lower bounded by their ELBOs. Standard practice is then to treat $\mathbf{y}$ as a latent variable to marginalize over whenever the

label is not provided. More specifically, most approaches consider splitting the latent space in $\mathbf{z} = \{\mathbf{z_y}, \mathbf{z}_{\setminus \mathbf{y}}\}$ and then directly fix $\mathbf{z_y} = \mathbf{y}$ whenever the label is provided, such that each dimension of $\mathbf{z_y}$ explicitly represents a predicted value of a label, with this value known exactly only for the labeled datapoints. Much of the original motivation for this (Kingma et al., 2014) was based around performing semi–supervised classification of the labels, with the encoder being used to impute the values of $\mathbf{z_y}$ for the unlabeled datapoints. However, the framework is also regularly used as a basis for learning meaningful representations and performing manipulations, exploiting the presence of the decoder to generate new datapoints after intervening on the labels via changes to $\mathbf{z_y}$. Our focus lies on the latter, for which we show this standard formulation leads to serious pathologies. Our primary goal is not to improve the fidelity of generations, but instead to demonstrate how label information can be used to structure the latent space such that it encapsulates and disentangles the characteristics associated with the labels.

## 3.3 Rethinking Supervision

As we explained in the last section, the de facto assumption for most approaches to supervision in VAEs is that the labels correspond to a partially observed augmentation of the latent space, $\mathbf{z_y}$. However, this can cause a number of issues if we want the latent space to encapsulate not just the labels themselves, but also the characteristics *associated* with these labels. For example, encapsulating the youthful characteristics of a face, not just the fact that it is a "young" face. At an abstract level, such an approach fails to capture the relationship between the inputs and labels: it fails to isolate characteristic information associated with each label from the other information required to reconstruct data. More specifically, it fails to deal with the following issues.

Firstly, the information in a datapoint associated with a label is richer than stored by the (typically categorical) label itself. That is not to say such information is absent when we impose $\mathbf{z_y} = \mathbf{y}$, but here it is *entangled* with the other latent variables $\mathbf{z}_{\setminus \mathbf{y}}$, which simultaneously contain the associated information for *all* the

labels. Moreover, when $\mathbf{y}$ is categorical, it can be difficult to ensure that the VAE actually uses $\mathbf{z}_{\mathbf{y}}$, rather than just capturing information relevant to reconstruction in the higher-capacity, continuous, $\mathbf{z}_{\setminus\mathbf{y}}$. Overcoming this is challenging and generally requires additional heuristics and hyper-parameters.

Second, we may wish to manipulate characteristics without fully changing the categorical label itself. For example, making a CelebA image depict more or less 'smiling' without fully changing its `"smile"` label. Here we do not know how to manipulate the latents to achieve this desired effect: we can only do the binary operation of changing the relevant variable in $\mathbf{z}_{\mathbf{y}}$. Also, we often wish to keep a level of diversity when carrying out conditional generation and, in particular, interventions. For example, if we want to add a smile, there is no single correct answer for how the smile would look, but taking $\mathbf{z}_{\mathbf{y}} = \texttt{"smile"}$ only allows for a single point estimate for the change.

Finally, taking the labels to be explicit latent variables can cause a mismatch between the VAE prior $p(\mathbf{z})$ and the pushforward distribution of the data to the latent space $q(\mathbf{z}) = \mathbb{E}_{p_{\mathcal{D}}(\mathbf{x})}[q_\phi(\mathbf{z} \mid \mathbf{x})]$. During training, latents are effectively generated according to $q(\mathbf{z})$, but once learned, $p(\mathbf{z})$ is used to make generations; variations between the two effectively corresponds to a train-test mismatch. As there is a ground truth data distribution over the labels (which are typically not independent), taking the latents as the labels themselves implies that there will be a ground truth $q(\mathbf{z}_{\mathbf{y}})$. However, as this is not generally known a priori, we will inevitably end up with a mismatch.

**What do we want from supervision?** Given these issues, it is natural to ask whether having latents directly correspond to labels is actually necessary. To answer this, we need to think about exactly what it is we are hoping to achieve through the supervision itself. Along with uses of VAEs more generally, the three most prevalent tasks are: **a) Classification**, predicting the labels of inputs where these are not known a priori; **b) Conditional Generation**, generating new examples conditioned on those examples conforming to certain desired labels; and **c)**

**Intervention**, manipulating certain desired characteristics of a data point before reconstructing it.

Inspecting these tasks, we see that for classification we need a classifier from $\mathbf{z}$ to $\mathbf{y}$, for conditional generation we need a mechanism for sampling $\mathbf{z}$ given $\mathbf{y}$, and for inventions we need to know how to manipulate $\mathbf{z}$ to bring about a desired change. None of these require us to have the labels directly correspond to latent variables. Moreover, as we previously explained, this assumption can be actively harmful, such as restricting the range of interventions that can be performed.

## 3.4 Characteristic Capturing Variational Autoencoders

To correct the issues discussed in the last section, we suggest eschewing the treatment of labels as direct components of the latent space and instead employ them to condition latent variables which are designed to capture the characteristics. To this end, we similarly split the latent space into two components, $\mathbf{z} = \{\mathbf{z_c}, \mathbf{z}_{\setminus \mathbf{c}}\}$, but where $\mathbf{z_c}$, the *characteristic latent*, is now designed to capture the characteristics associated with labels, rather than directly encode the labels themselves. In this breakdown, $\mathbf{z}_{\setminus \mathbf{c}}$ is intended only to capture information not directly associated with any of the labels, unlike $\mathbf{z}_{\setminus \mathbf{y}}$ which was still tasked with capturing the characteristic information.

For the purposes of exposition and purely to demonstrate how one might apply this schema, we first consider a standard VAE, with a latent space $\mathbf{z} = \{\mathbf{z}_c, \mathbf{z}_s\}$. The latent representation of the VAE will implicitly contain characteristic information required to perform classification, however the structure of the latent space will be arranged to optimize for reconstruction and characteristic information may be *entangled* between $\mathbf{z_c}$ and $\mathbf{z}_{\setminus \mathbf{c}}$. If we were now to jointly learn a classifier—from $\mathbf{z_c}$ to $\mathbf{y}$—with the VAE, resulting in the following objective:

$$\mathcal{J} = \sum_{\mathbf{x} \in \mathcal{U}} \mathcal{L}_{VAE}(\mathbf{x}) + \sum_{(\mathbf{x},\mathbf{y}) \in \mathcal{S}} \left( \mathcal{L}_{VAE}(\mathbf{x}) + \alpha \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})} \left[ \log q_\varphi(\mathbf{y} \mid \mathbf{z}_c) \right] \right), \quad (3.2)$$

where $\alpha$ is a hyperparameter, there will be pressure on the encoder to place characteristic information in $\mathbf{z_c}$, which can be interpreted as a stochastic layer containing the information needed for classification *and* reconstruction[1]. The classifier thus acts as a tool allowing $\mathbf{y}$ to influence the structure of $\mathbf{z}$, it is this high level concept, i.e. using $\mathbf{y}$ to structure $\mathbf{z}$, that we utilize in this work.

However, in general, the characteristics of different labels will be *entangled* within $\mathbf{z}_c$. Though it will contain the required information, the latents will typically be uninterpretable, and it is unclear how we could perform conditional generation or interventions. To *disentangle* the characteristics of different labels, we further partition the latent space, such that the classification of particular labels $y^i$ only has access to particular latents $\mathbf{z}_c^i$ and thus $\log q_\varphi(\mathbf{y} \mid \mathbf{z}_c) = \sum_i \log q_{\varphi^i}(y^i \mid \mathbf{z}_c^i)$. This has the critical effect of forcing the characteristic information needed to classify $y^i$ to be stored only in the corresponding $\mathbf{z}_c^i$, providing a means to encapsulate such information for each label separately. We further see that it addresses many of the prior issues: there are no measure-theoretic issues as $\mathbf{z}_c^i$ is not discrete, diversity in interventions is achieved by sampling different $\mathbf{z}_c^i$ for a given label, $\mathbf{z}_c^i$ can be manipulated while remaining within class decision boundaries, and a mismatch between $p(\mathbf{z}_c)$ and $q(\mathbf{z}_c)$ does not manifest as there is no ground truth for $q(\mathbf{z}_c)$.

How to conditionally generate or intervene when training with (3.2) is not immediately obvious though. However, the classifier *implicitly* contains the requisite information to do this via *inference* in an implied Bayesian model. For example, conditional generation needs samples from $p(\mathbf{z}_c)$ that classify to the desired labels, e.g. through rejection sampling. See Appendix A.1 for further details.

### 3.4.1 The Characteristic Capturing VAE

One way to address the need for inference is to introduce a conditional generative model $p_\psi(\mathbf{z}_c \mid \mathbf{y})$, simultaneously learned alongside the classifier introduced in (3.2),

[1]Though, for convenience, we implicitly assume here, and through the rest of the paper, that the labels are categorical such that the mapping $\mathbf{z_c} \to \mathbf{y}$ is a classifier, we note that the ideas apply equally well if some labels are actually continuous, such that this mapping is now a probabilistic regression.

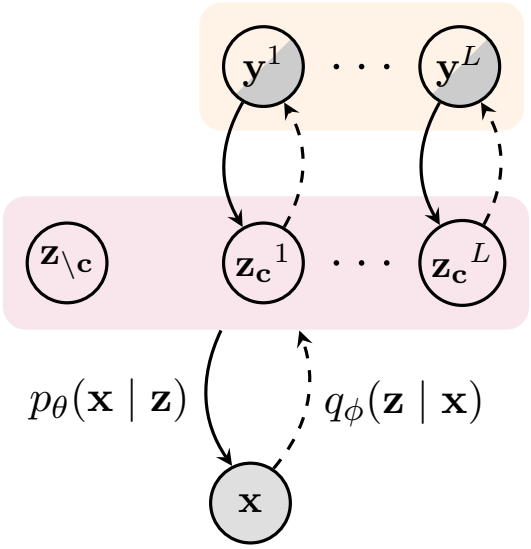



**Figure 3.2:** CCVAE graphical model.

along with a prior $p(\mathbf{y})$. This approach, which we term the CCVAE, allows the required sampling for conditional generations and interventions directly. Further, by persisting with the latent partitioning above, we can introduce a factorized set of generative models $p(\mathbf{z}_c \mid \mathbf{y}) = \prod_i p(\mathbf{z}_c^i \mid y^i)$, enabling easy generation and manipulation of $\mathbf{z}_c^i$ individually. CCVAE ensures that labels remain a part of the model for unlabeled datapoints, which transpires to be important for effective learning in practice.

To address the issue of learning, we perform variational inference, treating $\mathbf{y}$ as a partially observed auxiliary variable. The final graphical model is illustrated in Figure 3.2. The CCVAE can be seen as a way of combining top-down and bottom-up information to obtain a structured latent representation. However, it is important to highlight that CCVAE does not contain a hierarchy of latent variables. Unlike a hierarchical VAE, reconstruction is performed only from $\mathbf{z} \sim q_\phi(\mathbf{z} \mid \mathbf{x})$ *without* going through the "deeper" $\mathbf{y}$, as doing so would lead to a loss of information due to the bottleneck of $\mathbf{y}$. By enforcing each label variable to link to different characteristic-latent dimensions, we are able to isolate the generative factors corresponding to different label characteristics.

### 3.4.2 Model Objective

We now construct an objective function that encapsulates the model described above, by deriving a lower bound on the full model log-likelihood which factors over the supervised and unsupervised subsets as discussed in Section 3.2. The supervised objective can be defined as

$$\log p_{\theta,\psi}(\mathbf{x},\mathbf{y}) \geq \mathbb{E}_{q_{\varphi,\phi}(\mathbf{z}|\mathbf{x},\mathbf{y})}\left[\log \frac{p_\theta(\mathbf{x} \mid \mathbf{z})p_\psi(\mathbf{z} \mid \mathbf{y})p(\mathbf{y})}{q_{\varphi,\phi}(\mathbf{z} \mid \mathbf{x},\mathbf{y})}\right] \equiv \mathcal{L}_{CCVAE}(\mathbf{x},\mathbf{y}), \quad (3.3)$$

with $p_\psi(\mathbf{z} \mid \mathbf{y}) = p(\mathbf{z}_{\setminus \mathbf{c}})p_\psi(\mathbf{z_c} \mid \mathbf{y})$. Here, we avoid directly modeling $q_{\varphi,\phi}(\mathbf{z} \mid \mathbf{x},\mathbf{y})$; instead leveraging the conditional independence $\mathbf{x} \perp\!\!\!\perp \mathbf{y} \mid \mathbf{z}$, along with Bayes rule, to give

$$q_{\varphi,\phi}(\mathbf{z} \mid \mathbf{x},\mathbf{y}) = \frac{q_\varphi(\mathbf{y} \mid \mathbf{z}_c)q_\phi(\mathbf{z} \mid \mathbf{x})}{q_{\varphi,\phi}(\mathbf{y} \mid \mathbf{x})}, \quad \text{where} \quad q_{\varphi,\phi}(\mathbf{y} \mid \mathbf{x}) = \int q_\varphi(\mathbf{y} \mid \mathbf{z}_c)q_\phi(\mathbf{z} \mid \mathbf{x})\mathrm{d}\mathbf{z}.$$

Using this equivalence in (3.3) yields (see Appendix A.2.1 for a derivation and numerical details)

$$\mathcal{L}_{CCVAE}(\mathbf{x},\mathbf{y}) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\frac{q_\varphi(\mathbf{y} \mid \mathbf{z_c})}{q_{\varphi,\phi}(\mathbf{y} \mid \mathbf{x})} \log \frac{p_\theta(\mathbf{x} \mid \mathbf{z})p_\psi(\mathbf{z} \mid \mathbf{y})}{q_\varphi(\mathbf{y} \mid \mathbf{z}_c)q_\phi(\mathbf{z} \mid \mathbf{x})}\right] + \log q_{\varphi,\phi}(\mathbf{y} \mid \mathbf{x}) + \log p(\mathbf{y}). \quad (3.4)$$

Note that a classifier term $\log q_{\varphi,\phi}(\mathbf{y} \mid \mathbf{x})$ falls out naturally from the derivation, unlike previous models (e.g. Kingma et al. (2014); Siddharth et al. (2017)). Not placing the labels directly in the latent space is crucial for this feature. When defining latents to directly correspond to labels, observing both $\mathbf{x}$ and $\mathbf{y}$ *detaches* the mapping $q_{\varphi,\phi}(\mathbf{y} \mid \mathbf{x})$ between them, resulting in the parameters $(\varphi, \phi)$ not being learned—motivating addition of an explicit (weighted) classifier. Here, however, observing both $\mathbf{x}$ and $\mathbf{y}$ does not detach any mapping, since they are always connected via an unobserved random variable $\mathbf{z}_c$, and hence do not need additional terms. From an implementation perspective, this classifier strength can be increased, we experimented with this, but found that adjusting the strength had little effect on the overall classification accuracies. We consider this insensitivity to be a significant strength of this approach, as the model is able to apply enough pressure to the latent space to obtain high classification accuracies without having to hand tune

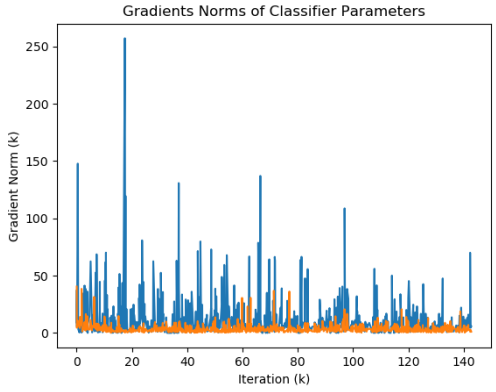



**Figure 3.3:** Gradient norms of classifier.

parameter values. We find that the gradient norm of the classifier parameters suffers from a high variance during training. This is not necessarily surprising, as using only a single sample to estimate $q_{\varphi,\phi}(\mathbf{y} \mid \mathbf{x})$, will produce biased gradient estimates with high variance. We find that not reparameterizing through $\mathbf{z_c}$ in $q_\varphi(\mathbf{y} \mid \mathbf{z}_c)$ reduces this affect and aides training changing the objective to

$$\mathcal{L}_{CCVAE}(\mathbf{x},\mathbf{y}) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\frac{q_\varphi(\mathbf{y} \mid \bar{\mathbf{z}_c})}{q_{\varphi,\phi}(\mathbf{y} \mid \mathbf{x})} \log \frac{p_\theta(\mathbf{x} \mid \mathbf{z}) p_\psi(\mathbf{z} \mid \mathbf{y})}{q_\varphi(\mathbf{y} \mid \bar{\mathbf{z}_c}) q_\phi(\mathbf{z} \mid \mathbf{x})}\right] + \log q_{\varphi,\phi}(\mathbf{y} \mid \mathbf{x}) + \log p(\mathbf{y})$$

where $\bar{\mathbf{z}_c}$ indicates that we do not reparameterize the sample. This significantly reduces the variance of the magnitude of the gradient norm $\nabla_\varphi$, allowing the classifier to learn appropriate weights and structure the latent space. This can be seen in Figure 3.3, where we plot the gradient norm of $\varphi$ for when we **do** reparameterize $\mathbf{z_c}$ (blue) and when we **do not** (orange). Clearly not reparameterizing leads to a lower variance in the gradient norm of the classifier, which aides learning. To a certain extent these gradients can be viewed as redundant, as there is already gradients to update the predictive distribution due to the $\log q_{\varphi,\phi}(\mathbf{y} \mid \mathbf{x})$ term anyway. It is worth noting that the true posterior $p(\mathbf{y}|\mathbf{z})$ can in fact be computed exactly using Bayes rule, $p(\mathbf{y}|\mathbf{z}) = \frac{p(\mathbf{z}|\mathbf{y})p(\mathbf{y})}{p(\mathbf{z})}$, but we chose against this approach as it does not enable scaling to a large number of classes, instead reverting to performing inference on the random variable $\mathbf{y}$.

For the datapoints without labels, we can again perform variational inference, treating the labels as random variables. Specifically, the unsupervised objective, $\mathcal{L}_{CCVAE}(\mathbf{x})$, derives as the standard (unsupervised) ELBO. However, it requires

marginalising over labels as $p(\mathbf{z}) = p(\mathbf{z}_c)p(\mathbf{z}_{\setminus c}) = p(\mathbf{z}_{\setminus c})\sum_{\mathbf{y}} p(\mathbf{z}_c|\mathbf{y})p(\mathbf{y})$. This can be computed exactly, but doing so can be prohibitively expensive if the number of possible label combinations is large. In such cases, we apply Jensen's inequality a second time to the expectation over $\mathbf{y}$ (see Appendix A.2.2) to produce a looser, but cheaper to calculate, ELBO given as

$$\mathcal{L}_{CCVAE}(\mathbf{x}) = E_{q_\phi(\mathbf{z}|\mathbf{x})q_\varphi(\mathbf{y}|\mathbf{z}_c)}\left[\log\left(\frac{p_\theta(\mathbf{x} \mid \mathbf{z})p_\psi(\mathbf{z} \mid \mathbf{y})p(\mathbf{y})}{q_\varphi(\mathbf{y} \mid \mathbf{z}_c)q_\phi(\mathbf{z} \mid \mathbf{x})}\right)\right]. \tag{3.5}$$

Combining (3.4) and (3.5), we get the following lower bound on the log probability of the data

$$\log p\left(\mathcal{D}\right) \geq \sum_{(\mathbf{x},\mathbf{y})\in\mathcal{S}} \mathcal{L}_{CCVAE}(\mathbf{x},\mathbf{y}) + \sum_{\mathbf{x}\in\mathcal{U}} \mathcal{L}_{CCVAE}(\mathbf{x}), \tag{3.6}$$

that unlike prior approaches faithfully captures the variational free energy of the model. As shown in Section 3.6, this enables a range of new capabilities and behaviors to encapsulate label characteristics.

## 3.5 Related Work

The seminal work of Kingma et al. (2014) was the first to consider supervision in the VAEs setting, introducing the M2 model for semi–supervised classification which placed labels directly in the latent space. The related approach of Maaløe et al. (2016) augments the encoding distribution with an additional, unobserved latent variable, enabling better semi-supervised classification accuracies. Siddharth et al. (2017) extended the above work to automatically derive the regularised objective for models with arbitrary (pre-defined) latent dependency structures. The approach of placing labels directly in the latent space was also adopted in Li et al. (2019). Regarding the disparity between continuous and discrete latent variables in the typical semi-supervised VAEs, Dupont (2018) provide an approach to enable effective *unsupervised* learning in this setting.

From a purely modeling perspective, there also exists prior work on VAEs involving hierarchies of latent variables, exploring richer higher-order inference and issues with redundancy among latent variables both in unsupervised (Ranganath et al., 2016;

Zhao et al., 2017) and semi-supervised (Maaløe et al., 2017, 2019) settings. In the unsupervised case, these hierarchical variables do not have a direct interpretation, but exist merely to improve the flexibility of the encoder. The semi-supervised approaches extend the basic M2 model to hierarchical VAEs by incorporating the labels as an additional latent (see Appendix F in Maaløe et al., 2019, for example), and hence must incorporate additional regularisers in the form of classifiers as in the case of M2. Moreover, by virtue of the typical dependencies assumed between labels and latents, it is difficult to disentangle the characteristics just associated with the label from the characteristics associated with the rest of the data—something we capture using our simpler split latents $(\mathbf{z_c}, \mathbf{z}_{\setminus \mathbf{c}})$.

From a more conceptual standpoint, Mueller et al. (2017) introduces interventions (called revisions) on VAEs for text data, regressing to auxiliary sentiment scores as a means of influencing the latent variables. This formulation is similar to (3.2) in spirit, although in practice they employ a range of additional factoring and regularizations particular to their domain of interest, in addition to training models in stages, involving different objective terms. Nonetheless, they share our desire to enforce meaningfulness in the latent representations through auxiliary supervision.

Another related approach involves explicitly treating labels as another data *modality* (Vedantam et al., 2018b; Suzuki et al., 2016; Wu and Goodman, 2018a; Shi et al., 2019b). This work is motivated by the need to learn latent representations that *jointly encode* data from different modalities. Looking back to (3.3), by refactoring $p(\mathbf{z} \mid \mathbf{y})p(\mathbf{y})$ as $p(\mathbf{y} \mid \mathbf{z})p(\mathbf{z})$, and taking $q(\mathbf{z} \mid \mathbf{x}, \mathbf{y}) = \mathcal{G}(q(\mathbf{z} \mid \mathbf{x}), q(\mathbf{z} \mid \mathbf{y}))$, one derives *multi-modal* VAEs, where $\mathcal{G}$ can construct a product (Wu and Goodman, 2018a) or mixture (Shi et al., 2019b) of experts. Of these, the MVAE (Wu and Goodman, 2018a) is more closely related to our setup here, as it explicitly targets cases where alternate data modalities are labels. However, they differ in that the latent representations are not structured explicitly to map to distinct classifiers, and do not explore the question of explicitly capturing the label characteristics. The JLVM model of Adel et al. (2018) is similar to the MVAE, but is motivated from

an interpretability perspective—with labels providing 'side-channel' information to constrain latents. They adopt a flexible normalising-flow posterior from data $\mathbf{x}$, along with a multi-component objective that is additionally regularised with the information bottleneck between data $\mathbf{x}$, latent $\mathbf{z}$, and label $\mathbf{y}$.

DIVA (Ilse et al., 2019) introduces a similar graphical model to ours, but is motivated to learn a generalized classifier for different domains. The objective is formed of a classifier which is regularized by a variational term, requiring additional hyper-parameters and preventing the ability to disentangle the representations. In Appendix A.3.4 we propose some modifications to DIVA that allow it to be applied in our problem domain. Obtaining the *true* joint distribution can also be obtained when following a similar graphical model (Khemakhem et al., 2020), but is restricted to using simple transformations.

In terms of interoperability, the work of Ainsworth et al. (2018) is closely related to ours, but they focus primarily on group data and not introducing labels. Here the authors employ sparsity in the multiple linear transforms for each decoder (one for each group) to encourage certain latent dimensions to encapsulate certain factors in the sample, thus introducing interoperability into the model. Tangentially to VAEs, similar objectives of structuring the latent space using GANs also exist Xiao et al. (2017, 2018), although they focus purely on interventions and cannot perform conditional generations, classification, or estimate likelihoods.

## 3.6 Experiments

Following our reasoning in Section 3.3 we now showcase the efficacy of CCVAE for the three broad aims of *(a) intervention*, *(b) conditional generation* and *(c) classification* for a variety of supervision rates, denoted by $f$. Specifically, we demonstrate that CCVAE is able to: encapsulate characteristics for each label in an isolated manner; introduce diversity in the conditional generations; permit a finer control on interventions; and match traditional metrics of baseline models. Furthermore, we demonstrate that no existing method is able to perform all of the

above,[2] highlighting its sophistication over existing methods. We compare against: M2 (Kingma et al., 2014); MVAE (Wu and Goodman, 2018a); and our modified version of DIVA (Ilse et al., 2019). See Appendix A.3.4 for details.

To demonstrate the capture of label characteristics, we consider the multi-label setting and utilise the Chexpert (Irvin et al., 2019) and CelebA (Liu et al., 2015) datasets.[3] For CelebA, we restrict ourselves to the 18 labels which are distinguishable in reconstructions; see Appendix A.3.1 for details. We use the architectures from Higgins et al. (2016) for the encoder and decoder. The label-predictive distribution $q_\varphi(\mathbf{y} \mid \mathbf{z}_c)$ is defined as $\text{Ber}(\mathbf{y} \mid \boldsymbol{\pi}_\varphi(\mathbf{z_c}))$ with a diagonal transformation $\boldsymbol{\pi}_\varphi(\cdot)$ enforcing $q_\varphi(\mathbf{y} \mid \mathbf{z}_c) = \prod_i q_{\varphi^i}(y_i \mid \mathbf{z_c}^i)$. The conditional prior $p_\psi(\mathbf{z}_c \mid \mathbf{y})$ is then defined as $\mathcal{N}(\mathbf{z_c}|\boldsymbol{\mu}_\psi(\mathbf{y}), \text{diag}(\boldsymbol{\sigma}^2_\psi(\mathbf{y})))$ with appropriate factorization, and has its parameters also derived through MLPs. See Appendix A.3.3 for further details.

### 3.6.1 Interventions

If CCVAE encapsulates characteristics of a label in a single latent (or small set of latents), then it should be able to smoothly manipulate these characteristics without severely affecting others. This allows for finer control during interventions, which is not possible when the latent variables directly correspond to labels. To demonstrate this, we traverse two dimensions of the latent space and display the reconstructions in Figure 3.4. These examples indicate that CCVAE is indeed able to smoothly manipulate characteristics. For example, in **b)** we are able to induce varying skin tones rather than have this be a binary intervention on `pale skin`, unlike DIVA in **a)**. In **c)**, the $\mathbf{z}_c^i$ associated with the `necktie` label has also managed to encapsulate information about whether someone is wearing a shirt or is bare-necked. No such traversals are possible for M2 and it is not clear how one would do them for MVAE; additional results, including traversals for DIVA, are given in Appendix A.4.2.

[2] DIVA can perform the same tasks as CCVAE but only with the modifications we ourselves suggest and still not to a comparable quality.

[3] CCVAE is well-suited to multi-label problems, but also works on multi-class problems. See Appendix A.4.6 for results and analyses on MNIST and FashionMNIST.

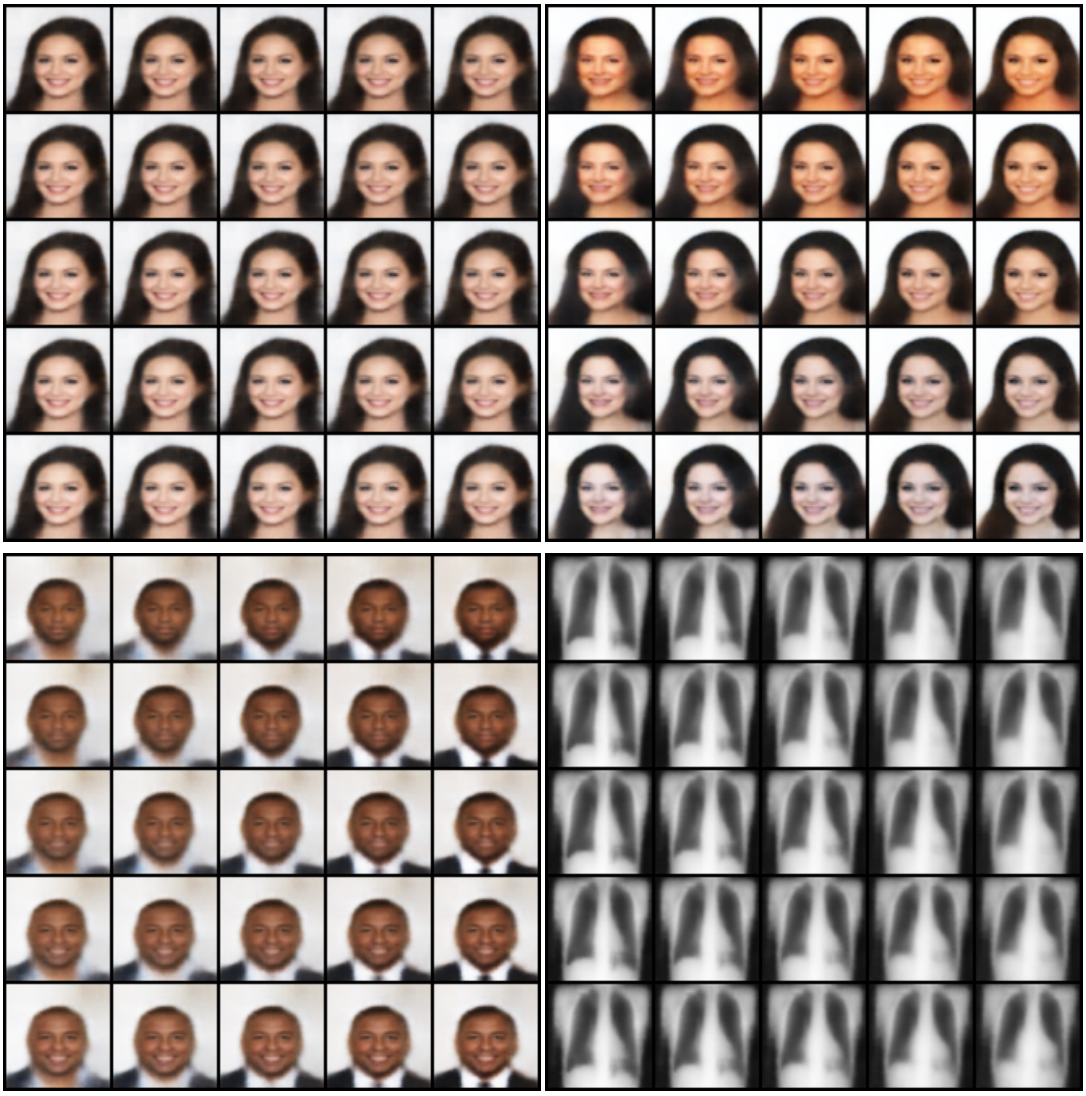


**Figure 3.4:** Continuous interventions through traversal of $\mathbf{z}_c$. From top left clockwise: a) DIVA `pale skin` and `young`; b) CCVAE `pale skin` and `young`; c) CCVAE `Pleural Effusion` and `Cardiomegaly`. d) CCVAE `smiling` and `necktie`;

### 3.6.2 Diversity of Generations

Label characteristics naturally encapsulate diversity (e.g. there are many ways to smile) which should be present in the learned representations. By virtue of the structured mappings between labels and characteristic latents, and since $\mathbf{z_c}$ is parameterized by continuous distributions, CCVAE is able to capture diversity in representations, allowing exploration for an attribute (e.g. smile) while preserving other characteristics. This is not possible with labels directly defined as latents, as only discrete choices can be made—diversity can only be introduced here by sampling

**Figure 3.5:** Diverse conditional generations for CCVAE, $\mathbf{y}$ is held constant along each row and each column represents a different sample for $\mathbf{z_c} \sim p(\mathbf{z_c}|\mathbf{y})$. $\mathbf{z}_{\setminus \mathbf{c}}$ is held constant over the entire figure.

**Figure 3.6:** Variance in reconstructions when intervening on a single label. [Top two] CelebA, from left to right: reconstruction, `bangs`, `eyeglasses`, `pale skin`, `smiling`, `necktie`.. [Bottom] Chexpert: reconstruction, `cardiomegaly`, `edema`, `consolidation`, `atelectasis`, `pleural effusion`.

from the unlabeled latent space—which necessarily affects all other characteristics. To demonstrate this, we reconstruct multiple times with $\mathbf{z} = \{\mathbf{z_c} \sim p_\psi(\mathbf{z}_c \mid \mathbf{y}), \mathbf{z}_{\setminus \mathbf{c}}\}$ for a fixed $\mathbf{z}_{\setminus \mathbf{c}}$. We provide qualitative results in Figure 3.5.

If several samples are taken from $\mathbf{z_c} \sim p_\psi(\mathbf{z}_c \mid \mathbf{y})$ when intervening on only a single characteristic, the resulting variations in pixel values should be focused around the locations relevant to that characteristic, e.g. pixel variations should be focused around the neck when intervening on `necktie`. To demonstrate this, we perform single interventions on each class, and take multiple samples of $\mathbf{z_c} \sim p_\psi(\mathbf{z}_c \mid \mathbf{y})$. We then display the variance of each pixel in the reconstruction in green in Figure 3.6, where it can be seen that generally there is only variance in the spatial locations expected. Interestingly, for the class `smile` (2nd from right), there is variance in the jaw line, suggesting that the model is able capture more subtle components of variation that just the mouth.

### 3.6.3 Classification

To demonstrate that reparameterizing the labels in the latent space does not hinder classification accuracy, we inspect the predictive ability of CCVAE across a range of supervision rates, given in Table 3.1. It can be observed that CCVAE generally obtains prediction accuracies slightly superior to other models. We emphasize here that CCVAE's primary purpose is not to achieve better classification accuracies; we are simply checking that it does not harm them, which it most clearly does not.

**Table 3.1:** Classification accuracies.

| | CelebA | | | | Chexpert | | | |
|---|---|---|---|---|---|---|---|---|
| Model | $f = 0.004$ | $f = 0.06$ | $f = 0.2$ | $f = 1.0$ | $f = 0.004$ | $f = 0.06$ | $f = 0.2$ | $f = 1.0$ |
| CCVAE | **0.832** | **0.862** | **0.878** | **0.900** | **0.809** | **0.792** | **0.794** | **0.826** |
| M2 | 0.794 | 0.862 | 0.877 | 0.893 | 0.799 | 0.779 | 0.777 | 0.774 |
| DIVA | 0.807 | 0.860 | 0.867 | 0.877 | 0.747 | 0.786 | 0.781 | 0.775 |
| MVAE | 0.793 | 0.828 | 0.847 | 0.864 | 0.759 | 0.787 | 0.767 | 0.715 |

### 3.6.4 Disentanglement of labeled and unlabeled latents

If a model can correctly disentangle the label characteristics from other generative factors, then manipulating $\mathbf{z}_{\setminus \mathbf{c}}$ should not change the label characteristics of the reconstruction. To demonstrate this, we perform "characteristic swaps," where we first obtain $\mathbf{z} = \{\mathbf{z}_\mathbf{c}, \mathbf{z}_{\setminus \mathbf{c}}\}$ for a given image, then swap in the characteristics $\mathbf{z}_\mathbf{c}$ to another image before reconstructing. This should apply the exact characteristics, not just the label, to the scene/background of the other image (cf. Figure 3.7).


**Figure 3.7:** Characteristic swap, where the characteristics of the first image (`blond hair, smiling, heavy makeup, female, no necktie, no glasses` etc.) are transferred to the unlabelled characteristics of the second (`red background` etc.).

Comparing CCVAE to our baselines in Figure 3.8, we see that CCVAE is able to transfer the exact characteristics to a greater extent than other models. Particular attention is drawn to the preservation of labeled characteristics in each row, where CCVAE is able to preserve characteristics, like the precise skin tone and hair color of the pictures on the left. We see that M2 is only able to preserve the label and not the exact characteristic, while MVAE performs very poorly, effectively ignoring the attributes entirely. Our modified DIVA variant performs reasonably well, but less reliably and at the cost of reconstruction fidelity compared to CCVAE.

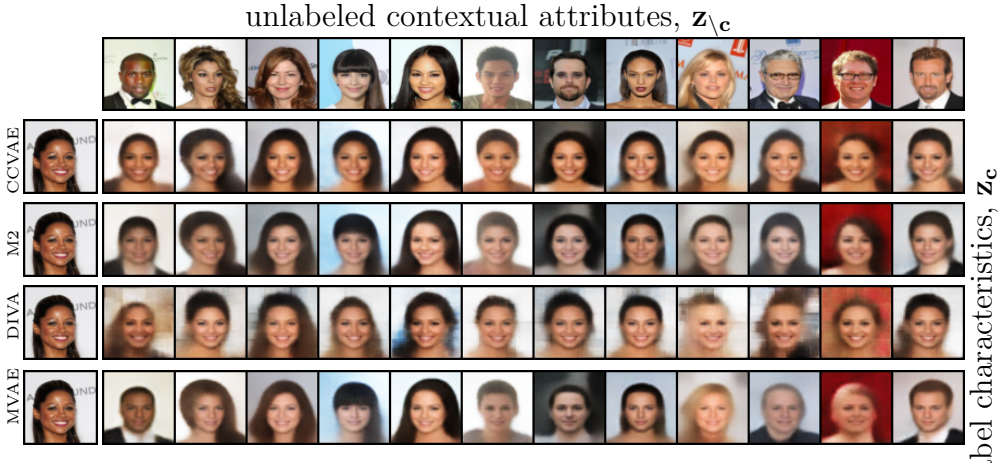



**Figure 3.8:** Characteristic swaps. Characteristics (`smiling, brown hair, skin tone`, etc) of the left image should be preserved along the row while background information should be preserved along the column.

An ideal characteristic swap should not change the probability assigned by a pre-trained classifier between the original image and a swapped one. We employ this as a quantitative measure, reporting the average difference in log probabilities for multiple swaps in Table 3.2. CCVAE is able to preserve the characteristics to a greater extent than other models. DIVA's performance is largely due to its heavier weighting on the classifier, which adversely affects reconstructions, as seen earlier.

## 3.7 Discussion

We have presented a novel mechanism for faithfully capturing label characteristics in VAEs, the *Characteristic Capturing* VAE (CCVAE), which captures label

**Table 3.2:** Difference in log-probabilities of pre-trained classifier from denotation swaps, lower is better.

| | CelebA | | | | Chexpert | | | |
|---|---|---|---|---|---|---|---|---|
| Model | $f = 0.004$ | $f = 0.06$ | $f = 0.2$ | $f = 1.0$ | $f = 0.004$ | $f = 0.06$ | $f = 0.2$ | $f = 1.0$ |
| CCVAE | **1.177** | **0.890** | **0.790** | **0.758** | **1.142** | **1.221** | **1.078** | **1.084** |
| M2 | 2.118 | 1.194 | 1.179 | 1.143 | 1.624 | 1.43 | 1.41 | 1.415 |
| DIVA | 1.489 | 0.976 | 0.996 | 0.941 | 1.36 | 1.25 | 1.199 | 1.259 |
| MVAE | 2.114 | 2.113 | 2.088 | 2.121 | 1.618 | 1.624 | 1.618 | 1.601 |

characteristics explicitly in the latent space while eschewing direct correspondences between label values and latents. This has allowed us to encapsulate and disentangle the *characteristics* associated with labels, rather than just the label values. We are able to do so without affecting the ability to perform the tasks one typically does in the (semi-)supervised setting—namely classification, conditional generation, and intervention. In particular, we have shown that, not only does this lead to more effective conventional label-switch interventions, it also allows for more fine-grained interventions to be performed, such as producing diverse sets of samples consistent with an intervened label value, or performing characteristic swaps between datapoints that retain relevant features.

# 4

# Multi-Modal learning through Mutual Supervision

# Abstract

Multimodal Variational Autoencoders (VAEs) seek to model the joint distribution over heterogeneous data (e.g. vision, language), whilst also capturing a shared representation across such modalities. Prior work has typically combined information from the modalities by reconciling idiosyncratic representations directly in the recognition model through explicit products, mixtures, or other such factorisations. Here we introduce a novel alternative, the **M**utually sup**E**rvised **M**ultimodal VA**E** (MEME), that avoids such explicit combinations by repurposing semi-supervised VAEs to combine information between modalities *implicitly* through mutual supervision. This formulation naturally allows learning from partially-observed data where some modalities can be entirely missing—something that most existing approaches either cannot handle, or do so to a limited extent. We demonstrate that MEME outperforms baselines on standard metrics across *both* partial and complete observation schemes on the MNIST-SVHN (image–image) and CUB (image–text) datasets. We also contrast the quality of the representations learnt by mutual supervision against standard approaches and observe interesting trends in its ability to capture relatedness between data.

