INVARIANT-COVARIANT REPRESENTATION LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Representations learnt through deep neural networks tend to be highly informative, but opaque in terms of what information they learn to encode. We introduce an approach to probabilistic modelling that learns to represent data with two separate deep representations: an invariant representation that encodes the information of the class from which the data belongs, and a covariant representation that encodes the symmetry transformation defining the particular data point within the class manifold (covariant in the sense that the representation varies naturally with symmetry transformations). This approach to representation learning is conceptually transparent, easy to implement, and in-principle generally applicable to any data comprised of discrete classes of continuous distributions (e.g. objects in images, topics in language, individuals in behavioural data). We demonstrate qualitatively compelling representation learning and competitive quantitative performance, in both supervised and semi-supervised settings, versus comparable modelling approaches in the literature with little fine tuning.

1 INTRODUCTION

Representation learning (Bengio et al., 2013) is part of the foundation of deep learning; powerful deep neural network models appear to derive their performance from sequentially representing data in more-and-more refined structures, tailored to the training task.

However, representation learning has a broader impact than just model performance. Transferable representations are leveraged efficiently for new tasks (Mikolov et al., 2013), representations are used for human interpretation of machine learning models (Mahendran & Vedaldi, 2015), and meaningfully structured (disentangled) representations can be used for model control (e.g. semi-supervised learning as in Kingma et al. (2014), topic modelling as in Blei et al. (2003)).

Consequently, it is often preferable to have interpretable data representations within a model, in the sense that the information contained in the representation is easily understood and the representation can be used to control the output of the model (e.g. to generate data of a given class or with a particular characteristic). Unfortunately, there is often a tension between optimal model performance and cleanly disentangled or controllable representations.

To overcome this, some practitioners have proposed modifying their model's objective functions by inserting parameters in front of particular terms (Bowman et al., 2016; Higgins et al., 2017), while others have sought to modify the associated generative models (Mansbridge et al., 2018). Further still, attempts have been made to build the symmetries of the data directly into the neural network architecture in order to force the learning of latent variables that transform meaningfully under those symmetries (Sabour et al., 2017). The diversity and marginal success of these approaches point to the importance and difficulty of learning meaningful representations in deep generative modelling.

In this work we present a general framework for probabilistic modelling of data comprised of a finite number of distinct classes, each described by a smooth manifold of instantiations of that class. For convenience, we call our approach CoVAE for Covariant Variational Autoencoder. CoVAE is a probabilistic model with 2 latent variables: an invariant latent that represents the global class information, and a covariant latent that smoothly interpolates between all of the members of that class. The CoVAE framework is completely general in that the symmetry group of the manifold must not be specified (as it is explicitly in Falorsi et al. (2018) and implicitly in Sabour et al. (2017)), and it can be used for any number of classes and any dimensionality of both underlying representations.

The price that must be paid for this level of model control and flexibility is that some labelled data is needed in order to provide the concept of class invariance versus covariance to the model.

CoVAE attempts to solve the same problem as Sabour et al. (2017), namely to separately learn the 'pose' of the object (what we call the covariant latent), as well as its class (the information stored in what we call the invariant latent). However, we seek a less-engineered approach where the model learns invariance and covariance on its own, instead of building it into vectors and vector norms explicitly throughout the neural-network architecture.

CoVAE is more similar to Siddharth et al. (2017), where the authors leverage labelled data explicitly in their generative model in order to force the VAE latent to learn the non-class information (Makhzani et al. (2016) do similarly using adversarial training). The primary difference between those works and ours is that CoVAE provides a non-trivial representation of the global information instead of simply using the integer-valued label. Furthermore, this invariant representation can be deterministically evaluated directly on unlabelled data. Practitioners can reuse this embedding on unlabelled data in downstream tasks, along with the covariant encoder if needed. The invariant representation provides much more information than a simple prediction of the class-label distribution.

The encoding procedure for the invariant representation in CoVAE is partially inspired by Eslami et al. (2018), who use images from various, known coordinates in a scene in order to reconstruct new images of that scene at a new, known coordinates. However, we do not have access to the exact coordinates of the class instance, which in our case corresponds to the unknown, non-trivial manifold structure of the class; we must infer these manifold coordinates in an unsupervised way.

2 Covariant Variational Autoencoders

We consider a generative model for data comprised of a finite set of distinct classes, each of which occupies a smooth manifold of instantiations. For example, images of distinct objects where each object might be in any pose, or sentences describing distinct sets of topics. Such data should be described by a generative model with two latent variables, the first describing which of the objects the data belongs to, and the second describing the particular instantiation of the object (e.g. its pose).

In this way the object-identity latent variable r would be *invariant* under the transformations that cover the set of possible instantiations of the object, and the instantiation-specifc latent variable v should be *covariant* under such transformations. Note that the class label y is itself an invariant representation of the class, however, we seek a higher-dimensional latent vector r that has the capacity to represent rich information relevant to the class of the data point, rather than just its label.

