Variational Autoencoders For Semi-Supervised Deep Metric Learning

Nathan Safir[‡*], Meekail Zain[§], Curtis Godwin[‡], Eric Miller[‡], Bella Humphrey[§], Shannon P Quinn[§¶]

Abstract—Deep metric learning (DML) methods generally do not incorporate unlabelled data. We propose borrowing components of the variational autoencoder (VAE) methodology to extend DML methods to train on semi-supervised datasets. We experimentally evaluate the atomic benefits to the perform- ing DML on the VAE latent space such as the enhanced ability to train using unlabelled data and to induce bias given prior knowledge. We find that jointly training DML with an autoencoder and VAE may be potentially helpful for some semi-suprevised datasets, but that a training routine of alternating between the DML loss and an additional unsupervised loss across epochs is generally unviable.

Index Terms—Variational Autoencoders, Metric Learning, Deep Learning, Representation Learning, Generative Models

Introduction

Within the broader field of representation learning, metric learning is an area which looks to define a distance metric which is smaller between similar objects (such as objects of the same class) and larger between dissimilar objects. Oftentimes, a map is learned from inputs into a low-dimensional latent space where euclidean distance exhibits this relationship, encouraged by training said map against a loss (cost) function based on the euclidean distance between sets of similar and dissimilar objects in the latent space. Existing metric learning methods are generally unable to learn from unlabelled data, which is problematic because unlabelled data is often easier to obtain and is potentially informative.

We take inspiration from variational autoencoders (VAEs), a generative representation learning architecture, for using unlabelled data to create accurate representations. Specifically, we look to evaluate three atomic improvement proposals that detail how pieces of the VAE architecture can create a better deep metric learning (DML) model on a semi-supervised dataset. From here, we can ascertain which specific qualities of how VAEs process unlabelled data are most helpful in modifying DML methods to train with semi-supervised datasets.

First, we propose that the autoencoder structure of the VAE helps the clustering of unlabelled points, as the reconstruction loss may help incorporate semantic information from unlabelled sources. Second, we propose that the structure of the VAE latent space, as it is confined by a prior distribution, can be used to induce bias in the latent space of a DML system. For instance, if we know a dataset contains N -many classes, creating a prior distribution that is a learnable mixture of N gaussians may help produce better representations. Third, we propose that performing DML on the latent space of the VAE so that the DML task can be jointly optimized with the VAE to incorporate unlabelled data may help produce better representations.

Each of the three improvement proposals will be evaluated experimentally. The improvement proposals will be evaluated by comparing a standard DML implementation to the same DML implementation:

- jointly optimized with an autoencoder
- while structuring the latent space around a prior distribution using the VAE's KL-divergence loss term between the approximated posterior and prior
- jointly optimized with a VAE

Our primary contribution is evaluating these three improvement proposals. Our secondary contribution is presenting the results of the joint approaches for VAEs and DML for more recent metric losses that have not been jointly optimized with a VAE in previous literature.

Related Literature

The goal of this research is to investigate how components of the variational autoencoder can help the performance of deep metric learning in semi supervised tasks. We draw on previous literature to find not only prior attempts at this specific research goal but also work in adjacent research questions that proves insightful. In this review of the literature, we discuss previous related work in the areas of Semi-Supervised Metric Learning and VAEs with Metric Losses.

Semi-Supervised Metric Learning

There have been previous approaches to designing metric learning architectures which incorporate unlabelled data into the metric learning training regimen for semi-supervised datasets. One of the original approaches is the MPCK-MEANS algorithm proposed by Bilenko et al. ([BBM04]), which adds a penalty for placing labelled inputs in the same cluster which are of a different class or in different clusters if they are of the same class. This penalty is proportional to the metric distance between the pair of inputs.

* *Corresponding author: nssafir@gmail.com*
‡ *Institute for Artificial Intelligence, University of Georgia, Athens, GA 30602 USA*
§ *Department of Computer Science, University of Georgia, Athens, GA 30602 USA*
¶ *Department of Cellular Biology, University of Georgia, Athens, GA 30602 USA*

Baghshah and Shouraki ([BS09]) also looks to impose similar constraints by introducing a loss term to preserve locally linear relationships between labelled and unlabelled data in the input space. Wang et al. ([WYF13]) also use a regularizer term to preserve the topology of the input space. Using VAEs, in a sense, draws on this theme: though there is not explicit term to enforce that the topology of the input space is preserved, a topology of the inputs is intended to be learned through a low-dimensional manifold in the latent space.