## 4.1 Introduction

Modelling the generative process underlying heterogenous data, particularly data spanning multiple perceptual modalities such as vision or language, can be enormously challenging. Consider for example, the case where data spans across photographs and sketches of objects. Here, a data point, comprising of an instance from each modality, is constrained by the fact that the instances are related and must depict the *same* underlying abstract concept. An effective model not only needs to faithfully generate data in each of the different modalities, it also needs to do so in a manner that preserves the underlying relation between modalities. Learning a model over multimodal data thus relies on the ability to bring together information from idiosyncratic sources in such a way as to overlap on aspects they relate on, while remaining disjoint otherwise.

VAEs (Kingma and Welling, 2014) are a class of deep generative models that are particularly well-suited for multimodal data as they employ the use of *encoders*—learnable mappings from high-dimensional data to lower-dimensional representations—that provide the means to combine information across modalities. They can also be adapted to work in situations where instances are missing for some modalities; a common problem where there are inherent difficulties in obtaining and curating heterogenous data. Much of the work in multimodal VAEs involves exploring different ways to model and formalise the combination of information with a view to improving the quality of the learnt models (see Section 4.2).

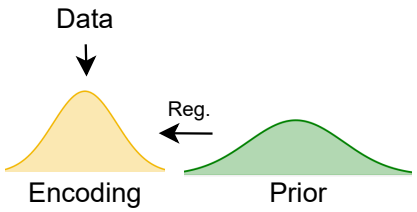
Prior approaches typically combine information through *explicit* specification as products (Wu and Goodman, 2018b), mixtures (Shi et al., 2019a), combinations of such (Sutter et al., 2021), or through additional regularisers on the representations (Suzuki et al., 2016; Sutter et al., 2020). Here, we explore an alternative approach that leverages advances in semi-supervised VAEs (Siddharth et al., 2017; Joy et al., 2021) to repurpose existing regularisation in the VAE framework as an *implicit* means by which information is combined across modalities (see Figure 4.1).

We develop a novel formulation for multimodal VAEs that views the combination of information through a semi-supervised lens, as *mutual supervision* between modalities. We term this approach MEME. Our approach not only avoids the need for additional explicit combinations, but it also naturally extends to *learning* in the partially-observed setting—something that most prior approaches cannot handle. We evaluate MEME on standard metrics for multimodal VAEs across both partial *and* complete data settings, on the typical multimodal data domains, MNIST-SVHN (image-image) and the less common but notably more complex CUB (image-text), and show that it outperforms prior work on both. We additionally investigate the capability of MEMEs ability to capture the 'relatedness', a notion of semantic similarity, between modalities in the latent representation; in this setting we also find that MEME outperforms prior work considerably.

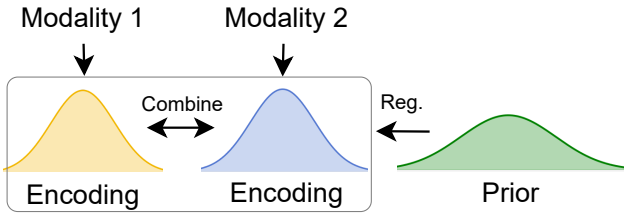
## 4.2 Related work

Prior approaches to multimodal VAEs can be broadly categorised in terms of the explicit combination of representations (distributions), namely concatenation and factorization.

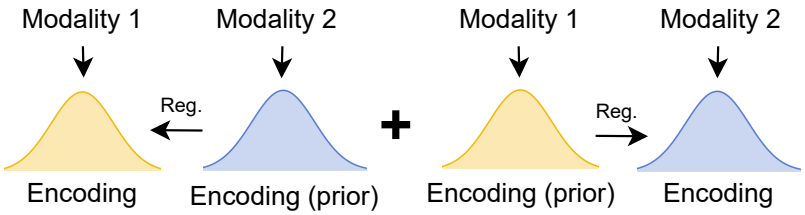
**Concatenation**: Models in this category learn joint representation by either concatenating the inputs themselves or their modality-specific representations. Examples for the former includes early work in multimodal VAEs such as the JMVAE (Suzuki et al., 2016), triple ELBO (Vedantam et al., 2018a) and MFM (Tsai et al., 2019), which define a joint encoder over concatenated multimodal data. Such approaches usually require the training of auxiliary modality-specific components to handle the partially-observed setting, with missing modalities, at test time. They also cannot learn from partially-observed data. In very recent work, Gong et al. (2021) propose VSAE where the latent representation is constructed as the concatenation of modality-specific encoders. Inspired by VAEs that deal with imputing pixels in images such as VAEAC (Ivanov et al., 2019), Partial VAE (Ma et al., 2018), MIWAE (Mattei and Frellsen, 2019), HI-VAE (Nazábal et al., 2020) and pattern-set mixture model (Ghalebikesabi et al., 2021), VSAE can learn in




(a) VAE




(b) Typical Multimodal VAE




(c) MEME (ours)

**Figure 4.1:** Constraints on the representations. ***(a) VAE***: A prior regularises the data encoding distribution through KL. ***(b) Typical multimodal VAE***: Encodings for different modalities are first explicitly combined, with the result regularised by a prior through KL. ***(c) MEME (ours)***: Leverage semi-supervised VAEs to cast one modality as a conditional prior, implicitly supervising/regularising the other through the VAE's KL. Mirroring the arrangement to account for KL asymmetry enables multimodal VAEs through mutual supervision.

the partially-observed setting by incorporating a modality mask. This, however, introduces additional components such as a collective proposal network and a mask generative network, while ignoring the need for the joint distribution over data to capture some notion of the relatedness between modalities.

**Factorization**: In order to handle missing data at test time without auxiliary components, recent work propose to factorize the posterior over all modalities as the product (Wu and Goodman, 2018b) or mixture (Shi et al., 2019a) of modality-specific posteriors (experts). Following this, Sutter et al. (2021) proposes to combine the two approaches (MoPoE-VAE) to improve learning in settings where the number

of modalities exceeds two. In contrast to these methods, mmJSD (Sutter et al., 2020) combines information not in the posterior, but in a "dynamic prior", defined as a function (either mixture or product) over the modality-specific posteriors as well as pre-defined prior.

Table 4.1 provides a high-level summary of prior work. Note that all the prior approaches have some explicit form of joint representation or distribution, where some of them induces the need for auxiliary components to deal with missing data at test time, while others are established without significant theoretical benefits. By building upon a semi-supervised framework, our method MEME circumvents this issue to learn representations through mutual supervision between modalities, and is able to deal with missing data at train or test time naturally without additional components.

**Table 4.1:** We examine four characteristics: The ability to handle partial observation at test and train time, the form of the joint distribution or representation in the bi-modal case ($\mathbf{s}$, $\mathbf{t}$ are modalities), and additional components. (✓) indicates a theoretical capability that is not verified empirically.

| | Partial Test | Partial Train | Joint repr./dist. | Additional |
|---|---|---|---|---|
| JMVAE | ✓ | ✗ | $q_\Phi(\mathbf{z}\|\mathbf{s},\mathbf{t})$ | $q_{\phi_s}(\mathbf{z}\|\mathbf{s}), q_{\phi_t}(\mathbf{z}\|\mathbf{t})$ |
| tELBO | ✓ | ✗ | $q_\Phi(\mathbf{z}\|\mathbf{s},\mathbf{t})$ | $q_{\phi_s}(\mathbf{z}\|\mathbf{s}), q_{\phi_t}(\mathbf{z}\|\mathbf{t})$ |
| MFM | ✓ | ✗ | $q_\Phi(\mathbf{z}\|\mathbf{s},\mathbf{t})$ | $q_{\phi_s}(\mathbf{z}\|\mathbf{s}), q_{\phi_t}(\mathbf{z}\|\mathbf{t})$ |
| VSVAE | ✓ | ✓ | $\texttt{concat}(z_s, z_t)$ | mask generative network |
| MVAE | ✓ | (✓) | $q_{\phi_s}(\mathbf{z}\|\mathbf{s})q_{\phi_t}(\mathbf{z}\|\mathbf{t})p(\mathbf{z})$ | sub-sampling |
| MMVAE | ✓ | ✗ | $q_{\phi_s}(\mathbf{z}\|\mathbf{s}) + q_{\phi_t}(\mathbf{z}\|\mathbf{t})$ | - |
| MoPoE | ✓ | (✓) | $q_{\phi_s}(\mathbf{z}\|\mathbf{s}) + q_{\phi_t}(\mathbf{z}\|\mathbf{t}) + q_{\phi_s}(\mathbf{z}\|\mathbf{s})q_{\phi_t}(\mathbf{z}\|\mathbf{t})$ | - |
| mmJSD | ✓ | ✗ | $f(q_{\phi_s}(\mathbf{z}\|\mathbf{s}), q_{\phi_t}(\mathbf{z}\|\mathbf{t}), p(\mathbf{z}))$ | - |
| **Ours** | ✓ | ✓ | - | - |

## 4.3 Method

Consider a scenario where we are given data spanning two modalities, $\mathbf{s}$ and $\mathbf{t}$, curated as pairs $(\mathbf{s}, \mathbf{t})$. For example this could be an "image" and associated "caption" of an observed scene. We will further assume that some proportion of observations have one of the modalities missing, leaving us with partially-observed data. Using $\mathcal{D}_{\mathbf{s},\mathbf{t}}$ to denote the proportion containing fully observed pairs from both modalities, and $\mathcal{D}_\mathbf{s}$, $\mathcal{D}_\mathbf{t}$ for the proportion containing observations only from modality $\mathbf{s}$ and $\mathbf{t}$ respectively, we can decompose the data as $\mathcal{D} = \mathcal{D}_\mathbf{s} \cup \mathcal{D}_\mathbf{t} \cup \mathcal{D}_{\mathbf{s},\mathbf{t}}$.

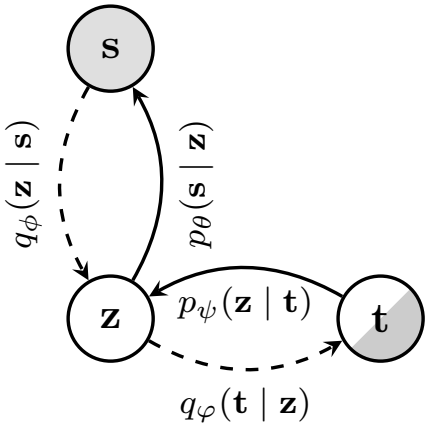



**Figure 4.2:** Simplified graphical model from Chapter 3.

In aid of clarity, we will introduce our method by confining attention to this bi-modal case, providing a discussion on generalising beyond two modalities later. Following established notation in the literature on VAEs, we will denote the generative model using $p$, latent variable using $\mathbf{z}$, and the encoder, or recognition model, using $q$. Subscripts for the generative and recognition models, where indicated, denote the parameters of deep neural networks associated with that model.

### 4.3.1 Semi-Supervised VAE

To develop our approach we draw inspiration from semi-supervised VAEs which use additional information, typically data labels, to extend the generative model. This facilitates learning tasks such as disentangling latent representations and performing intervention through conditional generation. In particular, we will build upon Chapter 3, where we supervise latent representations in VAEs with partial label information by forcing the encoder, or recognition model, to channel the flow of information as $\mathbf{s} \rightarrow \mathbf{z} \rightarrow \mathbf{t}$. They demonstrate that the model learns latent representations, $\mathbf{z}$, of data, $\mathbf{s}$, that can be faithfully identified with label information $\mathbf{t}$.

Figure 4.2 shows a modified version of the graphical model from Chapter 3, extracting just the salient components, and avoiding additional constraints therein. The label, here $\mathbf{t}$, is denoted as partially observed as not all observations $\mathbf{s}$ have associated labels. Note that, following the information flow argument, the generative model factorises

as $p_{\theta,\psi}(\mathbf{s}, \mathbf{z}, \mathbf{t}) = p_\theta(\mathbf{s} \mid \mathbf{z})\, p_\psi(\mathbf{z} \mid \mathbf{t})\, p(\mathbf{t})$ (solid arrows) whereas the recognition model factorises as $q_{\phi,\varphi}(\mathbf{t}, \mathbf{z} \mid \mathbf{s}) = q_\varphi(\mathbf{t} \mid \mathbf{z})\, q_\phi(\mathbf{z} \mid \mathbf{s})$ (dashed arrows). This autoregressive formulation of both the generative and recognition models is what enables the "supervision" of the latent representation of $\mathbf{s}$ by the label, $\mathbf{t}$, via the conditional prior $p_\psi(\mathbf{z} \mid \mathbf{t})$ as well as the classifier $q_\varphi(\mathbf{t} \mid \mathbf{z})$. It is worth noting that we have chosen to remove the inductive bias of introducing $\mathbf{z}_{\setminus \mathbf{c}}$ in Chapter 3 and instead provide the model with full flexibility to learn the relationship between latent factors.

The corresponding objective for *supervised* data, derived as the (negative) variational free energy or Evidence Lower Bound (ELBO) of the model is

$$\log p_{\theta,\psi}(\mathbf{s}, \mathbf{t}) \geq \mathcal{L}_{\{\boldsymbol{\Theta},\boldsymbol{\Phi}\}}(\mathbf{s}, \mathbf{t}) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{s})}\left[\frac{q_\varphi(\mathbf{t}|\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{s})} \log \frac{p_\theta(\mathbf{s}|\mathbf{z}) p_\psi(\mathbf{z}|\mathbf{t})}{q_\phi(\mathbf{z}|\mathbf{s}) q_\varphi(\mathbf{t}|\mathbf{z})}\right] + \log q_{\phi,\varphi}(\mathbf{t}|\mathbf{s}) + \log p(\mathbf{t}), \tag{4.1}$$

with the generative and recognition model parameterised by $\boldsymbol{\Theta} = \{\theta, \psi\}$ and $\boldsymbol{\Phi} = \{\phi, \varphi\}$ respectively. A derivation of this objective can be found in Appendix B.1.

### 4.3.2 Mutual Supervision

Procedurally, a semi-supervised VAE is already multimodal. Beyond viewing labels as a separate data modality, for more typical multimodal data (vision, language), one would just need to replace labels with data from the appropriate modality, and adjust the corresponding encoder and decoder to handle such data. Conceptually however, this simple replacement can be problematic.

Supervised learning encapsulates a very specific imbalance in information between observed data and the labels—that labels do not encode information beyond what is available in the observation itself. This is a consequence of the fact that labels are typically characterised as projections of the data into some lower-dimensional conceptual subspace such as the set of object classes one may encounter in images, for example. Such projections cannot introduce additional information into the system, implying that the information in the data subsumes the information in the

labels, i.e. that the conditional entropy of label $\mathbf{t}$ given data $\mathbf{s}$ is zero: $H(\mathbf{t} \mid \mathbf{s}) = 0$. Supervision-based models typically incorporate this information imbalance as a feature, as observed in the specific correspondences and structuring enforced between their label $\mathbf{y}$ and latent $\mathbf{z}$.

Multimodal data of the kind considered here, on the other hand, does not exhibit this feature. Rather than being characterised as a projection from one modality to another, they are better understood as idiosyncratic projections of an abstract concept into distinct modalities—for example, as an image of a bird or a textual description of it. In this setting, no one modality has *all* the information, as each modality can encode unique perspectives opaque to the other. More formally, this implies that both the conditional entropies $H(\mathbf{t} \mid \mathbf{s})$ and $H(\mathbf{s} \mid \mathbf{t})$ are finite.

Based on this insight we symmetrise the semi-supervised VAE formulation by additionally constructing a mirrored version, where we swap $\mathbf{s}$ and $\mathbf{t}$ along with their corresponding parameters, i.e. the generative model now uses the parameters $\boldsymbol{\Phi}$ and the recognition model now uses the parameters $\boldsymbol{\Theta}$. This has the effect of also incorporating the information flow in the opposite direction to the standard case as $\mathbf{t} \to \mathbf{z} \to \mathbf{s}$, ensuring that the modalities are now *mutually supervised*. This approach forces each encoder to act as an encoding distribution when information flows one way, but also act as a prior distribution when the information flows the other way. Extending the semi-supervised VAE objective (4.1), we construct a bi-directional objective for MEME

$$\mathcal{L}_{\mathrm{Bi}}(\mathbf{s}, \mathbf{t}) = \frac{1}{2}\Big[\mathcal{L}_{\{\boldsymbol{\Theta},\boldsymbol{\Phi}\}}(\mathbf{s}, \mathbf{t}) + \mathcal{L}_{\{\boldsymbol{\Phi},\boldsymbol{\Theta}\}}(\mathbf{t}, \mathbf{s})\Big], \tag{4.2}$$

where both information flows are weighted equally. On a practical note, we find that it is important to ensure that parameters are shared appropriately when mirroring the terms, and that the variance in the gradient estimator is controlled effectively. Please see Appendix B.4 and Appendix B.5 for further details.

### 4.3.3 Learning from Partial Observations

In practice, prohibitive costs on multimodal data collection and curation imply that observations can frequently be partial, i.e., have missing modalities. One of the main benefits of the method introduced here is its natural extension to the case of partial observations on account of its semi-supervised underpinnings. Consider, without loss of generality, the case where we observe modality $\mathbf{s}$, but not its pair $\mathbf{t}$. Recalling the autoregressive generative model $p(\mathbf{s}, \mathbf{z}, \mathbf{t}) = p(\mathbf{s} \mid \mathbf{z})p(\mathbf{z} \mid \mathbf{t})p(\mathbf{t})$ we can derive a lower bound on the log-evidence

$$\log p_{\theta,\psi}(\mathbf{s}) = \log \int p_\theta(\mathbf{s} \mid \mathbf{z}) p_\psi(\mathbf{z} \mid \mathbf{t}) p(\mathbf{t}) \, d\mathbf{z} \, d\mathbf{t} \geq \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{s})}\left[\log \frac{p_\theta(\mathbf{s} \mid \mathbf{z}) \int p_\psi(\mathbf{z} \mid \mathbf{t}) p(\mathbf{t}) \, d\mathbf{t}}{q_\phi(\mathbf{z} \mid \mathbf{s})}\right]. \tag{4.3}$$

Estimating the integral $p(\mathbf{z}) = \int p(\mathbf{z} \mid \mathbf{t}) p(\mathbf{t}) \, d\mathbf{t}$ highlights another conceptual difference between a (semi-)supervised setting and a multimodal one. When $\mathbf{t}$ is seen as a label, this typically implies that one could possibly compute the integral *exactly* by explicit marginalisation over its support, or at the very least, construct a reasonable estimate through simple Monte-Carlo integration. In Chapter 3, we extend the latter approach through importance sampling with the "inner" encoder $q(\mathbf{t} \mid \mathbf{z})$, to construct a looser lower bound to (4.3).

In the multimodal setting however, this poses serious difficulties as the domain of the variable $\mathbf{t}$ is not simple categorical labels, but rather complex continuous-valued data. This rules out exact marginalisation, and renders further importance-sampling practically infeasible on account of the quality of samples one can expect from the encoder $q(\mathbf{t} \mid \mathbf{z})$ which itself is being learnt from data. To overcome this issue and to ensure a flexible alternative, we adopt an approach inspired by the VampPrior (Tomczak and Welling, 2018b). Noting that our formulation includes a conditional prior $p_\psi(\mathbf{z} \mid \mathbf{t})$, we introduce learnable pseudo-samples $\lambda^{\mathbf{t}} = \{\mathbf{u}_i^{\mathbf{t}}\}_{i=1}^N$ to estimate the prior as $p_{\lambda^{\mathbf{t}}}(\mathbf{z}) = \frac{1}{N} \sum_{i=1}^N p_\psi(\mathbf{z} \mid \mathbf{u}_i^{\mathbf{t}})$. Our objective for when $\mathbf{t}$ is unobserved is thus

$$\mathcal{L}(\mathbf{s}) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{s})}\left[\log \frac{p_\theta(\mathbf{s} \mid \mathbf{z}) p_{\lambda^{\mathbf{t}}}(\mathbf{z})}{q_\phi(\mathbf{z} \mid \mathbf{s})}\right] = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{s})}\left[\log \frac{p_\theta(\mathbf{s} \mid \mathbf{z})}{q_\phi(\mathbf{z} \mid \mathbf{s})} + \log \frac{1}{N} \sum_{i=1}^N p_\psi(\mathbf{z} \mid \mathbf{u}_i^{\mathbf{t}})\right], \tag{4.4}$$

where the equivalent objective for when $\mathbf{s}$ is missing can be derived in a similar way. For a dataset $\mathcal{D}$ containing partial observations the overall objective (to maximise) becomes

$$\sum_{\mathbf{s},\mathbf{t}\in\mathcal{D}} \log p_{\theta,\psi}(\mathbf{s},\mathbf{t}) \geq \sum_{\mathbf{s}\in\mathcal{D}_{\mathbf{s}}} \mathcal{L}(\mathbf{s}) + \sum_{\mathbf{t}\in\mathcal{D}_{\mathbf{t}}} \mathcal{L}(\mathbf{t}) + \sum_{\mathbf{s},\mathbf{t}\in\mathcal{D}_{\mathbf{s},\mathbf{t}}} \mathcal{L}_{\text{Bi}}(\mathbf{s},\mathbf{t}), \tag{4.5}$$

This treatment of unobserved data distinguishes our approach from alternatives such as that of Shi et al. (2019a), where model updates for missing modalities are infeasible. Whilst there is the possibility to perform multimodal learning in the weakly supervised case as introduced by Wu and Goodman (2018b), their approach directly affects the posterior distribution, whereas ours only affects the regularization of the embedding during training. At test time, Wu and Goodman (2018b) will produce different embeddings depending on whether all modalities are present, which is typically at odds with the concept of placing the embeddings of related modalities in the same region of the latent space. Our approach does not suffer from this issue as the posterior remains unchanged regardless of whether the other modality is present or not.

**Learning with MEME** Given the overall objective in (4.5), we train MEME through maximum-likelihood estimation of the objective over a dataset $\mathcal{D}$. Each observation from the dataset is optimised using the relevant term in the right-hand side of (4.5), through the use of standard stocastic gradient descent methods. Note that training the objective involves learning *all* the (neural network) parameters $(\theta, \psi, \phi, \varphi)$ in the fully-observed, bi-directional case. When training with a partial observation, say just $\mathbf{s}$, all parameters except the relevant likelihood parameter $\varphi$ (for $q_\varphi(\mathbf{t} \mid \mathbf{z})$) are learnt. Note that the encoding for data in the domain of $\mathbf{t}$ is still computed through the learnable pseudo-samples $\lambda^{\mathbf{t}}$. This is reversed when training on an observation with just $\mathbf{t}$.

**Generalisation beyond two modalities** We confine our attention here to the bi-modal case for two important reasons. Firstly, the number of modalities one typically encounters in the multimodal setting is fairly small to begin with. This

is often a consequence of its motivation from embodied perception, where one is restricted by the relatively small number of senses available (e.g. sight, sound, proprioception). Furthermore, the vast majority of prior work on multimodal VAEs only really consider the bimodal setting (cf. Section 4.2). Secondly, it is quite straightforward to extend MEME to settings beyond the bimodal case, by simply incorporating existing explicit combinations (e.g. mixtures or products) *on top of* the implicit combination discussed here, we provide further explanation in Appendix B.6. Our focus in this work lies in exploring and analysing the utility of implicit combination in the multimodal setting, and our formulation and experiments reflect this focus.

## 4.4 Experiments

### 4.4.1 Learning from Partially Observed Data

In this section, we evaluate the performance of MEME following standard multimodal VAE metrics as proposed in Shi et al. (2019a). Since our model benefits from its implicit latent regularisation and is able to learn from partially-observed data, here we evaluate MEME's performance when different proportions of data are missing in either or both modalities during training. The two metrics used are *cross coherence* to evaluate the semantic consistency in the reconstructions, as well as *latent accuracy* in a classification task to quantitatively evaluate the representation learnt in the latent space. We demonstrate our results on two datasets, namely an image $\leftrightarrow$ image dataset MNIST-SVHN (LeCun et al., 2010; Netzer et al., 2011), which is commonly used to evaluate multimodal VAEs (Shi et al., 2019a; Shi et al., 2021; Sutter et al., 2020, 2021); as well as the more challenging, but less common, image $\leftrightarrow$ caption dataset CUB (Welinder et al., 2010).

Following standard approaches, we represented image likelihoods using Laplace distributions, and a categorical distribution for caption data. The latent variables are parameterised by Gaussian distributions. In line with previous research (Shi et al., 2019a; Massiceti et al., 2018), simple convolutional architectures were used for both MNIST-SVHN and for CUB images *and* captions. For the cpations data,

we first fit a FastText model on all sentences, resulting in a 300-$d$ projection for each word (Bojanowski et al., 2017), these projections are then stacked to form an 'image', permitting convolutions to be performed. For details on training and exact architectures see Appendix B.12; we also provide tabularised results in Appendix B.9.

**Cross Coherence** Here, we focus mainly on the model's ability to reconstruct one modality, say, $\mathbf{t}$, given another modality, $\mathbf{s}$, as input, while preserving the conceptual commonality between the two. In keeping with Shi et al. (2019a), we report the cross coherence score on MNIST-SVHN as the percentage of matching digit predictions of the input and output modality obtained from a pre-trained classifier. On CUB we perform Canonical Correlation Analysis (CCA) on input-output pairs of cross generation to measure the correlation between these samples. For more details on the computation of CCA values we refer to Appendix B.8.

In Figure 4.5 we plot cross coherence for MNIST-SVHN and display correlation results for CUB in Figure 4.6, across different partial-observation schemes. The $x$-axis represents the proportion of data that is paired, while the subscript to the method (see legends) indicates the modality that is presented. For instance, MEME_MNIST with $f = 0.25$ indicates that only 25% of samples are paired, and the other 75% only contain MNIST digits, and MEME_SPLIT with $f = 0.25$ indicates that the 75% contains a mix of MNIST and SVHN samples that are unpaired and never observed together, i.e we alternate depending on the iteration, the remaining 25% contain paired samples. We provide qualitative results in Figure 4.3 and Figure 4.4.

We can see that our model is able to obtain higher coherence scores than the baselines including MVAE (Wu and Goodman, 2018b) and MMVAE (Shi et al., 2019a) in the fully observed case, $f = 1.0$, as well as in the case of partial observations, $f < 1.0$. This holds true for both MNIST-SVHN and CUB[1]. It is worth pointing out that the coherence between SVHN and MNIST is similar for both partially observing

[1]We note that some of the reported results of MMVAE in our experiments do not match those seen in the original paper, please visit Appendix B.10 for more information.


**Figure 4.3:** MEME cross-modal generations for MNIST-SVHN.

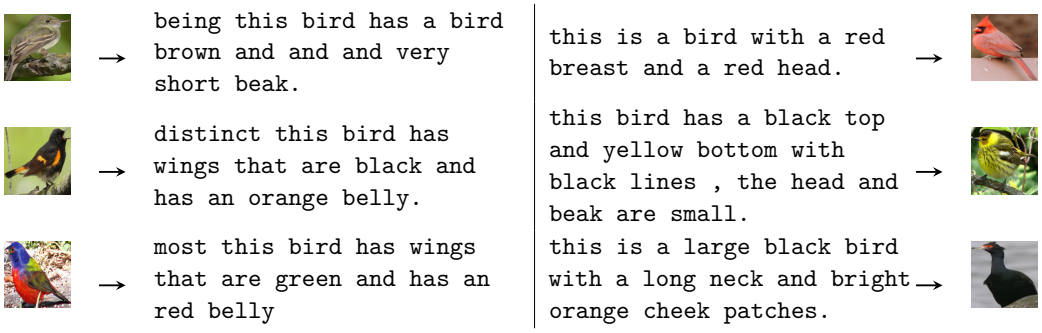



**Figure 4.4:** MEME cross-modal generations for CUB.

MNIST or SVHN, i.e. generating MNIST digits from SVHN is more robust to which modalities are observed during training (Figure 4.5 Right). However, when generating SVHN from MNIST, this is not the case, as when partially observing MNIST during training the model struggles to generate appropriate SVHN digits. This behaviour is somewhat expected since the information needed to generate an MNIST digit is typically subsumed within an SVHN digit (e.g. there is little style information associated with MNIST), making generation from SVHN to MNIST easier, and from MNIST to SVHN more difficult. Moreover, we also hypothesise that observing MNIST during training provides greater clustering in the latent space, which seems to aid cross generating SVHN digits. We provide additional t-SNE plots in Appendix B.9.3 to justify this claim.

For CUB we can see in Figure 4.6 that MEME consistently obtains higher correlations than MVAE across all supervision rates, and higher than MMVAE in the fully supervised case. Generally, cross-generating images yields higher correlation values, possibly due to the difficulty in generating semantically meaningful text with relatively simplistic convolutional architectures. We would like to highlight that partially observing captions typically leads to poorer performance when cross-generating captions. We hypothesise that is due to the difficulty in generating the

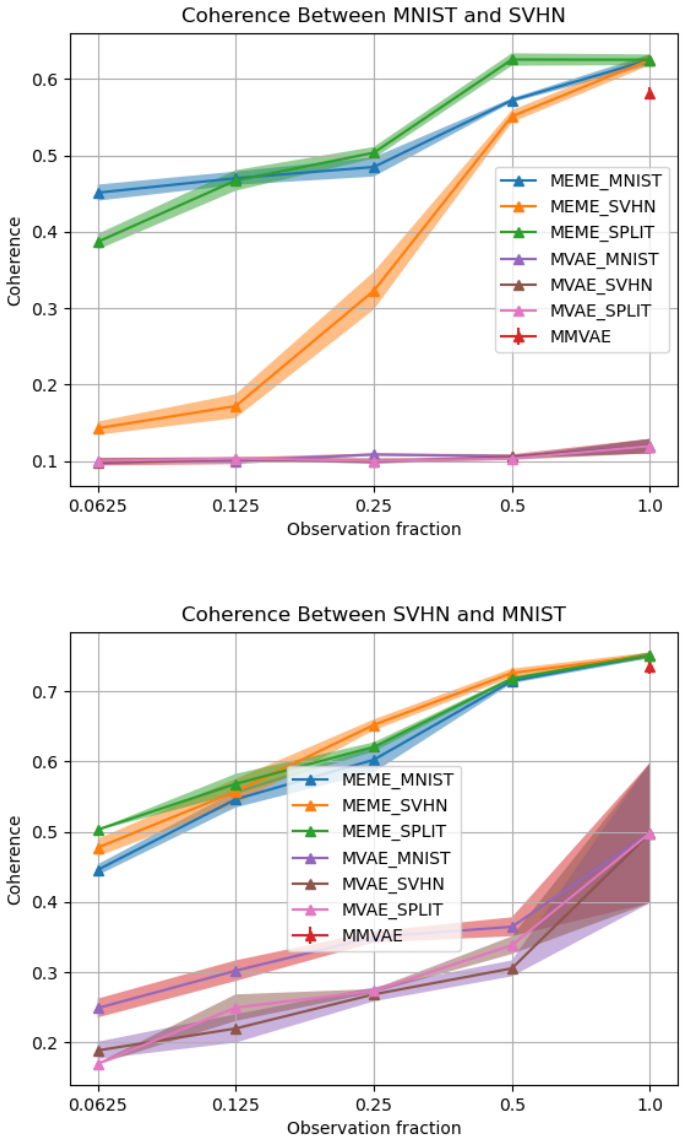



**Figure 4.5:** Coherence between MNIST and SVHN (Top) and SVHN and MNIST (Bottom). Shaded area indicates one-standard deviation of runs with different seeds.

captions and the fact there is a limited amount of captions data in this setting.

**Latent Accuracy** To gauge the quality of the learnt representations we follow previous work (Higgins et al., 2017; Kim and Mnih, 2018; Shi et al., 2019a; Sutter et al., 2021) and fit a linear classifier that predicts the input digit from the latent samples. The accuracy of predicting the input digit using this classifier indicates how well the latents can be separated in a linear manner.

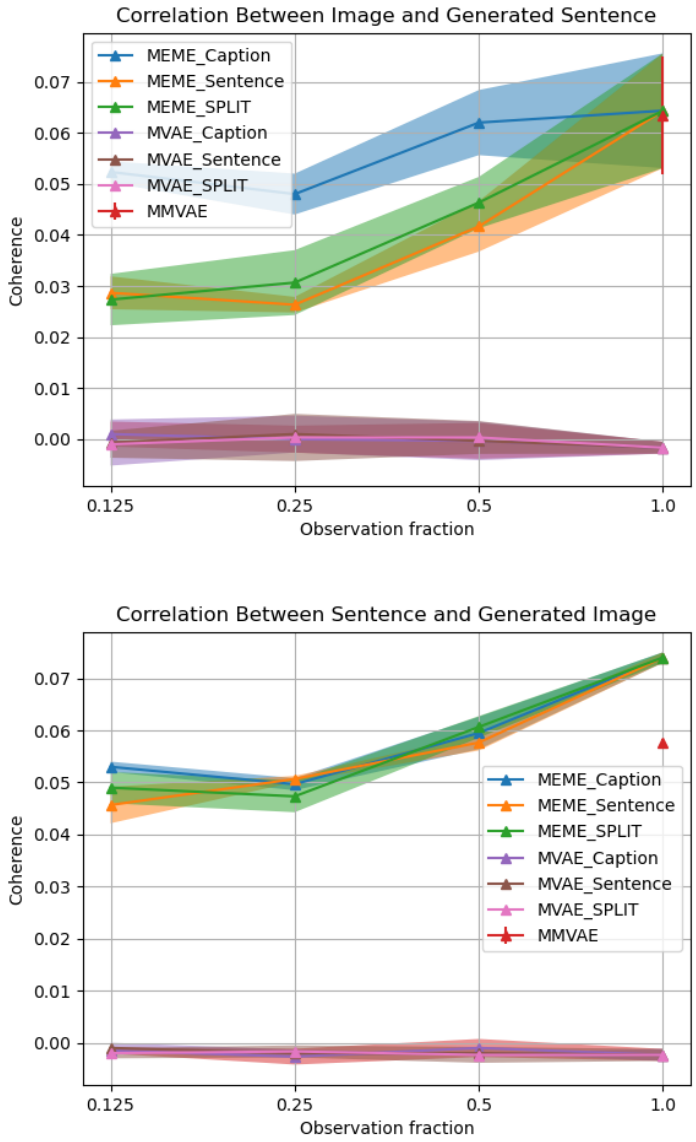



**Figure 4.6:** Correlation between Image and Sentence (Top) and Sentence and Image (Bottom). Shaded area indicates one-standard deviation of runs with different seeds.

In Figure 4.7, we plot the latent accuracy on MNIST and SVHN against the fraction of observation. We can see that MEME outperforms MVAE on both MNIST and SVHN under the fully-observed scheme (i.e. when observation fractions is 1.0). We can also notice that the latent accuracy of MVAE is rather lopsided, with the performance on MNIST to be as high as 0.88 when only 1/16 of the data is observed, while SVHN predictions remain almost random even when all data are used; this indicates that MVAE relies heavily on MNIST to extract digit information. On

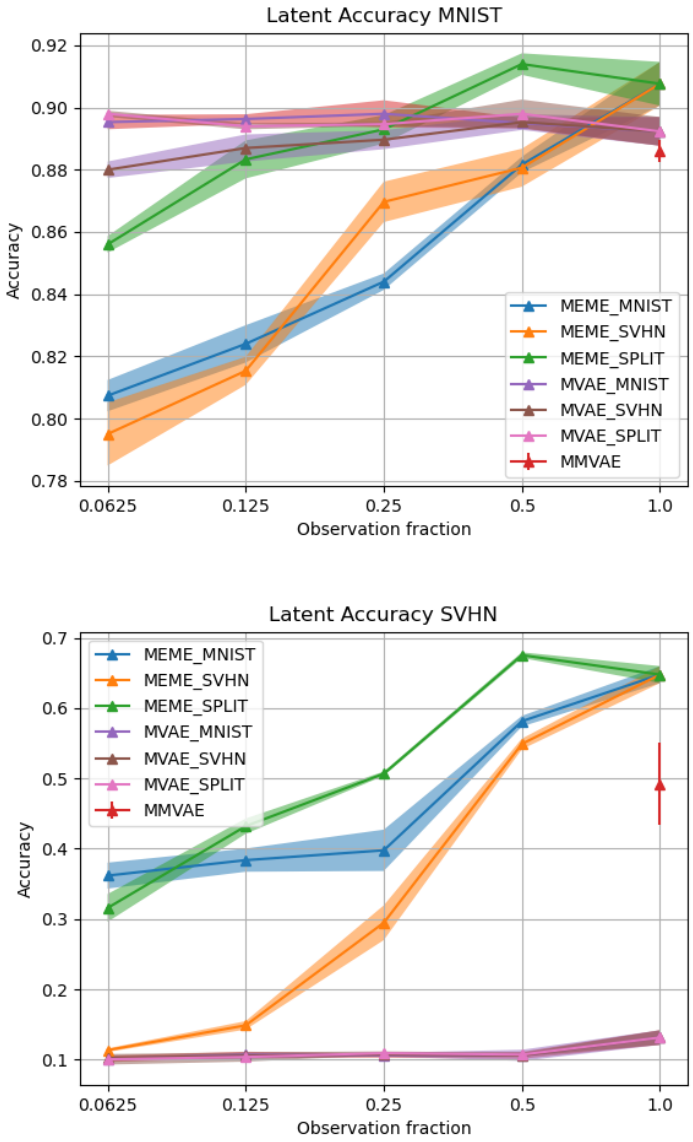



**Figure 4.7:** Latent accuracies for MNIST and SVHN (Top) and SVHN and MNIST (Bottom). Shaded area indicates one-standard deviation of runs with different seeds.

the other hand, MEME's latent accuracy observes a steady increase as observation fractions grow in both modalities. It is worth noting that both models performs better on MNIST than SVHN in general—this is unsurprising as it is easier to disentangle digit information from MNIST, however our experiments here show that MEME does not completely disregard the digits in SVHN like MVAE does, resulting in more balanced learned representations. It is also interesting to see that MVAE obtains a higher latent accuracy than MEME for low supervision rates.

This is due to MVAE learning to construct representations for each modality in a completely separate sub-space in the latent space, we provide a t-SNE plot to demonstrate this in Appendix B.9.1.

**Ablation Studies** To study the effect of modelling and data choices on performance, we perform two ablation studies: one varying the number of pseudo-samples for the prior, and the other evaluating how well the model leverages partially observed data over fully observed data. We find that performance degrades, as expected, with fewer pseudo-samples, and that the model trained with additional partially observed data does indeed improve. See Appendix B.11 for details.

### 4.4.2 Evaluating Relatedness

Now that we have established that the representation learned by MEME contains rich class information from the inputs, we also wish to analyse the relationship between the encodings of different modalities by studying their "relatedness", i.e. semantic similarity. The probabilistic nature of the learned representations suggests the use of probability distance functions as a measure of relatedness, where a low distance implies closely related representations and vice versa.

In the following experiments we use the 2-Wasserstein distance, $\mathcal{W}_2$, a probability metric with a closed-form expression for Gaussian distributions (see Appendix B.7 for more details). Specifically, we compute $d_{ij} = \mathcal{W}_2(\, q(\mathbf{z}|\mathbf{s}_i) \,\|\, q(\mathbf{z}|\mathbf{t}_j)\,)$, where $q(\mathbf{z}|\mathbf{s}_i)$ and $q(\mathbf{z}|\mathbf{t}_j)$ are the individual encoders, for all combination of pairs $\{\mathbf{s}_i, \mathbf{t}_j\}$ in the mini-batch, i.e $\{\mathbf{s}_i, \mathbf{t}_j\}$, for $i, j \in \{1 \ldots, M\}$ where $M$ is the number of elements in the mini-batch.

**General Relatedness** In this experiment we wish to highlight the disparity in measured relatedness between paired vs. unpaired multimodal data. To do so, we plot $d_{ij}$ on a histogram and color-code the histogram by whether the corresponding data pair $\{\mathbf{s}_i, \mathbf{t}_j\}$ shows the same concept, e.g. same digit for MNIST-SVHN and same image-caption pair for CUB. Ideally, we should observe smaller distances between encoding distributions for data pairs that are related, and larger for ones that are not.