Denoting an individual data point as x_n with associated class label y_n, and the full set of class-y labeled data $\{x_n|\text{label}(x_n) = y\}$ as $\mathcal{D}^y_{\text{lab}}$, we write such a generative model as:

$$p\big(\{x_n, y_n\}_{n=1}^N\big) \;=\; \prod_{n=1}^{N} \int dv_n\, dr_n\, p_\theta(x_n|r_n, v_n)\, \delta\big(r_n - r(y_n, \mathcal{D}^{y_n}_{\text{lab}} \setminus \{x_n\})\big)\, p(v_n)\, p(y_n) \quad (1)$$

where θ are the parameters of the generative model. We make explicit with a δ function the conditional dependency of p_θ on the deterministically calculable representation r_n of the global properties of class y_n. The prior distribution $p(y_n)$ is a categorical distribution with weights given by the relative frequency of each class and the prior distribution $p(v_n)$ is taken to be a unit normal describing the set of smooth transformations that cover the class-y_n manifold.

To guarantee that r_n will learn a meaningful, invariant representation of the class-y_n data, we use a technique inspired by Generative Query Networks (Eslami et al., 2018). Instead of encoding the information of a single data point (x_n, y_n) into r_n, we provide samples from the whole class-y_n manifold. That is, we compute the invariant latent as:

$$r\big(y_n, \mathcal{D}^{y_n}_{\text{lab}} \setminus \{x_n\}\big) \overset{\substack{\text{concise}\\ \text{notation}}}{=} r_{y_n} \;=\; \mathbb{E}_{x\sim\mathcal{D}^{y_n}_{\text{lab}}\setminus\{x_n\}}\big[f_{\theta_{\text{inv}}}(x)\big] \;\approx\; \frac{1}{m}\sum_{i=1}^{m} f_{\theta_{\text{inv}}}(x^i) \quad (2)$$

where θ_{inv} are the parameters of this embedding. We explicitly exclude the data point at hand x_n from this expectation value; in the infinite labelled data limit, the probability of sampling x_n from $\mathcal{D}^{y_n}_{\text{lab}}$ would vanish. We include the simplified notation r_{y_n} for subsequent mathematical clarity.

This procedure invalidates the assumption that the data is generated i.i.d. conditioned on a set of model parameters, since r_y is computed using a number of other data points x^i with label y. For notational simplicity, we will suppress this fact, and consider the likelihood $p(x, y)$ as if it were i.i.d. per data point. It is not difficult to augment the equations that follow to incorporate the full dependencies, but we find this to obfuscate the discussion.

The primary purpose of our approach to the invariant representation used in Equation 2 is to provide exactly the information needed to learn a global-class embedding: namely, to learn what the elements of the class manifold have in common. However, our approach provides a secondary advantage. During training we will use values of m (see Equation 2) sampled uniformly between 1 and some small maximal value. The r_y embedding will thus learn to work well for various m, including $m = 1$. Consequently, at inference time, any unlabelled data point x can be immediately embedded via $f_{\theta_{\text{inv}}}(x)$. This is ideal for downstream usage; we will use this technique in Section 3.1 to competitively classify unlabelled test-set data using only $f_{\theta_{\text{inv}}}(x)$.

In order to approximate the integral in Equation 1 over the covariant latent, we use variational inference following the standard VAE approach (Kingma & Welling, 2014; Rezende et al., 2014):

$$q_{\phi_{\text{cov}}}(v|r_y, x) = \mathcal{N}\big(\mu_{\phi_{\text{cov}}}(r_y, x), \sigma^2_{\phi_{\text{cov}}}(r_y, x)I\big) \tag{3}$$

where ϕ_{cov} are the parameters of the variational distribution over the covariant latent. Note that v is inferred from r_y, not y, since r_y is posited to be a multi-dimensional latent vector that represents the rich set of global properties of the class y, rather than just its label.

We thus arrive at a lower bound on $\log p(x, y)$ given in Equation 1 following the standard arguments:

$$\mathcal{L}_{\text{lab}} \;=\; \mathbb{E}_{q(v|r_y,x)} \log p(x|r_y, v) - D_{\text{KL}}\big[q(v|r_y, x)\big|\big|p(v)\big] + \log p(y) \tag{4}$$

where the various model parameters are suppressed for clarity.

CoVAE is designed to efficiently learn a representation that stores global-class information, and as such, it is a natural framework for semi-supervised learning. Thus, an objective function for unlabelled data must be specified to accompany the labelled-data objective function given in Equation 4.

We marginalise over the label in $p(x, y)$ (Equation 1) when a data point is unlabelled. In order to perform variational inference with this additional marginalisation, we use a variational distribution of the form:

$$q(v, y|x) \;=\; q_{\phi_{\text{cov}}}(v|r_y, x)\, q_{\phi_{y\text{-post}}}(y|x) \tag{5}$$

where $q_{\phi_{\text{cov}}}(v|r_y, x)$ is the same distribution as is used in the labelled case, given in Equation 3. The unlabelled setting requires an additional inference distribution to infer the label y, which is achieved with $q_{\phi_{y\text{-post}}}(y|x)$, parametrised by $\phi_{y\text{-post}}$. Once y is inferred from $q_{\phi_{y\text{-post}}}(y|x)$, r_y can be deterministically calculated using Equation 2 from the labelled data set $\mathcal{D}^y_{\text{lab}}$ for class y, of which x is no longer a part. With r_y and x, the covariant latent v is inferred via $q_{\phi_{\text{cov}}}(v|r_y, x)$.

Using this variational inference procedure, we arrive at a lower bound for $\log p(x)$:

$$\mathcal{L}_{\text{unlab}} = \mathbb{E}_{q(y|x)}\Big[\mathbb{E}_{q(v|r_y,x)} \log p(x|r_y, v) - D_{\text{KL}}\big[q(v|r_y, x)\big|\big|p(v)\big]\Big] - D_{\text{KL}}\big[q(y|x)\big|\big|p(y)\big] \tag{6}$$

where the model parameters are again suppressed for clarity.

We will compute the expectation over the discrete distribution $q(y|x)$ in Equation 6 exactly in order to avoid the problem of back propagating through discrete variables. However, this expectation could be calculated by sampling using standard techniques (Brooks et al., 2011; Jang et al., 2017; Maddison et al., 2017).

2.1 Discussion

The motivation for this approach to CoVAE is that the information that the latent variables encode is controlled by the information flow in the modelling setup. The invariant latent variable r_y gets access to multiple samples from the full manifold of class y so it has the capacity to represent the data of class y in a global way. Moreover, the particular data point x from the class-y manifold is withheld from r_y explicitly, so r_y cannot learn any information particular to x. The latter is why we call it an invariant representation: r_y will be the same (or very similar, depending on the sample size m in Equation 2) for a different x coming from the same class y.

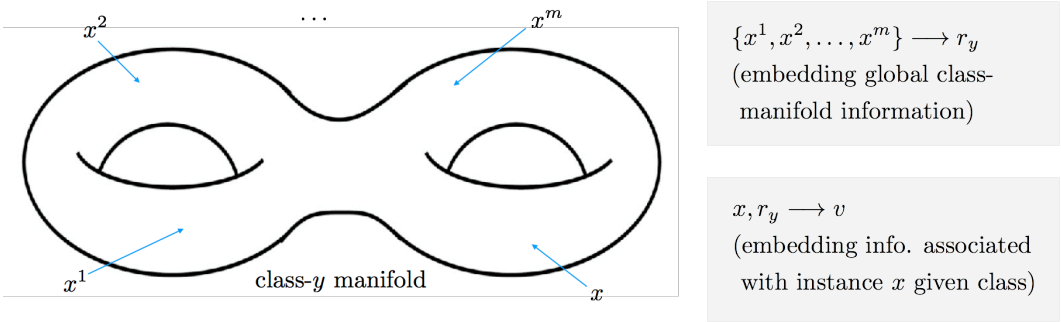

Figure 1: Heuristic depiction of class-y manifold. The invariant latent r_y will encode global-manifold information, whereas covariant latent v will encode coordinates of x on the manifold.

Conversely, the covariant representation v only has access to the particular instantiation x. It is modelled as a smooth, stochastic latent with prior $\mathcal{N}(0, I)$ in order to capture the smoothness corresponding to moving around the class-y manifold. We call v covariant because, for a fixed r_y (representing class y), variations of v will correspond to the variations of data x that cover the class-y manifold.

The intuition presented here is depicted heuristically in Figure 1. We will see that our expectations from this intuition are indeed borne out in the results we present in Section 3.

The goal of CoVAE is to provide an approach to probabilistic modelling that enables meaningful representations of both the covariant and the invariant information in the data. Aside from this primary objective, we consider CoVAE to possess four helpful attributes:

- *Conceptual transparency:* Reconstructing data from samples of the same-class manifold makes it clear that the information stored in the invariant latent is that which is shared between samples within a given class (i.e. the global information). The covariant latent will encode the remaining information required to reconstruct x, namely, the instantiation-specific information. Given that this information is smoothly distributed across the manifold, a stochastic latent is appropriate.
- *General applicability:* The CoVAE approach in-principle applies to modelling any data set comprised of classes of smoothly distributed data. All that is required is some labelled data. This is no different than techniques like Siddharth et al. (2017) and Makhzani et al. (2016), however, techniques such as Sabour et al. (2017) and Falorsi et al. (2018) are appropriate to particular symmetry groups.
- *Ease of implementation:* Our implementation of CoVAE is essentially a standard VAE with an additional deterministic latent; the idea of feeding complementary same-class data to the invariant representation does the heavy lifting. Much more engineering is required in many other approaches.
- *Robustness of performance:* We find competitive results on MNIST to other similar approaches using essentially no hyperparameter tuning.

We now turn to the last point, experimental performance.

3 Results

We carry out experiments on the MNIST data set (LeCun et al., 1998). As well as being a standard benchmarking data set, MNIST provides a meaningful test of the CoVAE framework, since digits from a particular class do not live on an a-priori known manifold, unlike images of rigid objects which have a clear rotational symmetry group, among others.