To investigate this, we plot $d_{ij}$ on a histogram for every mini-batch; ideally we should see higher densities at closer distances for points that are paired, and higher densities at further distances for unpaired points. In Figure 4.8, we see that MEME (left) does in fact yields higher mass at lower distance values for paired multimodal samples (orange) than it does for unpaired ones (blue). This effect is not so pronounced in

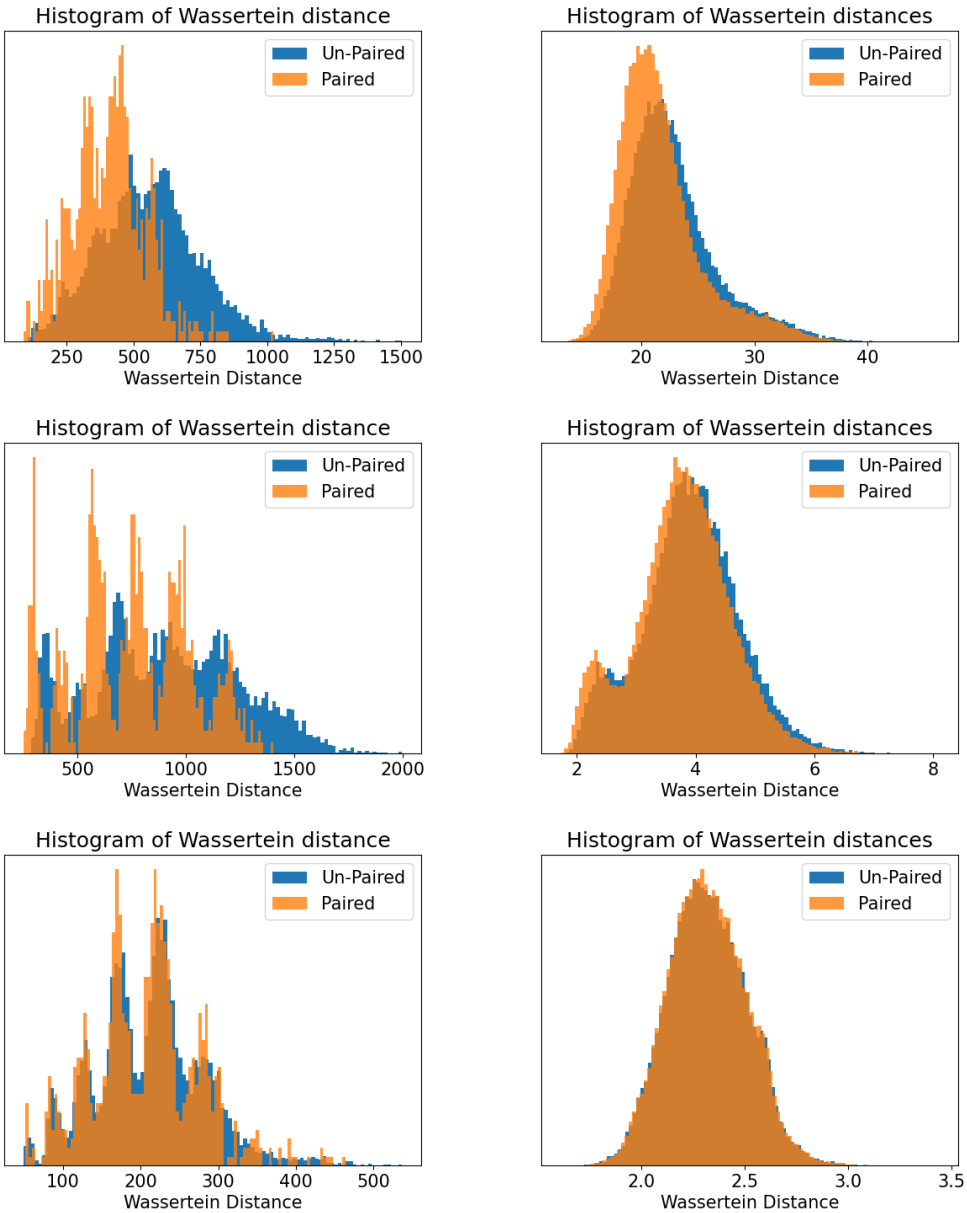



**Figure 4.8:** Histograms of Wassertein distance for SVHN and MNIST (Left) and CUB (Right): MEME (Top), MMVAE (middle) and MVAE (Bottom). Blue indicates *unpaired* samples and orange *paired* samples. We expect to see high densities of blue at further distances and visa-versa for orange.

MMVAE and not present at all in MVAE. This demonstrates MEME's capability of capturing relatedness between multimodal samples in its latent space, and the quality of its representation.

**Class-contextual Relatedness** To offer more insights on the relatedness of representations within classes, we construct a distance matrix $\mathbf{K} \in \mathbb{R}^{10\times10}$ for the MNIST-SVHN dataset, where each element $\mathbf{K}_{i,j}$ corresponds to the average $\mathcal{W}_2$ distance between encoding distributions of class $i$ of MNIST and $j$ of SVHN. A perfect distance matrix will consist of a diagonal of all zeros and positive values in the off-diagonal.

See the class distance matrix in Figure 4.9 (right), generated with models trained on fully observed multimodal data. It is clear that our model (top) produces much lower distances on the diagonal, i.e. when input classes for the two modalities are the same, and higher distances off diagonal where input classes are different. A clear, lower-valued diagonal can also be observed for MMVAE (middle), however it is less distinct compared to MEME, since some of the mismatched pairs also obtains smaller values. The distance matrix for MVAE (bottom), on the other hand, does not display a diagonal at all, reflecting poor ability to identify relatedness or extract class information through the latent.

To closely examine which digits are considered similar by the model, we construct dendrograms to visualise the hierarchical clustering of digits by relatedness, as seen in Figure 4.9 (right). To do this, we first obtain the latent representation for each class and subsequently create a linking matrix between the classes based on the distances between the clusters. Using this linkage function, the dendrogram can be produced which represents how close certain clusters are form one another. We see that our model (left) is able to obtain a clustering of conceptually similar digits. In particular, digits with smoother writing profile such as `3`, `5`, `8`, along with `6` and `9` are clustered together (right hand side of dendrogram), and the digits with sharp angles, such as `4` and `7` are clustered together. The same trend is not observed for MMVAE nor MVAE. It is also important to note the height of each bin, where

higher values indicate greater distance between clusters. Generally the clusters obtained in MEME are further separated for MMVAE, demonstrating more distinct clustering across classes.

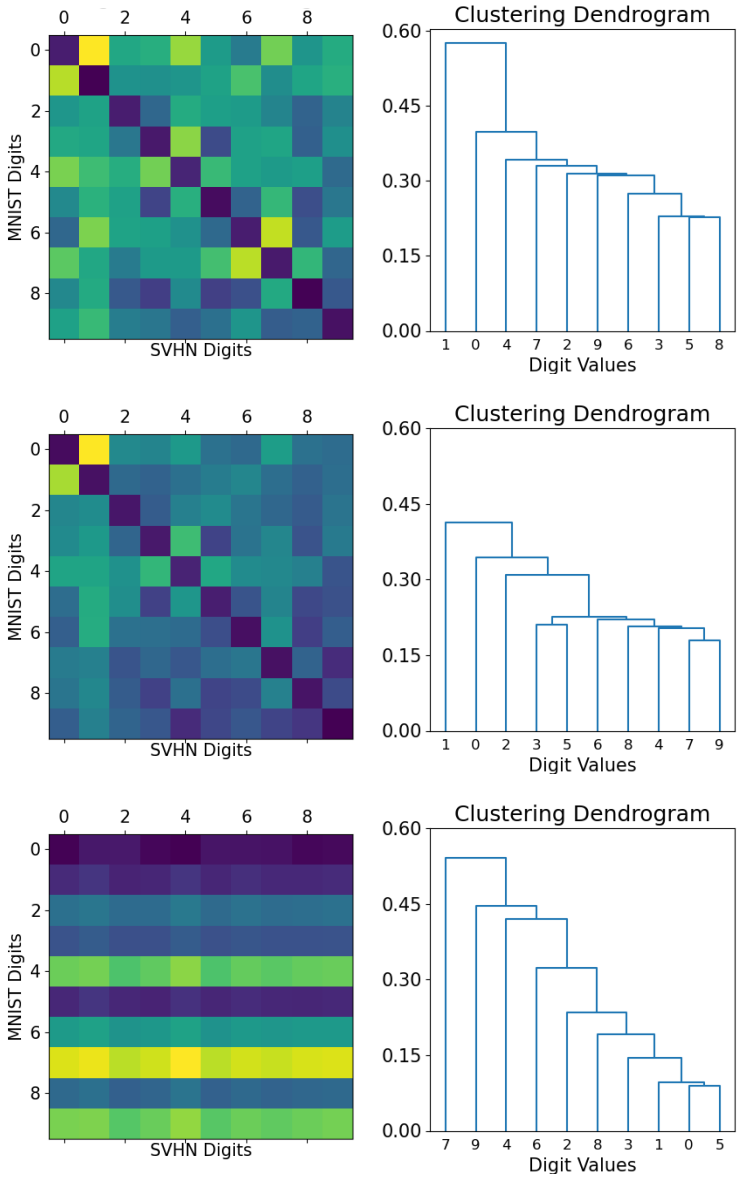



**Figure 4.9:** Distance matrices for Wasserstein divergence between classes for SVHN and MNIST (Top) and dendrogram (Bottom) for: Ours (Left), MMVAE (middle) and MVAE (Right).

## 4.5 Discussion

Here we have presented a method which faithfully deals with partially observed modalities in VAEs. Through leveraging recent advances in semi-supervised VAEs,

we construct a model which is amenable to multi-modal learning when modalities are partially observed. Specifically, our method employs mutual supervision by treating the uni-modal encoders individually and minimizing a KL between them to ensure embeddings for are pertinent to one another. This approach enables us to successfully learn a model when either of the modalites are partially observed. Furthermore, our model is able to naturally extract an indication of relatedness between modalities. We demonstrate our approach on the MNIST-SVHN and CUB datasets, where training is performed on a variety of different observations rates.

**Ethics Statement** We believe there are no inherent ethical concerns within this work, as all datasets and motivations do not include or concern humans. As with every technological advancement there is always the potential for miss-use, for this work though, we can not see a situation where this method may act adversarial to society. In fact, we believe that multi-modal representation learning in general holds many benefits, for instance in language translation which removes the need to translate to a base language (normally English) first.

# 5

# Sample-dependent Temperature Scaling for Improved Calibration

# Abstract

It is now well known that neural networks can be wrong with high confidence in their predictions, leading to poor calibration. The most common post-hoc approach to compensate for this is to perform temperature scaling, which adjusts the confidences of the predictions on any input by scaling the logits by a fixed value. Whilst this approach typically improves the average calibration across the *whole* test dataset, this improvement typically reduces the individual confidences of the predictions irrespective of whether the classification of a given input is correct or incorrect. With this insight, we base our method on the observation that different samples contribute to the calibration error by varying amounts, with some needing to increase their confidence and others needing to decrease it. Therefore, for each input, we propose to predict a different temperature value, allowing us to adjust the mismatch between confidence and accuracy at a finer granularity. Our method is applied post-hoc, consequently using very little computation time and with a negligible memory footprint and is applied to off-the-shelf pre-trained classifiers. We test our method on the ResNet50 and WideResNet28-10 architectures using the CIFAR10/100 and Tiny-ImageNet datasets, showing that producing per-data-point temperatures is beneficial also for the expected calibration error across the whole test set.

## 5.1 Introduction

For neural networks to be employed in real-world safety-critical applications, we do not only require them to produce correct predictions, but also provide reliable confidence estimates in their predictions (i.e. they are calibrated). Limiting our scope to neural classifiers, using the maximum probability of the predictive distribution as a confidence measure, literature has established that a mismatch exists between such notion of confidence and the expected accuracy. Indeed, such models generally suffer from being *on average* overconfident over the test-set.

A simple approach to rectify this issue is to perform *temperature scaling* (Guo et al., 2017), a post-hoc method which scales the logits by a single scalar value, obtained through cross validation. This approach improves the classifier's performance on standard calibration metrics across a test dataset. However, from a per-sample point of view there are significant issues. Since the temperature is found by minimising the calibration error (in expectation) over the *entire* validation set, and since neural networks are overconfident on average, practically speaking, the effect of temperature scaling is to reduce the confidence for every prediction. However, as we will discuss, different samples contribute by varying amounts to the calibration error.

This issue can be seen in Figure 5.1, which shows the histogram of the individual contributions to the calibration errors; i.e. the distribution of $|p(\mathbf{y}|\boldsymbol{p}_i) - \boldsymbol{p}_i|$, where $p(\mathbf{y}|\boldsymbol{p}_i)$ is the accuracy and $\boldsymbol{p}_i$ is the softmax probability for the data point $i$, the calibration error can be obtained by taking the weighted average over all the values.[1] Here the mismatch between per data-point confidence and accuracy is not constant across all the data-points, and hence miscalibration cannot be fixed by scaling the logits by a single fixed value, a key assumption in vanilla temperature scaling. The calibration error varies significantly, with a small (but not insignificant) number of samples on which the network is overconfident. Consequently, scaling the

[1] Here $p(\mathbf{y}|\boldsymbol{p}_i)$ is obtained through histogram binning and represents the accuracy of each bin, and the weights are proportional to the number of samples in each bin.

predictions with a single temperature value will adjust *all* of the errors in the same way. Typically, the temperature values obtained are greater than 1, resulting in a reduction of confidence of *all* predictions, regardless of whether they are correct with low confidence or incorrect with high confidence.

To combat this, we propose a method which produces per-data-point predictions of the temperature, permitting an adequate decrease in the confidence on samples which the classifier is *likely* to get wrong, and an increase in the confidence on predictions it is *likely* to get correct. As a result, we obtain better test Expected Calibration Error (ECE) guo2017calibration both on in-distribution sets (i.e. the test set is i.i.d. with respect to the training set) and under covariate-shifted sets (i.e. the test set shares the same set of labels of the training set, but the inputs are not i.i.d. with respect to the training set).

Like temperature scaling, our method is applied post-hoc and is very fast to train and test. We extensively test the calibration of ResNet50 (He et al., 2016) and WideResNet28 (Zagoruyko and Komodakis, 2016) when using our method on CIFAR10/CIFAR100 and TinyImageNet, including results under data-shift (Hendrycks and Dietterich, 2019). Specifically our contributions is to identify a limitation in using a constant temperature for temperature scaling and propose a novel method to predict temperature values on a per-data-point basis to address this limitation. Our method produces a temperature value that is sample dependent, allowing the method to reduce the confidence of incorrect predictions, but also increase the confidence of correct ones.

## 5.2 Problem formulation

### 5.2.1 Network Overconfidence and Temperature Scaling

Given an input $\mathbf{x}$, a standard $K$-class neural classifier first extracts a feature embedding $\Phi(\mathbf{x})$ before computing the logits $\mathbf{s} = f(\Phi(\mathbf{x})) \in \mathbb{R}^K$ and finally applying the softmax operator $\boldsymbol{p} = \sigma(\mathbf{s}) = \frac{\exp(\mathbf{s})}{\sum_i \exp(s_i)}$ to obtain the class probabilities for the categorical distribution, the prediction is then given as $\hat{y} = \arg\max \boldsymbol{p}$. A classifier is said to be calibrated if the confidence in its prediction (usually taken to be

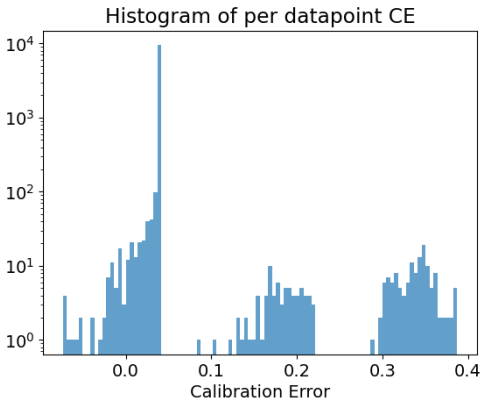



**Figure 5.1:** Histogram of per sample contribution to calibration error, positive numbers indicate overconfidence. Here we can see that the samples contribute by different amounts to the overall calibration error. Predictions are for CIFAR-10 using a ResNet-50.

$\max \boldsymbol{p}$) matches its accuracy on expectation, i.e. if a classifier makes predictions with a confidence of 80% for a certain set of points, then it also has an accuracy of 80% on such set of points. Typically, the predictions of neural networks produce overconfident, i.e. the probability of the predicted class is higher than their expected accuracy (Guo et al., 2017).

Temperature scaling (Guo et al., 2017) consists of re-scaling the logits by a constant factor $T \in \mathbb{R}^+$ before applying the softmax, i.e. $\boldsymbol{p}' = \sigma(\frac{\mathbf{s}}{T})$. The value of $T$ can drastically affect the entropy of the predicted distribution, which is demonstrated in Figure 5.2, where a value of $T > 1$ leads to a higher entropy distribution (the higher $T$, the higher the entropy); a value of $T < 1$ leads to a lower entropy distribution (the lower $T$, the more "peaky" the distribution).

The temperature $T$ is usually found by minimising the ECE or the Negative Log-Likelihood (NLL) using a validation set. Typical optimal values for $T$ are usually greater than 1 (Mukhoti et al., 2020), indicating that, on average, optimising the ECE or NLL across the validation set leads to a higher entropy of the predictions. However, this approach decreases the confidences of *all* the predictions without considering that the miscalibration error can vary widely on a data-point basis. For correct predictions, temperature scaling will make the predictions more under-confident, whilst for incorrect predictions, the temperature may not be the right value to bring the confidences down to a level which will make it calibrated.

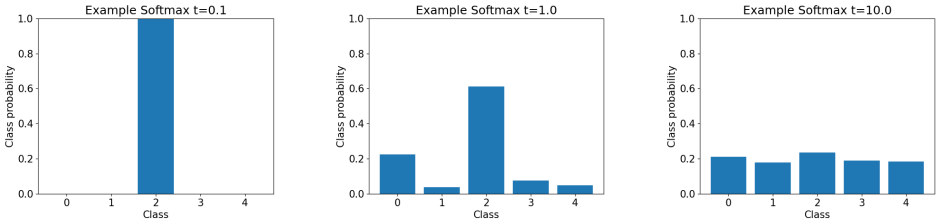



**Figure 5.2:** Example plots of Softmax distribution with different temperature values for fixed logits. Left to right: $T = 0.1, T = 1.0$ and $T = 10.0$.

This suggests that further improvements in calibration can be achieved by using a variable temperature $T$, predicted on a per data-point basis (i.e. $T = g(\mathbf{x})$), permitting $T > 1$ for over-confident or incorrect predictions, and $T < 1$ for under-confident or correct predictions. Moreover, this approach can be applied *without* affecting a classifiers accuracy[2].

### 5.2.2 Why Jointly Learning Temperature Alongside the Network Weights Might Go Wrong?

Here we outline why learning to predict the temperature values $T$ and predictive probabilities $\boldsymbol{p}$ cannot be performed at the same time. Consider the last layer of a NN with parameters $\boldsymbol{w} \in \mathbb{R}^{D \times K}$ for a feature space of size $D$ and the cross entropy loss $\mathcal{L} : \mathbb{R}^K \to \mathbb{R}$. The gradient for the layer is given as

$$\frac{\partial \mathcal{L}}{\partial \bar{\boldsymbol{w}}} = \frac{\partial \mathbf{s}}{\partial \bar{\boldsymbol{w}}}(\sigma(\mathbf{s}) - \boldsymbol{q}), \tag{5.1}$$

where $\boldsymbol{q} \in \mathbb{R}^K$ is the one-hot logit of the label and $\sigma(\mathbf{s}) - \boldsymbol{q} = \{\sigma(s_j) - q_j : j \in \{1 \dots K\}$[3], $\bar{\boldsymbol{w}}$ indicates the network weights are flattened to column vector form. Inspecting the gradients indicates that the gradient starts to vanish when $s_k \to \infty$ and $s_{\setminus k} \to -\infty$, where $k$ is the correct class. Or to put it simply, the optimisation does not converge until the network produces one-hot logits.

This forces the magnification of the network weights (Mukhoti et al., 2020), which subsequently leads to an overconfident network and hence miss-calibrated predictions. A mechanism to achieve the desired one-hot prediction without magnifying the weights could be instead to naïvely learn the temperature alongside the logits,

[2]For a proof please see Appendix C.2
[3]See Appendix C.1 for a derivation

assuming the model is trainable and converges. In this case, gradient updates would decrease the value of $T$, resulting in a lower-entropy distribution that is more "peaky". This has been attempted before in Neumann et al. (2018) but with limited success. We now outline why this approach does not work well in practice.

If we consider the gradient of the temperature, which is given as

$$\frac{\partial \mathcal{L}}{\partial T} = \sum_k \frac{q_k}{T^2} \left( s_k \sum_{i \setminus k} \exp\left(\frac{s_i}{T}\right) - \sum_{j \setminus k} s_j \exp\left(\frac{s_j}{T}\right) \right), \tag{5.2}$$

which decreases the value of $T$ for a correct prediction ($\tilde{k} = \arg\max_k s_k$), leading to more confident predictions, see Appendix C.3 for a proof. Typically, the train accuracy will approach 100%, meaning that gradient updates to $T$ cause it to decrease without any moderation, preventing the network from learning how to predict $T$ appropriately. In short, there is essentially only data for correct predictions, preventing crucial information on how the network should behave when it's wrong. Consequently, learning $T$ naïvely is not a feasible option as the network just learns to be confident everywhere. Empirically we found this to be the case when experimenting with the technique described in Neumann et al. (2018).

### 5.2.3 Learning to Calibrate

We propose to use a separate training regime, with an objective to learn how to calibrate the confidences for each prediction. Doing so requires learning a temperature prediction module on a data-set consisting of data-points $\mathcal{X}_{cal} = \{\mathbf{x}_n\}^N, \mathcal{X}_{train} \cap \mathcal{X}_{cal} = \varnothing$, neural network predictions $\mathcal{P}_{cal} = \{\boldsymbol{p}_n\}^N$ and labels $\mathcal{Y}_{cal} = \{\mathbf{y}_n\}^N$. It is important to note that the objective here is to learn to assign low confidences to data points which are *likely* to be incorrect and high confidences to those which are likely to be correct.

Specifically for a given data-point $\mathbf{x} \in \mathcal{X}_{cal}$, we propose to optimise the temperature prediction module over $T$ by maximising the log probability of the label $\mathbf{y}$ under the Categorical probability distribution parametrised by the $T$-scaled logits $\mathbf{s}$, i.e.

$T^* = \arg\max_T \log \text{Cat}(\mathbf{y}; \text{softmax}(\mathbf{s}/T))$[4]. Here we do not optmise $\mathbf{s}$ is fixed, we are only optimising w.r.t to $T$.

In situations where $\mathbf{y} = \arg\max_k \boldsymbol{p}_k$ (i.e. correct prediction), the target function is maximised when $T \to 0$, as we want the predicted probabilities to match the one-hot logits, e.g. see $T = 0.1$ in Figure 5.2. This is equivalent to minimising the entropy of the predictive distribution by only manipulating $T$, which is the desired outcome for a correct prediction.

In situations where the prediction is incorrect, $\mathbf{y} \neq \arg\max_k \boldsymbol{p}_k$, to maximise the target function we need to maximise $\boldsymbol{p}_{\mathbf{y}}$ and minimise $\boldsymbol{p}_{\tilde{k}}$, where $\tilde{k} = \arg\max_k \boldsymbol{p}_k$. As the temperature prediction module cannot change the predicted label, the optimzation accepts the incorrect prediction and maximise the target function by flattening the Softmax outputs with $T \gg 1$, which is equivalent to maximising the entropy of the predictive distribution. This effect can be seen by considering the case where predicting class 2 in Figure 5.2 is the incorrect prediction; among the three cases shown, $T = 10$ maximises the $\text{Cat}(\mathbf{y} \neq 2; \text{softmax}(\mathbf{s}/T))$.

## 5.3 Representing Uncertainty with the Variational Autoencoder (VAE)

VAEs (Kingma and Welling, 2013) act as an efficient model to obtain representations of data; the representations encapsulate the generative factors in a lower-dimensional subspace and are rich enough to reconstruct the data sample. In the specification of the generative model, the user has to specify a prior over the latent variables (typically an isotropic Gaussian) where the KL distance between the prior and approximate posterior is minimised during training. Unlike a standard autoencoder (Hinton and Zemel, 1994), there is now a mechanism to obtain a likelihood on the latent codes. In reality this value forms part of the importance weight and can be used as a proxy to the true likelihood but avoids issues associated with deep generative models (Nalisnick et al., 2019).

[4]Which is equivalent to the cross entropy loss.

From a mechanistic point of view, we expect samples which are much more common to be placed in the centre of the prior. Here we leverage this idea and use the latent likelihoods as a basis to predict the temperature value. Indeed, we find empirically that this approach works well in practice. Rather than using an isotropic Guassian as the prior, we instead introduce a Gaussian mixture prior, with component for each class specified by the *learnable* parameters $\lambda_k = \{\mu_k, \sigma_k\} \in \mathbb{R}^{D_{\mathbf{z}}}$. This allows for an individual unimodal prior for each class; preventing any issue with clusters for individual classes being placed in lower likelihood regions of the latent space, as would be the case if an isotropic prior is used. With this mixture prior, the evidence lower bound is given as

$$\mathbb{ELBO}[\Phi(\mathbf{x})] = \mathbb{E}_{q_\varphi(\mathbf{z}|\Phi(\mathbf{x}))} \log \frac{p_\vartheta(\Phi(\mathbf{x})|\mathbf{z}) p_\lambda(\mathbf{z}|\mathbf{y})}{q_\varphi(\mathbf{z}|\Phi(\mathbf{x}))}, \tag{5.3}$$

where $q_\varphi(\mathbf{z}|\Phi(\mathbf{x}))$, $p_\vartheta(\Phi(\mathbf{x})|\mathbf{z})$ and $p_\lambda(\mathbf{z}|\mathbf{y})$ represent the encoder, decoder and mixture prior component, the parameters of the VAE are given as $\mathbf{\Theta} = \{\vartheta, \varphi, \lambda_1, \ldots, \lambda_K\}$. The parameters of the mixture prior are learnt alongside the parameters of the encoder and decoder (Tomczak and Welling, 2018a). The use of this mixture prior forces the aggregate posterior for *each* class to match a Gaussian distribution, i.e. $\sum_{\mathbf{x} \in \mathcal{X}_k} q(\mathbf{z}|\Phi(\mathbf{x})) \approx \mathcal{N}(\mathbf{z}; \mu_k, \sigma_k)$. This encourages the representations for each class to cluster around a known distribution $p_\lambda(\mathbf{z}|\mathbf{y})$, which we will use to obtain a pseudo likelihood to predict the temperature value. The choice of the VAE was in part down to the motivation that samples which contribute significantly to the ECE will have lower latent-likelihood but also because empirically we found it worked well in practice. Before outlining the details of the approach in the next two subsections, we first provide evidence of this empirical motivation to use a VAE.

We now perform a preliminary experiment, which serves to investigate which samples in the latent representation contribute the most to the calibration error. Specifically, we construct a t-SNE plot for each class of CIFAR-10 but colour code the points depending on their per-data-point contributions to the calibration error. This provides a visual method for us to inspect where samples which harm calibration are placed, which can be seen in Fig. 5.3. Here we can see that data-points which

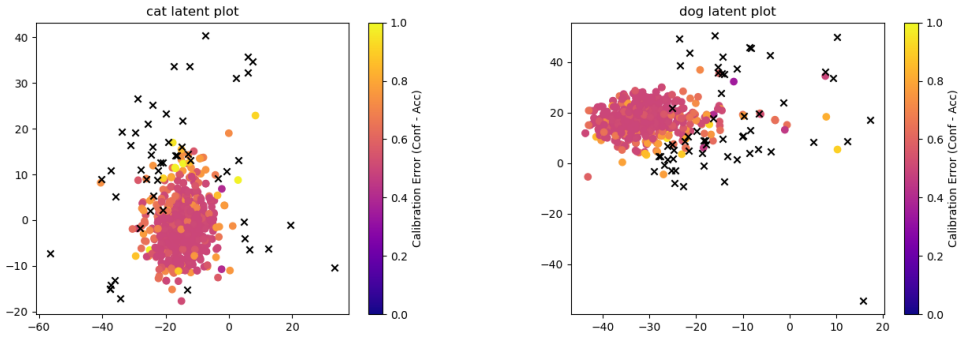



**Figure 5.3:** t-SNE plot for classes `cat` and `dog`, colour indicates per data-point contribution to ECE, 0.5 indicates no contribution. Generally, samples with little contribution to calibration error (pink) are placed around the centre of the cluster, unlike samples with a high contribution (yellow and orange) which are placed near the edges. Furthermore, incorrect samples (black cross) are placed significantly far away from the cluster centre.

do not contribute to the calibration error tend to be placed near the centre of the cluster, and ones which do, or are incorrect, indicated by a black cross, are placed far from the centre. This highlights that the VAE is able, to some extent, to provide a basis to predict the temperature, we then utilise this representation to predict $T$ through a simple Multi Layer Perceptron, rather than predicting $T$ directly from the latent space, which we found not to work in practice. Other approaches such as performing Linear Discriminant Analysis on the feature space could be considered, but we chose not to perform these experiments due to the complexity of the feature space.

### 5.3.1 Temperature Prediction Network

Given that the VAE is able to encapsulate the information needed for confidence prediction, we learn a very simple MLP parameterised by $\theta$, which predicts the temperature based on the latent embeddings, using the cross entropy loss as an objective. Rather than using the latent samples as input to the MLP, given the observations in Figure 5.3, we choose to predict the temperature as a function of the vector of log-likelihoods on *all* of the conditional priors, specifically $T = g_\theta(\tilde{\boldsymbol{q}})$ where $g : \mathbb{R}^K \to \mathbb{R}$ is the MLP which predicts the temperature and $\tilde{\boldsymbol{q}} = \{\log p_\lambda(\mathbf{z}|\mathbf{y})|\forall \mathbf{y}\}$, i.e each element $\tilde{\boldsymbol{q}}_i$ contains the log-likelihood of $\mathbf{z}$ on the corresponding conditional

prior $p_\lambda(\mathbf{z}|\mathbf{y}=i)$. Evaluating $\log p_\lambda(\mathbf{z}|\mathbf{y})$ can be viewed as a pseudo likelihood of $\mathbf{x}$, consequently the module predicts the temperature as a non-linear transform of a pseudo-likelihood of the sample. It is also important to point out that due to the use of feature space as the input, we able to use small architectures, making this approach very fast during training and at test time. We represent a high level overview and the graphical model in Figure 5.4.

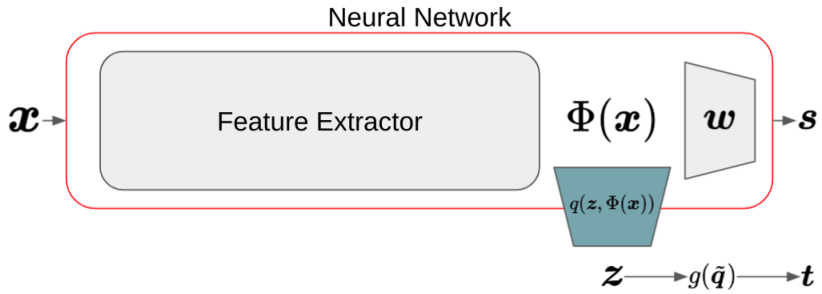



**Figure 5.4:** High level architecture. The off shelf neural-network is represented by the red box, where the parameters are left unchanged, the learnable VAE encoder is indicated by $q(\mathbf{z}|\Phi(\mathbf{x}))$, with the $g_\theta(\tilde{\boldsymbol{q}})$ as the MLP predicting $T$.

### 5.3.2 Calibrated Training Details

The overall post-hoc learning algorithm is very simple and the module can be trained in under a minute on an 8Gb Titan Xp for most datasets, depending on the validation set and feature space size, we give an overview of the procedure in algorithm 1. We combine learning the VAE and the temperature prediction network into one objective. Specifically, we maximise the following objective

$$\mathcal{L}(\mathbf{x}, \mathbf{y}) = \mathbb{ELBO}[\Phi(\mathbf{x})] + \log \text{Cat}(\mathbf{y} \mid \text{softmax}(\mathbf{s}/g_\theta(\tilde{\boldsymbol{q}})) \tag{5.4}$$

with $\tilde{\boldsymbol{q}} = \{\log p_\lambda(\mathbf{z}|\mathbf{y})|\forall \mathbf{y}\}$ $\mathbf{z} \sim q_\phi(\mathbf{z} \mid \mathbf{x})$ and using Normal distributions for $\mathbf{z}$; Laplace distribution over $\Phi(\mathbf{x})$ (L1 loss); and a Categorical over $\mathbf{y}$. To train the VAE, we use a held-out dataset from training the network, i.e. $\mathcal{X}_{train} \cap \mathcal{X}_{cal} = \varnothing$. We used the Adam optimiser with a learning rate of 0.001 and trained for 50 epochs. This is an additional benefit of training the VAE on the $\Phi(\mathbf{x})$, as the feature space has a lower dimensionallity and simpler structure than the image space, leading to much faster training and the ability to use simpler networks.

**Algorithm 1** Learning Adaptive Temperature

**Require:** $\mathcal{X}_{cal}, \mathcal{Y}_{cal}, \mathcal{P}_{cal}$
  **while** not converged **do**
    $\mathbf{x}, \mathbf{y} \leftarrow$ Random batch
    $\nabla_{VAE} \leftarrow \nabla \mathbb{ELBO}[\Phi(\mathbf{x})]$ ▷ Standard VAE gradient
    $\tilde{\boldsymbol{q}} = \{\log p_\lambda(\mathbf{z}|\mathbf{y})|\forall \mathbf{y}\} \quad \mathbf{z} \sim q(\mathbf{z}|\Phi(\mathbf{x}))$ ▷ Vector of pseudo likelihoods
    $\nabla_T \leftarrow \nabla \log \mathrm{Cat}(\mathbf{y}; \mathrm{softmax}(\mathbf{s}/g_\theta(\tilde{\boldsymbol{q}}))$ ▷ CE loss for temperature
    $\{\boldsymbol{\Theta}, \theta\}_{t+1} \leftarrow \{\boldsymbol{\Theta}, \theta\}_t - \alpha(\nabla_{VAE} + \nabla_T)$ ▷ Update parameters $\{\boldsymbol{\Theta}, \theta\}$
  **end while**

**Calibration at Test Time** During test time, the features of the data point $\Phi(\mathbf{x})$ and predicted logits are computed from the classifier $\mathbf{s} = f(\Phi(\mathbf{x}))$, the temperature can then predicted through $T = g_\theta(\tilde{\boldsymbol{q}})$. The calibrated predictions are then computed as $\boldsymbol{p} = \sigma(\mathbf{s}/T)$.

## 5.4 Related Work

**Uncertainty Estimation** In deep learning, the most typical way to address uncertainty estimation is to make the networks output a distribution, and to extract an uncertainty measure as a function of the predictive distribution. Bayesian approaches define a prior distribution over the weights of the network and apply inference techniques to update such distributions given the training set. Given the intractability of exact inference for neural networks, several approximate variational inference schemes have been proposed (Gal and Ghahramani, 2016; Blundell et al., 2015; Kingma et al., 2015; Welling and Teh, 2011; Kim et al., 2020; Chen et al., 2014). Recent literature tries to combine the benefits of Bayesian deep learning with the training of deterministic neural networks trained via standard optimisation algorithms. Some methodologies suggest using a Laplace approximation of a trained network to approximate a Gaussian using a Laplace approximation around the optimal parameters (Ritter et al., 2018; Kristiadi et al., 2020). Others suggest replacing the head of the network with a Gaussian Process (Liu et al., 2020) or a head parametrising a Dirichlet distribution (Malinin and Gales, 2018; Joo et al., 2020), or just performing Bayesian inference on the final layer (Riquelme et al., 2018). Another family of models leverages ensembles (Lakshminarayanan et al., 2016) to output distributions. Given the extreme computational and memory

requirements of ensembles, several techniques have been suggested to obtain the ensembling benefits more efficiently (Havasi et al., 2020; Wen et al., 2020)

**Calibration** Deep Neural Networks suffer from overconfident classification scores, Which can be alleviated through temperature scaling in post-processing (Guo et al., 2017)—a modern variant of Platt scaling (Platt et al., 1999). As previously mentioned, this typically comes at the cost of decreasing the confidence in correct predictions (Kumar et al., 2018). Other approaches include histogram binning (Zadrozny and Elkan, 2001); isotonic regression (Zadrozny and Elkan, 2002); and Bayesian binning (Naeini et al., 2015; Naeini and Cooper, 2016). Overconfidence is caused by over-fitting to the cross-entropy loss, which can be alleviated by instead using a focal loss (Lin et al., 2017; Mukhoti et al., 2020). In a similar fashion, Kumar et al. (2018) utilised a differentiable proxy during training to improve calibration. Label smoothing was also shown to improve calibration (Müller et al., 2019). It has also been shown that recomputing the coefficients of batch normalization improves calibration (Nado et al., 2020). Tangentially, Ovadia et al. (2019) performed a large scale comparison of methods under dataset-shift.

## 5.5 Results

Before evaluating the model, we define the hypothesis we are trying to test. Specifically, we want to evaluate if predicting the temperature on a per-data-point basis leads to improved calibration over vanilla temperature scaling. Secondly, we wish to investigate how adaptive temperature performs under dataset shift.

We performed our experiments on WideResNet28-10 (Zagoruyko and Komodakis, 2016) and ResNet50 (He et al., 2016) architectures. We report calibration results on CIFAR10/CIFAR100 (Krizhevsky et al., 2009) and Tiny-ImageNet (Torralba et al., 2008). We conducted distribution-shit experiments using variants CIFAR10-C/CIFAR100-C to test for domain shift (Hendrycks and Dietterich, 2019). We used the following as models for our evaluation:

- Cross Entropy Loss, due to it's popularity and wide adoption.

- Brier Score (Brier et al., 1950), due to it's ability to obtain well calibrated predictions (Mukhoti et al., 2020).
- Deep Ensembles (Lakshminarayanan et al., 2016), as it achieves state of the art results.[5]

Results are obtained for multiple seeds for Cross Entropy and Brier Score, but only one seed for Deep Ensembles due to the number of models needed.

### 5.5.1 Calibration

Here we evaluate how adaptive temperature scaling affects standard calibration metrics compared to vanilla temperature scaling. We report results using the ECE, which divides the probability into equally sized bins and then computes the absolute difference between confidence and accuracy for each bin before taking the average. However, the ECE is known to be a biased estimator of the theoretical probabilistic expectation (Ding et al., 2019), whose performance depends on the binning size and on the distribution of samples in each bin. For this reason, the ECE reliability as a good miscalibration metric is being questioned and several alternatives have been proposed (e.g. (Nixon et al., 2019; Roelofs et al., 2020; Mukhoti et al., 2020)). Among these, we choose to also use the AdaECE (Mukhoti et al., 2020), which uses adaptive bin sizes to ensure each bin contains the same number of samples.

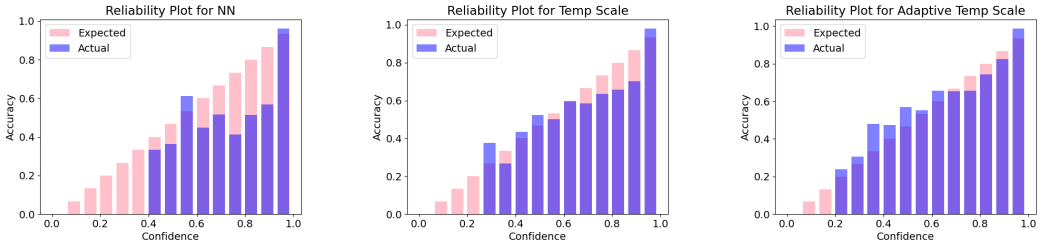



**Figure 5.5:** Reliability plots for: left) vanilla predictions; middle) temperature scaling; right) adaptive temperature scaling (ours). Temperature scaling was optimised through cross validating in the range 0 - 10 and optimised the ECE. CIFAR-10 on ResNet50.