The CoVAE framework requires some labelled data. Forcing the model to reconstruct x through a representation r_y that only has access to other members of the y class is what forces r_y to be an abstract representation of that class, rather than a representation of the particular instantiation x. Thus,

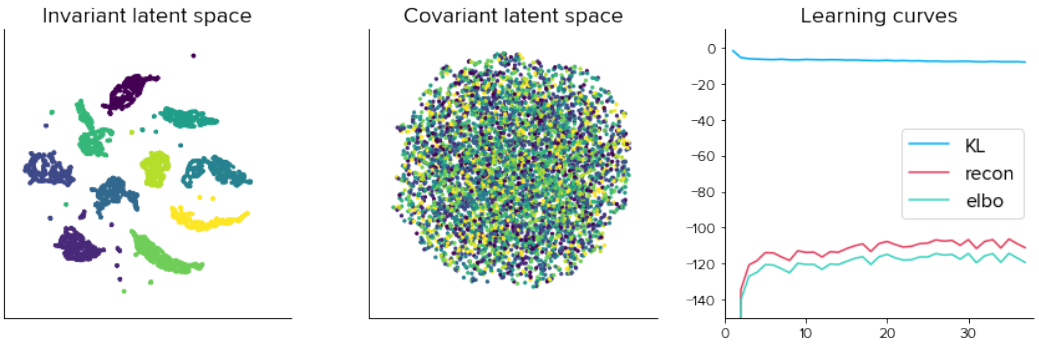

Figure 2: Validation-set results for CoVAE. Invariant (left) and covariant (middle) latent representations are shown reduced to 2D using UMAP. Learning curves are shown (right) with the ELBO broken down into the sum of the reconstruction and (negative) KL terms, as in Equation 4.

the requirement of some labelled data is at the heart of CoVAE. Indeed, we trained several versions of the CoVAE generative model in the unsupervised setting, allowing r to receive x directly. The results were as expected: the covariant latent is completely unused, with the model unable to reconstruct the structure in each class, as it essentially becomes a deterministic autoencoder.

In Section 3.1, we study the invariant-covariant properties of the representations learnt with CoVAE in the supervised setting, where we have access to the full set of labelled training data. The semi-supervised learning setting is discussed in Section 3.2, where we show the power of CoVAE to learn meaningful representations quickly and with very little labelled data. The details of the various experimental setups used in this section are provided in Appendix A.

3.1 Supervised learning

With the full training data set labelled, CoVAE is able to learn to optimise both covariant and invariant representations at every training step. The supervised CoVAE (objective given in Equation 4) converges in approximately 40 epochs on the MNIST training datset of 55,000 data points.

We show the training curves on the right in Figure 2. The covariant latent is learning to represent non-trivial information from the data as evidenced by the KL between the covariant variational distribution and its prior not vanishing at convergence.

However, when visualised in 2 dimensions using UMAP for dimensional reduction (McInnes & Healy, 2018), the covariant latent v appears not to distinguish between digit classes, as is seen in the uniformity of the class-coloured labels in the middle plot of Figure 2. The apparent uniformity of v reflects two facts: the first is that, given that the generative model gets access to another latent containing the global class information, the covariant latent does not need to distinguish between classes. The second is that the covariant manifolds should be similar across all digits. Indeed, they all include rotations, stretches, stroke thickness, among smooth transformations.

Finally, on the left in Figure 2, the invariant representation vectors r_y are shown, dimensionally reduced to 2 dimensions using UMAP, and coloured according to the label of the class. It is clear that these representations are well separated for each class. Each class has some spread in its invariant representations due to the fact that we choose relatively small numbers of complementary samples, m (see Equation 2). The model shown in Figure 2 had m randomly selected between 1 and 7 during training, with $m = 5$ used for visualisation. The outlier points in this plot are exaggerated by the dimensional reduction; we will show this below in Figure 4 by considering a CoVAE with 2D latent.

All visualisations in this work, including those in Figure 2, use data from the validation set of 5,000 images. We reserve the remaining 10,000 MNIST data points as a test set for computing the accuracy values provided. We did not look at this test set during training and hyperparameter tuning. In terms of hyperparameter tuning, we only tuned the number of epochs for training, the range of m values (i.e. $m < 7$), and chose between 8 and 16 for the dimensionality of both latents. We fixed the architecture at the outset to have ample capacity for this task, but did not vary it in our experiments.

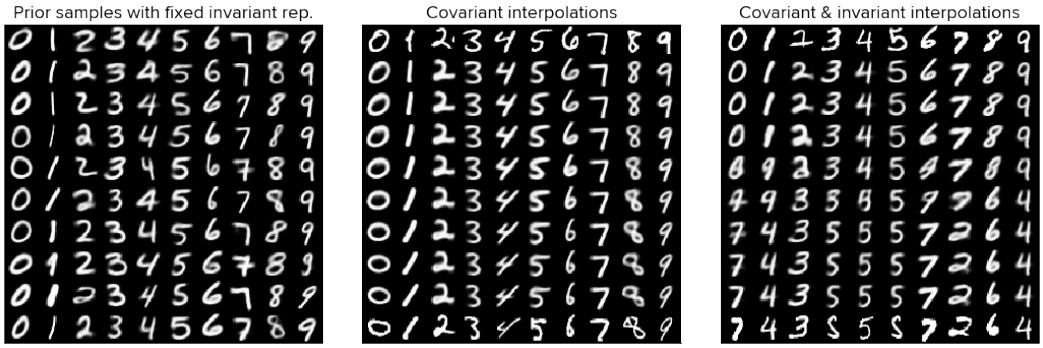

Figure 3: Generated images sampled from the prior $p(v)$ for each r_y (left), reconstructed from covariant interpolations between the embeddings of same-class digits with fixed r_y (middle), and reconstructed from inter-class interpolations of both v and r_y.