We report the results in Table 5.1 along with reliability plots in Figure 5.5, where it can be seen that adaptive temperature scaling improves calibration compared to

[5]Applying adaptive temperature scaling to deep ensembles does not necessarily preseve accuracy, however as we show in our experiments the difference is negligible.

standard temperate scaling. In all cases our method is able to outperform vanilla temperature scaling, with large improvements obtained when using the cross entropy loss, e.g. $0.93 \rightarrow 0.76$ and $3.76 \rightarrow 2.95$ ECE for CIFAR10 and CIFAR100 when using the WideResNet2810 Network.

| Method | Scaling | Accuracy (↑) | ECE (↓) | AdaECE (↓) | Accuracy (↑) | ECE (↓) | AdaECE (↓) |
|---|---|---|---|---|---|---|---|
| | | **CIFAR10** | | | | | |
| | | **WideResNet2810** | | | **ResNet50** | | |
| CE | None | 95.52 ± 0.43 | 2.15 ± 0.18 | 2.13 ± 0.18 | 93.13 ± 1.97 | 3.75 ± 1.32 | 3.74 ± 1.32 |
| CE | Vanilla TS | 95.52 ± 0.43 | 0.93 ± 0.20 | 0.98 ± 0.30 | 93.13 ± 1.97 | **1.41 ± 0.43** | **1.45 ± 0.44** |
| CE | Adaptive TS | 95.52 ± 0.43 | **0.76 ± 0.07** | **0.86 ± 0.20** | 93.13 ± 1.97 | **1.13 ± 0.60** | **1.09 ± 0.57** |
| Brier | None | 95.84 ± 0.10 | 0.92 ± 0.13 | 1.50 ± 0.16 | 94.59 ± 0.23 | 2.03 ± 0.13 | 2.27 ± 0.12 |
| Brier | Vanilla TS | 95.84 ± 0.10 | 1.88 ± 0.23 | 1.94 ± 0.19 | 94.59 ± 0.23 | **1.67 ± 0.24** | 2.08 ± 0.30 |
| Brier | Adaptive TS | 95.84 ± 0.10 | **1.65 ± 0.15** | **1.61 ± 0.13** | 94.59 ± 0.23 | **1.61 ± 0.40** | **1.53 ± 0.44** |
| Ensmbls | None | 96.35 | 1.68 | 1.61 | 95.62 | 1.92 | 1.89 |
| Ensmbls | Vanilla TS | 96.35 | 0.61 | 0.68 | 95.62 | 0.93 | 0.84 |
| Ensmbls | Adaptive TS | 96.37 | **0.51** | **0.46** | 95.64 | **0.60** | **0.58** |
| | | **CIFAR100** | | | | | |
| | | **WideResNet2810** | | | **ResNet50** | | |
| CE | None | 80.71 ± 0.17 | 5.76 ± 0.16 | 5.70 ± 0.16 | 77.91 ± 0.33 | 9.39 ± 0.42 | 9.37 ± 0.43 |
| CE | Vanilla TS | 80.71 ± 0.17 | 3.76 ± 0.29 | 3.68 ± 0.28 | 77.91 ± 0.33 | **3.63 ± 0.21** | **3.61 ± 0.26** |
| CE | Adaptive TS | 80.71 ± 0.17 | **2.95 ± 0.41** | **2.90 ± 0.47** | 77.99 ± 0.33 | **3.30 ± 0.50** | **3.32 ± 0.47** |
| Brier | None | 79.25 ± 0.14 | 4.19 ± 0.24 | 4.13 ± 0.22 | 76.03 ± 0.55 | 4.15 ± 0.22 | 4.04 ± 0.25 |
| Brier | Vanilla TS | 79.25 ± 0.14 | **3.87 ± 0.62** | **3.90 ± 0.62** | 76.03 ± 0.55 | **3.34 ± 0.46** | **3.41 ± 0.39** |
| Brier | Adaptive TS | 79.25 ± 0.14 | **3.67 ± 0.82** | **3.64 ± 0.74** | 76.03 ± 0.55 | **3.30 ± 0.50** | **3.32 ± 0.47** |
| Ensmbls | None | 83.19 | 4.24 | 4.21 | 80.90 | 6.59 | 6.29 |
| Ensmbls | Vanilla TS | 83.18 | 3.71 | 3.55 | 80.90 | 3.22 | 3.16 |
| Ensmbls | Adaptive TS | 83.22 | **2.95** | **2.66** | 80.86 | **2.79** | **2.77** |
| | | **Tiny-ImageNet** | | | | | |
| | | **WideResNet2810** | | | **ResNet50** | | |
| CE | None | 60.47 ± 0.17 | 7.54 ± 4.00 | 7.53 ± 4.09 | 55.27 ± 2.19 | 8.92 ± 2.72 | 8.93 ± 2.74 |
| CE | Vanilla TS | 60.47 ± 1.06 | 6.28 ± 2.43 | 6.15 ± 2.47 | 55.27 ± 2.19 | 7.64 ± 1.53 | 7.57 ± 1.58 |
| CE | Adaptive TS | 60.04 ± 1.20 | **5.18 ± 1.40** | **5.17 ± 1.32** | 55.27 ± 2.19 | **4.51 ± 1.76** | **4.45 ± 1.78** |
| Brier | None | 50.23 ± 0.45 | 5.56 ± 0.63 | 5.52 ± 0.64 | 42.38 ± 1.21 | 5.33 ± 1.47 | 5.37 ± 1.47 |
| Brier | Vanilla TS | 50.23 ± 0.45 | **4.55 ± 0.28** | **4.43 ± 0.63** | 42.38 ± 1.21 | 3.08 ± 0.59 | 3.12 ± 0.55 |
| Brier | Adaptive TS | 50.23 ± 0.45 | **4.43 ± 0.47** | **4.21 ± 0.51** | 42.38 ± 1.21 | **2.71 ± 0.08** | **2.60 ± 0.23** |
| Ensmbls | None | 66.16 | 6.21 | 6.19 | 61.90 | 8.89 | 9.00 |
| Ensmbls | Vanilla TS | 66.16 | 5.12 | 5.06 | 61.90 | 4.29 | 4.43 |
| Ensmbls | Adaptive TS | 66.00 | **4.58** | **4.41** | 61.76 | **4.26** | **4.17** |

**Table 5.1:** Calibration results, here we can see that adaptive temperate scaling is able to improve calibration on a variety of models. Bold indicates best results, or with in one standard deviaiton of best results.

### Data-Shift

A key hypothesis we want to test is how adaptive temperature scaling behaves under data-shift. Specifically, we use the widely used CIFAR10-C and CIFAR100-C datasets, which are corrupted versions of the CIFAR10 and CIFAR100 (Hendrycks and Dietterich, 2019). The dataset consists of standard CIFAR images which have

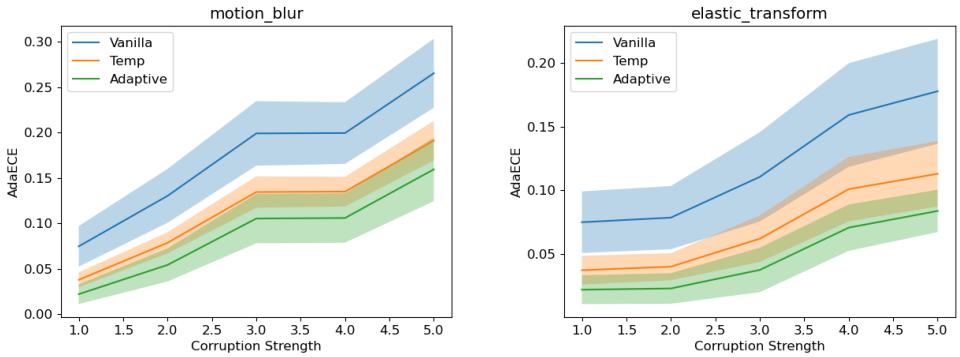



**Figure 5.6:** How AdaECE changes with varying levels of `motion-blur` (left) and `elastic transform` (right) corruptions. Adaptive temperature consistently produces lower error rates. CIFAR10-C on ResNet50.

undergone 15 synthetic corruptions (e.g. noise, weather conditions, image properties) at varying levels. Within this scenario, the classifier should either be robust to such corruptions (retaining accuracy) or if the accuracy is compromised, reduce the confidences accordingly. As such, we report the test accuracy as well as ECE and AdaECE in Table 5.2, where adaptive temperature scaling shows improvements over temperature scaling.

We also expect to see adaptive temperature scaling provide improvement over temperature scaling as the intensity of corruptions are increased for CIFAR-10-C. We generate plots highlighting the AdaECE calibration metric as the level of the corruption intensity is increased; the plot for `motion-blur` is displayed in Figure 5.6. Here despite a general increase in error for all methods adaptive temperature scaling consistently produces lower error rates than vanilla temperature scaling (orange) and vanilla predictions (blue). More examples are in Appendix C.4.

**Temperature Variation Along Feature Interpolations** If the temperature module is successfully able to predict a high temperature in uncertain regions, then we should see a change in the temperature as we traverse the feature the space. To conduct this experiment, we obtain the average feature representation for each class $\boldsymbol{\phi}_k = \frac{1}{|\mathcal{X}_k|} \sum_{\mathbf{x} \in \mathcal{X}_k} \Phi(\mathbf{x})$ and measure the temperature when interpolating between two classes. i.e. we predict the temperature for the features $\{\alpha \boldsymbol{\phi}_{k^{(i)}} + (1-\alpha)\boldsymbol{\phi}_{k^{(j)}}\} \quad \alpha \in [0, 1]$. We plot the interpolation results in Figure 5.7 (Left) for the classes in

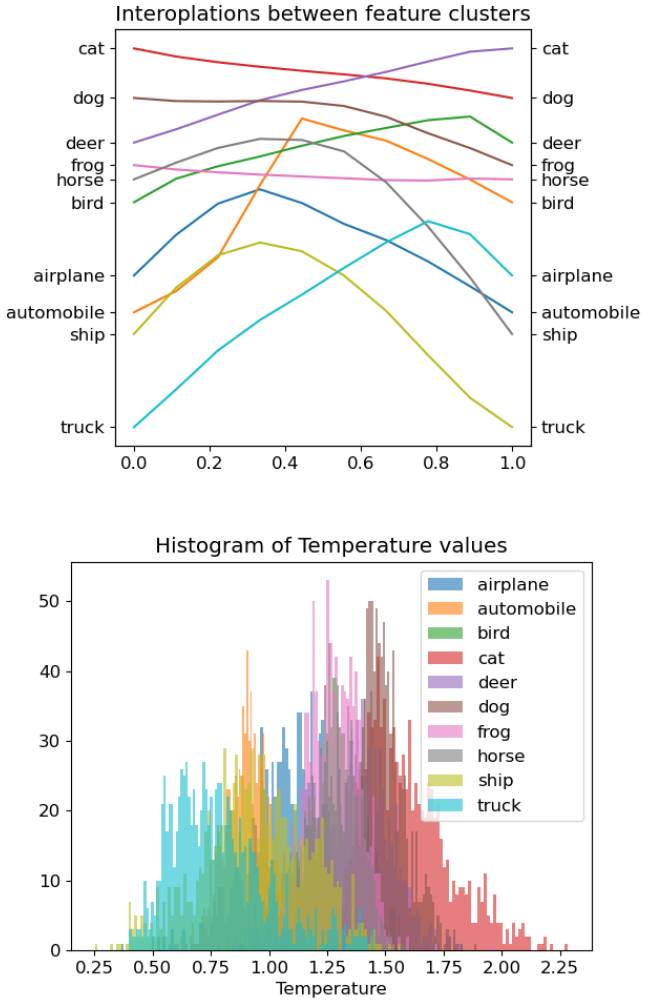



**Figure 5.7:** Top: How temperature varies when interpolating between class feature means. Here we can see that temperature increases between classes or remains high for classes who's embeddings are close together. Pairs were chosen to improve visual clarity. Dataset: CIFAR-10; architecture: ResNet50. Bottom: Histogram of temperature values for each image in CIFAR-10, here we can see that typically objects have a lower temperature than animals, indicating they are easier to classify. Dataset: CIFAR-10; architecture: ResNet50.

CIFAR-10, where the horizontal axis represent $\alpha$ and the vertical axis represents the temperature. For some classes we see a significant rise in the temperature as we interpolate between two classes, e.g. `automobile` and `bird`. This highlights the temperature prediction models ability to assign a low temperature in regions that the classifier is certain about, e.g. around the mean and a higher temperature in less certain regions, e.g. near a decision boundary.

| Method | Scaling | Accuracy (↑) | ECE (↓) | AdaECE (↓) | Accuracy (↑) | ECE (↓) | AdaECE (↓) |
|---|---|---|---|---|---|---|---|
| | | **CIFAR10-C** | | | | | |
| | | **WideResNet2810** | | | **ResNet50** | | |
| CE | None | 75.07 ± 1.46 | 15.70 ± 1.14 | 15.68 ± 1.14 | 71.45 ± 2.96 | 18.48 ± 1.70 | 18.47 ± 1.70 |
| CE | Vanilla TS | 75.07 ± 1.46 | **12.19 ± 0.91** | **12.17 ± 0.91** | 71.45 ± 2.96 | 12.72 ± 0.64 | 12.70 ± 0.63 |
| CE | Adaptive TS | 75.07 ± 1.46 | **12.03 ± 1.31** | **12.02 ± 1.31** | 71.45 ± 2.96 | **10.86 ± 1.88** | **10.83 ± 1.87** |
| Brier | None | 75.27 ± 0.73 | 16.21 ± 0.80 | 16.45 ± 0.78 | 74.19 ± 0.28 | 15.34 ± 0.63 | 15.34 ± 0.65 |
| Brier | Vanilla TS | 75.27 ± 0.73 | 15.87 ± 0.46 | 15.86 ± 0.46 | 74.19 ± 0.28 | 14.66 ± 0.83 | 14.67 ± 0.86 |
| Brier | Adaptive TS | 75.27 ± 0.73 | **14.84 ± 0.88** | **14.81 ± 0.89** | 74.19 ± 0.28 | **13.39 ± 1.18** | **13.35 ± 1.18** |
| Ensmbls | None | 77.28 | 13.45 | 13.43 | 74.84 | 13.95 | 13.93 |
| Ensmbls | Vanilla TS | 77.28 | 10.12 | 10.09 | 74.84 | 10.37 | 10.33 |
| Ensmbls | Adaptive TS | 77.21 | **9.29** | **9.25** | 74.80 | **9.15** | **9.12** |
| | | **CIFAR100-C** | | | | | |
| | | **WideResNet2810** | | | **ResNet50** | | |
| CE | None | 51.74 ± 0.39 | 18.63 ± 0.70 | 18.58 ± 0.70 | 49.67 ± 0.28 | 24.27 ± 0.89 | 24.25 ± 0.90 |
| CE | Vanilla TS | 51.74 ± 0.39 | 12.28 ± 1.14 | **12.25 ± 1.14** | 49.67 ± 0.28 | **11.78 ± 0.91** | **11.76 ± 0.91** |
| CE | Adaptive TS | 51.74 ± 0.39 | **12.17** ± 0.10 | **12.15 ± 0.11** | 49.72 ± 0.29 | **11.69 ± 0.74** | **11.67 ± 0.71** |
| Brier | None | 50.58 ± 0.28 | 15.04 ± 1.36 | 15.02 ± 1.36 | 48.14 ± 0.83 | 13.43 ± 1.06 | 13.41 ± 1.06 |
| Brier | Vanilla TS | 50.58 ± 0.28 | **9.81 ± 0.84** | **9.81 ± 0.85** | 48.14 ± 0.83 | 10.12 ± 0.67 | 10.10 ± 0.67 |
| Brier | Adaptive TS | 50.58 ± 0.28 | **9.56 ± 0.82** | **9.64 ± 0.74** | 48.62 ± 0.55 | **8.83 ± 0.48** | **8.86 ± 0.48** |
| Ensmbls | None | 54.61 | 14.81 | 14.78 | 52.94 | 19.12 | 19.07 |
| Ensmbls | Vanilla TS | 54.61 | 12.66 | 12.62 | 52.94 | 11.36 | 11.33 |
| Ensmbls | Adaptive TS | 54.61 | **12.02** | **12.00** | 53.91 | **9.15** | **9.15** |

**Table 5.2:** Corrupted calibration results. Here we can see that adaptive temperate scaling is able to improve calibration on a variety of models. Bold indicates best results, or within one standard deviation of best results.

Interestingly, this feature is not present for all class pairs; for some, e.g. `cat` and `dog`, where the temperature remains high between classes. We hypothesise that this is due to an interpolation between these classes being a plausible realisation of an image, unlike for `automobile` and `bird`.

We further show a histogram in Figure 5.7 (Right), of the temperature values for each sample in CIFAR-10 and colour code according to class. Again we see a similar pattern where the animal based classes typically have a higher temperature than the objects, indicating that the network should be more uncertain. This higher temperature is obtained from the VAE learning that samples in this region are often incorrect, which is where the pressure comes from to increase the temperature.

### 5.5.2 Misclassification Rejection

Calibrated uncertainty estimates should render that the models are able to reject samples in order to preserve the accuracy. In this setting we report results for AURRA, which computes the area under the rejection ratio curve(Nadeem

| Methods | AURRA-C (↑) | AURRA-DS (↑) | AURRA-E (↑) | AURRA-C (↑) | AURRA-DS (↑) | AURRA-E (↑) |
|---|---|---|---|---|---|---|
| | WideResNet2810 | | | ResNet50 | | |
| | CIFAR-100 | | | | | |
| None | 93.07 ± 4.28 | 91.84 ± 5.24 | 92.95 ± 4.23 | 93.96 ± 0.15 | 92.94 ± 0.21 | 93.87 ± 0.16 |
| Vanilla TS | 92.97 ± 4.25 | 91.70 ± 5.40 | 92.67 ± 4.41 | 93.62 ± 0.06 | 92.68 ± 0.26 | 93.35 ± 0.11 |
| Adaptive TS | **93.20 ± 4.25** | **92.00 ± 5.39** | **92.99 ± 4.35** | **94.03 ± 0.18** | **93.18 ± 0.19** | **93.84 ± 0.19** |
| | Tiny-ImageNet | | | | | |
| None | 84.21 ± 1.09 | 81.83 ± 0.64 | 83.69 ± 1.16 | 79.84 ± 2.10 | 76.71 ± 1.64 | 79.35 ± 2.25 |
| Vanilla TS | 84.05 ± 1.09 | 81.63 ± 0.64 | 83.31 ± 1.11 | 79.58 ± 2.06 | 76.27 ± 1.56 | 78.59 ± 2.07 |
| Adaptive TS | **84.68 ± 0.18** | **81.93 ± 0.28** | **84.09 ± 0.20** | **81.26 ± 0.49** | **77.60 ± 0.50** | **80.44 ± 0.48** |

**Table 5.3:** AURRA scores for based on: confidence (**AURRA-C**), Demster-Schafer (Sensoy et al., 2018) (**AURRA-DS**) and entropy (**AURRA-E**). Unlike temperature scaling, adaptive temperature scaling does not suffer a reduction in rejection ability.

et al., 2009). We display the results in Table 5.3, where we see that temperature scaling provides a slight improvement over normal predictions and also vanilla temperature scaling. Furthermore, we would like to highlight that even though vanilla temperature scaling improves calibration, it does so at the expense of being able to reject samples; unlike adaptive temperature scaling which is able to provide the best of both worlds. It is important to stress that this is a significant advantage, as we are able to provide better calibrated predictions whilst also increasing the models ability to reject samples.

### 5.5.3 Evaluating Hardness

Given our models ability to predict the temperature, it should naturally extract a notion of hardness, that is how difficult is it to classify. One would expect hard samples to have a high temperature and easy ones to have a low temperature. To conduct this experiment, we utilise the CIFAR-10.1 Recht et al. (2018); Torralba et al. (2008) datset, which contains "harder", but statistically similar images to CIFAR-10; conseqently this experiments is not examining data-shift, but is instead measuring the performance on challenging samples. We report the standard metrics: accuracy, ECE and AdaECE in Table 5.4, where we see that adaptive temperature is able to obtain a lower calibration error than vanilla temperature scaling for both ResNet50 and WideResNet28-10 when trained using cross entropy loss.

A key hypothesis we wish to test is "does the model assign higher temperatures to harder samples?"; harder samples should naturally contain a greater amount of

| Methods | Accuracy (↑) | ECE (↓) | AdaECE (↓) | Accuracy (↑) | ECE (↓) | AdaECE (↓) |
|---|---|---|---|---|---|---|
| None | 85.86 ± 2.48 | 8.35 ± 1.58 | 8.15 ± 1.68 | 89.55 ± 0.81 | 5.66 ± 0.28 | 5.56 ± 0.32 |
| Vanilla TS | 85.86 ± 2.48 | 4.57 ± 0.32 | 4.24 ± 0.43 | 89.55 ± 0.81 | 3.64 ± 0.25 | 3.38 ± 0.35 |
| Adptive TS | 85.86 ± 2.48 | **3.67 ± 1.41** | **3.35 ± 1.39** | 89.55 ± 0.81 | **3.53 ± 0.22** | **3.35 ± 0.20** |

**Table 5.4:** CIFAR-10.1 Results for ResNet50 and WideResNet28-10, here we see that adaptive temperature scaling is able to provide slightly improved calibration on the harder CIFAR-10.1 dataset.

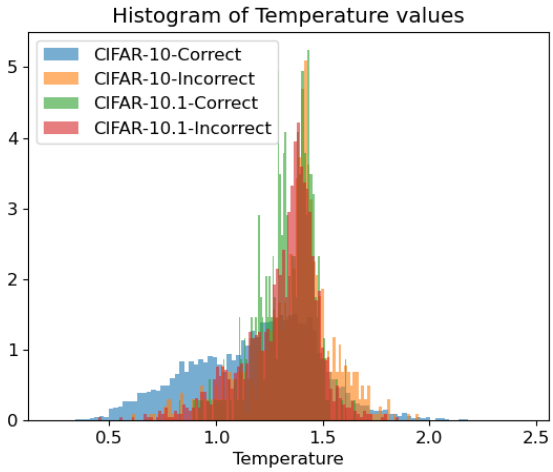



**Figure 5.8:** Histograms of temperature for correct predictions for CIFAR-10 and CIFAR-10.1 on ResNet50. Lower temperatures are typically assigned to correct (blue) samples from CIFAR-10 but higher for incorrect samples (orange). We also see that hard samples are assigned higher values, regardless of whether they are correct or not (red and green) for CIFAR-10.1.

uncertainty in their predictions. Consequently, we should see higher temperature values assigned to harder samples (CIFAR-10.1) than to easier ones (CIFAR-10). We test this hypothesis by plotting the histogram of temperature values for CIFAR-10 and CIFAR-10.1, for both correct and incorrect predictions in Figure 5.8 (Right).

Here we see that generally, correct samples for CIFAR-10 (blue) are assigned a lower temperature than for CIFAR-10.1 (green), indicating that the adaptive temperature is able to recognise harder samples and assign a higher temperature increasing the uncertainty. Furthermore, we also see that adaptive temperature predicts higher temperatures for incorrect predictions for CIFAR-10 (orange), highlighting adaptive temperatures ability to reduce the confidence of samples which are likely to be incorrect. Interestingly, the same is not true for CIFAR-10.1, this is due to the

fact that the samples from CIFAR-10.1 are by design harder, adaptive temperature predicts higher values of $T$ than for the easier CIFAR10 counterpart.

## 5.6 Discussion

Here we have presented a novel post-hoc method for predicting the temperature of the softmax distribution in neural network classification. Given a data-point, our method is able predict how confident the classifier should be about its prediction, improving the calibration error, furthermore, adaptive temperature is also able to obtain better results under distribution shifts. This is achieved by leveraging the latent space of a VAE, which we found to naturally encapsulate and structure the information relating to confidence appropriately. As the model is applied post-hoc, training is very fast, requiring little computational overhead, furthermore it is very easy to implement.

# 6 Conclusion

Here we summarise and highlight the contributions made in this thesis, before providing a critical discussion on these contributions.

## 6.1 Summary

Variational Autoencoders (VAEs) offer a principled way to learn representations of data. In this thesis, we have explored the utility of such representations when incorporating additional information such as labels or different modalities. Specifically, we have explored the following problems: utilising partially observed label information to capture the characteristics of an image and disentangle them within the latent space (Chapter 3); how shared representations of two modalities can be learned in the situation where one of the modalities may not always be present (Chapter 4); and how the latent space of the VAE can be used to obtain reliable confidence estimates in a neural-classifier (Chapter 5). We will first outline these contributions in more detail and the subsequently provide a discussion on their limitations.

In Chapter 3, we presented *Characteristic Capturing* VAE (CCVAE); a semi-supervised VAE which encapsulated an image's characteristics in the latent space and structured it accordingly utilising the label information. Specifically, by placing

a classifier on the latent space with an appropriate inductive bias, the generative factors within the image are forced to align themselves with the axes of the latent space. This forces the information associated with a label to be encapsulated within a given latent dimension, allowing us to perform additional tasks such as latent traversals and fine grained manipulation. Moreover, this functionality can be learned in the semi-supervised case, where the labels are not present for a vast majority of the dataset. In practice we found that good performance can be obtained, even at very low supervision rates where only 0.4% of the data is labelled.

In Chapter 4, we extended the motivation to learn VAEs with label information to the case where the additional data source represents another modality of the same underlying object, which typically contains unstructured and higher dimensional data. In this work the focus was on learning shared representation, allowing the model to perform tasks such as cross-generation—mapping from one modality to the other whilst preserving the underlying class. Moreover, we wanted to achieve this in the partially-observed setting, where one of the modalities may be missing during training. To achieve this, we repurposed CCVAE by reformulating it as a bi-directional objective; which ensures that the resulting graphical model is symmetric. Another significant advantage of this approach is the ability to extract relatedness between two samples.

In Chapter 5, we explored the utility of VAEs in calibrating a neural-classifiers predictions, which are known to be generally overconfident. In this setting, we utilised the latent space of the VAE to obtain temperature values for the resulting softmax predictive probability distribution. We found that empirically, the latent space of the VAE is able to structure samples such that those which contribute significantly to the Expected Calibration Error (ECE) are placed in lower likelihood regions of the latent space than those which do not. Based on this empirical finding, we used the likelihoods as inputs to predict the temperature value for the predictive probabilities. This allows the model to assign high confidences to samples that typically do not contribute to ECE and low values to ones that contribute

significantly. Adaptive Temperature Scaling is a post-hoc method and can be applied to almost any pre-existing neural-classifier.

## 6.2 Discussion

In this section we present a critical discussion of the methods presented in this thesis.

### 6.2.1 Capturing Label Characteristics in VAEs

One of the major limitation of CCVAE is that the characteristics are forced to be captured in a single latent dimension. This clearly limits to models ability to represent the characteristics, and could subsequently lead to poorer representations or entangle the characteristics. Moreover, there may be more than one generative feature associated with a label; an example of this could be `smiling`, where the showing of the teeth does not necessarily mean that the person is smiling to a greater extent than someone who has their mouth closed. Whilst it is feasible to use more than one dimension, it is not immediately obvious what inductive bias should be placed on the classifier.

Another issue that is somewhat linked to the aforementioned issue, is how to deal with the situation where some of the labels follow a Categorical distribution. So far, we have assumed that the labels are independent and can be modelled by a Bernoulli distribution; which for some of the examples is not a valid assumption to make. An example of this scenario could be describing facial hair, where it is implausible for someone to have a `beard` and a `5 o'clock shadow` (stubble). Dealing with this issue again requires us to think careful about the inductive-bias and how to place it into the model.

A further issue is the case when the model is completely unsupervised and no label information is present. Empirically we found the model to completely fail, which is hardly a surprising result.

### 6.2.2 Learning Multimodal VAEs through mutual supervision

One clear problem is the fidelity of the resulting cross-generations, which tend to be poor compared to other multimodal method which are based on transformer architectures . Fidelity was not a primary focus for this work and instead we used simpler datasets for the purpose of exposition. Generating high fidelity images is of course an important goal and a highly sought after feature in models; it would be interesting to see how this approach could be used in much larger models such as Child (2021).

Another issue with this approach is the training time, which typically takes 5 times longer to train than it's unimodal counter part. Whilst this is not a major deficiency, it is still an undesirable outcome in the model and if possible should be reduced.

### 6.2.3 Sample-dependent Temperature Scaling for Improved Calibration

With any deep learning model, there is always an inherent amount of variation in the resulting parameters due to stochasticity in the training regime. Unfortunately, this uncertainty propagates into Adaptive Temperature Scaling, resulting in different calibration errors for the same neural-classifier. This can be alleviated by training multiple temperature prediction networks and choosing the best ones on the validation set. We opted not to do this, due to the additional computational burden and the impact on the time taken to calibrate the model.

Another further issue is comes with the need for additional data. The calibration phase requires additional data to train temperature prediction model which was *not* present in the training set. This potentially limits the ability to apply this method to off-the-shelf classifiers. However, we would also like to note that this is an issue which also plagues most other post-hoc calibration methods.

# Bibliography

T. Adel, Z. Ghahramani, and A. Weller. Discovering interpretable representations for both deep generative and discriminative models. In *International Conference on Machine Learning*, pages 50–59, 2018.

S. K. Ainsworth, N. J. Foti, A. K. Lee, and E. B. Fox. oi-vae: Output interpretable vaes for nonlinear group factor analysis. In *International Conference on Machine Learning*, pages 119–128. PMLR, 2018.

M. I. Bauer and P. N. Johnson-Laird. How diagrams can improve reasoning. *Psychological science*, 4(6):372–378, 1993.

Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(8):1798–1828, Aug. 2013a. ISSN 0162-8828.

Y. Bengio, L. Yao, G. Alain, and P. Vincent. Generalized denoising auto-encoders as generative models. *Advances in neural information processing systems*, 26, 2013b.

R. v. d. Berg, L. Hasenclever, J. M. Tomczak, and M. Welling. Sylvester normalizing flows for variational inference. *arXiv preprint arXiv:1803.05649*, 2018.

C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra. Weight uncertainty in neural networks. May 2015.

P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146, 2017.

J. Bornschein and Y. Bengio. Reweighted wake-sleep. *arXiv preprint arXiv:1406.2751*, 2014.

S. R. Bowman, L. Vilnis, O. Vinyals, A. M. Dai, R. Jozefowicz, and S. Bengio. Generating sentences from a continuous space. *arXiv preprint arXiv:1511.06349*, 2015.

G. W. Brier et al. Verification of forecasts expressed in terms of probability. *Monthly weather review*, 78(1):1–3, 1950.

R. A. Brooks. Intelligence without representation. *Artificial intelligence*, 47(1-3): 139–159, 1991.

Y. Burda, R. Grosse, and R. Salakhutdinov. Importance weighted autoencoders. *arXiv preprint arXiv:1509.00519*, 2015.

T. Chen, E. Fox, and C. Guestrin. Stochastic gradient hamiltonian monte carlo. In E. P. Xing and T. Jebara, editors, *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 1683–1691, Bejing, China, 22–24 Jun 2014. PMLR. URL `http://proceedings.mlr.press/v32/cheni14.html`.

X. Chen, D. P. Kingma, T. Salimans, Y. Duan, P. Dhariwal, J. Schulman, I. Sutskever, and P. Abbeel. Variational lossy autoencoder. *arXiv preprint arXiv:1611.02731*, 2016.

R. Child. Very deep {vae}s generalize autoregressive models and can outperform them on images. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=RLRXCV6DbEJ`.

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

T. G. Dietterich. Ensemble methods in machine learning. In *International workshop on multiple classifier systems*, pages 1–15. Springer, 2000.

Y. Ding, J. Liu, J. Xiong, and Y. Shi. Revisiting the evaluation of uncertainty estimation and its application to explore model Complexity-Uncertainty Trade-Off. Mar. 2019.

E. Dupont. Learning disentangled joint continuous and discrete representations. In *Advances in Neural Information Processing Systems*, pages 710–720, 2018.

C. Durkan, A. Bekasov, I. Murray, and G. Papamakarios. Neural spline flows. *Advances in neural information processing systems*, 32, 2019.

J. E. Fan, D. Yamins, and N. B. Turk-Browne. Common object representations for visual recognition and production. In *CogSci*, 2015.

P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE transactions on pattern analysis and machine intelligence*, 32(9):1627–1645, 2010.

Y. Gal. *Uncertainty in deep learning*. PhD thesis, University of Cambridge, 2016. Unpublished doctoral dissertation.

Y. Gal and Z. Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059. PMLR, 2016.

S. Ghalebikesabi, R. Cornish, C. Holmes, and L. J. Kelly. Deep generative missingness pattern-set mixture models. In *AISTATS*, pages 3727–3735, 2021.

C. R. Givens and R. M. Shortt. A class of Wasserstein metrics for probability distributions., 2002. ISSN 0026-2285.

P. Glasserman. *Monte Carlo methods in financial engineering*, volume 53. Springer, 2004.

P. Glasserman and Y.-C. Ho. *Gradient estimation via perturbation analysis*, volume 116. Springer Science & Business Media, 1991.

Y. Gong, H. Hajimirsadeghi, J. He, T. Durand, and G. Mori. Variational selective autoencoder: Learning from partially-observed heterogeneous data. In *AISTATS*, pages 2377–2385, 2021.

I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

W. Grathwohl, R. T. Chen, J. Bettencourt, I. Sutskever, and D. Duvenaud. Ffjord: Free-form continuous dynamics for scalable reversible generative models. *arXiv preprint arXiv:1810.01367*, 2018.

I. Gulrajani, K. Kumar, F. Ahmed, A. A. Taiga, F. Visin, D. Vazquez, and A. Courville. Pixelvae: A latent variable model for natural images. *arXiv preprint arXiv:1611.05013*, 2016.

C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger. On calibration of modern neural networks. In *International Conference on Machine Learning*, pages 1321–1330. PMLR, 2017.

M. Havasi, R. Jenatton, S. Fort, J. Z. Liu, J. Snoek, B. Lakshminarayanan, A. M. Dai, and D. Tran. Training independent subnetworks for robust prediction. *arXiv preprint arXiv:2010.06610*, 2020.

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick. Masked autoencoders are scalable vision learners. *arXiv preprint arXiv:2111.06377*, 2021.

D. Hendrycks and T. Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*, 2019.

M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in neural information processing systems*, pages 6626–6637, 2017.

I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner. beta-VAE: Learning basic visual concepts with a constrained variational framework. In *Proceedings of the International Conference on Learning Representations*, 2016.

I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *ICLR 2017 : International Conference on Learning Representations 2017*, 2017.

G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006.

G. E. Hinton and R. S. Zemel. Autoencoders, minimum description length and helmholtz free energy. In *Advances in neural information processing systems*, pages 3–10, 1994.

G. E. Hinton, P. Dayan, B. J. Frey, and R. M. Neal. The" wake-sleep" algorithm for unsupervised neural networks. *Science*, 268(5214):1158–1161, 1995.

M. D. Hoffman, D. M. Blei, C. Wang, and J. Paisley. Stochastic variational inference. *The Journal of Machine Learning Research*, 14(1):1303–1347, 2013.

C.-W. Huang, A. Touati, L. Dinh, M. Drozdzal, M. Havaei, L. Charlin, and A. Courville. Learnable explicit density for continuous latent space and variational inference. *arXiv preprint arXiv:1710.02248*, 2017.

C.-W. Huang, D. Krueger, A. Lacoste, and A. Courville. Neural autoregressive flows. In *International Conference on Machine Learning*, pages 2078–2087. PMLR, 2018.

M. Ilse, J. M. Tomczak, C. Louizos, and M. Welling. Diva: Domain invariant variational autoencoders. *arXiv preprint arXiv:1905.10427*, 2019.

J. Irvin, P. Rajpurkar, M. Ko, Y. Yu, S. Ciurea-Ilcus, C. Chute, H. Marklund, B. Haghgoo, R. Ball, K. Shpanskaya, et al. Chexpert: A large chest radiograph dataset with uncertainty labels and expert comparison. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 590–597, 2019.

O. Ivanov, M. Figurnov, and D. P. Vetrov. Variational autoencoder with arbitrary conditioning. In *International Conference on Learning Representations*, pages 1–25, 2019.

T. Joo, U. Chung, and M.-G. Seo. Being bayesian about categorical probability. Feb. 2020.

T. Joy, A. Desmaison, T. Ajanthan, R. Bunel, M. Salzmann, P. Kohli, P. H. Torr, and M. P. Kumar. Efficient relaxations for dense CRFs with sparse higher-order potentials. *SIAM journal on imaging sciences*, 12(1):287–318, 2019.

T. Joy, S. M. Schmon, P. H. S. Torr, N. Siddharth, and T. Rainforth. Capturing label characteristics in VAEs. In *International Conference on Learning Representations*, 2021.

T. Joy, Y. Shi, P. H. S. Torr, T. Rainforth, S. M. Schmon, and N. Siddharth. Learning multimodal VAEs through mutual supervision. In *International Conference on Learning Representations*, 2022.

T. Joy, F. Pinto, S. Lim, P. H. S. Torr, and P. K. Dokania. Sample-dependent adaptive temperature scaling for improved calibration. *AAAI Conference on Artificial Intelligence*, 2023.

I. Khemakhem, D. Kingma, R. Monti, and A. Hyvarinen. Variational autoencoders and nonlinear ica: A unifying framework. In *International Conference on Artificial Intelligence and Statistics*, pages 2207–2217. PMLR, 2020.

D. Kiela, H. Firooz, A. Mohan, V. Goswami, A. Singh, P. Ringshia, and D. Testuggine. The hateful memes challenge: Detecting hate speech in multimodal memes. *Advances in Neural Information Processing Systems*, 33:2611–2624, 2020.

H. Kim and A. Mnih. Disentangling by factorising. In *International Conference on Machine Learning*, pages 2649–2658, 2018.

S. Kim, Q. Song, and F. Liang. Stochastic gradient langevin dynamics algorithms with adaptive drifts. Sept. 2020.

D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

D. P. Kingma and M. Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations*, 2014.

D. P. Kingma, D. J. Rezende, S. Mohamed, and M. Welling. Semi-supervised learning with deep generative models. *arXiv preprint arXiv:1406.5298*, 2014.

D. P. Kingma, T. Salimans, and M. Welling. Variational dropout and the local reparameterization trick. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28, pages 2575–2583. Curran Associates, Inc., 2015. URL `https://proceedings.neurips.cc/paper/2015/file/bc7316929fe1545bf0b98d114ee3ecb8-Paper.pdf`.

D. P. Kingma, T. Salimans, R. Jozefowicz, X. Chen, I. Sutskever, and M. Welling. Improving variational inference with inverse autoregressive flow. *arXiv preprint arXiv:1606.04934*, 2016.

A. Kristiadi, M. Hein, and P. Hennig. Being bayesian, even just a bit, fixes overconfidence in ReLU networks. In H. D. III and A. Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 5436–5446. PMLR, 13–18 Jul 2020. URL `http://proceedings.mlr.press/v119/kristiadi20a.html`.

A. Krizhevsky, G. Hinton, et al. Learning multiple layers of features from tiny images. 2009.

A. Kumar, S. Sarawagi, and U. Jain. Trainable calibration measures for neural networks from kernel mean embeddings. In *International Conference on Machine Learning*, pages 2805–2814. PMLR, 2018.

B. M. Lake, R. Salakhutdinov, and J. B. Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015.

B. Lakshminarayanan, A. Pritzel, and C. Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Neural Information Processing System*, 2016.

T. A. Le, A. R. Kosiorek, N. Siddharth, Y. W. Teh, and F. Wood. Revisiting reweighted wake-sleep for models with stochastic control flow. In *Uncertainty in Artificial Intelligence*, pages 1039–1049. PMLR, 2020.

Y. LeCun, C. Cortes, and C. Burges. Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010.

Y. Li, Q. Pan, S. Wang, H. Peng, T. Yang, and E. Cambria. Disentangled variational auto-encoder for semi-supervised learning. *Information Sciences*, 482:73–85, 2019.

T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

J. Z. Liu, Z. Lin, S. Padhy, D. Tran, T. Bedrax-Weiss, and B. Lakshminarayanan. Simple and principled uncertainty estimation with deterministic deep learning via distance awareness. *arXiv preprint arXiv:2006.10108*, 2020.

Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

F. Locatello, S. Bauer, M. Lucic, G. Raetsch, S. Gelly, B. Schölkopf, and O. Bachem. Challenging common assumptions in the unsupervised learning of disentangled

representations. In *International Conference on Machine Learning*, pages 4114–4124, 2019.

J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

C. Louizos, K. Swersky, Y. Li, M. Welling, and R. Zemel. The variational fair autoencoder. *arXiv preprint arXiv:1511.00830*, 2015.

C. Ma, S. Tschiatschek, K. Palla, J. M. Hernández-Lobato, S. Nowozin, and C. Zhang. Eddi: Efficient dynamic discovery of high-value information with partial vae. In *International Conference on Machine Learning*, pages 4234–4243, 2018.

L. Maaløe, C. K. Sønderby, S. K. Sønderby, and O. Winther. Auxiliary deep generative models. *arXiv preprint arXiv:1602.05473*, 2016.

L. Maaløe, M. Fraccaro, and O. Winther. Semi-supervised generation with cluster-aware generative models. *arXiv preprint arXiv:1704.00637*, 2017.