In order to really see what information is stored in the covariant and invariant latents, we consider them in the context of the generative model. We show reconstructed images in various latent-variable configurations in Figure 3. To show samples from the covariant prior $p(v)$ we fix a single invariant representation r_y for each class y by taking the mean r_y over each class in the validation set. On the left in Figure 3 we show random samples from $p(v)$ reconstructed along with r_y for each class y ascending from 0 to 9 in each column. Two properties stand out from these samples: firstly, the samples are all images from the correct class, showing that the invariant latent is representing the class information well. Secondly, there is meaningful variance within each class, showing that samples from the prior $p(v)$ are able to represent the intra-class variations of MNIST digits.

The middle plot in Figure 3 shows interpolations between actual digits of the same class (the top and bottom rows), with the invariant representation fixed throughout. These interpolations are smooth, as is expected from interpolations of a VAE latent, and cover the trajectory between the two images well. This again supports the argument that the covariant representation v, as a stochastic latent variable, is appropriate for representing the smooth intra-class transformations.

Finally, on the right of Figure 3, interpolations between images from different classes are shown. This requires an interpolation in the invariant representation as well as the covariant representation, since the covariant representation alone cannot interpolate between classes. As is expected, these interpolations are not smooth, since the invariant representation is not stochastic (implying that the model is not forced to perform well in open sets around the training data). We do not consider the lack of smoothness of the invariant interpolations to be problematic, since the invariant representation is meant to represent non-smooth information, namely, the global-class information.

Though non-smooth, these interpolations exhibit two non-trivial properties: firstly, the invariant interpolation exhibits rapid transition between classes indicating that the generative model has learnt to be somewhat robust to changes in r_y except when crossing class thresholds. The second is that r_y and v are independent from each other, which can be seen by the apparent smoothness of the covariant interpolations even across digits. These two results are made more explicit in Figure 4, where a CoVAE trained with 2-dimensional invariant and covariant latents is shown. On the left the invariant latent space is shown; without dimensional reduction, it is clear that the invariant latents are tightly clustered and highly separated. The middle shows reconstructions for $v = \vec{0}$ and r_y ranging evenly by ± 10 cluster standard deviations. These reconstructions vary imperceptibly. Finally, on the right, reconstructions are shown with fixed r_y for each $y = 3, 4, 5, 6$ and with the identical set of evenly spaced v over a grid spanning from -2 to $+2$ in each coordinate (2 prior standard deviations). Indeed, the stylistic variations appear to be similar for the same values of v across different digits.

We have thus seen that the invariant representation r_y in CoVAE learns to represent global-class information and the covariant representation v learns to represent local, smooth, intra-class information. This is exactly what we expected from the theoretical considerations given in Section 2.

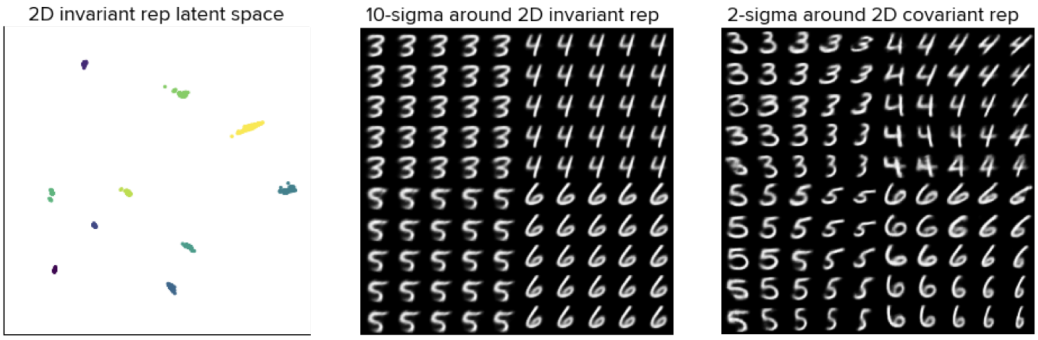

Figure 4: Latent variables for CoVAE with 2 dimensional latents. The invariant latent space (left), reconstructions from 10-cluster-standard-deviation variations of r_y at fixed $v = \vec{0}$ (middle), and reconstructions from 2-prior-standard-deviation variations of v at fixed r_y (right) are shown.

Table 1: Supervised error rates on MNIST 10,000 image test set.

Technique	**Error rate**
CoVAE Benchmark neural classifier	0.84 ± 0.03
CoVAE (neural classifier using $f_{\theta_{\text{inv}}}(x)$)	0.82 ± 0.03
CoVAE (distance based on $f_{\theta_{\text{inv}}}(x)$)	0.82 ± 0.05
Stacked VAE (M1+M2) (Kingma et al., 2014)	0.96
Adversarial Autoencoders (Makhzani et al., 2016)	0.85 ± 0.02
Capsule Networks (Sabour et al., 2017)	0.25 ± 0.005

We now show quantitatively that the invariant representation r_y learns the class information by showing that it alone can predict the class y as well as a dedicated classifier. In order to predict an unknown label, we employ a direct technique to compute the invariant representation r_y from x. We simply use $f_{\theta_{\text{inv}}}(x)$ from Equation 2. We can then pass $f_{\theta_{\text{inv}}}(x)$ into a neural classifier, or we can find the nearest cluster mean r_y from the training set and assign class probabilities according to $p(\text{label}(x) = y) \propto \exp(-||f_{\theta_{\text{inv}}}(x) - r_y||^2)$. We find that classifying test-set images using this 0-parameter distance metric performs as well as using a neural classifier (2-layer dense dropout network with 128, 64 neurons per layer) with $f_{\theta_{\text{inv}}}(x)$ as input. Note that using $p(y|x) \propto p(x, y)$ is roughly equivalent to our distance-based classifier, as $p(y|x)$ will be maximal when r_y (computed from the training data with label y) is most similar to $f_{\theta_{\text{inv}}}(x)$. Thus, our distance-based technique is a more-direct and efficient approach to classification than using $p(y|x)$.

Our results are shown in Table 1, along with the error rate of a dedicated, end-to-end neural network classifier, identical in architecture to that of $f_{\theta_{\text{inv}}}(x)$, with 2 dropout layers added. This benchmark classifier performs similarly to the simple classifier based on finding the nearest training-set cluster to $f_{\theta_{\text{inv}}}(x)$. This is a strong result, as our classification algorithm based on $f_{\theta_{\text{inv}}}(x)$ has no direct classification objective in its training, see Equation 4. Our uncertainty bands quoted in Table 1 use the standard error on the mean (standard deviation divided by $\sqrt{N-1}$), with $N = 5$ trials.

Results from selected, relevant works from the literature are also shown in Table 1. Kingma et al. (2014) do not provide error bars, and they only provide a fully supervised result for their most-powerful, stacked / pre-trained VAE M1+M2, but it appears as if their learnt representation is less accurate in its classification of unlabelled data. Makhzani et al. (2016) perform slightly worse than CoVAE, but within error bars. Makhzani et al. (2016) train for more than 100 times more epochs than we do, and use over 10 times more parameters in their model, although they use shallower dense networks. Finally, we include Sabour et al. (2017) in our comparisons, as they are seeking to achieve a similar goal of invariant-covariant representations in their modelling. Their performance is significantly superior to ours. However, they use data augmentation and well over 10 times more

Table 2: Semi-supervised error rates on MNIST for various labelled-data-set sizes

Labels	CoVAE	Benchmark	Siddharth et al. (2017)	Kingma et al. (2014) (M2)
100	8.90 ± 0.70	21.91 ± 0.66	9.71 ± 0.91	11.97 ± 1.71
600	3.99 ± 0.17	6.64 ± 0.35	3.84 ± 0.86	4.94 ± 0.13
1000	3.34 ± 0.17	5.43 ± 0.31	2.88 ± 0.79	3.60 ± 0.56
3000	2.23 ± 0.14	2.96 ± 0.11	1.57 ± 0.93	3.92 ± 0.63

parameters than we do, not to mention fine-tuning their model specifically for this task. Note also that Kingma et al. (2014) and Makhzani et al. (2016) train on a training set of 50,000 images, we use 55,000, and Sabour et al. (2017) use 60,000. We are unable to compare to Siddharth et al. (2017) as they do not provide fully supervised results.

3.2 Semi-supervised learning

For semi-supervised learning we combine the labelled and unlabelled objective functions of Equations 4 and 6, respectively, as follows:

$$\sum_{x,\text{ unlab.}} \log p(x) + \sum_{(x,y),\text{ lab.}} \log p(x,y) \;\geq\; \mathcal{L}_{\text{semi}} \;=\; \sum_{x,\text{ unlab.}} \mathcal{L}_{\text{unlab}} + \sum_{(x,y),\text{ lab.}} \mathcal{L}_{\text{lab}} \tag{7}$$

In order to ensure that $q_{\phi_{y\text{-post}}}(y|x)$ does not collapse into the local minimum of predicting a single label for every x value, we add $\log q_{\phi_{y\text{-post}}}(y|x)$ to $\mathcal{L}_{\text{lab}}$. This is done in Kingma et al. (2014) and Siddharth et al. (2017), however, we do not add any hyperparameter in front of this term unlike those works. We also do not add a hyperparameter up-weighting $\mathcal{L}_{\text{lab}}$ overall, as is done in Siddharth et al. (2017). The only hyperparameter tuning we do is to choose 8 versus 16 dimensional latents and to choose the range of m (see Equation 2) to be between 1 and 4, but otherwise no tuning is performed.