L. Maaløe, M. Fraccaro, V. Liévin, and O. Winther. Biva: A very deep hierarchy of latent variables for generative modeling. In *Advances in Neural Information Processing Systems*, volume 32, pages 6551–6562. Curran Associates, Inc., 2019.

A. Makhzani, J. Shlens, N. Jaitly, I. Goodfellow, and B. Frey. Adversarial autoencoders. *arXiv preprint arXiv:1511.05644*, 2015.

A. Malinin and M. Gales. Predictive uncertainty estimation via prior networks. *Neural Information Processing System*, 2018.

J. Mao, C. Gan, P. Kohli, J. B. Tenenbaum, and J. Wu. The neuro-symbolic concept learner: Interpreting scenes, words, and sentences from natural supervision. *arXiv preprint arXiv:1904.12584*, 2019.

D. Massiceti, N. Siddharth, P. K. Dokania, and P. H. Torr. FlipDial: a generative model for two-way visual dialogue. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

E. Mathieu, T. Rainforth, N. Siddharth, and Y. W. Teh. Disentangling disentanglement in variational autoencoders. In *International Conference on Machine Learning*, pages 4402–4412, 2019.

P.-A. Mattei and J. Frellsen. Miwae: Deep generative modelling and imputation of incomplete data sets. In *International Conference on Machine Learning*, pages 4413–4423, 2019.

J. Mueller, D. Gifford, and T. Jaakkola. Sequence to better sequence: continuous revision of combinatorial structures. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 2536–2544. JMLR. org, 2017.

J. Mukhoti, V. Kulharia, A. Sanyal, S. Golodetz, P. H. Torr, and P. K. Dokania. Calibrating deep neural networks using focal loss. *arXiv preprint arXiv:2002.09437*, 2020.

R. Müller, S. Kornblith, and G. Hinton. When does label smoothing help? *NeurIPS*, 2019.

M. S. A. Nadeem, J.-D. Zucker, and B. Hanczar. Accuracy-rejection curves (arcs) for comparing classification methods with a reject option. In *Machine Learning in Systems Biology*, pages 65–81. PMLR, 2009.

Z. Nado, S. Padhy, D. Sculley, A. D'Amour, B. Lakshminarayanan, and J. Snoek. Evaluating prediction-time batch normalization for robustness under covariate shift. *arXiv preprint arXiv:2006.10963*, 2020.

M. P. Naeini and G. F. Cooper. Binary classifier calibration using an ensemble of near isotonic regression models. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pages 360–369. IEEE, 2016.

M. P. Naeini, G. Cooper, and M. Hauskrecht. Obtaining well calibrated probabilities using bayesian binning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 29, 2015.

E. Nalisnick, A. Matsukawa, Y. W. Teh, D. Gorur, and B. Lakshminarayanan. Do deep generative models know what they don't know? *ICLR*, 2019.

A. Nazábal, P. M. Olmos, Z. Ghahramani, and I. Valera. Handling incomplete heterogeneous data using vaes. *Pattern Recognition*, 107:107501, 2020.

Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng. Reading digits in natural images with unsupervised feature learning. 2011.

L. Neumann, A. Zisserman, and A. Vedaldi. Relaxed softmax: Efficient confidence auto-calibration for safe pedestrian detection. 2018.

J. Nixon, M. Dusenberry, G. Jerfel, L. Zhang, and D. Tran. Measuring calibration in deep learning. Sept. 2019.

Y. Ovadia, E. Fertig, J. Ren, Z. Nado, D. Sculley, S. Nowozin, J. V. Dillon, B. Lakshminarayanan, and J. Snoek. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. *NeurIPS*, 2019.

J. Paisley, D. Blei, and M. Jordan. Variational bayesian inference with stochastic search. *arXiv preprint arXiv:1206.6430*, 2012.

G. C. Pflug. *Optimization of stochastic models: the interface between simulation and optimization*, volume 373. Springer Science & Business Media, 2012.

J. Platt et al. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, 10(3): 61–74, 1999.

R. Q. Quiroga, A. Kraskov, C. Koch, and I. Fried. Explicit encoding of multimodal percepts by single neurons in the human brain. *Current Biology*, 19(15):1308–1313, 2009.

R. Ranganath, S. Gerrish, and D. Blei. Black box variational inference. In *Artificial Intelligence and Statistics*, pages 814–822. PMLR, 2014.

R. Ranganath, D. Tran, and D. Blei. Hierarchical variational models. In *International Conference on Machine Learning*, pages 324–333, 2016.

B. Recht, R. Roelofs, L. Schmidt, and V. Shankar. Do cifar-10 classifiers generalize to cifar-10? 2018. `https://arxiv.org/abs/1806.00451`.

D. Rezende and S. Mohamed. Variational inference with normalizing flows. In *International conference on machine learning*, pages 1530–1538. PMLR, 2015.

D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International Conference on Machine Learning*, pages 1278–1286, 2014.

C. Riquelme, G. Tucker, and J. Snoek. Deep bayesian bandits showdown. In *International Conference on Learning Representations*, 2018.

H. Ritter, A. Botev, and D. Barber. A scalable laplace approximation for neural networks. In *International Conference on Learning Representations*, 2018. URL `https://openreview.net/forum?id=Skdvd2xAZ`.

G. Roeder, Y. Wu, and D. K. Duvenaud. Sticking the landing: Simple, lower-variance gradient estimators for variational inference. *Advances in Neural Information Processing Systems*, 30:6925–6934, 2017.

R. Roelofs, N. Cain, J. Shlens, and M. C. Mozer. Mitigating bias in calibration error estimation. Sept. 2020.

M. Sensoy, L. Kaplan, and M. Kandemir. Evidential deep learning to quantify classification uncertainty. *NeurIPS*, 2018.

Y. Shi, N. Siddharth, B. Paige, and P. H. Torr. Variational mixture-of-experts autoencoders for multi-modal deep generative models. *arXiv*, (NeurIPS), 2019a. ISSN 23318422. URL `https://arxiv.org/pdf/1911.03393.pdf`.

Y. Shi, N. Siddharth, B. Paige, and P. H. S. Torr. Variational Mixture-of-Experts Autoencoders for Multi-Modal Deep Generative Models. In *Advances in Neural Information Processing Systems ({NeurIPS})*, pages 15692–15703, dec 2019b.

Y. Shi, B. Paige, P. Torr, and S. N. Relating by contrasting: A data-efficient framework for multimodal generative models. In *ICLR 2021: The Ninth International Conference on Learning Representations*, 2021.

N. Siddharth, T. B. Paige, J.-W. Van de Meent, A. Desmaison, N. Goodman, P. Kohli, F. Wood, and P. Torr. Learning disentangled representations with semi-supervised deep generative models. In *Advances in Neural Information Processing Systems*, pages 5925–5935, 2017.

K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

L. Smith and Y. Gal. Understanding measures of uncertainty for adversarial example detection. *arXiv preprint arXiv:1803.08533*, 2018.

C. K. Sønderby, T. Raiko, L. Maaløe, S. K. Sønderby, and O. Winther. Ladder variational autoencoders. In *Advances in neural information processing systems*, pages 3738–3746, 2016.

B. E. Stein, T. R. Stanford, and B. A. Rowland. The neural basis of multisensory integration in the midbrain: its organization and maturation. *Hearing research*, 258(1-2):4–15, 2009.

T. M. Sutter, I. Daunhawer, and J. E. Vogt. Multimodal generative learning utilizing jensen-shannon divergence. In *Workshop on Visually Grounded Interaction and Language at the 33rd Conference on Neural Information Processing Systems (NeurIPS 2019)*, volume 33, pages 6100–6110, 2020.

T. M. Sutter, I. Daunhawer, and J. E. Vogt. Generalized multimodal elbo. In *ICLR 2021: The Ninth International Conference on Learning Representations*, 2021.

M. Suzuki, K. Nakayama, and Y. Matsuo. Joint multimodal learning with deep generative models. In *International Conference on Learning Representations Workshop*, 2016.

C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

J. B. Tenenbaum. Mapping a manifold of perceptual observations. In *Advances in neural information processing systems*, pages 682–688, 1998.

J. B. Tenenbaum and W. T. Freeman. Separating style and content with bilinear models. *Neural computation*, 12(6):1247–1283, 2000.

J. Tomczak and M. Welling. Vae with a vampprior. In *International Conference on Artificial Intelligence and Statistics*, pages 1214–1223. PMLR, 2018a.

J. M. Tomczak and M. Welling. VAE with a vampprior. *Proceedings of Machine Learning Research*, 2018b.

A. Tonioni, O. Rahnama, T. Joy, L. D. Stefano, T. Ajanthan, and P. H. Torr. Learning to adapt for stereo. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9661–9670, 2019.

A. Torralba, R. Fergus, and W. T. Freeman. 80 million tiny images: A large data set for nonparametric object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(11):1958–1970, 2008.

Y. H. Tsai, P. P. Liang, A. A. Bagherzade, L.-P. Morency, and R. Salakhutdinov. Learning factorized multimodal representations. In *International Conference on Learning Representations*, 2019.

A. Van den Oord, N. Kalchbrenner, L. Espeholt, O. Vinyals, A. Graves, et al. Conditional image generation with pixelcnn decoders. In *Advances in neural information processing systems*, pages 4790–4798, 2016.

A. Van Den Oord, O. Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017.

R. Vedantam, I. Fischer, J. Huang, and K. Murphy. Generative models of visually grounded imagination. *CoRR*, abs/1705.10762, 2017.

R. Vedantam, I. Fischer, J. Huang, and K. Murphy. Generative models of visually grounded imagination. In *International Conference on Learning Representations*, 2018a.

R. Vedantam, I. Fischer, J. Huang, and K. Murphy. Generative models of visually grounded imagination. In *Proceedings of the International Conference on Learning Representations*, 2018b.

P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, P.-A. Manzagol, and L. Bottou. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of machine learning research*, 11(12), 2010.

M. J. Wainwright and M. I. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning*, 1(1-2): 1–305, 2008.

P. Welinder, S. Branson, T. Mita, C. Wah, F. Schroff, S. Belongie, and P. Perona. Caltech-UCSD Birds 200. Technical Report CNS-TR-2010-001, California Institute of Technology, 2010.

M. Welling and Y. W. Teh. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, page 681–688, Madison, WI, USA, 2011. Omnipress. ISBN 9781450306195.

Y. Wen, D. Tran, and J. Ba. Batchensemble: an alternative approach to efficient ensemble and lifelong learning. In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=Sklf1yrYDr`.

R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3):229–256, 1992.

M. Wu and N. Goodman. Multimodal generative models for scalable weakly-supervised learning. In *Advances in Neural Information Processing Systems*, pages 5580–5590, 2018a.

M. Wu and N. Goodman. Multimodal generative models for scalable weakly-supervised learning. *Adv. Neural Inf. Process. Syst.*, 2018-Decem(Nips):5575–5585, 2018b. ISSN 10495258. URL `https://arxiv.org/pdf/1802.05335.pdf`.

T. Xiao, J. Hong, and J. Ma. Dna-gan: Learning disentangled representations from multi-attribute images. *arXiv preprint arXiv:1711.05415*, 2017.

T. Xiao, J. Hong, and J. Ma. Elegant: Exchanging latent encodings with gan for transferring multiple face attributes. In *Proceedings of the European conference on computer vision (ECCV)*, pages 168–184, 2018.

B. Zadrozny and C. Elkan. Obtaining calibrated probability estimates from decision trees and naive bayesian classifiers. In *Icml*, volume 1, pages 609–616. Citeseer, 2001.

B. Zadrozny and C. Elkan. Transforming classifier scores into accurate multiclass probability estimates. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 694–699, 2002.

S. Zagoruyko and N. Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

S. Zhao, J. Song, and S. Ermon. Learning hierarchical features from deep generative models. In *International Conference on Machine Learning*, pages 4091–4099, 2017.

# Appendices

# A

# Appendix: Capturing Label Characteristics in VAEs

## A.1 Conditional Generation and Intervention for Equation (3.2)

For the model trained using (3.2) as the objective to be usable, we must consider whether it can carry out the classification, conditional generation, and intervention tasks outlined previously. Of these, classification is straightforward, but it is less apparent how the others could be performed. The key here is to realize that the classifier itself *implicitly* contains the information required to perform these tasks.

Consider first conditional generation and note that we still have access to the prior $p(\mathbf{z})$ as per a standard VAE. One simple way of performing conditional generation would be to conduct a rejection sampling where we draw samples $\hat{\mathbf{z}} \sim p(\mathbf{z})$ and then accept these if and only if they lead to the classifier predicting the desired labels up to a desired level of confidence, i.e. $q_\phi(\mathbf{y} \mid \hat{\mathbf{z}}_c) > \lambda$ where $0 < \lambda < 1$ is some chosen confidence threshold. Though such an approach is likely to be highly inefficient for any general $p(\mathbf{z})$ due to the curse of dimensionality, in the standard setting where each dimension of $\mathbf{z}$ is independent, this rejection sampling can be

performed separately for each $\mathbf{z}_c^i$, making it relatively efficient. More generally, we have that conditional generation becomes an inference problem where we wish to draw samples from

$$p\left(\mathbf{z} \mid \{q_\phi(\mathbf{y} \mid \mathbf{z}_c) > \lambda\}\right) \propto p(\mathbf{z}) \mathbb{I}\left(q_\phi(\mathbf{y} \mid \mathbf{z}_c) > \lambda\right).$$

Interventions can also be performed in an analogous manner. Namely, for a conventional intervention where we change one or more labels, we can simply resample the $\mathbf{z}_c^i$ associated we those labels, thereby sampling new characteristics to match the new labels. Further, unlike prior approaches, we can perform alternative interventions too. For example, we might attempt to find the closest $\mathbf{z}_c^i$ to the original that leads to the class label changing; this can be done in a manner akin to how adversarial attacks are performed. Alternatively, we might look to manipulate the $\mathbf{z}_c^i$ without actually changing the class itself to see what other characteristics are consistent with the labels.

To summarize, (3.2) yields an objective which provides a way of learning a semi-supervised VAEs that avoids the pitfalls of directly fixing the latents to correspond to labels. It still allows us to perform all the tasks usually associated with semi-supervised VAEs and in fact allows a more general form of interventions to be performed. However, this comes at the cost of requiring inference to perform conditional generation or interventions. Further, as the label variables $\mathbf{y}$ are absent when the labels are unobserved, there may be empirical complications with forcing all the denotational information to be encoded to the appropriate characteristic latent $\mathbf{z}_c^i$. In particular, we still have a hyperparameter $\alpha$ that must be carefully tuned to ensure the appropriate balance between classification and reconstruction.

## A.2 Model Formulation

### A.2.1 Variational Lower Bound

In this section we provide the mathematical details of our objective functions. We show how to derive it as a lower bound to the marginal model likelihood and show how we estimate the model components.

The variational lower bound for the generative model in Figure 3.2, is given as

$$\begin{aligned}
\mathcal{L}_{CCVAE} &= \sum_{\mathbf{x}\in\mathcal{U}} \mathcal{L}_{CCVAE}(\mathbf{x}) + \sum_{(\mathbf{x},\mathbf{y})\in\mathcal{S}} \mathcal{L}_{CCVAE}(\mathbf{x},\mathbf{y}) \\
\mathcal{L}_{CCVAE}(\mathbf{x},\mathbf{y}) &= E_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\frac{q_\varphi(\mathbf{y}\mid\mathbf{z}_c)}{q_{\varphi,\phi}(\mathbf{y}\mid\mathbf{x})}\log\left(\frac{p_\theta(\mathbf{x}\mid\mathbf{z})p_\psi(\mathbf{z}\mid\mathbf{y})}{q_\varphi(\mathbf{y}\mid\mathbf{z}_c)q_\phi(\mathbf{z}\mid\mathbf{x})}\right)\right] + \log q_{\varphi,\phi}(\mathbf{y}\mid\mathbf{x}) + \log p(\mathbf{y}), \\
\mathcal{L}_{CCVAE}(\mathbf{x}) &= E_{q_\phi(\mathbf{z}|\mathbf{x})q_\varphi(\mathbf{y}|\mathbf{z}_c)}\left[\log\left(\frac{p_\theta(\mathbf{x}\mid\mathbf{z})p_\psi(\mathbf{z}_c\mid\mathbf{y})p(\mathbf{y})}{q_\varphi(\mathbf{y}\mid\mathbf{z}_c)q_\phi(\mathbf{z}\mid\mathbf{x})}\right)\right].
\end{aligned}$$

The overall likelihood in the semi-supervised case is given as

$$p_\theta(\mathbf{x},\mathbf{y}) = \prod_{(\mathbf{x},\mathbf{y})\in\mathcal{S}} p_\theta(\mathbf{x},\mathbf{y}) \prod_{\mathbf{x}\in\mathcal{U}} p_\theta(\mathbf{x}),$$

To derive a lower bound for the overall objective, we need to obtain lower bounds on $\log p_\theta(\mathbf{x})$ and $\log p_\theta(\mathbf{x},\mathbf{y})$. When the labels are unobserved the latent state will consist of $\mathbf{z}$ and $\mathbf{y}$. Using the factorization according to the graph in Figure 3.2 yields

$$\log p_\theta(\mathbf{x}) \geq E_{q_\phi(\mathbf{z}|\mathbf{x})q_\varphi(\mathbf{y}|\mathbf{z}_c)}\left[\log\left(\frac{p_\theta(\mathbf{x}\mid\mathbf{z})p_\psi(\mathbf{z}\mid\mathbf{y})p(\mathbf{y})}{q_\varphi(\mathbf{y}\mid\mathbf{z}_c)q_\phi(\mathbf{z}\mid\mathbf{x})}\right)\right],$$

where $p_\psi(\mathbf{z}\mid\mathbf{y}) = p(\mathbf{z}_{\setminus\mathbf{c}})p_\psi(\mathbf{z}_\mathbf{c}\mid\mathbf{y})$. For supervised data points we consider a lower bound on the likelihood $p_\theta(\mathbf{x},\mathbf{y})$,

$$\log p_\theta(\mathbf{x},\mathbf{y}) \geq \int \log \frac{p_\theta(\mathbf{x}\mid\mathbf{z})p_\psi(\mathbf{z}\mid\mathbf{y})p(\mathbf{y})}{q_{\varphi,\phi}(\mathbf{z}\mid\mathbf{x},\mathbf{y})} q_{\varphi,\phi}(\mathbf{z}\mid\mathbf{x},\mathbf{y})\mathrm{d}\mathbf{z},$$

in order to make sense of the term $q_{\varphi,\phi}(\mathbf{z}\mid\mathbf{x},\mathbf{y})$, which is usually different from $q_\phi(\mathbf{z}\mid\mathbf{x})$ we consider the inference model

$$q_{\varphi,\phi}(\mathbf{z}\mid\mathbf{x},\mathbf{y}) = \frac{q_\varphi(\mathbf{y}\mid\mathbf{z}_c)q_\phi(\mathbf{z}\mid\mathbf{x})}{q_{\varphi,\phi}(\mathbf{y}\mid\mathbf{x})}, \quad \text{where} \quad q_{\varphi,\phi}(\mathbf{y}\mid\mathbf{x}) = \int q_\varphi(\mathbf{y}\mid\mathbf{z}_c)q_\phi(\mathbf{z}\mid\mathbf{x})\mathrm{d}\mathbf{z}.$$

Returning to the lower bound on $\log p_\theta(\mathbf{x},\mathbf{y})$ we obtain

$$\begin{aligned}
\log p_\theta(\mathbf{x},\mathbf{y}) &\geq \int \log \frac{p_\theta(\mathbf{x}\mid\mathbf{z})p_\psi(\mathbf{z}\mid\mathbf{y})p(\mathbf{y})}{q(\mathbf{z}\mid\mathbf{x},\mathbf{y})} q(\mathbf{z}\mid\mathbf{x},\mathbf{y})\mathrm{d}\mathbf{z} \\
&= \int \log\left(\frac{p_\theta(\mathbf{x}\mid\mathbf{z})p_\psi(\mathbf{z}\mid\mathbf{y})p(\mathbf{y})q_{\varphi,\phi}(\mathbf{y}\mid\mathbf{x})}{q_\varphi(\mathbf{y}\mid\mathbf{z}_c)q_\phi(\mathbf{z}\mid\mathbf{x})}\right)\frac{q_\varphi(\mathbf{y}\mid\mathbf{z}_c)q_\phi(\mathbf{z}\mid\mathbf{x})}{q_{\varphi,\phi}(\mathbf{y}\mid\mathbf{x})}\mathrm{d}\mathbf{z} \\
&= E_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\frac{q_\varphi(\mathbf{y}\mid\mathbf{z}_c)}{q_{\varphi,\phi}(\mathbf{y}\mid\mathbf{x})}\log\left(\frac{p(\mathbf{x}\mid\mathbf{z})p_\psi(\mathbf{z}_c\mid\mathbf{y})}{q_\varphi(\mathbf{y}\mid\mathbf{z}_c)q_\phi(\mathbf{z}\mid\mathbf{x})}\right)\right] + \log q_{\varphi,\phi}(\mathbf{y}\mid\mathbf{x}) + \log p(\mathbf{y}),
\end{aligned}$$

where $q_\varphi(\mathbf{y}\mid\mathbf{z}_c)/q_{\varphi,\phi}(\mathbf{y}\mid\mathbf{x})$ denotes the Radon-Nikodym derivative of $q_{\varphi,\phi}(\mathbf{z}\mid\mathbf{x},\mathbf{y})$ with respect to $q_\phi(\mathbf{z}\mid\mathbf{x})$.

### A.2.2 Alternative Derivation of Unsupervised Bound

The bound for the unsupervised case can alternatively be derived by applying Jensen's inequality twice. First, use the standard (unsupervised) Evidence Lower Bound (ELBO)

$$\log p_\theta(\mathbf{x}) \geq \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log \frac{p_\theta(\mathbf{x} \mid \mathbf{z})p(\mathbf{z})}{q_\phi(\mathbf{z} \mid \mathbf{x})}\right].$$

Now, since calculating $p(\mathbf{z}) = p(\mathbf{z}_c)p(\mathbf{z}_{\setminus c}) = p(\mathbf{z}_{\setminus c})\sum_{\mathbf{y}} p(\mathbf{z}_c \mid \mathbf{y})p(\mathbf{y})$ can be expensive we can apply Jensen's inequality a second time to the expectation over $\mathbf{z}_c$ to obtain

$$\log p(\mathbf{z}_c) \geq \mathbb{E}_{q_\varphi(\mathbf{y}|\mathbf{z}_c)}\left[\log \frac{p_\psi(\mathbf{z}_s \mid \mathbf{y})p(\mathbf{y})}{q_\varphi(\mathbf{y} \mid \mathbf{z}_s)}\right].$$

Substituting this bound into the unsupervised ELBO yields again our bound

$$\log p(\mathbf{x}) \geq \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})q_\varphi(\mathbf{y}|\mathbf{z}_c)}\left[\log \frac{p_\theta(\mathbf{x} \mid \mathbf{z})p(\mathbf{z} \mid \mathbf{y})}{q_\phi(\mathbf{z} \mid \mathbf{x})q_\varphi(\mathbf{y} \mid \mathbf{z}_c)}\right] + \log p(\mathbf{y}) \tag{A.1}$$

## A.3 Implementation

### A.3.1 CelebA

We chose to use only a subset of the labels present in CelebA, since not all attributes are visually distinguishable in the reconstructions e.g. (`earrings`). As such we limited ourselves to the following labels: `arched eyebrows`, `bags under eyes`, `bangs`, `black hair`, `blond hair`, `brown hair`, `bushy eyebrows`, `chubby`, `eyeglasses`, `heavy makeup`, `male`, `no beard`, `pale skin`, `receding hairline`, `smiling`, `wavy hair`, `wearing necktie`, `young`. No images were omitted or cropped, the only modifications were keeping the aforementioned labels and resizing the images to be $64 \times 64$ in dimension.

### A.3.2 Chexpert

The Chexpert dataset comprises of chest X-rays taken from a variety of patients. We down-sampled each image to be $64 \times 64$ and used the same networks from the CelebA experiments. The five main attributes for Chexpert are: `cardiomegaly`, `edema`, `consolidation`, `atelectasis`, `pleural effusion`. Which for non medical experts can be interpreted as: enlargement of the heart; fluid in the alveoli; fluid in the lungs; collapsed lung; fluid in the corners of the lungs.

### A.3.3 Implementation Details

For our experiments we define the generative and inference networks as follows. The approximate posterior is represented as $q_\phi(\mathbf{z} \mid \mathbf{x}) = \mathcal{N}(\mathbf{z_c}, \mathbf{z}_{\setminus \mathbf{c}} \mid \boldsymbol{\mu}_\phi(\mathbf{x}), \text{diag}(\boldsymbol{\sigma}^2_\phi(\mathbf{x})))$ with $\boldsymbol{\mu}_\phi(\mathbf{x})$ and $\text{diag}(\boldsymbol{\sigma}^2_\phi(\mathbf{x}))$ being the architecture from Higgins et al. (2016). The generative model $p_\theta(\mathbf{x} \mid \mathbf{z})$ is represented by a Laplace distribution, again parametrized using the architecture from Higgins et al. (2016). The label predictive distribution $q_\varphi(\mathbf{y} \mid \mathbf{z}_c)$ is represented as $\text{Ber}(\mathbf{y} \mid \boldsymbol{\pi}_\varphi(\mathbf{z_c}))$ with $\boldsymbol{\pi}_\varphi(\mathbf{z_c})$ being a diagonal transformation forcing the factorisation $q_\varphi(\mathbf{y} \mid \mathbf{z}_c) = \prod_i q_{\psi^i}(y_i \mid \mathbf{z_c}^i)$. The conditional prior is given as $p_\psi(\mathbf{z}_c \mid \mathbf{y}) = \mathcal{N}(\mathbf{z_c} \mid \boldsymbol{\mu}_\psi(\mathbf{y}), \text{diag}(\boldsymbol{\sigma}^2_\psi(\mathbf{y})))$, with the appropriate factorisation, where the parameters are represented by an MLP. Finally, the prior placed on the portion of the latent space reserved for unlabelled latent variables is $p(\mathbf{z}_{\setminus \mathbf{c}}) = \mathcal{N}(\mathbf{z}_{\setminus \mathbf{c}} \mid \mathbf{0}, \mathbf{I}))$. For the latent space $\mathbf{z_c} \in \mathbb{R}^{m_c}$ and $\mathbf{z}_{\setminus \mathbf{c}} \in \mathbb{R}^{m_{\setminus c}}$, where $m = m_c + m_{\setminus c}$ with $m_c = 18$ and $m_{\setminus c} = 27$ for CelebA. The architectures are given in and Table A.1.

| **Encoder** | **Decoder** |
|---|---|
| Input 32 x 32 x 3 channel image | Input $\in \mathbb{R}^m$ |
| $32 \times 3 \times 4 \times 4$ Conv2d stride 2 & ReLU | $m \times 256$ Linear layer |
| $32 \times 32 \times 4 \times 4$ Conv2d stride 2 & ReLU | $128 \times 256 \times 4 \times 4$ ConvTranspose2d stride 1 & ReLU |
| $64 \times 32 \times 4 \times 4$ Conv2d stride 2 & ReLU | $64 \times 128 \times 4 \times 4$ ConvTranspose2d stride 2 & ReLU |
| $128 \times 64 \times 4 \times 4$ Conv2d stride 2 & ReLU | $32 \times 64 \times 4 \times 4$ ConvTranspose2d stride 2 & ReLU |
| $256 \times 128 \times 4 \times 4$ Conv2d stride 1 & ReLU | $32 \times 32 \times 4 \times 4$ ConvTranspose2d stride 2 & ReLU |
| $256 \times (2 \times m)$ Linear layer | $3 \times 32 \times 4 \times 4$ ConvTranspose2d stride 2 & Sigmoid |

| **Classifier** | **Conditional Prior** |
|---|---|
| Input $\in \mathbb{R}^{m_c}$ | Input $\in \mathbb{R}^{m_c}$ |
| $m_c \times m_c$ Diagonal layer | $m_c \times m_c$ Diagonal layer |

**Table A.1:** Architectures for CelebA and Chexpert.

**Optimization** We trained the models on a GeForce GTX Titan GPU. Training consumed $\sim$ 2Gb for CelebA and Chexpert, taking around 2 hours to complete 100 epochs respectively. Both models were optimized using Adam with a learning rate of $2 \times 10^{-4}$ for CelebA respectively.

### High variance of classifier gradients

The gradients of the classifier parameters $\varphi$ suffer from a high variance during training. We find that not reparameterizing $\mathbf{z_c}$ for $q_\varphi(\mathbf{y} \mid \mathbf{z}_c)$ reduces this issue:

$$\mathcal{L}_{CCVAE}(\mathbf{x}, \mathbf{y}) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\frac{q_\varphi(\mathbf{y} \mid \bar{\mathbf{z}_c})}{q_{\varphi,\phi}(\mathbf{y} \mid \mathbf{x})} \log \frac{p_\theta(\mathbf{x} \mid \mathbf{z}) p_\psi(\mathbf{z} \mid \mathbf{y})}{q_\varphi(\mathbf{y} \mid \bar{\mathbf{z}_c}) q_\phi(\mathbf{z} \mid \mathbf{x})}\right] + \log q_{\varphi,\phi}(\mathbf{y} \mid \mathbf{x}) + \log p(\mathbf{y}). \tag{A.2}$$

where $\bar{\mathbf{z}_c}$ indicates that we do not reparameterize the sample. This significantly reduces the variance of the magnitude of the gradient norm $\nabla_\varphi$, allowing the classifier to learn appropriate weights and structure the latent space. This can be seen in Figure A.1, where we plot the gradient norm of $\varphi$ for when we **do** reparameterize $\mathbf{z_c}$ (blue) and when we **do not** (orange). Clearly not reparameterizing leads to a lower variance in the gradient norm of the classifier, which aides learning. To a certain extent these gradients can be viewed as redundant, as there is already gradients to update the predictive distribution due to the $\log q_{\varphi,\phi}(\mathbf{y} \mid \mathbf{x})$ term anyway.

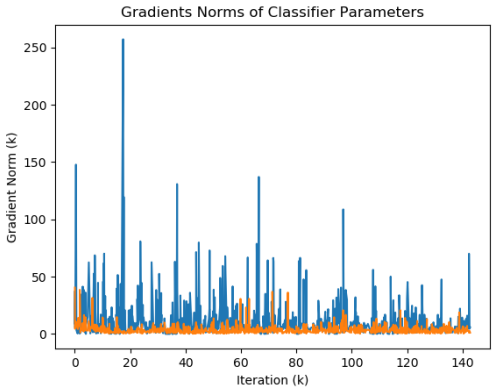



**Figure A.1:** Gradient norms of classifier.

### A.3.4 Modified DIVA

The primary goal of DIVA is domain invariant classification and not to obtain representations of individual characteristics like we do here. The objective is essentially a classifier which is regularized by a variational objective. However, to achieve domain generalization, the authors aim to disentangle the domain, class and other generative factors. This motivation leads to a graphical model that is similar in spirit to ours ( Figure A.2), in that the latent variables are used to predict labels, and the introduction of the inductive bias to partition the latent space. As such, DIVA can be modified to suit our problem of encapsulating characteristics. The first modification we need to consider is the removal of $\mathbf{z}_d$, as we are not considering multi-domain problems. Secondly, we introduce the factorization present in CCVAE, namely $q_\varphi(\mathbf{y} \mid \mathbf{z}_c) = \prod_i q_{\psi^i(y_i}|\mathbf{z_c}^i)$. With these two modifications an alternative objective can now be constructed, with the supervised given as

$$\begin{aligned}\mathcal{L}_{SDIVA}(\mathbf{x}, \mathbf{y}) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})} \log p_\theta(\mathbf{x} \mid \mathbf{z}) - \beta KL(q_\phi(\mathbf{z_{\setminus c}}|x)||p(\mathbf{z_{\setminus c}})) \\ - \beta KL(q_\phi(\mathbf{z_c}|x)||p_\psi(\mathbf{z}_c \mid \mathbf{y})),\end{aligned}$$

and the unsupervised as

$$\begin{aligned}\mathcal{L}_{UDIVA}(\mathbf{x}) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})} \log p_\theta(\mathbf{x} \mid \mathbf{z}) - \beta KL(q_\phi(\mathbf{z_{\setminus c}}|x)||p(\mathbf{z_{\setminus c}})) \\ + \beta \mathbb{E}_{q_\phi(\mathbf{z_c}|x)q_\varphi(\mathbf{y}|\mathbf{z}_c)}[\log p_\psi(\mathbf{z}_c \mid \mathbf{y}) - \log q_\phi(\mathbf{z_c}|x)], \\ + \beta \mathbb{E}_{q_\phi(\mathbf{z_c}|x)q_\varphi(\mathbf{y}|\mathbf{z}_c)}[\log p(\mathbf{y}) - \log q_\varphi(\mathbf{y} \mid \mathbf{z}_c)],\end{aligned}$$

where $\mathbf{y}$ has to be imputed. The final objective for DIVA is then given as

$$\log p_\theta\left(\mathcal{D}\right) \geq \sum\nolimits_{(\mathbf{x},\mathbf{y})\in\mathcal{S}} \mathcal{L}_{SDIVA}(\mathbf{x}, \mathbf{y}) + \sum\nolimits_{\mathbf{x}\in\mathcal{U}} \Big[\mathcal{L}_{UDIVA}(\mathbf{x}) + \alpha \mathbb{E}_{q(\mathbf{z_c}|\mathbf{x})} \log q_\varphi(\mathbf{y} \mid \mathbf{z}_c)\Big].$$

It is interesting to note the differences to the objective of CCVAE, namely, there is no emergence of a natural classifier in the supervised case, and $\mathbf{y}$ has to be imputed in the unsupervised case instead of relying on variational inference as in CCVAE. Clearly such differences have a significant impact on performance as demonstrated by the main results of this paper.

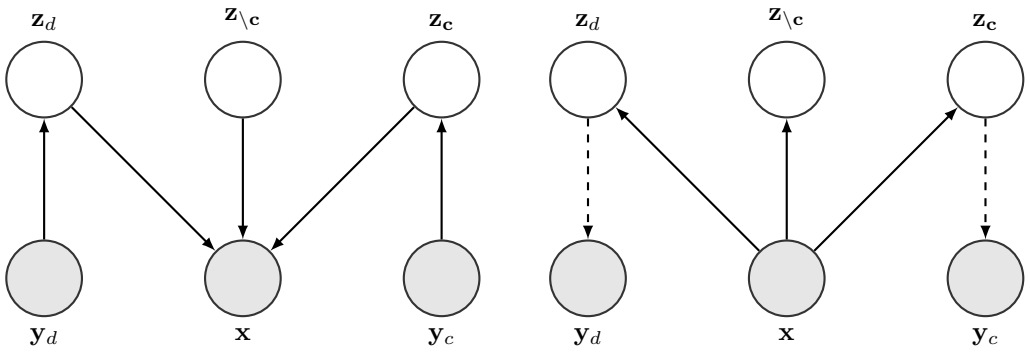



**Figure A.2:** Left: Generative model for DIVA, Right: Inference model where dashed line indicates auxiliary classifier.

## A.4 Additional Results

### A.4.1 Single Interventions

Here we demonstrate single interventions where we change the binary value for the desired attributes. To quantitatively evaluate the single interventions, we intervene on a single label and report the changes in log-probabilities assigned by a pre-trained classifier. If the single intervention only affects the characteristics of the chosen label, then there should be no change in other classes and only a change on the chosen label. Intervening on all possible labels yields a confusion matrix, with the optimal results being a diagonal matrix with zero off-diagonal elements. We also report the condition number for the confusion matrices, given in the titles.

It is interesting to note that the interventions for CCVAE are subtle, this is due to the latent $z_c^i \sim p(z_c^i|y_i)$, which will be centered around the mean. More striking intervention can be achieved by traversing along $z_c^i$.

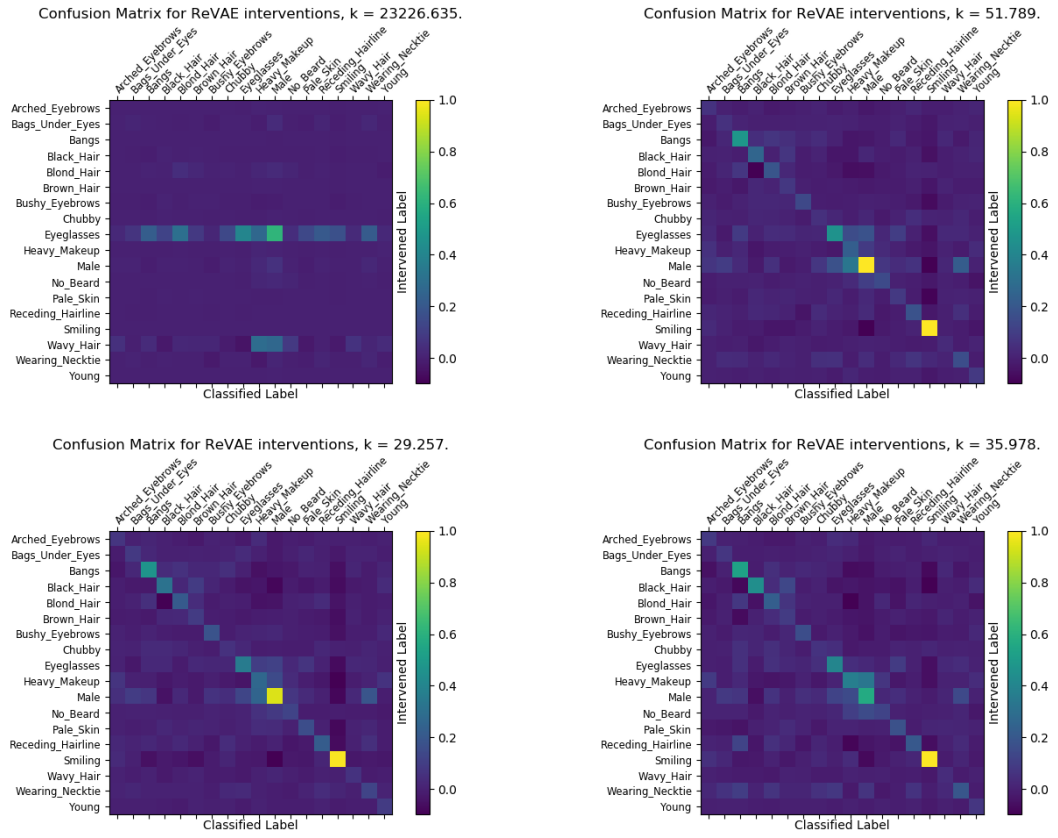



**Figure A.3:** Confusion matrices for CCVAE for (from top left clockwise) $f = 0.004, 0.06, 0.2, 1.0$

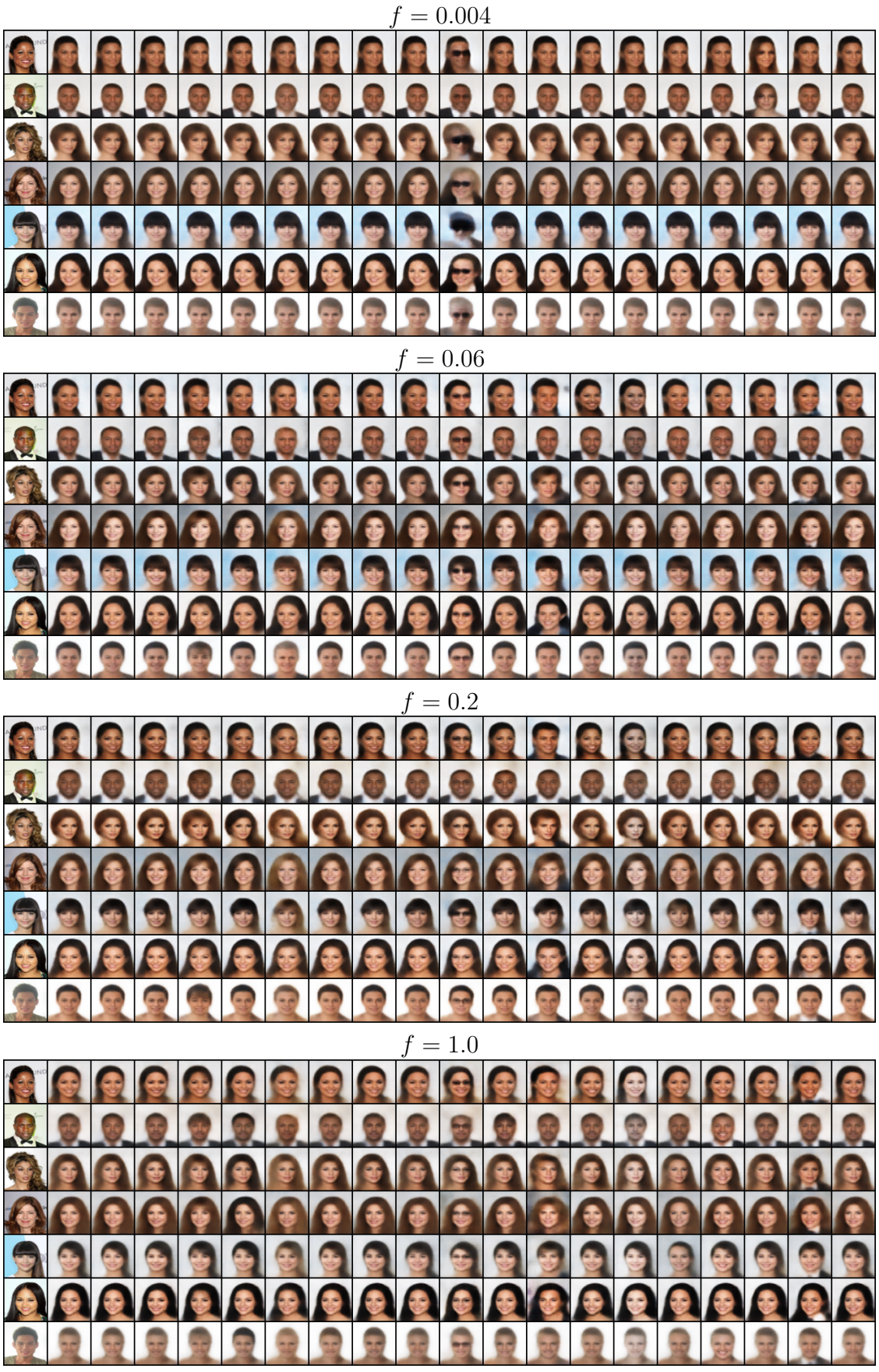



**Figure A.4:** CCVAE. From left to right: original, reconstruction, then interventions from switching on the following labels: `arched eyebrows`, `bags under eyes`, `bangs`, `black hair`, `blond hair`, `brown hair`, `bushy eyebrows`, `chubby`, `eyeglasses`, `heavy makeup`, `male`, `no beard`, `pale skin`, `receding hairline`, `smiling`, `wavy hair`, `wearing necktie`, `young`.