Our results from maximising $\mathcal{L}_{\text{semi}}$ in Equation 7 are presented in Table 2. We compare to a benchmark classifier with similar architecture to $q_{\phi_{y\text{-post}}}(y|x)$ trained only on the labelled data, as well as to similar VAE-based semi-supervised work. The number of training epochs are chosen to be 20, 25, 30, and 35, for data set sizes 100, 600, 1000, and 3000, respectively. Each of our experiments is run 5 times to get the mean and (standard) error on the estimate of the mean in Table 2.

We find that CoVAE performs significantly better than the benchmark classifier with the same architecture trained only on the labelled data, especially for small labelled data sets. Furthermore, CoVAE performs within error bars of its most similar comparison, Siddharth et al. (2017). Given its relative simplicity, rapid convergence (20-35 epochs), and lack of hyperparameter tuning performed, we consider this to be a strong indication that CoVAE is an effective approach to jointly learning invariant and covariant representations, including in the regime of minimal labelled data.

4 Conclusions

We have introduced a general technique for jointly learning invariant and covariant representations of data comprised of discrete classes of continuous values. The invariant representation encodes global information about the given class manifold which is ensured by the procedure of reconstructing a data point through complementary samples from the same class. The covariant representation is a stochastic VAE latent that learns the smooth set of transformations that cover the instances of data on that class manifold. We showed that the invariant latents are so widely separated that a 99.18% accuracy can be achieved on MNIST with a simple 0-parameter distance metric based on the invariant embedding. The covariant latent learns to cover the manifold for each class of data with qualitatively excellent samples and interpolations for each class. Finally, we showed that semi-supervised learning based on such latent variable models is competitive with similar approaches in the literature with essentially no hyperparameter tuning.

REFERENCES

Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2013.

D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 2003.

S. R. Bowman, L. Vilnis, O. Vinyals, A. Dai, R. Jozefowicz, and S. Bengio. Generating sentences from a continuous space. In *Conference on Computational Natural Language Learning*, 2016.

S. Brooks, A. Gelman, G. Jones, and M. Xiao-Li. *Handbook of Markov chain monte carlo*. CRC press, 2011.

S. M. A. Eslami, D. J. Rezende, F. Besse, F. Viola, A. S. Morcos, M. Garnelo, A. Ruderman, A. A. Rusu, I. Danihelka, K. Gregor, D. P. Reichert, L. Buesing, T. Weber, O. Vinyals, D. Rosenbaum, N. Rabinowitz, H. King, C. Hillier, M. Botvinick, D. Wierstra, K. Kavukcuoglu, and D. Hassabis. Neural scene representation and rendering. *Science*, 2018.

L. Falorsi, P. de Haan, T. R. Davidson, N. D. Cao, M. Weiler, P. Forré, and T. S. Cohen. Explorations in homeomorphic variational auto-encoding. *arXiv preprint 1807.04689*, 2018.

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 *International Conference on Learning Representations*, 2017.

E. Jang, S. Gu, and B. Poole. Categorical reparameterization with gumbel-softmax. In *International Conference on Learning Representations*, 2017.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

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

D. P. Kingma, S. Mohamed, D. J. Rezende, and M. Welling. Semi-supervised learning with deep generative models. In *Advances in Neural Information Processing Systems*. 2014.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, 1998.

C. J. Maddison, A. Mnih, and Y. W. Teh. The concrete distribution: a continuous relaxation of discrete random variables. In *International Conference on Learning Representations*, 2017.

A. Mahendran and A. Vedaldi. Understanding deep image representations by inverting them. In *The IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

A. Makhzani, J. Shlens, N. Jaitly, and I. Goodfellow. Adversarial autoencoders. In *International Conference on Learning Representations*, 2016.

A. Mansbridge, R. Fierimonte, I. Feige, and D. Barber. Improving latent variable descriptiveness with autogen. *arXiv preprint 1806.04480*, 2018.

L. McInnes and J. Healy. UMAP: uniform manifold approximation and projection for dimension reduction. *arXiv preprint 1802.03426*, 2018.

T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*. 2013.

A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *International Conference on Learning Representations*, 2016.

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

S. Sabour, N. Frosst, and G. E. Hinton. Dynamic routing between capsules. In *Advances in Neural Information Processing Systems*. 2017.

N. Siddharth, B. T. 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*, 2017.

A EXPERIMENTAL SETUP

For our implementation of CoVAE, we use relatively standard neural networks. All of our experiments use implementations with well under 1 million parameters in total, converge within 1-2 hours (on a Tesla K80 GPU), and are exposed to minimal hyperparameter tuning.

In particular, for the deterministic class-representation vector r_y given in Equation 2, we parametrise $f_{\theta_{\text{inv}}}(x)$ using a 5-layer, stride-2 (stride-1 first layer), with 5x5 kernal size, convolution network, followed by a dense hidden layer. The mean of these m embeddings $f_{\theta_{\text{inv}}}(x_y^i)$ is taken, followed then by another dense hidden layer, and the final linear dense output layer. This is shown for a $y = 6$ MNIST digit in the top shaded box of Figure 5. Our implementation uses $(8, 16, 32, 64, 64)$ filters in the convolution layers, and $(128, 64)$ hidden units in the two subsequent dense layers for a 16 dimensional latent (the number of units in the dense layers are halved when using 8 dimensional latents, as in our semi-supervised experiments).

We parametrise the approximate posterior distribution $q_{\phi_{\text{cov}}}(v|r_y, x)$ over the covariant latent as a diagonal-covariance normal distribution, $\mathcal{N}(\mu_{\phi_{\text{cov}}}(r_y, x), \sigma^2_{\phi_{\text{cov}}}(r_y, x))$, following the SGVB algorithm (Kingma & Welling, 2014; Rezende et al., 2014). For $\mu_{\phi_{\text{cov}}}(r_y, x)$ and $\sigma^2_{\phi_{\text{cov}}}(r_y, x)$, we use the identical convolution architecture as for the invariant embedding network as an initial embedding for the data point x. This embedding is then concatenated with the output of a single dense layer that transforms r_y, the output of which is then passed to one more dense hidden layer for each μ and σ^2 separately. This is shown in the bottom shaded box of Figure 5.

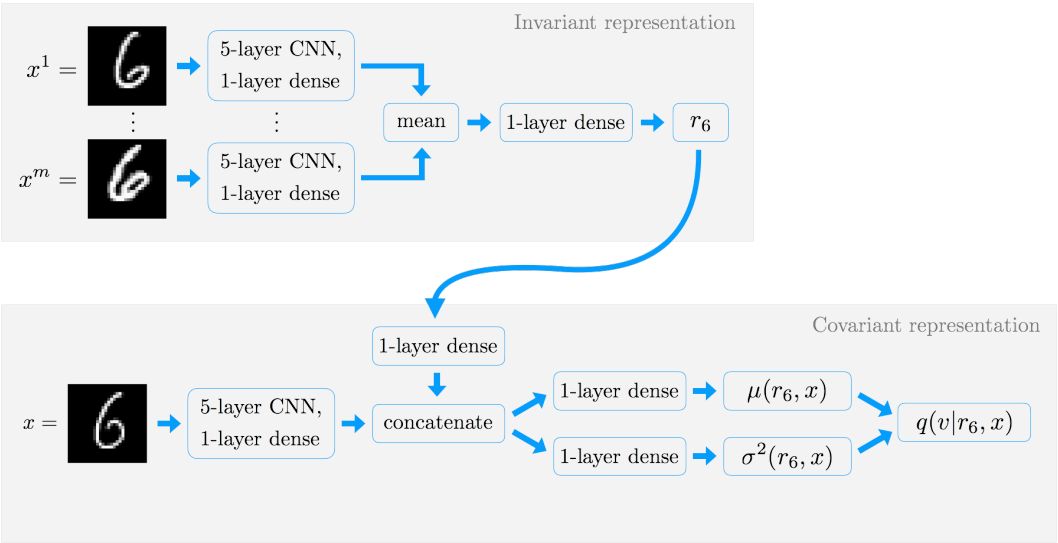

Figure 5: Example of variational encoding of a particular MNIST digit x (from the 6 class) both in terms of its invariant representation (top) and its covariant representation (bottom).

The generative model $p_\theta(x|r_y, v)$ is based on the DCGAN-style transposed convolutions (Radford et al., 2016), and is assumed to be a Bernoulli distribution over the conditionally independent image pixels (as in Kingma et al. (2014)). Both the invariant representation r_y and the covariant representation v, are separately passed through a single-layer dense network before being concatenated and passed through another dense layer. This flat embedding that combines both representations is then

transpose convolved to get the output image in a way the mirrors the 5-layer convolution network used to embed the representations in the first place. That is, we use $(64, 128)$ hidden units in the first two dense layers, and then $(64, 32, 16, 8, 1)$ filters in each transpose convolution layer, all with 5x5 kernals and stride 2, except the last layer, which is a stride-1 convolution layer.

In our semi-supervised experiments, we implement $q_{\phi_{y\text{-post}}}(y|x)$ using the same (5-CNN, 1-dense) encoding block to provide an initial embedding for x. This is then concatenated with stop_grad$(f_{\theta_{\text{inv}}}(x))$ and passed to a 2-layer dense dropout network with (256, 128) units. The use of stop_grad$(f_{\theta_{\text{inv}}}(x))$ is simply that $f_{\theta_{\text{inv}}}(x)$ is learning a highly relevant, invariant representation of x that $q_{\phi_{y\text{-post}}}(y|x)$ might as well get access to. However, we do not allow gradients to pass through this operation since $f_{\theta_{\text{inv}}}(x)$ is meant to learn from the complementary data of known same-class members only.

As discussed in Section 2, the number of complementary samples m used to reconstruct r_y (see Equation 2) is chosen randomly each training step in order to ensure that r_y is insensitive to m. For our supervised experiments where labelled data are plentiful, m is randomly select between 1 and 7, whereas in the semi-supervised case m is randomly selected between 1 and 4.

Finally, all activation functions that are not fixed by model outputs are taken to be rectified linear units. We use Adam (Kingma & Ba, 2015) for training with default settings, and choose a batch size of 32 at the beginning of training, which we double successively throughout training.