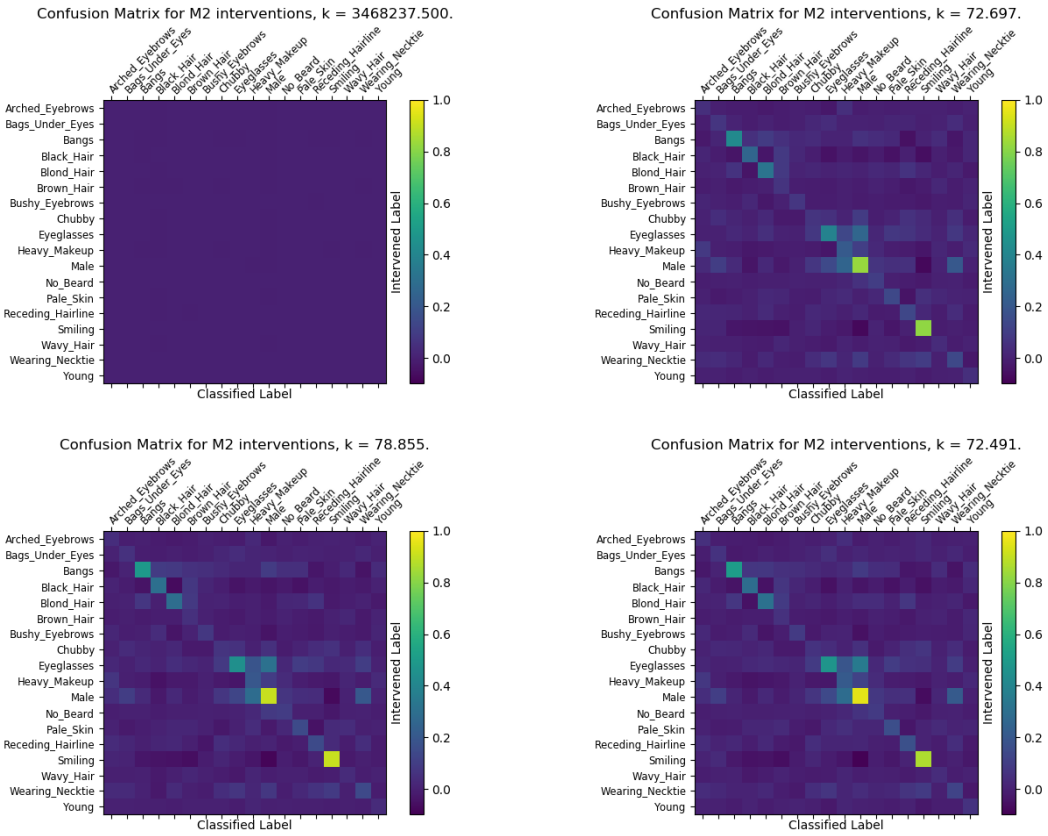



**Figure A.5:** Confusion matrices for M2 for (from top left clockwise) $f = 0.004, 0.06, 0.2, 1.0$

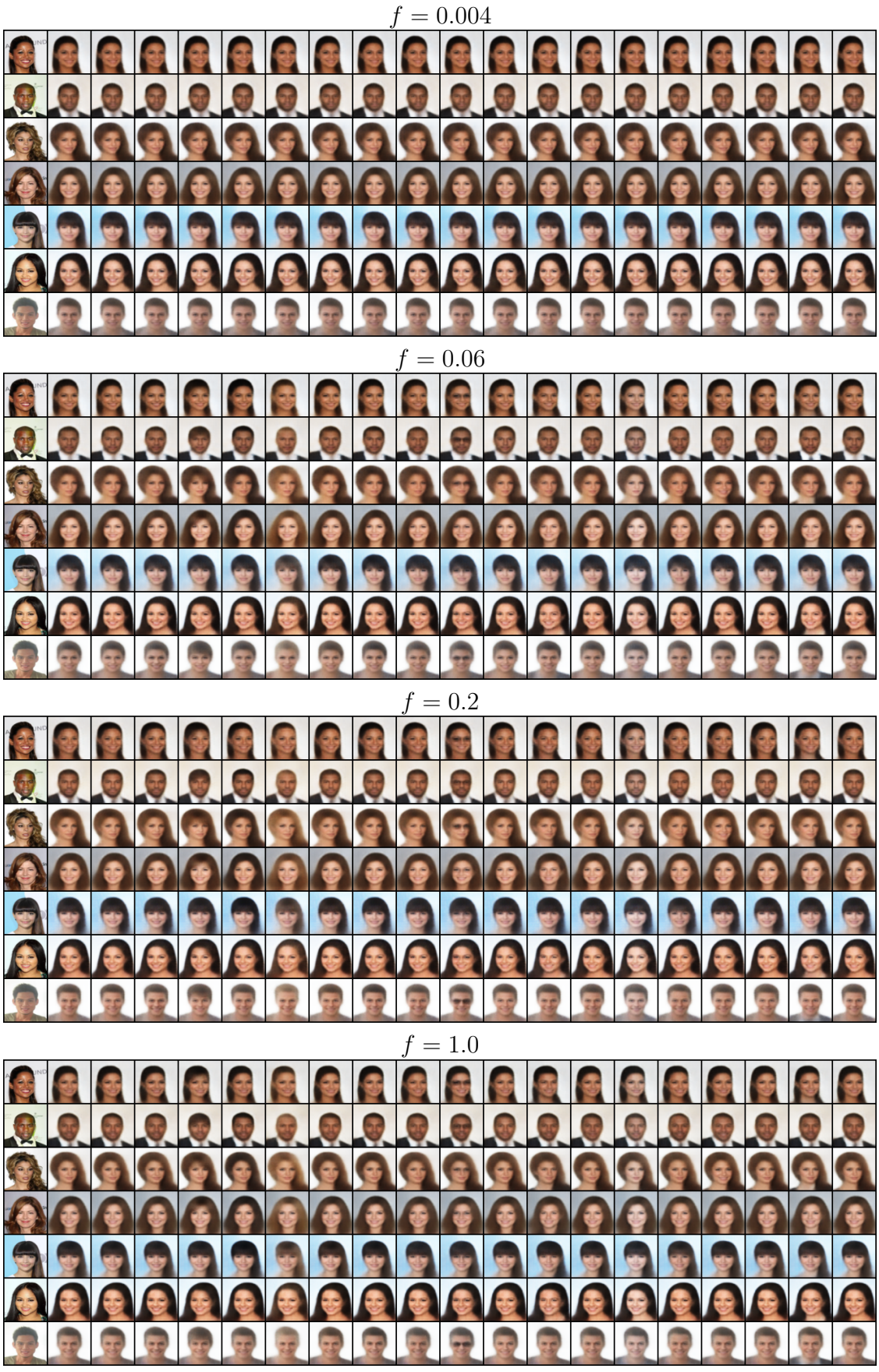



**Figure A.6:** M2. From left to right: original, reconstruction, then interventions from switching on the following labels: `arched eyebrows`, `bags under eyes`, `bangs`, `black hair`, `blond hair`, `brown hair`, `bushy eyebrows`, `chubby`, `eyeglasses`, `heavy makeup`, `male`, `no beard`, `pale skin`, `receding hairline`, `smiling`, `wavy hair`, `wearing necktie`, `young`.

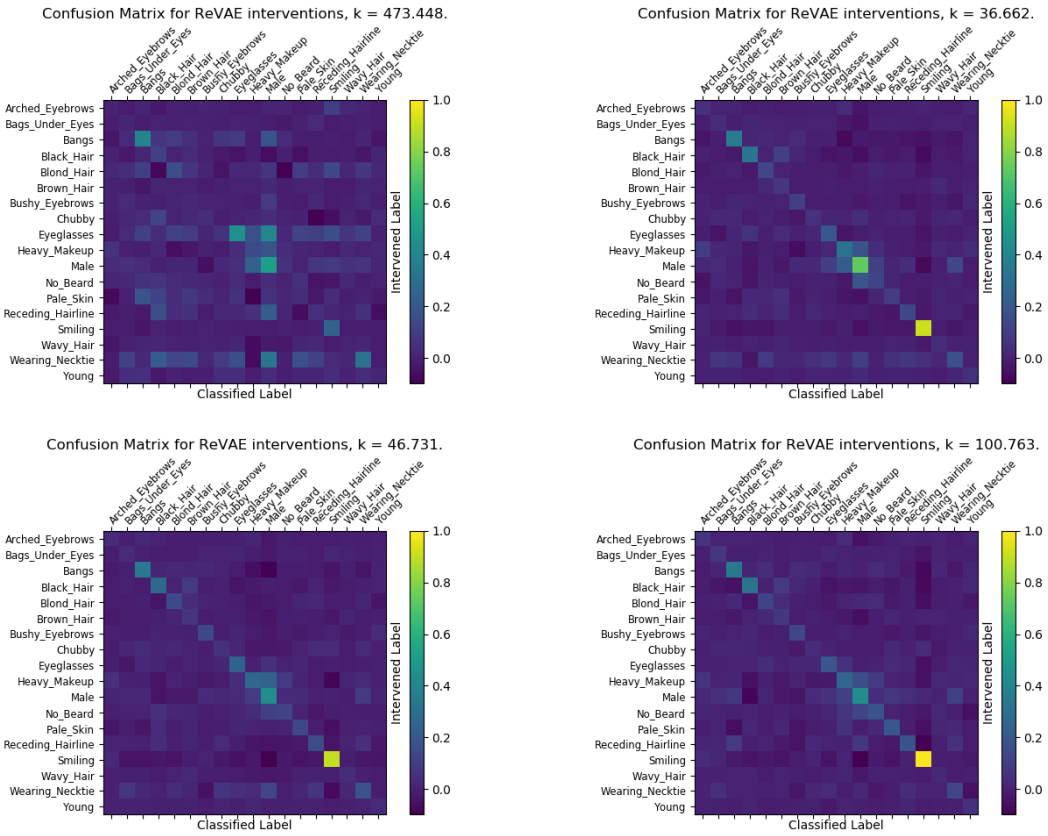



**Figure A.7:** Confusion matrices for DIVA for (from top left clockwise) $f = 0.004, 0.06, 0.2, 1.0$

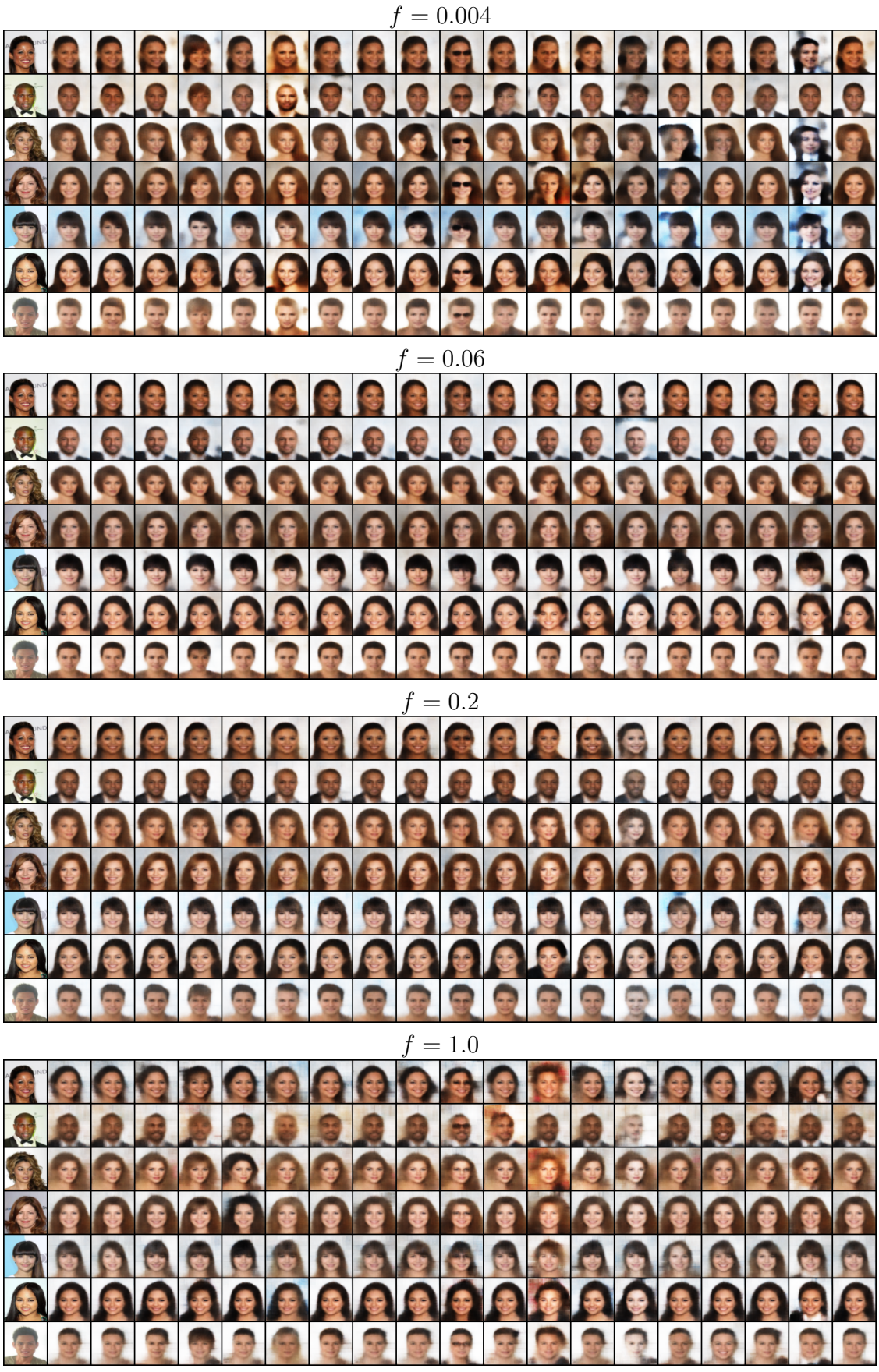



**Figure A.8:** DIVA. From left to right: original, reconstruction, then interventions from switching on the following labels: `arched eyebrows`, `bags under eyes`, `bangs`, `black hair`, `blond hair`, `brown hair`, `bushy eyebrows`, `chubby`, `eyeglasses`, `heavy makeup`, `male`, `no beard`, `pale skin`, `receding hairline`, `smiling`, `wavy hair`, `wearing necktie`, `young`.

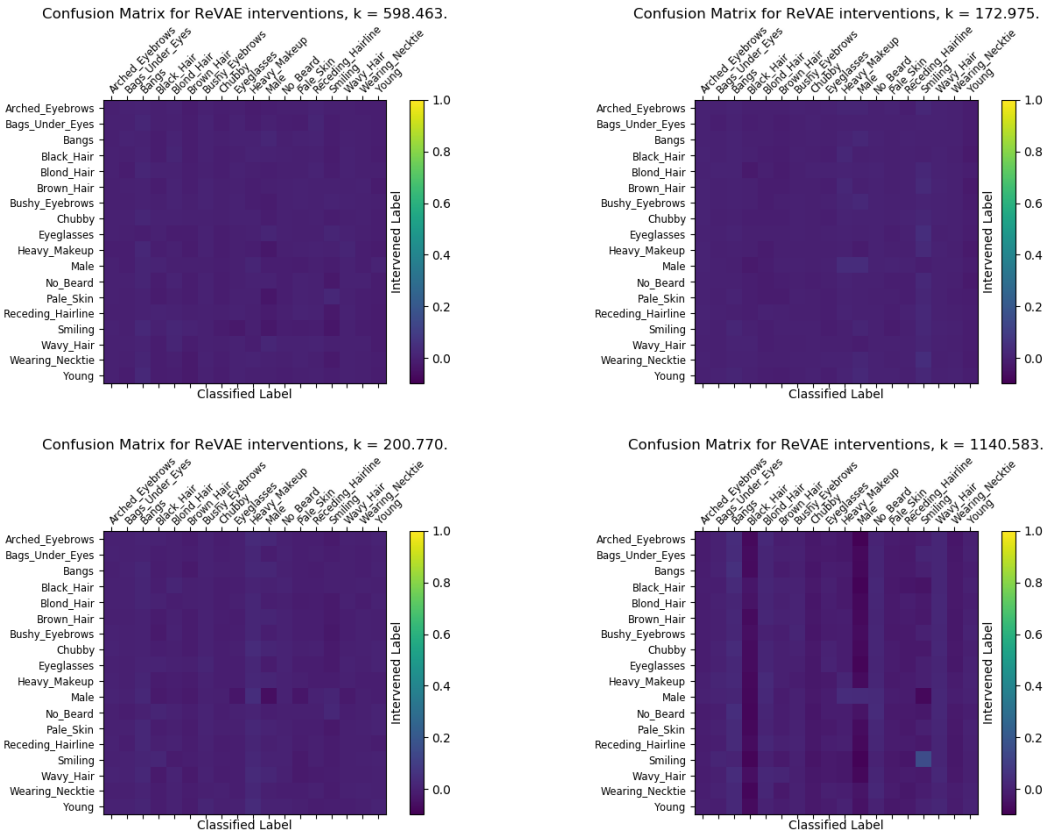



**Figure A.9:** Confusion matrices for MVAE for (from top left clockwise) $f = 0.004, 0.06, 0.2, 1.0$

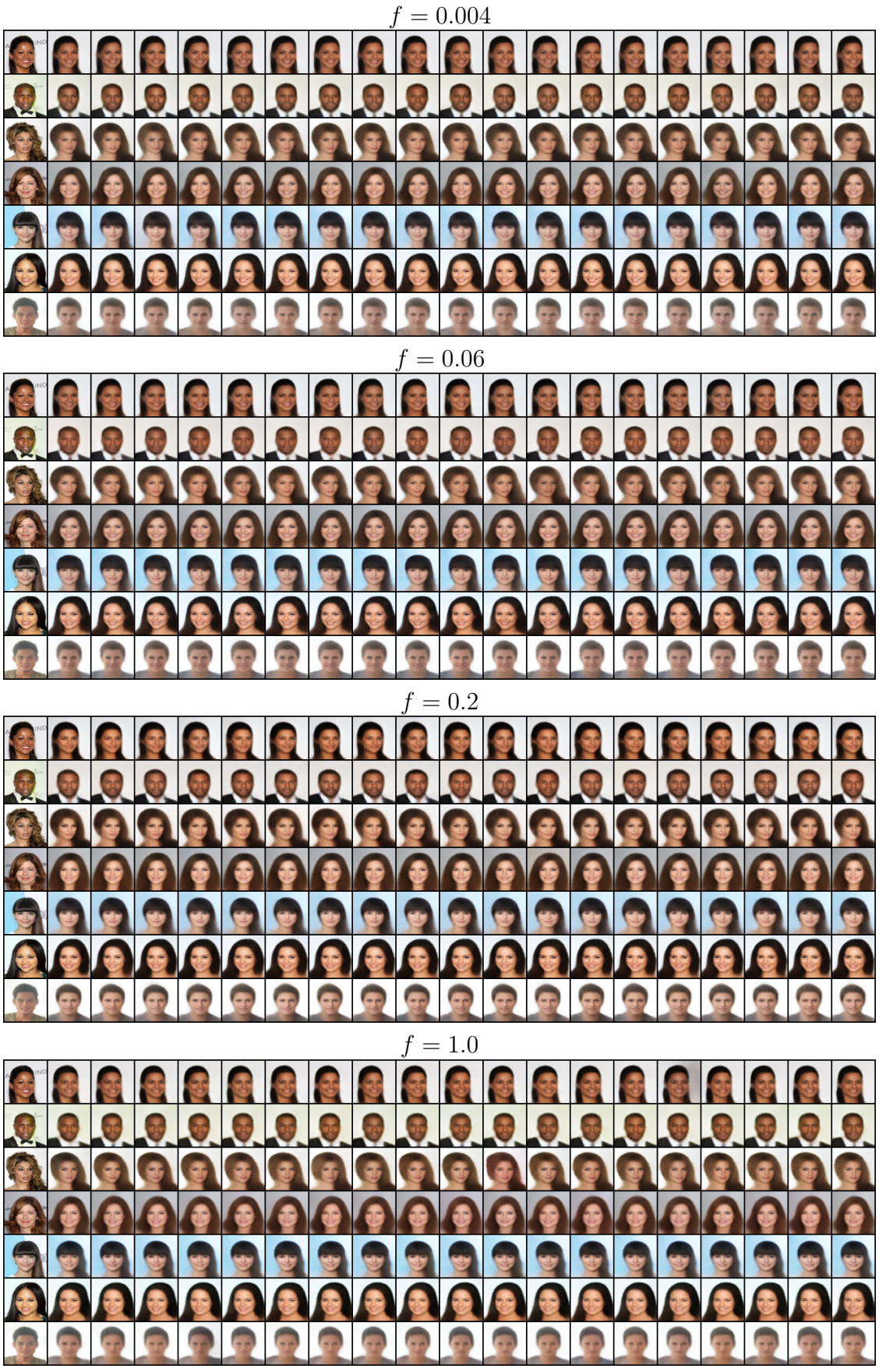



**Figure A.10:** MVAE. From left to right: original, reconstruction, then interventions from switching on the following labels: `arched eyebrows`, `bags under eyes`, `bangs`, `black hair`, `blond hair`, `brown hair`, `bushy eyebrows`, `chubby`, `eyeglasses`, `heavy makeup`, `male`, `no beard`, `pale skin`, `receding hairline`, `smiling`, `wavy hair`, `wearing necktie`, `young`.

### A.4.2 Latent Traversals

Here we provide more latent traversals for CCVAE in Figure A.11 and for DIVA in Figure A.12. CCVAE is able to smoothly alter characteristics, indicating that it is able to encapsulate characteristics in a single dimension, unlike DIVA which is unable to alter the characteristics effectively, suggesting it cannot encapsulate the characteristics.

### A.4.3 Generation

We provide results for the fidelity of image generation on CelebA. To do this we use the FID metric Heusel et al. (2017), we omitted results for Chexpert as the inception model used in FID has not been trained on the typical features associated with X-Rays. The results are given in Table A.2, interestingly for low supervision rates MVAE obtains the best performance but for higher supervision rates M2 outperforms MVAE. We posit that this is due to MVAE having little structure imposed on the latent space, as such the POE can structure the representation purely for reconstruction without considering the labels, something which is not possible as the supervision rate is increased. CCVAE obtains competitive results with respect to M2. It is important to note that generative fidelity is not the focus of this work as we focus purely on how to structure the latent space using labels. It is no surprise then that the generations are bad as structuring the latent space will potentially be at odds with the reconstruction term in the loss.

**Table A.2:** CelebA FID scores.

| Model | $f = 0.004$ | $f = 0.06$ | $f = 0.2$ | $f = 1.0$ |
|---|---|---|---|---|
| CCVAE | 127.956 | 121.84 | 121.751 | 120.457 |
| M2 | 127.719 | 122.521 | **120.406** | **119.228** |
| DIVA | 192.448 | 230.522 | 218.774 | 201.484 |
| MVAE | **118.308** | **115.947** | 128.867 | 137.461 |

### A.4.4 Conditional Generation

To asses conditional generation, we first train an independent classifier for both datasets. We then conditionally generate samples given labels and evaluate them using this pre-trained classifier. Results provided in Table A.3. CCVAE and M2 are

comparable in generative abilities, but DIVA and MVAE perform poorly, indicated by random guessing.

**Table A.3:** Generations accuracies.

| | CelebA | | | | Chexpert | | | |
|---|---|---|---|---|---|---|---|---|
| Model | $f = 0.004$ | $f = 0.06$ | $f = 0.2$ | $f = 1.0$ | $f = 0.004$ | $f = 0.06$ | $f = 0.2$ | $f = 1.0$ |
| CCVAE | **0.513** | 0.605 | **0.612** | 0.596 | **0.516** | **0.563** | **0.549** | 0.542 |
| M2 | 0.499 | **0.61** | 0.612 | **0.611** | 0.503 | 0.547 | 0.547 | **0.558** |
| DIVA | 0.501 | 0.501 | 0.501 | 0.501 | 0.499 | 0.503 | 0.503 | 0.503 |
| MVAE | 0.501 | 0.501 | 0.501 | 0.501 | 0.499 | 0.499 | 0.499 | 0.499 |

### A.4.5 Diversity of Conditional Generations

We also report more examples for diversity, as in Figure 3.6, in Figure A.13.

### A.4.6 Multi-class Setting

Here we provide results for the multi-class setting of MNIST and FashionMNIST. The multi-class setting is somewhat tangential to our work, but we include it for completeness. For CCVAE, we have some flexibility over the size of the latent space. Trying to encapsulate representations for each label is not well suited for this setting, as it's not clear how you could alter the representation of an image being a 6, whilst preserving the representation of it being an 8. In fact, there is really only one label for this setting, but it takes multiple values. With this in mind, we can now make an explicit choice about how the latent space will be structured, we can set $\mathbf{z_c} \in \mathbb{R}$ or $\mathbf{z_c} \in \mathbb{R}^N$, or conversely, store all of the representation in $\mathbf{z_c}$, i.e. $\mathbf{z}_{\setminus \mathbf{c}} = \emptyset$. Furthermore, we do not need to enforce the factorization $q_\varphi(\mathbf{y} \mid \mathbf{z}_c) = \prod_i q(y_i|z_c^i)$, and instead can be parameterized by a function $\mathcal{F} : \mathbb{R}^N \to \mathbb{R}^M$ where M is the number of possible classes.

**Classification** We provide the classification results in Table A.4.

**Table A.4:** Additional classification accuracies.

| | MNIST | | | | FashionMNIST | | | |
|---|---|---|---|---|---|---|---|---|
| Model | $f = 0.004$ | $f = 0.06$ | $f = 0.2$ | $f = 1.0$ | $f = 0.004$ | $f = 0.06$ | $f = 0.2$ | $f = 1.0$ |
| CCVAE | **0.927** | **0.974** | **0.979** | **0.988** | 0.741 | **0.865** | **0.879** | **0.901** |
| M2 | 0.918 | 0.962 | 0.968 | 0.981 | **0.756** | 0.848 | 0.860 | 0.892 |

**Conditional Generation** We provide classification accuracies for pre-trained classifier using conditional generated samples as input and the condition as a label. We also report the mutual information to give an indication of how *out-of-distribution* the samples are. In order to estimate the uncertainty, we transform a fixed pre-trained classifier into a Bayesian predictive classifier that integrates over the posterior distribution of parameters $\omega$ as $p(\mathbf{y} \mid \mathbf{x}, \mathcal{D}) = \int p(\mathbf{y} \mid \mathbf{x}, \omega) p(\omega \mid \mathcal{D}) \mathrm{d}\omega$. The utility of classifier uncertainties for out-of-distribution detection has previously been explored Smith and Gal (2018), where dropout is also used at test time to estimate the mutual information (MI) between the predicted label $\mathbf{y}$ and parameters $\omega$ (Gal, 2016; Smith and Gal, 2018) as

$$I(\mathbf{y}, \omega \mid \mathbf{x}, \mathcal{D}) = H[p(\mathbf{y} \mid \mathbf{x}, \mathcal{D})] - \mathbb{E}_{p(\omega \mid \mathcal{D})} \left[ H[p(\mathbf{y} \mid \mathbf{x}, \omega)] \right].$$

However, the Monte Carlo (MC) dropout approach has the disadvantage of requiring *ensembling* over multiple instances of the classifier for a robust estimate and repeated forward passes through the classifier to estimate MI. To mitigate this, we instead employ a sparse variational GP (with 200 inducing points) as a replacement for the last linear layer of the classifier, fitting just the GP to the data and labels while holding the rest of the classifier fixed. This, in our experience, provides a more robust and cheaper alternative to MC-dropout for estimating MI. Results are provided in Table A.5.

**Table A.5:** Pre-trained classifier accuracies and MI for MNIST (top) and FashionMNIST (bottom).

| | Model | $f = 0.004$ | | $f = 0.06$ | | $f = 0.2$ | | $f = 1.0$ | |
|---|---|---|---|---|---|---|---|---|---|
| | | Acc | MI | Acc | MI | Acc | MI | Acc | MI |
| M | CCVAE | **0.910** | **0.020** | **0.954** | **0.014** | **0.961** | **0.013** | **0.973** | **0.010** |
| | M2 | 0.883 | 0.035 | 0.929 | 0.026 | 0.934 | 0.024 | 0.948 | 0.020 |
| F | CCVAE | 0.734 | **0.025** | **0.806** | **0.024** | **0.801** | **0.028** | **0.798** | **0.029** |
| | M2 | **0.750** | 0.032 | 0.792 | 0.032 | 0.787 | 0.032 | 0.789 | 0.031 |

**Latent Traversals** We can also perform latent traversals for the multi-class setting. Here, we perform linear interpolation on the polytope where the corners are obtained from the network $\boldsymbol{\mu}_{\psi}(\mathbf{y})$ for four different classes. We provide the reconstructions in Figure A.14.

**Diversity in Conditional Generations** Here we show how we can introduce diversity in the conditional generations whilst keeping attributes such as pen-stroke and orientation constant. Inspecting the M2 results Figure A.15 and Figure A.16, where we have to sample from $\mathbf{z}$ to introduce diversity, indicates that we are unable to introduce diversity without affecting other attributes.

**Interventions** We can also perform interventions on individual classes, as showed in Figure A.17.

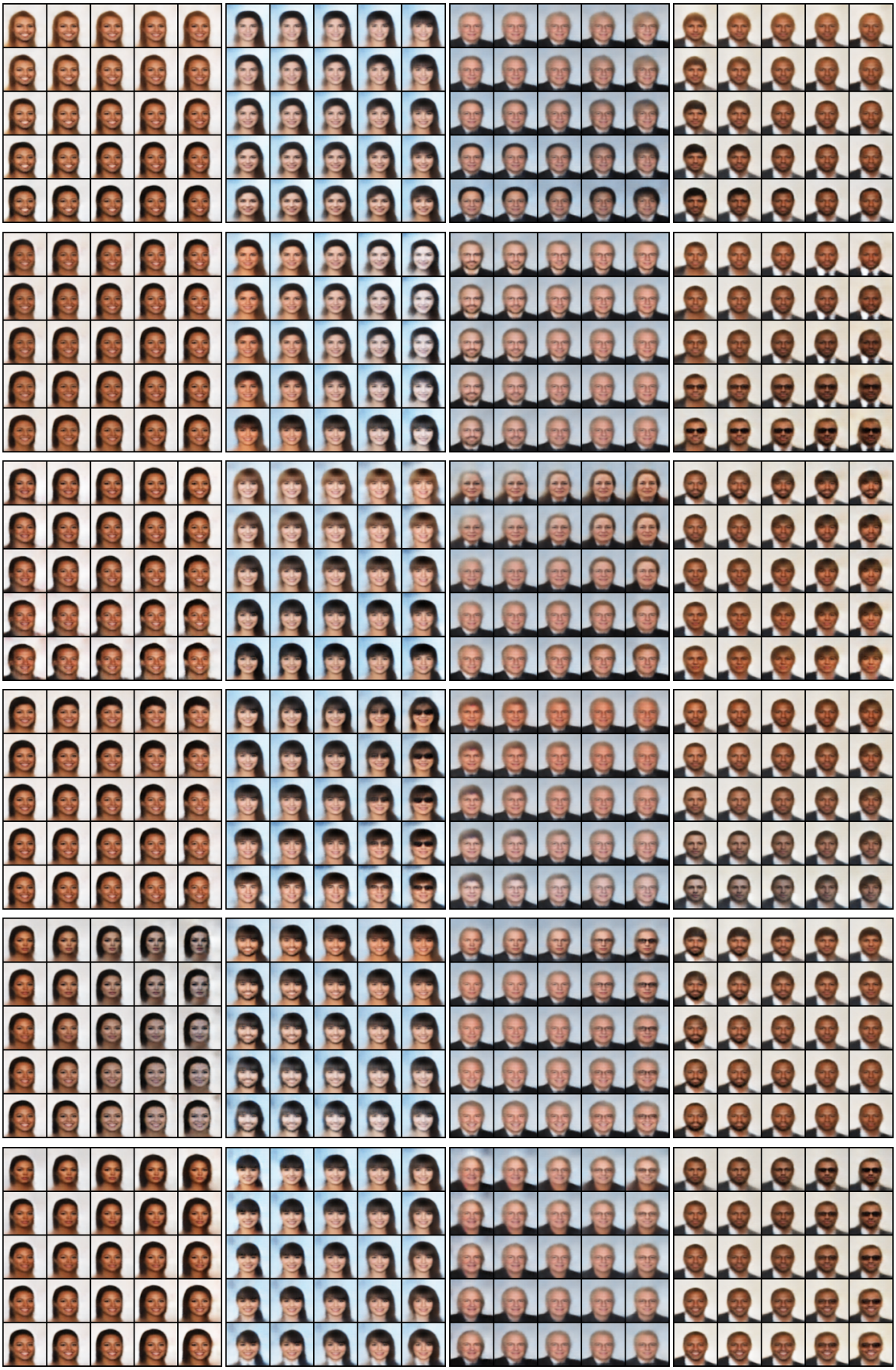


**Figure A.11:** Various latent traversals for CCVAE.

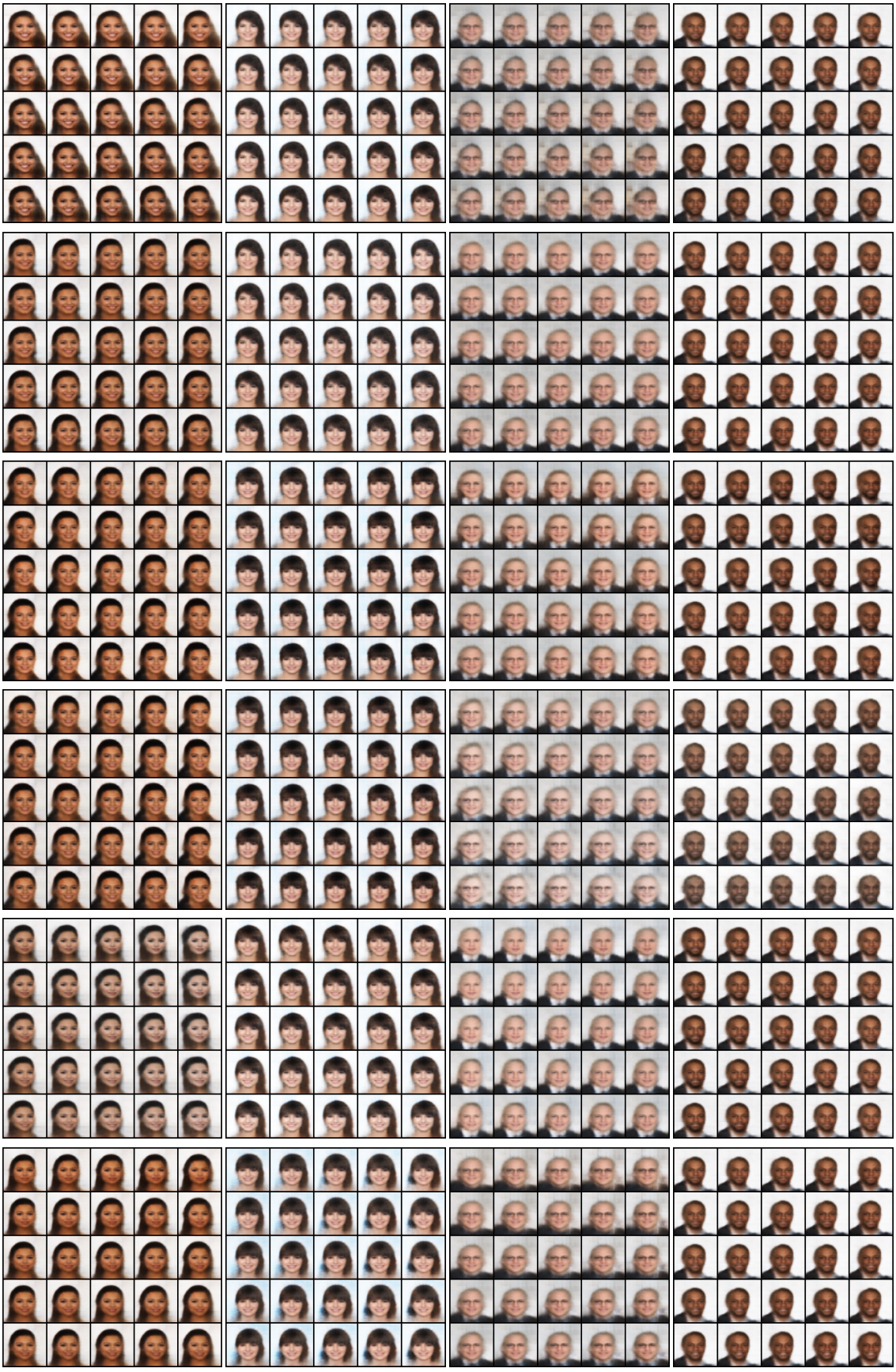


**Figure A.12:** Various latent traversals for DIVA.

**Figure A.13:** CCVAE, variance in reconstructions when intervening on a single label. From left to right: reconstruction, then interventions from switching on the following labels: `arched eyebrows`, `bags under eyes`, `bangs`, `black hair`, `blond hair`, `brown hair`, `bushy eyebrows`, `chubby`, `eyeglasses`, `heavy makeup`, `male`, `no beard`, `pale skin`, `receding hairline`, `smiling`, `wavy hair`, `wearing necktie`, `young`.

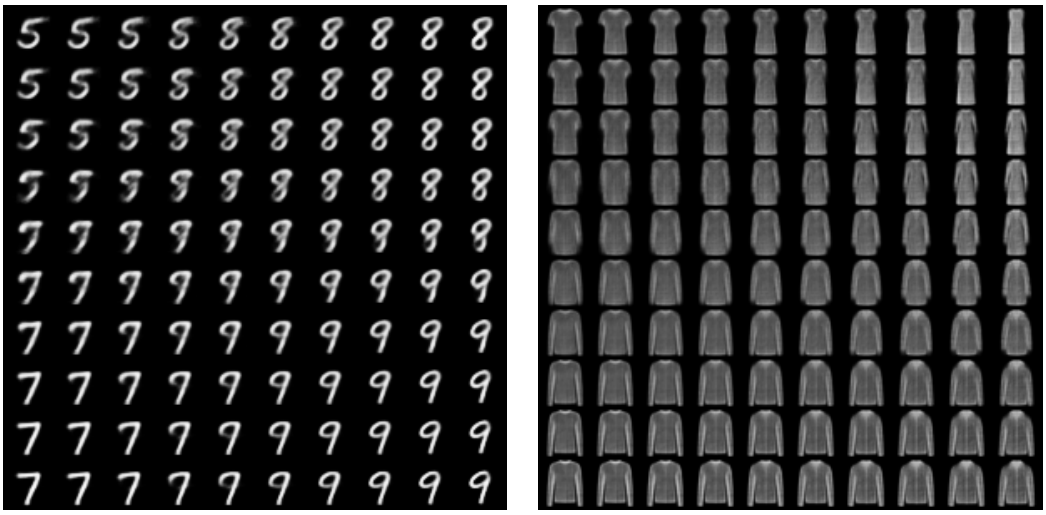


**Figure A.14:** CCVAE latent traversals for MNIST and FashionMNIST. It is interesting to see how one class transforms into another, e.g. for MNIST we see the end of the 5 curling around to form an 8 and a steady elongation of the torso when traversing from `t-shirt` to `dress`.

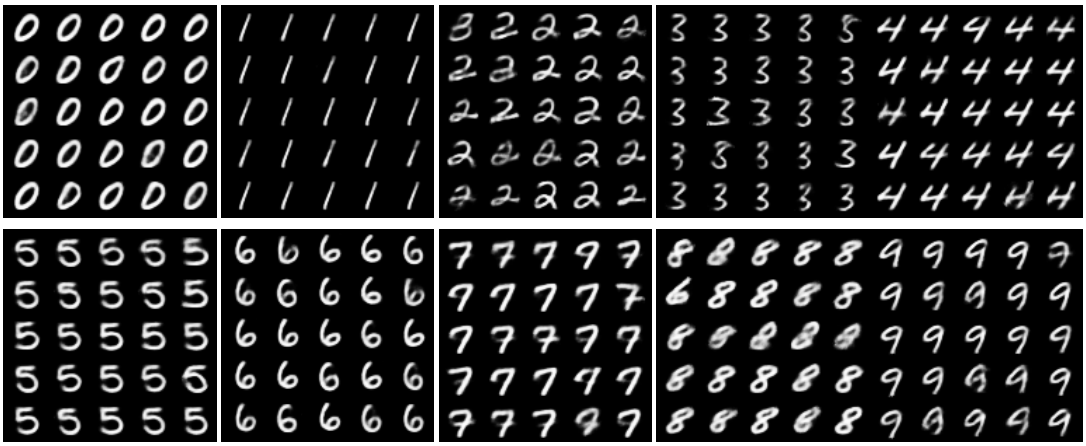


**Figure A.15:** CCVAE conditional generations with $\mathbf{z}_{\setminus \mathbf{c}}$ fixed. Here we can see that CCVAE is able to introduce diversity whilst preserving the "style" of the digit, e.g. pen width and tilt.

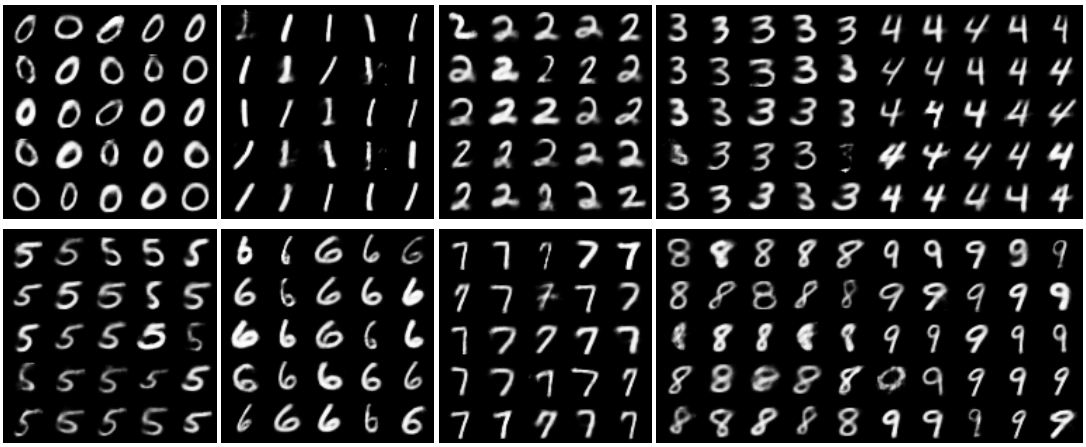


**Figure A.16:** M2 conditional generations. Here we can see that M2 is unable to introduce diversity without altering the "style" of the digit, e.g. pen width and tilt.

**Figure A.17:** Left: CCVAE, right: M2. As with other approaches, we can also perform wholesale interventions on each class whilst preserving the style.

# B

# Appendix: MEME

## B.1 Derivation of the Objective

The variational lower bound for the case when $\mathbf{s}$ and $\mathbf{t}$ are both observed derives as:

$$\begin{aligned}\log p(\mathbf{t},\mathbf{s}) &= \log \int_{\mathbf{z}} p(\mathbf{t},\mathbf{s},\mathbf{z}) d\mathbf{z} \\ &\geq \int_{\mathbf{z}} \log \frac{p(\mathbf{s},\mathbf{t},\mathbf{z})}{q(\mathbf{z}|\mathbf{t},\mathbf{s})} q(\mathbf{z}|\mathbf{t},\mathbf{s}) d\mathbf{z}\end{aligned}$$

Following Chapter 3, assuming $\mathbf{s} \perp\!\!\!\perp \mathbf{t}|\mathbf{z}$ and applying Bayes rule we have

$$q(\mathbf{z}|\mathbf{t},\mathbf{s}) = \frac{q(\mathbf{z}|\mathbf{s})q(\mathbf{t}|\mathbf{z})}{q(\mathbf{t}|\mathbf{s})},$$

which can be substituted into the lower bound to obtain

$$\begin{aligned}\log p(\mathbf{t},\mathbf{s}) &\geq \int_{\mathbf{z}} \log \frac{p(\mathbf{s},\mathbf{t},\mathbf{z})q(\mathbf{t}|\mathbf{s})}{q(\mathbf{z}|\mathbf{s})q(\mathbf{t}|\mathbf{z})} \frac{q(\mathbf{z}|\mathbf{s})q(\mathbf{t}|\mathbf{z})}{q(\mathbf{t}|\mathbf{s})} d\mathbf{z} \\ &= \mathbb{E}_{q(\mathbf{z}|\mathbf{s})}\left[\frac{q(\mathbf{t}|\mathbf{z})}{q(\mathbf{t}|\mathbf{s})} \log \frac{p(\mathbf{s}|\mathbf{z})(\mathbf{z}|\mathbf{t})}{q(\mathbf{z}|\mathbf{s})q(\mathbf{t}|\mathbf{z})}\right] + \log q(\mathbf{t}|\mathbf{s}) + \log p(\mathbf{t}). \qquad \text{(B.1)}\end{aligned}$$

## B.2 Efficient Gradient Estimation

Given the objective in (B.1), note that the first term is quite complex, and requires estimating a weight ratio that involves an additional integral through $q(\mathbf{t} \mid \mathbf{s}) =$

$\int q_\varphi(\mathbf{t} \mid \mathbf{z}) q_\phi(\mathbf{z} \mid \mathbf{s}) d\mathbf{z}$. This has a significant effect, as the naive Monte-Carlo gradient estimator can be very noisy, and prohibit learning effectively. To mitigate this, we make a relatively simple modification that ensures correctness of both objective and gradient, but has the benefit of simplifying some of the gradient computations to avoid noise. We compute the first term of (B.1) as

$$\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{s})}\left[\frac{q_\varphi(\mathbf{t} \mid \bar{\mathbf{z}})}{q(\mathbf{t} \mid \mathbf{s})} \log \frac{p_\theta(\mathbf{s} \mid \mathbf{z}) p_\vartheta(\mathbf{z} \mid \mathbf{t})}{q_\phi(\mathbf{z} \mid \mathbf{s}) q_\varphi(\mathbf{t} \mid \bar{\mathbf{z}})}\right], \tag{B.2}$$

where $\bar{\mathbf{z}}$ denotes a "detached" variable—i.e., one that disallows gradient propagation upstream (i.e., backwards) through its location in the associated compute graph.

We will now show that this formulation simply enforces that components that cancel or are analytically zero in the gradient estimator are not unnecessarily computed. Note that the gradient of (B.2) with respect to $\phi$ can be written (following reparametrisation of $\mathbf{z} = \rho(\epsilon, \phi)$) as

$$\begin{aligned}
&\nabla_\phi \, \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{s})}\left[\frac{q_\varphi(\mathbf{t} \mid \mathbf{z})}{q(\mathbf{t}|\mathbf{s})} \log \frac{p_\theta(\mathbf{s} \mid \mathbf{z}) p_\vartheta(\mathbf{z} \mid \mathbf{t})}{q_\phi(\mathbf{z} \mid \mathbf{s}) q_\varphi(\mathbf{t} \mid \mathbf{z})}\right] \\
&\quad = \mathbb{E}_{p(\epsilon)}\left[\left(\nabla_\phi \frac{q_\varphi(\mathbf{t} \mid \mathbf{z})}{q(\mathbf{t} \mid \mathbf{s})}\right) \log \frac{p_\theta(\mathbf{s} \mid \mathbf{z}) p_\vartheta(\mathbf{z} \mid \mathbf{t})}{q_\phi(\mathbf{z} \mid \mathbf{s}) q_\varphi(\mathbf{t} \mid \mathbf{z})} + \frac{q_\varphi(\mathbf{t} \mid \mathbf{z})}{q(\mathbf{t} \mid \mathbf{s})} \nabla_\phi \log \frac{p_\theta(\mathbf{s} \mid \mathbf{z}) p_\vartheta(\mathbf{z} \mid \mathbf{t})}{q_\phi(\mathbf{z} \mid \mathbf{s}) q_\varphi(\mathbf{t} \mid \mathbf{z})}\right].
\end{aligned} \tag{B.3}$$

In order to validate the "detached" formulation in (B.2), it suffices to show that the two instances of $\bar{\mathbf{z}}$ employed do not change the gradient estimator. These two instances neatly map onto the two terms of the gradient estimator in (B.3).

**Part A:** We show that

$$\mathbb{E}_{p(\epsilon)}\left[\frac{q_\varphi(\mathbf{t} \mid \mathbf{z})}{q(\mathbf{t} \mid \mathbf{s})} \nabla_\phi \log \frac{p_\theta(\mathbf{s} \mid \mathbf{z}) p_\vartheta(\mathbf{z} \mid \mathbf{t})}{q_\phi(\mathbf{z} \mid \mathbf{s}) q_\varphi(\mathbf{t} \mid \mathbf{z})}\right] = \mathbb{E}_{p(\epsilon)}\left[\frac{q_\varphi(\mathbf{t} \mid \mathbf{z})}{q(\mathbf{t} \mid \mathbf{s})} \nabla_\phi \log \frac{p_\theta(\mathbf{s} \mid \mathbf{z}) p_\vartheta(\mathbf{z} \mid \mathbf{t})}{q_\phi(\mathbf{z} \mid \mathbf{s})}\right].$$

This becomes apparent when considering the missing term in the log, where the relevant factor (after moving the independent $q(\mathbf{t} \mid \mathbf{s})$ out) is

$$\int p(\epsilon) q_\varphi(\mathbf{t} \mid \mathbf{z}) \nabla_\phi \log q_\varphi(\mathbf{t} \mid \mathbf{z}) d\epsilon = \int p(\epsilon) \nabla_\phi q_\varphi(\mathbf{t} \mid \mathbf{z}) d\epsilon = \nabla_\phi \int p(\epsilon) q_\varphi(\mathbf{t} \mid \mathbf{z}) d\epsilon = \nabla_\phi q_\varphi(\mathbf{t}) = 0$$

Using $\log q_\varphi(\mathbf{t} \mid \bar{\mathbf{z}})$ simply enforces this zero gradient directly.

**Part B:** We show that

$$\mathbb{E}_{p(\epsilon)}\left[\left(\nabla_\phi \frac{q_\varphi(\mathbf{t} \mid \mathbf{z})}{q(\mathbf{t} \mid \mathbf{s})}\right) \log \frac{p_\theta(\mathbf{s} \mid \mathbf{z}) p_\vartheta(\mathbf{z} \mid \mathbf{t})}{q_\phi(\mathbf{z} \mid \mathbf{s}) q_\varphi(\mathbf{t} \mid \mathbf{z})}\right] = 0.$$

To show this, we make the following observations:

$$\frac{p(\mathbf{s}|\mathbf{z})p(\mathbf{z}|\mathbf{t})}{q(\mathbf{z}|\mathbf{s})q(\mathbf{t}|\mathbf{z})} = \frac{p(\mathbf{s},\mathbf{z} \mid \mathbf{t})}{q(\mathbf{t},\mathbf{z} \mid \mathbf{s})} = \frac{p(\mathbf{s},\mathbf{z} \mid \mathbf{t})p(\mathbf{s})}{q(\mathbf{s},\mathbf{z} \mid \mathbf{t})p(\mathbf{t})} \approx \frac{q(\mathbf{s})}{p(\mathbf{t})}$$
$$\text{(at convergence } p(\mathbf{s},\mathbf{z} \mid \mathbf{t}) \approx q(\mathbf{s},\mathbf{z} \mid \mathbf{t}))$$

which means the term within log is independent of $\mathbf{z}$. Moreover, we expand the term $q(\mathbf{t} \mid \mathbf{s})$, denoting $z' = \rho(\epsilon', \phi)$, as

$$q(\mathbf{t} \mid \mathbf{s}) = \int q_\phi(\mathbf{z}' \mid \mathbf{s}) q_\varphi(\mathbf{t} \mid \mathbf{z}') d\mathbf{z}' = \int p(\epsilon') q_\varphi(\mathbf{t} \mid \rho(\epsilon', \phi)) d\epsilon' = \mathbb{E}_{p(\epsilon')}\left[q_\varphi(\mathbf{t} \mid z' = \rho(\epsilon', \phi))\right]$$

Using these terms, we can now simplify the gradient as follows:

$$\begin{aligned}
&\mathbb{E}_{p(\epsilon)}\left[\left(\nabla_\phi \frac{q_\varphi(\mathbf{t} \mid \mathbf{z})}{q(\mathbf{t} \mid \mathbf{s})}\right) \log \frac{p_\theta(\mathbf{s} \mid \mathbf{z}) p_\vartheta(\mathbf{z} \mid \mathbf{t})}{q_\phi(\mathbf{z} \mid \mathbf{s}) q_\varphi(\mathbf{t} \mid \mathbf{z})}\right] \\
&= \mathbb{E}_{p(\epsilon)}\left[\left(\frac{q(\mathbf{t} \mid \mathbf{s})\nabla_\phi q_\varphi(\mathbf{t} \mid \mathbf{z}) - q_\varphi(\mathbf{t} \mid \mathbf{z})\nabla_\phi q(\mathbf{t} \mid \mathbf{s})}{q(\mathbf{t} \mid \mathbf{s})^2}\right) \log \frac{q(\mathbf{s})}{p(\mathbf{t})}\right] \\
&= \frac{1}{q(\mathbf{t} \mid \mathbf{s})^2} \log \frac{q(\mathbf{s})}{p(\mathbf{t})}\ \mathbb{E}_{p(\epsilon)}[q(\mathbf{t} \mid \mathbf{s})\nabla_\phi q_\varphi(\mathbf{t} \mid \mathbf{z}) - q_\varphi(\mathbf{t} \mid \mathbf{z})\nabla_\phi q(\mathbf{t} \mid \mathbf{s})] \\
&= \frac{1}{q(\mathbf{t} \mid \mathbf{s})^2} \log \frac{q(\mathbf{s})}{p(\mathbf{t})}\ \left[q(\mathbf{t} \mid \mathbf{s})\mathbb{E}_{p(\epsilon)}[\nabla_\phi q_\varphi(\mathbf{t} \mid \mathbf{z})] - \mathbb{E}_{p(\epsilon)}[q_\varphi(\mathbf{t} \mid \mathbf{z})]\ \nabla_\phi q(\mathbf{t} \mid \mathbf{s})\right] \\
&= \frac{1}{q(\mathbf{t} \mid \mathbf{s})^2} \log \frac{q(\mathbf{s})}{p(\mathbf{t})}\ \left[q(\mathbf{t} \mid \mathbf{s})\nabla_\phi \mathbb{E}_{p(\epsilon)}[q_\varphi(\mathbf{t} \mid \mathbf{z})] - \mathbb{E}_{p(\epsilon)}[q_\varphi(\mathbf{t} \mid \mathbf{z})]\ \nabla_\phi q(\mathbf{t} \mid \mathbf{s})\right] \\
&= \frac{1}{q(\mathbf{t} \mid \mathbf{s})^2} \log \frac{q(\mathbf{s})}{p(\mathbf{t})}\ \left[q(\mathbf{t} \mid \mathbf{s})\nabla_\phi q(\mathbf{t} \mid \mathbf{s}) - q(\mathbf{t} \mid \mathbf{s})\nabla_\phi q(\mathbf{t} \mid \mathbf{s})\right] \\
&= 0
\end{aligned}$$

Taken together, these justify the simplification proposed in (B.2). We note that in Chapter 3 we perform the same modification as in (B.2), motivated by an empirical study, but here, we prove that this change is both sound and correct, and does not introduce additional any bias. We plot the signal-to-noise ratios (SNR) for the "detached" case (blue) and the naive case (orange) in Figure B.1, which highlights the effect of the simplification.

## B.3 High Variance of the gradient estimator

During training, the term $\frac{q(\mathbf{t}|\mathbf{z})}{q(\mathbf{t}|\mathbf{s})}$ incurs a very low signal to noise ratio, which becomes untenable for learning. To combat this, we formulate an alternate estimate of the gradient of the objective, enabling us to learn effectively.

The gradient of the first term in the objective wrt $\phi$ is given as

$$\nabla_\phi \mathcal{L}(\mathbf{s},\mathbf{t};\phi,\varphi,\theta,\vartheta) = \mathbb{E}_{p(\epsilon)}\left[\left(\nabla_\phi \frac{q(\mathbf{t}|\mathbf{z})}{q(\mathbf{t}|\mathbf{s})}\right)\log\frac{p(\mathbf{s}|\mathbf{z})p(\mathbf{z}|\mathbf{t})}{q(\mathbf{z}|\mathbf{s})q(\mathbf{t}|\mathbf{z})} + \frac{q(\mathbf{t}|\mathbf{z})}{q(\mathbf{t}|\mathbf{s})}\nabla_\phi \log\frac{p(\mathbf{s}|\mathbf{z})p(\mathbf{z}|\mathbf{t})}{q(\mathbf{z}|\mathbf{s})q(\mathbf{t}|\mathbf{z})}\right] \tag{B.4}$$

with $\mathbf{z} = t(\epsilon,\mathbf{s};\phi)$ (reparameterization). We observe that the first term has a variance which is too high to learn anything meaningful from. Fortunately, we note that, under certain conditions, the expected value of the gradient in the first term is zero, which means it can be removed (Roeder et al., 2017) through a judicious application of a "stop gradient" on $\frac{q(\mathbf{t}|\mathbf{z})}{q(\mathbf{t}|\mathbf{s})}$.

Using reparemterisation, with $\mathbf{z} = t(\epsilon)$ and denoting $\log\frac{p(\mathbf{s}|\mathbf{z})p(\mathbf{z}|\mathbf{t})p(\mathbf{t})}{q(\mathbf{z}|\mathbf{s})q(\mathbf{t}|\mathbf{z})}$ as $\log A$ for convenience, the first term of (B.4) can be expanded as

$$\begin{aligned}&\mathbb{E}_{\epsilon\sim g(\epsilon)}\left[\frac{\int q_\varphi(\mathbf{t}\mid\mathbf{z})q_\phi(\mathbf{z}\mid\mathbf{s})d\mathbf{z}\nabla_{\mathbf{z}}q_\varphi(\mathbf{t}|\mathbf{z})\nabla_\phi t(\epsilon)}{[\int q_\varphi(\mathbf{t}\mid\mathbf{z})q_\phi(\mathbf{z}\mid\mathbf{s})d\mathbf{z}]^2}\log\frac{p_\theta(\mathbf{s}|\mathbf{z})p_\vartheta(\mathbf{z}|\mathbf{t})}{q_\varphi(\mathbf{t}|\mathbf{z})q_\phi(\mathbf{z}|\mathbf{s})}\right] \\ &-\mathbb{E}_{\epsilon\sim g(\epsilon)}\left[\frac{q_\varphi(\mathbf{t}|\mathbf{z})\nabla_\phi\int q_\varphi(\mathbf{t}\mid\mathbf{z})q_\phi(\mathbf{z}\mid\mathbf{s})d\mathbf{z}}{[\int q_\varphi(\mathbf{t}\mid\mathbf{z})q_\phi(\mathbf{z}\mid\mathbf{s})d\mathbf{z}]^2}\log\frac{p_\theta(\mathbf{s}|\mathbf{z})p_\vartheta(\mathbf{z}|\mathbf{t})}{q_\varphi(\mathbf{t}|\mathbf{z})q_\phi(\mathbf{z}|\mathbf{s})}\right]\end{aligned} \tag{B.5}$$

$$\begin{aligned}&\mathbb{E}_{\epsilon\sim g(\epsilon)}\left[\frac{\int q_\varphi(\mathbf{t}\mid\mathbf{z})q_\phi(\mathbf{z}\mid\mathbf{s})d\mathbf{z}\nabla_{\mathbf{z}}q_\varphi(\mathbf{t}|\mathbf{z})\nabla_\phi t(\epsilon)}{[\int q_\varphi(\mathbf{t}\mid\mathbf{z})q_\phi(\mathbf{z}\mid\mathbf{s})d\mathbf{z}]^2}\log\frac{p_\theta(\mathbf{s}|\mathbf{z})p_\vartheta(\mathbf{z}|\mathbf{t})}{q_\varphi(\mathbf{t}|\mathbf{z})q_\phi(\mathbf{z}|\mathbf{s})}\right] \\ &-\mathbb{E}_{\epsilon\sim g(\epsilon)}\left[\frac{q_\varphi(\mathbf{t}|\mathbf{z})\nabla_\phi\int q_\varphi(\mathbf{t}|\mathbf{z}')g(\alpha)d\alpha}{[\int q_\varphi(\mathbf{t}\mid\mathbf{z})q_\phi(\mathbf{z}\mid\mathbf{s})d\mathbf{z}]^2}\log\frac{p_\theta(\mathbf{s}|\mathbf{z})p_\vartheta(\mathbf{z}|\mathbf{t})}{q_\varphi(\mathbf{t}|\mathbf{z})q_\phi(\mathbf{z}|\mathbf{s})}\right]\end{aligned} \tag{B.6}$$

with $\mathbf{z}' = t(\alpha)$

$$\begin{aligned}&\mathbb{E}_{\epsilon\sim g(\epsilon)}\left[\frac{\int q_\varphi(\mathbf{t}|t(\alpha))g(\alpha)d\alpha\nabla_{\mathbf{z}}q_\varphi(\mathbf{t}|\mathbf{z})\nabla_\phi t(\epsilon)}{[\int q_\varphi(\mathbf{t}\mid\mathbf{z})q_\phi(\mathbf{z}\mid\mathbf{s})d\mathbf{z}]^2}\log\frac{p_\theta(\mathbf{s}|\mathbf{z})p_\vartheta(\mathbf{z}|\mathbf{t})}{q_\varphi(\mathbf{t}|\mathbf{z})q_\phi(\mathbf{z}|\mathbf{s})}\right] \\ &-\mathbb{E}_{\epsilon\sim g(\epsilon)}\left[\frac{q_\varphi(\mathbf{t}|\mathbf{z})\int\nabla_{\mathbf{z}'}q_\varphi(\mathbf{t}|\mathbf{z}')\nabla_\phi t(\alpha)d\alpha}{[\int q_\varphi(\mathbf{t}\mid\mathbf{z})q_\phi(\mathbf{z}\mid\mathbf{s})d\mathbf{z}]^2}\log\frac{p_\theta(\mathbf{s}|\mathbf{z})p_\vartheta(\mathbf{z}|\mathbf{t})}{q_\varphi(\mathbf{t}|\mathbf{z})q_\phi(\mathbf{z}|\mathbf{s})}\right]\end{aligned} \tag{B.7}$$

$$\int \mathbb{E}_{\epsilon\sim g(\epsilon)}\left[\frac{\int q_\varphi(\mathbf{t}|t(\alpha))g(\alpha)d\alpha\nabla_{\mathbf{z}}q_\varphi(\mathbf{t}|\mathbf{z})\nabla_\phi t(\epsilon)}{[\int q_\varphi(\mathbf{t}\mid\mathbf{z})q_\phi(\mathbf{z}\mid\mathbf{s})d\mathbf{z}]^2}\log\frac{p_\theta(\mathbf{s}|\mathbf{z})p_\vartheta(\mathbf{z}|\mathbf{t})}{q_\varphi(\mathbf{t}|\mathbf{z})q_\phi(\mathbf{z}|\mathbf{s})}\right]d\alpha \tag{B.8}$$
$$-\int \mathbb{E}_{\epsilon\sim g(\epsilon)}\left[\frac{q_\varphi(\mathbf{t}|\mathbf{z})\int\nabla_{\mathbf{z}'}q_\varphi(\mathbf{t}|\mathbf{z}')\nabla_\phi t(\alpha)d\alpha}{[\int q_\varphi(\mathbf{t}\mid\mathbf{z})q_\phi(\mathbf{z}\mid\mathbf{s})d\mathbf{z}]^2}\log\frac{p_\theta(\mathbf{s}|\mathbf{z})p_\vartheta(\mathbf{z}|\mathbf{t})}{q_\varphi(\mathbf{t}|\mathbf{z})q_\phi(\mathbf{z}|\mathbf{s})}\right]d\alpha$$

When the true posterior $p(\mathbf{z}|\mathbf{s})$ matches the approximate posterior $q_\phi(\mathbf{z}\mid\mathbf{s})$ and the predictive distribution $q_\varphi(\mathbf{t}\mid\mathbf{z})$ matches the true distribution $p(\mathbf{t}|\mathbf{z})$, by applying Bayes rule, the term inside the log is equivalent to $p(\mathbf{s})$, which is independent of $\epsilon$ and $\alpha$.

$$\int\int\left[\frac{q_\varphi(\mathbf{t}|g(\alpha))q(\alpha)\nabla_{g(\epsilon)}q_\varphi(\mathbf{t}|g(\epsilon))\nabla_\phi g(\epsilon)q(\epsilon)}{[\int q_\varphi(\mathbf{t}\mid\mathbf{z})q_\phi(\mathbf{z}\mid\mathbf{s})d\mathbf{z}]^2}\right]d\epsilon d\alpha\log p(\mathbf{s}) \tag{B.9}$$
$$-\int\int\left[\frac{q_\varphi(\mathbf{t}|g(\epsilon))q(\epsilon)\nabla_{g(\alpha)}q_\varphi(\mathbf{t}|g(\alpha))\nabla_\phi g(\alpha)q(\alpha)}{[\int q_\varphi(\mathbf{t}\mid\mathbf{z})q_\phi(\mathbf{z}\mid\mathbf{s})d\mathbf{z}]^2}\right]d\epsilon d\alpha\log p(\mathbf{s})$$

Which subsequently equals zero, leading to our choice of removing this term from the gradient. We are aware that this approach leads to a biased estimator and is only applicable under certain conditions, but we empirically observe that it is essential to training. We plot the resulting SNR ratios for the case when we apply the stop gradient (blue) and when do not (orange) in Figure B.1.

This modification can be viewed as the control variate strategy below

$$\hat{f}(\mathbf{z}) \coloneqq f(\mathbf{z}) - \alpha(h(\mathbf{z}) - \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{s})}[h(\mathbf{z})]), \tag{B.10}$$

with $\alpha = 1$ and $\mathbb{E}[f(\mathbf{z})] = \mathbb{E}[\hat{f}(\mathbf{z})]$ as required. Here, the definitions of $f(\mathbf{z})$ and $h(\mathbf{z})$ are

$$f(\mathbf{z}) = \nabla_{\phi,\varphi}\mathbb{E}_{q(\mathbf{z}|\mathbf{s})}\left[\frac{q(\mathbf{t}|\mathbf{z})}{q(\mathbf{t}|\mathbf{s})}\log\frac{p(\mathbf{s}|\mathbf{z})p(\mathbf{z}|\mathbf{t})p(\mathbf{t})}{q(\mathbf{z}|\mathbf{s})q(\mathbf{t}|\mathbf{z})}\right] + \nabla_{\phi,\varphi}q_{\varphi,\phi}(\mathbf{t}\mid\mathbf{s}) \tag{B.11}$$

$$h(\mathbf{z}) = \nabla_{\phi,\varphi}\left[\frac{q(\mathbf{t}|\mathbf{z})}{q(\mathbf{t}|\mathbf{s})}\right]\log\frac{p(\mathbf{s}|\mathbf{z})p(\mathbf{z}|\mathbf{t})p(\mathbf{t})}{q(\mathbf{z}|\mathbf{s})q(\mathbf{t}|\mathbf{z})}. \tag{B.12}$$

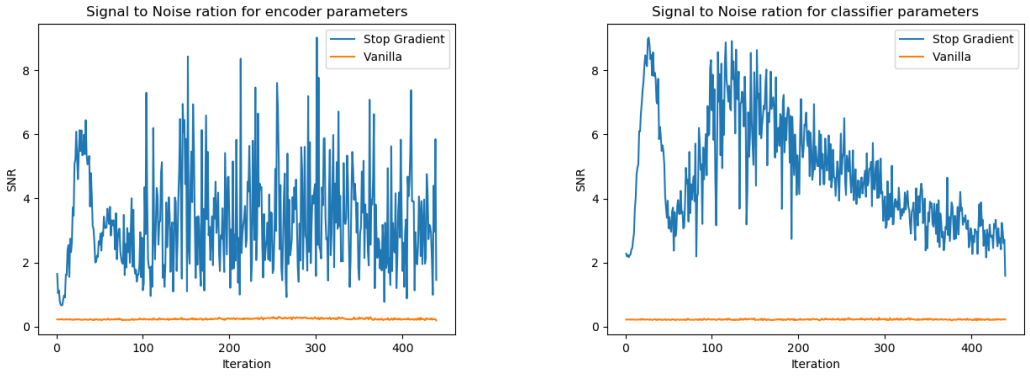



**Figure B.1:** SNR for encoder parameters (Left) and classifier parameters (Right), blue indicates that we apply the stop gradient in Appendix B.3, orange indicates we do not. A higher value typically leads to improved learning.

## B.4 Weight Sharing

Another critical issue with naïvely training using (B.1), is that in certain situations $q_\varphi(\mathbf{t} \mid \mathbf{z})$ struggles to learn features (typically style) for $\mathbf{t}$, consequently making it difficult to generate realistic samples. This is due to the information entering the latent space only coming from $\mathbf{s}$, which contains all of the information needed to reconstruct $\mathbf{s}$, but does not necessarily contain the information needed to reconstruct a corresponding $\mathbf{t}$. Consequently, the term $p_\theta(\mathbf{s} \mid \mathbf{z})$ will learn appropriate features (like a standard VAE decoder), but the term $q_\varphi(\mathbf{t} \mid \mathbf{z})$ will fail to do so. In situations like this, where the information in $\mathbf{t}$ is *not* subsumed by the information in $\mathbf{s}$, there is no way for the model to know how to reconstruct a $\mathbf{t}$. Introducing weight sharing into the bidirectional objective (4.2) removes this issue, as there is equal opportunity for information from both modalities to enter the latent space, consequently enabling appropriate features to be learned in the decoders $p_\theta(\mathbf{s} \mid \mathbf{z})$ and $p_\varphi(\mathbf{t} \mid \mathbf{z})$, which subsequently allow cross generations to be performed.

Furthermore, we also observe that when training with (4.2) we are able to obtain much more balanced likelihoods Table B.1. In this setting we train two models separately using (B.1) with $\mathbf{s}$ = MNIST and SVHN and then with $\mathbf{t}$ = SVHN and $\mathbf{s}$ = MNIST respectively. At test time, we then 'flip' the modalities and the corresponding networks, allowing us to obtain marginal likelihoods in each direction.

Clearly we see that we only obtain reasonable marginal likelihoods in the direction for which we train. Training with the bidirectional objective completely removes this deficiency, as we now introduce a balance between the modalities.

**Table B.1:** Marginal likelihoods.

| | Train Direction | | |
|---|---|---|---|
| Test Direction | $\mathbf{s} = \mathrm{M}, \mathbf{t} = \mathrm{S}$ | $\mathbf{s} = \mathrm{S}, \mathbf{t} = \mathrm{M}$ | Bi |
| $\mathbf{s} = \mathrm{M}, \mathbf{t} = \mathrm{S}$ | $-14733.6$ | $-40249.9^{\text{flip}}$ | $-14761.3$ |
| $\mathbf{s} = \mathrm{S}, \mathbf{t} = \mathrm{M}$ | $-428728.7^{\text{flip}}$ | $-11668.1$ | $-11355.4$ |

## B.5 Reusing Approximate Posterior MC Sample

When approximating $q_{\varphi,\phi}(\mathbf{t} \mid \mathbf{s})$ through MC sampling, we find that it is essential for numerical stability to include the sample from the approximate posterior. Before considering why, we must first outline the numerical implementation of $q_{\varphi,\phi}(\mathbf{t} \mid \mathbf{s})$, which for $K$ samples $\mathbf{z}_{1:K} \sim q_\phi(\mathbf{z} \mid \mathbf{s})$ is computed using the LogSumExp trick as:

$$\log q_{\varphi,\phi}(\mathbf{t} \mid \mathbf{s}) \approx \log \sum_{k=1}^{K} \exp \log q_\varphi(\mathbf{t}|\mathbf{z}_k), \tag{B.13}$$

where the ratio $\frac{q_\varphi(\mathbf{t}|\mathbf{z})}{q_{\varphi,\phi}(\mathbf{t}|\mathbf{s})}$ is computed as $\exp\{\log q_\varphi(\mathbf{t} \mid \mathbf{z}) - \log q_{\varphi,\phi}(\mathbf{t} \mid \mathbf{s})\}$. Given that the LogSumExp trick is defined as:

$$\log \sum_{n=1}^{N} \exp x_n = x^* + \log \sum_{n=1}^{N} \exp(x_n - x^*), \tag{B.14}$$

where $x^* = \max\{x_1, \ldots, x_N\}$. The ratio will be computed as

$$\frac{q_\varphi(\mathbf{t} \mid \mathbf{z})}{q_{\varphi,\phi}(\mathbf{t} \mid \mathbf{s})} = \exp\{\log q_\varphi(\mathbf{t} \mid \mathbf{z}) - \log q_\varphi(\mathbf{t}|\mathbf{z}^*) - \log \sum_{k=1}^{K} \exp[\log q_\varphi(\mathbf{t}|\mathbf{z}_k) - \log q_\varphi(\mathbf{t}|\mathbf{z}^*)]\}, \tag{B.15}$$

where $\mathbf{z}^* = \arg\max_{\mathbf{z}_{1:K}} \log q_\varphi(\mathbf{t}|\mathbf{z}_k)$. For numerical stability, we require that $\log q_\varphi(\mathbf{t} \mid \mathbf{z}) \not\gg \log q_\varphi(\mathbf{t}|\mathbf{z}^*)$, otherwise the computation may blow up when taking the exponent. To enforce this, we need to include the sample $\mathbf{z}$ into the LogSumExp function, doing so will cause the first two terms to either cancel if $\mathbf{z} = \mathbf{z}^*$ or yield a negative value, consequently leading to stable computation when taking the exponent.

## B.6 Extension beyond the Bi-Modal case

Here we offer further detail on how **M**utually sup**E**rvised **M**ultimodal VA**E** (MEME) can be extended beyond the bi-modal case, i.e. when the number of modalities $M > 2$. Note that the central thesis in MEME is that the evidence lower bound (ELBO) offers an implicit way to regularise different representations if viewed from the posterior-prior perspective, which can be used to build effective multimodal DGMs that are additionally applicable to partially-observed data. In MEME, we explore the utility of this implicit regularisation in the simplest possible manner to show that a direct application of this to the multi-modal setting would involve the case where $M = 2$.

The way to extend, say for $M = 3$, involves additionally employing an explicit combination for two modalities in the prior (instead of just 1). This additional combination could be something like a mixture or product, following from previous approaches. More formally, if we were to denote the implicit regularisation between posterior and prior as $R_i(.,.)$, and an explicit regularisation function $R_e(.,.)$, and the three modalities as $m_1, m_2$, and $m_3$, this would mean we would compute

$$\frac{1}{3}\left[R_i(m_1, R_e(m_2, m_3)) + R_i(m_2, R_e(m_1, m_3)) + R_i(m_3, R_e(m_1, m_2))\right], \tag{B.16}$$

assuming that $R_e$ was commutative, as is the case for products and mixtures. There are indeed more terms to compute now compared to $M = 2$, which only needs $R_i(m_1, m_2)$, but note that $R_i$ is still crucial—it does not diminish because we are additionally employing $R_e$.

As stated in prior work(Suzuki et al., 2016; Wu and Goodman, 2018b; Shi et al., 2019a), we follow the reasoning that the actual number of modalities, at least when considering embodied perception, is not likely to get much larger, so the increase in number of terms, while requiring more computation, is unlikely to become intractable. Note that prior work on multimodal VAEs also suffer when extending the number of modalities in terms of the number of paths information flows through.

We do not explore this setting empirically as our priary goal is to highlight the utility of this implicit regularisation for multi-modal DGMs, and its effectiveness at handling partially-observed data.

## B.7 Closed Form expression for Wassertein Distance between two Gaussians

The Wassertein-2 distance between two probability measures $\mu$ and $\nu$ on $\mathbb{R}^n$ is defined as

$$\mathcal{W}_2(\mu, \nu) \coloneqq \inf \mathbb{E}(||X - Y||_2^2)^{\frac{1}{2}},$$

with $X \sim \mu$ and $Y \sim \nu$. Given $\mu = \mathcal{N}(m_1, \Sigma_1)$ and $\nu = \mathcal{N}(m_2, \Sigma_2)$, the 2-Wassertein is then given as

$$d^2 = ||m_1 + m_2||_2^2 + \text{Tr}(\Sigma_1 + \Sigma_2 - 2(\Sigma_1^{\frac{1}{2}}\Sigma_2\Sigma_1^{\frac{1}{2}})^{\frac{1}{2}}).$$

For a detailed proof please see Givens and Shortt (2002).

## B.8 Canonical Correlation Analysis

Following Shi et al. (2019a); Massiceti et al. (2018), we report cross-coherence scores for CUB using Canonical Correlation Analysis (CCA). Given paired observations $\mathbf{x}_1 \in \mathbb{R}_1^n$ and $\mathbf{x}_2 \in \mathbb{R}_2^n$, CCA learns projection weights $W_1^T \in \mathbb{R}^{n_1 \times k}$ and $W_2^T \in \mathbb{R}^{n_2 \times k}$ which minimise the correlation between the projections $W_1^T\mathbf{x}_1$ and $W_2^T\mathbf{x}_2$. The correlations between a data pair $\{\tilde{\mathbf{x}}_1, \tilde{\mathbf{x}}_2\}$ can thus be calculated as

$$\text{corr}(\tilde{\mathbf{x}}_1, \tilde{\mathbf{x}}_2) = \frac{\phi(\tilde{\mathbf{x}}_1)^T\phi(\tilde{\mathbf{x}}_2)}{||\phi(\tilde{\mathbf{x}}_1)||_2||\phi(\tilde{\mathbf{x}}_2)||_2} \tag{B.17}$$

where $\phi(\mathbf{x}_n) = W_n^T\tilde{\mathbf{x}_n} - \text{avg}(W_n^T\tilde{\mathbf{x}_n})$.

Following Shi et al. (2019a), we use feature extractors to pre-process the data. Specifically, features for image data are generated from an off-the-shelf ResNet-101 network. For text data, we first fit a FastText model on all sentences, resulting in a 300-$d$ projection for each word (Bojanowski et al., 2017), the representation is then computed as the average over the words in the sentence.

**Figure B.2:** MNIST → SVHN (Left) and SVHN → MNIST (Right), for the fully observed case.

**Figure B.3:** MNIST → SVHN (Left) and SVHN → MNIST (Right), when SVHN is observed 50% of the time.

**Figure B.4:** MNIST → SVHN (Left) and SVHN → MNIST (Right), when MNIST is observed 50% of the time.

**Figure B.5:** MNIST → SVHN (Left) and SVHN → MNIST (Right), when SVHN is observed 25% of the time.

**Figure B.6:** MNIST → SVHN (Left) and SVHN → MNIST (Right), when MNIST is observed 25% of the time.

**Figure B.7:** MNIST → SVHN (Left) and SVHN → MNIST (Right), when SVHN is observed 12.5% of the time.

**Figure B.8:** MNIST → SVHN (Left) and SVHN → MNIST (Right), when MNIST is observed 12.5% of the time.

## B.9 Additional Results

### B.9.1 MVAE Latent Accuracies

The superior accuracy in latent accuracy when classifying MNIST from MVAE is due to a complete failure to construct a joint representation, which is evidenced in its failure to perform cross-generation. Failure to construct joint representations aids latent classification, as the encoders just learn to construct representations for single modalities, this then provides more flexibility and hence better classification. In Figure B.10, we further provide a t-SNE plot to demonstrate that MVAE places representations for MNIST modality in completely different parts of the latent space to SVHN. Here we can see that representations for each modality are completely separated, meaning that there is no shared representation. Furthermore, MNIST is well clustered, unlike SVHN. Consequently it is far easier for the classifier to predict the MNIST digit as the representations do not contain any information associated with SVHN.

### B.9.2 Generative Capability

We report the mutual information between the parameters $\omega$ of a pre-trained classifier and the labels $y$ for a corresponding reconstruction $\mathbf{x}$. The mutual information

**Table B.2:** Coherence Scores for MNIST → SVHN (Top) and for SVHN → MNIST (Bottom). Subscript indicates which modality is always present during training, $f$ indicates the percentage of matched samples. Higher is better.

| | MNIST → SVHN | | | | |
|---|---|---|---|---|---|
| Model | $f = 1.0$ | $f = 0.5$ | $f = 0.25$ | $f = 0.125$ | $f = 0.0625$ |
| $\text{MEME}_{\text{SVHN}}$ | **0.625 ± 0.007** | **0.551 ± 0.008** | **0.323 ± 0.025** | **0.172 ± 0.016** | **0.143 ± 0.009** |
| $\text{MMVAE}_{\text{SVHN}}$ | 0.581 ± 0.008 | - | - | - | - |
| $\text{MVAE}_{\text{SVHN}}$ | 0.123 ± 0.003 | 0.110 ± 0.014 | 0.112 ± 0.005 | 0.105 ± 0.005 | 0.105 ± 0.006 |
| $\text{MEME}_{\text{MNIST}}$ | **0.625 ± 0.007** | **0.572 ± 0.003** | **0.485 ± 0.013** | **0.470 ± 0.009** | **0.451 ± 0.011** |
| $\text{MMVAE}_{\text{MNIST}}$ | 0.581 ± 0.008 | - | - | - | - |
| $\text{MVAE}_{\text{MNIST}}$ | 0.123 ± 0.003 | 0.111 ± 0.007 | 0.112 ± 0.013 | 0.116 ± 0.012 | 0.116 ± 0.005 |
| $\text{MEME}_{\text{SPLIT}}$ | **0.625 ± 0.007** | 0.625 ± 0.008 | 0.503 ± 0.008 | 0.467 ± 0.013 | 0.387 ± 0.010 |
| $\text{MVAE}_{\text{SPLIT}}$ | 0.123 ± 0.003 | 0.108 ± 0.005 | 0.101 ± 0.005 | 0.101 ± 0.001 | 0.101 ± 0.002 |

| | SVHN → MNIST | | | | |
|---|---|---|---|---|---|
| Model | $f = 1.0$ | $f = 0.5$ | $f = 0.25$ | $f = 0.125$ | $f = 0.0625$ |
| $\text{MEME}_{\text{SVHN}}$ | **0.752 ± 0.004** | **0.726 ± 0.006** | **0.652 ± 0.008** | **0.557 ± 0.018** | **0.477 ± 0.012** |
| $\text{MMVAE}_{\text{SVHN}}$ | 0.735 ± 0.010 | - | - | - | - |
| $\text{MVAE}_{\text{SVHN}}$ | 0.498 ± 0.100 | 0.305 ± 0.011 | 0.268 ± 0.010 | 0.220 ± 0.020 | 0.188 ± 0.012 |
| $\text{MEME}_{\text{MNIST}}$ | **0.752 ± 0.004** | **0.715 ± 0.003** | **0.603 ± 0.018** | **0.546 ± 0.012** | **0.446 ± 0.008** |
| $\text{MMVAE}_{\text{MNIST}}$ | 0.735 ± 0.010 | - | - | - | - |
| $\text{MVAE}_{\text{MNIST}}$ | 0.498 ± 0.100 | 0.365 ± 0.014 | 0.350 ± 0.008 | 0.302 ± 0.015 | 0.249 ± 0.014 |
| $\text{MEME}_{\text{SPLIT}}$ | **0.752 ± 0.004** | 0.718 ± 0.002 | 0.621 ± 0.007 | 0.568 ± 0.014 | 0.503 ± 0.001 |
| $\text{MVAE}_{\text{SPLIT}}$ | 0.498 ± 0.100 | 0.338 ± 0.013 | 0.273 ± 0.003 | 0.249 ± 0.019 | 0.169 ± 0.001 |

gives us an indication of the amount of information we would gain about $\omega$ for a label $y$ given $\mathbf{x}$, this provides an indicator to how *out-of-distribution* $\mathbf{x}$ is. If $\mathbf{x}$ is a realistic reconstruction, then there will be a low MI, conversely, an un-realistic $\mathbf{x}$ will manifest as a high MI as there is a large amount of information to be gained about $\omega$. The MI for this setting is given as

$$I(y, \omega \mid \mathbf{x}, \mathcal{D}) = H[p(y \mid \mathbf{x}, \mathcal{D})] - \mathbb{E}_{p(\omega|\mathcal{D})}\left[H[p(y \mid \mathbf{x}, \omega)]\right].$$

Rather than using dropout (Gal, 2016; Smith and Gal, 2018) which requires an ensemble of multiple classifiers, we instead replace the last layer with a sparse variational GP. This allows us to estimate $p(y \mid x, \mathcal{D}) = \int p(y \mid x, \omega) p(\omega \mid \mathcal{D}) \mathrm{d}\omega$ using Monte Carlo samples and similarly estimate $\mathbb{E}_{p(\omega|\mathcal{D})}\left[H[p(y \mid \mathbf{x}, \omega)]\right]$. We display the MI scores in Table B.5, where we see that our model is able to obtain superior results.

**Table B.3:** Latent Space Linear Digit Classification.

| | MNIST | | | | |
|---|---|---|---|---|---|
| Model | 1.0 | 0.5 | 0.25 | 0.125 | 0.0625 |
| $\text{MEME}_{\text{SVHN}}$ | **0.908 ± 0.007** | 0.881 ± 0.006 | 0.870 ± 0.007 | 0.815 ± 0.005 | 0.795 ± 0.010 |
| $\text{MMVAE}_{\text{SVHN}}$ | 0.886 ± 0.003 | - | - | - | - |
| $\text{MVAE}_{\text{SVHN}}$ | 0.892 ± 0.005 | **0.895 ± 0.003** | **0.890 ± 0.003** | **0.887 ± 0.004** | **0.880 ± 0.003** |
| $\text{Ours}_{\text{MNIST}}$ | **0.908 ± 0.007** | 0.882 ± 0.003 | 0.844 ± 0.003 | 0.824 ± 0.006 | 0.807 ± 0.005 |
| $\text{MMVAE}_{\text{MNIST}}$ | 0.886 ± 0.003 | - | - | - | - |
| $\text{MVAE}_{\text{MNIST}}$ | 0.892 ± 0.005 | **0.895 ± 0.002** | **0.898 ± 0.004** | **0.896 ± 0.002** | **0.895 ± 0.002** |
| $\text{MEME}_{\text{SPLIT}}$ | **0.908 ± 0.007** | **0.914 ± 0.003** | 0.893 ± 0.005 | 0.883 ± 0.006 | 0.856 ± 0.003 |
| $\text{MVAE}_{\text{SPLIT}}$ | 0.892 ± 0.005 | 0.898 ± 0.005 | **0.895 ± 0.001** | **0.894 ± 0.001** | **0.898 ± 0.001** |
| | SVHN | | | | |
| Model | 1.0 | 0.5 | 0.25 | 0.125 | 0.0625 |
| $\text{MEME}_{\text{SVHN}}$ | **0.648 ± 0.012** | **0.549 ± 0.008** | **0.295 ± 0.025** | **0.149 ± 0.006** | **0.113 ± 0.003** |
| $\text{MMVAE}_{\text{SVHN}}$ | 0.499 ± 0.045 | - | - | - | - |
| $\text{MVAE}_{\text{SVHN}}$ | 0.131 ± 0.010 | 0.106 ± 0.008 | 0.107 ± 0.003 | 0.105 ± 0.005 | 0.102 ± 0.001 |
| $\text{Ours}_{\text{MNIST}}$ | **0.648 ± 0.012** | **0.581 ± 0.008** | **0.398 ± 0.029** | **0.384 ± 0.017** | **0.362 ± 0.018** |
| $\text{MMVAE}_{\text{MNIST}}$ | 0.499 ± 0.045 | - | - | - | - |
| $\text{MVAE}_{\text{MNIST}}$ | 0.131 ± 0.010 | 0.106 ± 0.005 | 0.106 ± 0.003 | 0.107 ± 0.005 | 0.101 ± 0.005 |
| $\text{MEME}_{\text{SPLIT}}$ | **0.648 ± 0.012** | **0.675 ± 0.004** | **0.507 ± 0.003** | **0.432 ± 0.011** | **0.316 ± 0.020** |
| $\text{MVAE}_{\text{SPLIT}}$ | 0.131 ± 0.010 | 0.107 ± 0.003 | 0.109 ± 0.003 | 0.104 ± 0.007 | 0.100 ± 0.008 |

### B.9.3 t-SNE Plots When Partially Observing both Modalities

In Figure B.11 we can see that partially observing MNIST leads to less structure in the latent space.

## B.10 MMVAE baseline with Laplace Posterior and Prior

The difference in results between our implementation of MVAE and the ones in the paper (Shi et al., 2019a), is becuase we restrict MEME to use Gaussian distributions for the posterior and prior, and therefore we adopt Gaussian posteriors and priors for all three models to ensure like-for-like comparison. Better results for MMVAE can be obtained by using Laplace posteriors and priors, and In Table B.6 we display coherence scores using our implementation of MMVAE using a Laplace posterior and prior. Our implementation is inline with the results reported in Shi et al. (2019a), indicating that the baseline for MMVAE is accurate.

**Table B.4:** Correlation Values for CUB cross generations. Higher is better.

| | Image → Captions | | | | |
|---|---|---|---|---|---|
| Model | GT | $f = 1.0$ | $f = 0.5$ | $f = 0.25$ | $f = 0.125$ |
| $\text{MEME}_{\text{Image}}$ | 0.106 ± 0.000 | **0.064 ± 0.011** | **0.042 ± 0.005** | **0.026 ± 0.002** | **0.029 ± 0.003** |
| $\text{MMVAE}_{\text{Image}}$ | 0.106 ± 0.000 | 0.060 ± 0.010 | - | - | - |
| $\text{MVAE}_{\text{Image}}$ | 0.106 ± 0.000 | -0.002 ± 0.001 | -0.000 ± 0.004 | 0.001 ± 0.004 | -0.001 ± 0.005 |
| $\text{MEME}_{\text{Captions}}$ | 0.106 ± 0.000 | **0.064 ± 0.011** | **0.062 ± 0.006** | **0.048 ± 0.004** | **0.052 ± 0.002** |
| $\text{MMVAE}_{\text{Captions}}$ | 0.106 ± 0.000 | 0.060 ± 0.010 | - | - | - |
| $\text{MVAE}_{\text{Captions}}$ | 0.106 ± 0.000 | -0.002 ± 0.001 | -0.000 ± 0.004 | 0.000 ± 0.003 | 0.001 ± 0.002 |
| $\text{MEME}_{\text{SPLIT}}$ | 0.106 ± 0.000 | **0.064 ± 0.011** | **0.046 ± 0.005** | **0.031 ± 0.006** | **0.027 ± 0.005** |
| $\text{MVAE}_{\text{SPLIT}}$ | 0.106 ± 0.000 | -0.002 ± 0.001 | 0.000 ± 0.003 | 0.000 ± 0.005 | -0.001 ± 0.003 |
| | Caption → Image | | | | |
| Model | GT | $f = 1.0$ | $f = 0.5$ | $f = 0.25$ | $f = 0.125$ |
| $\text{MEME}_{\text{Image}}$ | 0.106 ± 0.000 | **0.074 ± 0.001** | **0.058 ± 0.002** | **0.051 ± 0.001** | **0.046 ± 0.004** |
| $\text{MMVAE}_{\text{Image}}$ | 0.106 ± 0.000 | 0.058 ± 0.001 | - | - | - |
| $\text{MVAE}_{\text{Image}}$ | 0.106 ± 0.000 | -0.002 ± 0.001 | -0.002 ± 0.000 | -0.002 ± 0.001 | -0.001 ± 0.001 |
| $\text{Ours}_{\text{Captions}}$ | 0.106 ± 0.000 | **0.074 ± 0.001** | **0.059 ± 0.003** | **0.050 ± 0.001** | **0.053 ± 0.001** |
| $\text{MMVAE}_{\text{Captions}}$ | 0.106 ± 0.000 | 0.058 ± 0.001 | - | - | - |
| $\text{MVAE}_{\text{Captions}}$ | 0.106 ± 0.000 | 0.002 ± 0.001 | -0.001 ± 0.002 | -0.003 ± 0.002 | -0.002 ± 0.001 |
| $\text{MEME}_{\text{SPLIT}}$ | 0.106 ± 0.000 | **0.074 ± 0.001** | **0.061 ± 0.002** | **0.047 ± 0.003** | **0.049 ± 0.003** |
| $\text{MVAE}_{\text{SPLIT}}$ | 0.106 ± 0.000 | -0.002 ± 0.001 | -0.002 ± 0.002 | -0.002 ± 0.001 | -0.002 ± 0.001 |

## B.11 Ablation Studies

Here we carry out two ablation studies to test the hypotheses: 1) How sensitive is the model to the number of pseudo samples in $\lambda$ and 2) What is the effect of training the model using only paired data for a given fraction of the dataset.

### B.11.1 Sensitivity to number of pseudo-samples

In Figure B.12 we plot results where the number of pseudo samples is varied for different observation rates. Ideally we expect to see the results decrease in their performance as the number of pseudo-samples is minimised. This is due to the number of components being present in the mixture $p_{\lambda^{\mathbf{t}}}(\mathbf{z}) = \frac{1}{N}\sum_{i=1}^{N} p_\psi(\mathbf{z} \mid \mathbf{u}_i^{\mathbf{t}})$, also being decreased, thus reducing the its ability to approximate the true prior $p(\mathbf{z}) = \int_{\mathbf{t}} p_\vartheta(\mathbf{z} \mid \mathbf{t})p(\mathbf{t})dt$. As expected lower observation rates are more sensitive, due to a higher dependence on the prior approximation, and a higher number of pseudo samples typically leads to better results.

**Table B.5:** Mutual Information Scores. Lower is better.

| | MNIST | | | | |
|---|---|---|---|---|---|
| Model | 1.0 | 0.5 | 0.25 | 0.125 | 0.0625 |
| $\text{Ours}_{\text{SVHN}}$ | **0.075 ± 0.002** | **0.086 ± 0.003** | **0.101 ± 0.002** | **0.102 ± 0.004** | **0.103 ± 0.001** |
| $\text{MMVAE}_{\text{SVHN}}$ | 0.105 ± 0.004 | - | - | - | - |
| $\text{MVAE}_{\text{SVHN}}$ | 0.11 ± 0.00551 | 0.107 ± 0.007 | 0.106 ± 0.004 | 0.106 ± 0.012 | 0.142 ± 0.007 |
| $\text{Ours}_{\text{MNIST}}$ | **0.073 ± 0.002** | **0.087 ± 0.001** | **0.101 ± 0.001** | **0.099 ± 0.001** | **0.104 ± 0.002** |
| $\text{MMVAE}_{\text{MNIST}}$ | 0.105 ± 0.004 | - | - | - | - |
| $\text{MVAE}_{\text{MNIST}}$ | 0.11 ± 0.00551 | 0.102 ± 0.00529 | 0.1 ± 0.00321 | 0.1 ± 0.0117 | 0.0927 ± 0.00709 |
| $\text{MEME}_{\text{SPLIT}}$ | **0.908 ± 0.007** | **0.914 ± 0.003** | **0.893 ± 0.005** | **0.883 ± 0.006** | **0.856 ± 0.003** |
| $\text{MVAE}_{\text{SPLIT}}$ | 0.11 ± 0.00551 | 0.104 ± 0.006 | 0.099 ± 0.003 | 0.1 ± 0.0117 | 0.098 ± 0.005 |
| | SVHN | | | | |
| Model | 1.0 | 0.5 | 0.25 | 0.125 | 0.0625 |
| $\text{Ours}_{\text{SVHN}}$ | **0.036 ± 0.001** | **0.047 ± 0.002** | **0.071 ± 0.003** | **0.107 ± 0.007** | **0.134 ± 0.003** |
| $\text{MMVAE}_{\text{SVHN}}$ | 0.042 ± 0.001 | - | - | - | - |
| $\text{MVAE}_{\text{SVHN}}$ | 0.163 ± 0.003 | 0.166 ± 0.010 | 0.165 ± 0.003 | 0.164 ± 0.004 | 0.176 ± 0.004 |
| $\text{Ours}_{\text{MNIST}}$ | **0.036 ± 0.001** | **0.048 ± 0.001** | **0.085 ± 0.006** | **0.111 ± 0.004** | **0.142 ± 0.005** |
| $\text{MMVAE}_{\text{MNIST}}$ | 0.042 ± 0.001 | - | - | - | - |
| $\text{MVAE}_{\text{MNIST}}$ | 0.163 ± 0.003 | 0.175 ± 0.00551 | 0.17 ± 0.0102 | 0.174 ± 0.012 | 0.182 ± 0.00404 |
| $\text{MEME}_{\text{SPLIT}}$ | **0.648 ± 0.012** | **0.675 ± 0.004** | **0.507 ± 0.003** | **0.432 ± 0.011** | **0.316 ± 0.020** |
| $\text{MVAE}_{\text{SPLIT}}$ | 0.163 ± 0.003 | 0.165 ± 0.01 | 0.172 ± 0.015 | 0.173 ± 0.013 | 0.179 ± 0.005 |

**Table B.6:** Coherence Scores for MMVAE using Laplace posterior and prior.

| MNIST | SVHN |
|---|---|
| 91.8% | 65.2% |

### B.11.2 Training using only paired data

Here we test the models ability to leverage partially observed data to improve the results. If the model is successfully able to leverage the partially observed samples, then we should see a decrease in the efficacy if we train the model using only paired samples, i.e. a model trained with 25% paired and 75% partially observed should perform improve the results over a model trained with only the 25% paired data. In other words we omit, the first two partially observed terms in (4.5), discarding $\mathcal{D}_{\mathbf{s}}$ and $\mathcal{D}_{\mathbf{t}}$. In Figure B.13 we can see that the model is able to use the partially observed modalities to improve its results.

## B.12 Training Details

**MNIST-SVHN** We provide the architectures used in Table B.7b and Table B.7a. We used the Adam optimizer with a learning rate of 0.0005 and beta values of

$(0.9, 0.999)$ for 100 epochs, training consumed around 2Gb of memory.

**CUB** We provide the architectures used in Table B.7c and Table B.7d. We used the Adam optimizer with a learning rate of 0.0001 and beta values of $(0.9, 0.999)$ for 300 epochs, training consumed around 3Gb of memory.

| Encoder | Decoder |
|---|---|
| Input $\in \mathbb{R}^{1x28x28}$ | Input $\in \mathbb{R}^{L}$ |
| FC. 400 ReLU | FC. 400 ReLU |
| FC. $L$, FC. $L$ | FC. 1 x 28 x 28 Sigmoid |

**(a)** MNIST dataset.

| Encoder |
|---|
| Input $\in \mathbb{R}^{1x28x28}$ |
| 4x4 conv. 32 stride 2 pad 1 & ReLU |
| 4x4 conv. 64 stride 2 pad 1 & ReLU |
| 4x4 conv. 128 stride 2 pad 1 & ReLU |
| 4x4 conv. L stride 1 pad 0, 4x4 conv. L stride 1 pad 0 |
| **Decoder** |
| Input $\in \mathbb{R}^{L}$ |
| 4x4 upconv. 128 stride 1 pad 0 & ReLU |
| 4x4 upconv. 64 stride 2 pad 1 & ReLU |
| 4x4 upconv. 32 stride 2 pad 1 & ReLU |
| 4x4 upconv. 3 stride 2 pad 1 & Sigmoid |

**(b)** SVHN dataset.

| Encoder | Decoder |
|---|---|
| Input $\in \mathbb{R}^{2048}$ | Input $\in \mathbb{R}^{L}$ |
| FC. 1024 ELU | FC. 256 ELU |
| FC. 512 ELU | FC. 512 ELU |
| FC. 256 ELU | FC. 1024 ELU |
| FC. $L$, FC. $L$ | FC. 2048 |

**(c)** CUB image dataset.

| Encoder |
|---|
| Input $\in \mathbb{R}^{1590}$ |
| Word Emb. 256 |
| 4x4 conv. 32 stride 2 pad 1 & BatchNorm2d & ReLU |
| 4x4 conv. 64 stride 2 pad 1 & BatchNorm2d & ReLU |
| 4x4 conv. 128 stride 2 pad 1 & BatchNorm2d & ReLU |
| 1x4 conv. 256 stride 1x2 pad 0x1 & BatchNorm2d & ReLU |
| 1x4 conv. 512 stride 1x2 pad 0x1 & BatchNorm2d & ReLU |
| 4x4 conv. L stride 1 pad 0, 4x4 conv. L stride 1 pad 0 |
| **Decoder** |
| Input $\in \mathbb{R}^{L}$ |
| 4x4 upconv. 512 stride 1 pad 0 & ReLU |
| 1x4 upconv. 256 stride 1x2 pad 0x1 & BatchNorm2d & ReLU |
| 1x4 upconv. 128 stride 1x2 pad 0x1 & BatchNorm2d & ReLU |
| 4x4 upconv. 64 stride 2 pad 1 & BatchNorm2d & ReLU |
| 4x4 upconv. 32 stride 2 pad 1 & BatchNorm2d & ReLU |
| 4x4 upconv. 1 stride 2 pad 1 & ReLU |
| Word Emb.$^{T}$ 1590 |

**(d)** CUB-Language dataset.

**Table B.7:** Encoder and decoder architectures.

Fully observed.

 → this is a white bird with a wings and and black beak.

 → a small brown bird with a white belly.

a grey bird with darker brown mixed in and a short brown beak. → 

yellow bird with a black and white wings with a black beak. → 

Captions observed 50% of the time.

 → than a particular a wings a with and wings has on yellow.

 → below a small has has that bill yellow and.

the bird has two large , grey wingbars , and orange feet. → 

yellow bird with a black and white wings with a black beak. → 

Images observed 50% of the time.

 → blacks this bird has has that are that and has a yellow belly.

 → bird this bird has red with bird with a a and a pointy.

tiny brown bird with white breast and a short stubby bill. → 

this is a yellow bird with a black crown on its head. → 

Captions observed 25% of the time.

 → throat a is is yellow with yellow gray chest has medium belly belly throat.

 → bird green small has crown as wing crown white white abdomen and crown and and.

this is a puffy bird with a bright yellow chest with white streaks along the feathers. → 

this bird has a small bill with a black head and wings but white body. → 

Images observed 25% of the time.

 → crest this small with looking with a brown with a and face body.

 → rotund white bill large bird , brown black back mostly with with breast flying.

the bird had a large white breast in <exc> to its head size. → 

this bird has a black belly , breast and head , gray and white wings , and red tarsus and feet. → 

Captions observed 12.5% of the time.

 → a the bird wings a is and nape the and , crown , rectrices grey , , beak its and , a tail.

 → red this bird skinny black black crown red feet beak downwards short and a.

white belly with a brown body and a very short , small brown beak. → 

a bird with a short , rounded beak which ends in a point , stark white eyes , and white throat. → 

Images observed 12.5% of the time.

 → an a is colorful bird , short with and black and crown small short orange light small light over.

 → a this is white a all is with flat beak and black a , 's a curved for light has a body is.

small bird with a long beak and blue wing feathers with brown body. → 

this is a large black bird with a long neck and bright orange cheek patches. → 

**Figure B.9:** MEME cross-modal generations for CUB.

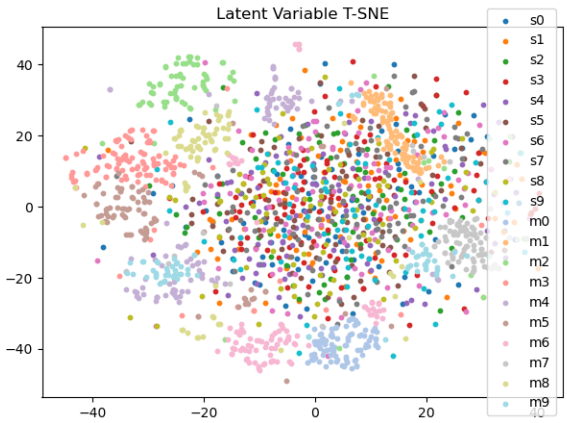



**Figure B.10:** T-SNE plot indicating the complete failure of MVAE to construct joint representations. s indicates SVHN (low transparency), m indicates MNIST (high transparency).

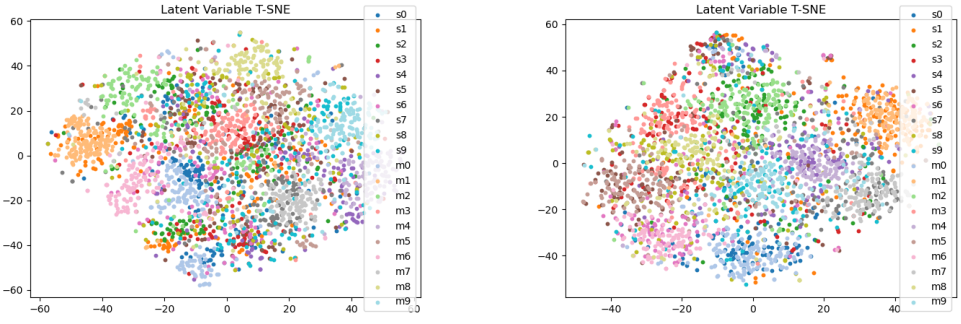



**Figure B.11:** f = 0.25, Left) t-SNE when partially observing MNIST. Right) t-SNE when partially observing SVHN.

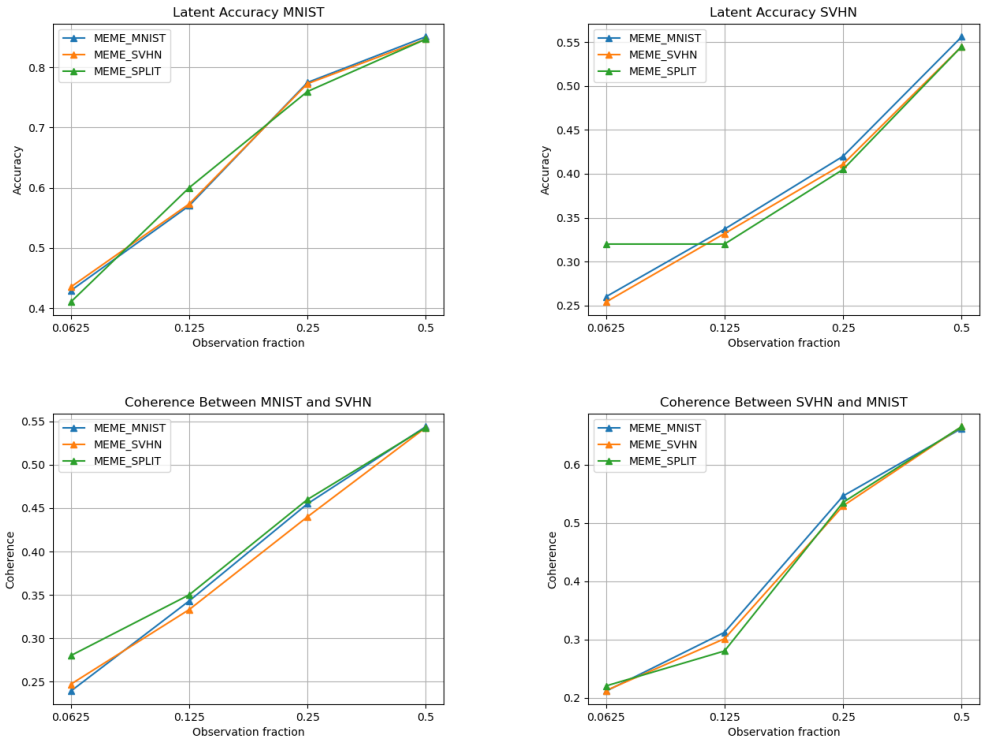



**Figure B.12:** How performance varies for different numbers of psuedo samples. Number of pseudo samples ranges from 1 to 100 on the x axis.

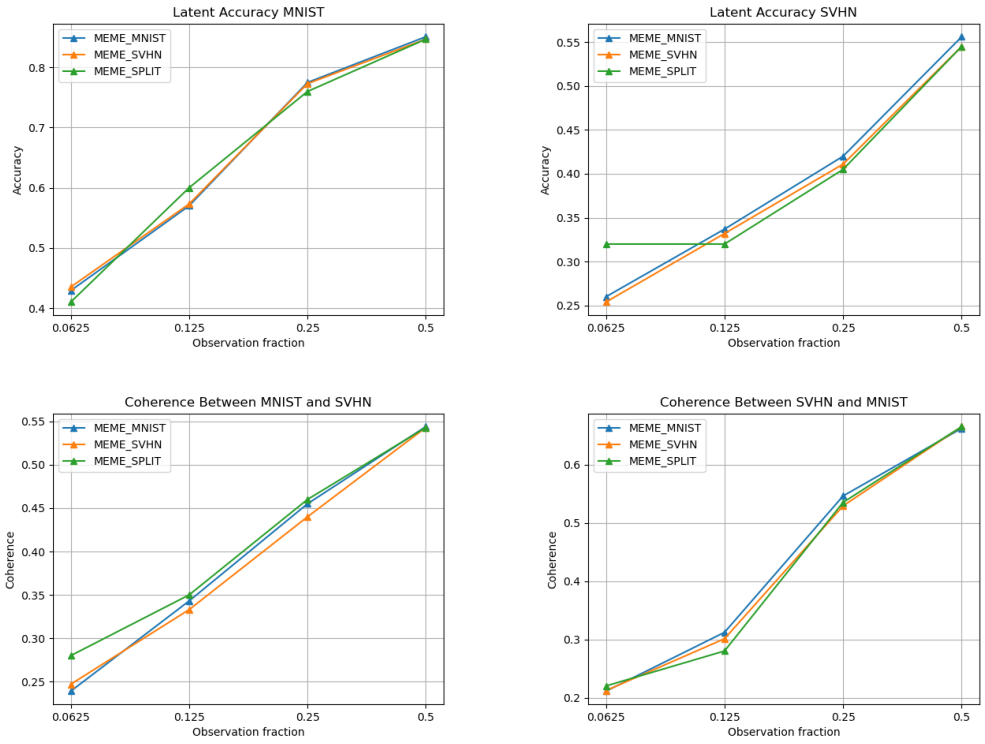



**Figure B.13:** How performance varies when training using only a fraction of the partially observed data.

# C

# Appendix: Adaptive Temperature Scaling

## C.1 Gradient of Network Weights

Consider the last layer of a Neural Network with parameters $\boldsymbol{w}$ and the cross entropy loss $\mathcal{L} : \mathbb{R}^K \to \mathbb{R}$. The gradient of the parameters is given as $\frac{\partial \mathcal{L}}{\partial \boldsymbol{w}} = \frac{\partial \mathbf{s}}{\partial \boldsymbol{w}} \frac{\partial \sigma(\mathbf{s})}{\partial \mathbf{s}} \frac{\partial \mathcal{L}}{\partial \sigma(\mathbf{s})}$, where

$$\frac{\partial \mathcal{L}}{\partial \sigma(s_k)} = -\frac{q_k}{\sigma(s_k)} \tag{C.1}$$

$$\frac{\partial \sigma(s_k)}{\partial s_j} = \sigma(s_k)(\delta_{jk} - \sigma(s_j)). \tag{C.2}$$

the gradient for the last layers is thus given as

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{w}} = \frac{\partial \mathbf{s}}{\partial \boldsymbol{w}}(\sigma(\mathbf{s}) - \boldsymbol{q}), \tag{C.3}$$

where $\sigma(\mathbf{s}) - \boldsymbol{q} = \{\sigma(s_j) - q_j : j \in \{1 \dots K\}$.

## C.2 Predictions are unaffected by temperature

In neural network classification problems, the parameters of the Categorical distribution are obtained through the Softmax operator

$$\sigma(\mathbf{s}) = \frac{\exp\left(\frac{\mathbf{s}}{T}\right)}{\sum_i \exp\left(\frac{s_i}{T}\right)},$$

with the predicted class given as $\tilde{k} = \arg\max_k \sigma(\mathbf{s}_k)$. The temperature value has no effect on the resulting prediction

$$\arg\max_k \sigma(\mathbf{s}_k) = \arg\max_k \frac{\mathbf{s}_k}{T} \tag{C.4}$$

$$= \arg\max_k \mathbf{s}_k. \tag{C.5}$$

Hence, the value of $T$ does not affect the class prediction.

## C.3 Gradient of Temperature

The gradient of the loss w.r.t the temperature is $\frac{\partial \mathcal{L}}{\partial T} = \frac{\partial \boldsymbol{p}}{\partial T} \frac{\partial \mathcal{L}}{\boldsymbol{p}}^T$. The gradient of the individual class probabilities from the softmax output is

$$\begin{aligned}\frac{\partial \boldsymbol{p}_k}{\partial T} &= \frac{\sum_i \exp\left(\frac{\mathbf{s}_i}{T}\right) \frac{\partial}{\partial T} \exp\left(\frac{\mathbf{s}_k}{T}\right)}{\left(\sum_i \exp\left(\frac{\mathbf{s}_i}{T}\right)\right)^2} - \frac{\exp\left(\frac{\mathbf{s}_k}{T}\right) \sum_k \frac{\partial}{\partial T} \exp\left(\frac{\mathbf{s}_k}{T}\right)}{\left(\sum_i \exp\left(\frac{\mathbf{s}_i}{T}\right)\right)^2} \\ &= \frac{\sigma(\mathbf{s}_k)}{T^2}\Big(\sum_{i \setminus k} \mathbf{s}_i \exp\left(\frac{\mathbf{s}_i}{T}\right) - \mathbf{s}_k \exp\left(\frac{\mathbf{s}_i}{T}\right)\Big).\end{aligned}$$

Given that $\frac{\partial \mathcal{L}}{\partial \sigma(s_k)} = -\frac{q_k}{\sigma(s_k)}$, using the chain rule we have

$$\frac{\partial \mathcal{L}}{\partial T} = \sum_k \frac{q_k}{T^2}\left(s_k \sum_{i \setminus k} \exp\left(\frac{s_i}{T}\right) - \sum_{j \setminus k} s_j \exp\left(\frac{s_j}{T}\right)\right), \tag{C.6}$$

thus concluding the proof.

## C.4 Corruptions

We display additional plots for how AdaECE varies with corruption strength for CIFAR10-C in Figure C.1, where we can see that adaptive temperature scaling consistently obtains better results than vanilla temperature scaling.

## C.5 Temperature Values

We display the average temperature values in Table C.1, here we see that adaptive temperature obtains a similar average temperature to vanilla temperature scaling.

## C.6 Training Details

We followed standard training protocols when training the neural networks. Models trained on CIFAR-10/CIFAR-100 required 350 epochs, with an initial learning

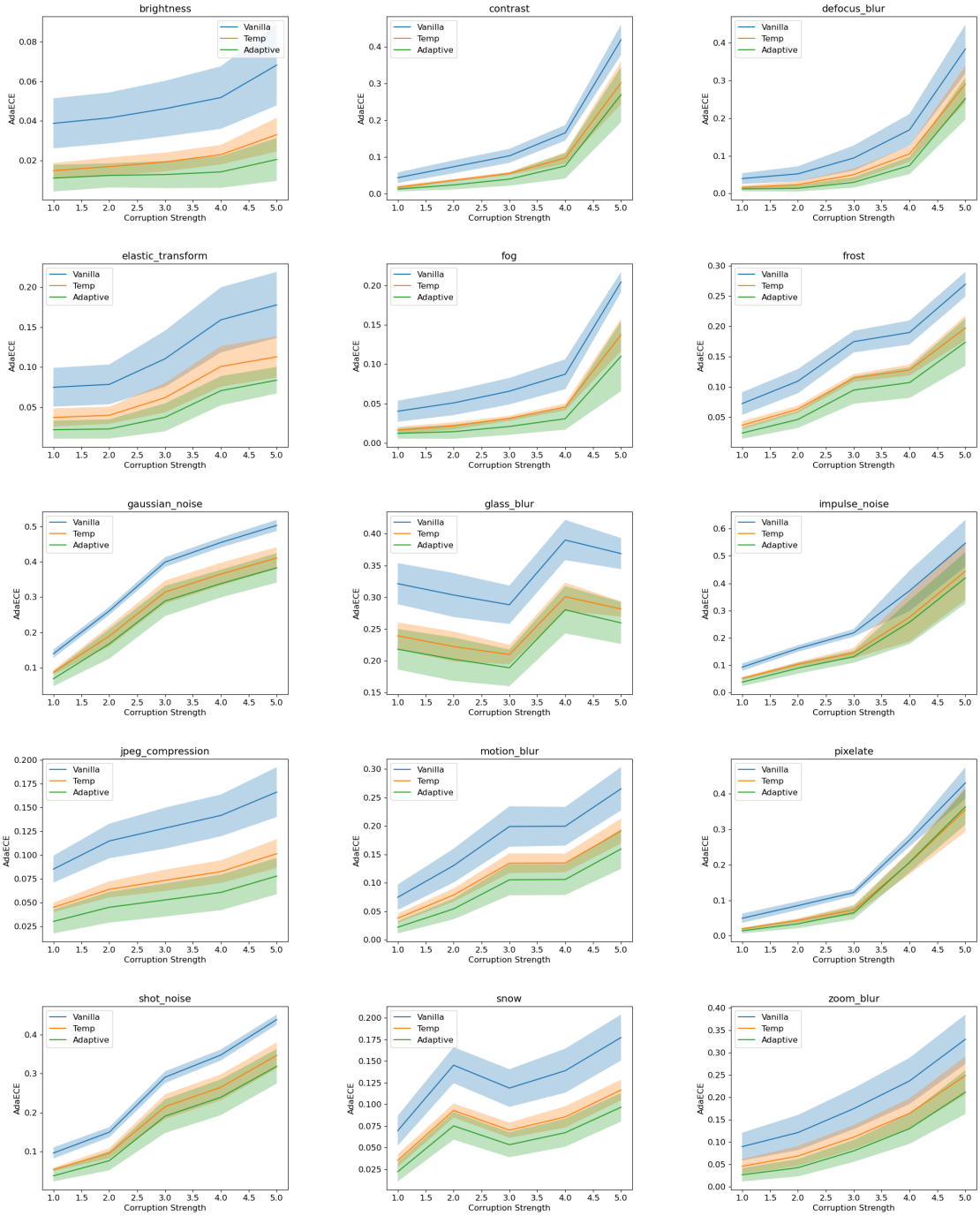



**Figure C.1:** Additional plots for how AdaECE varies with corruption strength for CIFAR10-C.

| **Network** | **Dataset** | **Avg. Temp** | |
|---|---|---|---|
| | | *Vanilla* | *Adaptive* |
| ResNet50 | CIFAR-10 | 1.484 ± 0.123 | 1.506 ± 0.296 |
| | CIFAR-100 | 1.398 ± 0.034 | 1.313 ± 0.076 |
| | TinyImageNet | 1.296 ± 0.234 | 1.131 ± 0.179 |
| WideResNet2810 | CIFAR-10 | 1.310 ± 0.035 | 1.294 ± 0.072 |
| | CIFAR-100 | 1.220 ± 0.010 | 1.134 9 |
| | TinyImageNet | 1.174 ± 0.084 | 1.017 ± 0.117 |

**Table C.1:** Average temperature values for neural different models.

rate of 0.1, the learning rate was decreased by a factor of 10 at the milestones 150 and 250 epochs. Models trained on TinyImageNet required 100 epochs, with an initial learning rate of 0.1, the learning rate was decreased by a factor of 10 at the milestones 43 and 72 epochs.