One more recent common general approach to this problem is to use the unlabelled data's proximity to the labelled data to estimate labels for unlabelled data, effectively transforming unlabelled data into labelled data. Dutta et al. ([DHS21]) and Li et al. ([LYZ+19]) propose a model which uses affinity propagation on a k-Nearest-Neighbors graph to label partitions of unlabelled data based on their closest neighbors in the latent space. Wu et al. ([WFZ20]) also look to assign pseudo-labels to unlabelled data, but not through a graph-based approach. Instead, the proposed model looks to approximate "soft" pseudo-labels for unlabelled data from the metric learning similarity measure between the embedding of unlabelled data and the center of each input of each class of the labelled data.

Several of the recent graph based approaches can be considered state-of-the-art for semi supervised metric learning. Li et. al.'s paper states their methods achieve 98.9 percent clustering accuracy on the MNIST dataset with 10% labelled data, outperforming two similar state-of-the-art methods, DFCM ([ARJM18]) and SDEC ([RHD+19]), by roughly 8 points. Dutta et. al.'s method also outperforms 5 other state for the R@1 metric (the "percentage of test examples" that have at least one 1 "nearest neighbor from the same class.") by at leat 1.2 on the MNIST dataset, as well as the Fashion-MNIST and CIFAR-10 datasets. It is difficult to compare the two approaches as the evaluation metrics used in each paper differ. Li et al.'s paper has been cited rather heavily relative to other papers in the field and can be considered state of the art for semi-supervised DML on MNIST. The paper also provides a helpful metric (98.9 percent clustering accuracy on the MNIST dataset with 10% labelled data) to use as a reference point for the results in this paper.

VAEs with Metric Loss

Some approaches to incorporating labelled data into VAEs use a metric loss to govern the latent space more explicitly. Lin et al. ([LDD+18]) model the intra-class invariance (i.e. the class-related information of a data point) and intra-class variance (i.e. the distinct features of a data point not unique to it's class) seperately. Like several other models in this section, this paper's proposed model incorporates a metric loss term for the latent vectors representing intra-class invariance and the latent vectors representing both intra-class invariance and intra-class variance.

Kulkarni et al. ([KCJ20]) incorporate labelled information into the VAE methodology in two ways. First, a modified architecture called the CVAE is used in which the encoder and generator of the VAE is not only conditioned on the input X and latent vector z, respectively, but also on the label Y. The CVAE was introduced in previous papers ([SLY15]) ([DCGO19]). Second, the authors add a metric loss, specifically a multi-class N-pair loss ([Soh16]), in the overall loss function of the model. While it is unclear how the CVAE technique would be adapted in a semi-supervised setting, as there is not a label Y associated with each datapoint X, we also experiment with adding a (different) metric loss to the overall VAE loss function.

Most recently, Grosnit et al. ([GTM+21]) leverage a new training algorithm for combining VAEs and DML for Bayesian Optimization and said algorithm using simple, contrastive, and triplet metric losses. We look to build on this literature by also testing a combined VAE DML architecture on more recent metric losses, albeit using a simpler training regimen.

Deep Metric Learning (DML)

Metric learning attempts to create representations for data by training against the similarity or dissimilarity of samples. In a more technical sense, there are two notable functions in DML systems. Function f_θ is a neural network which maps the input data X to the latent points Z (i.e. $f_\theta : X \mapsto Z$, where θ is the network parameters). Generally, Z exists in a space of much lower dimensionality than X (eg. X is a set of 28×28 pixel pictures such that $X \subset \mathbb{R}^{28\times 28}$ and $Z \subset \mathbb{R}^{10}$).

The function $D_{f_\theta}(x,y) = D(f_\theta(x), f_\theta(y))$ represents the distance between two inputs $x, y \in X$. To create a useful embedding model f_θ, we would like for f_θ to produce large values of $D_{f_\theta}(x,y)$ when x and y are dissimilar and for f_θ to produce small values of $D_{f_\theta}(x,y)$ when x and y are similar. In some cases, dissimilarity and similarity can refer to when inputs are of different and the same classes, respectively.

It is common for the Euclidean metric (i.e. the L_2 metric) to be used as a distance function in metric learning. The generalized L_p metric can be defined as follows, where $z_0, z1 \in \mathbb{R}^d$.

$$D_p(z_0, z_1) = ||z_0 - z_1||_p = (\sum_{i=1}^{d} |z_{0_i} - z_{1_i}|^p)^{1/p}$$

If we have chosen f_θ (a neural network) and the distance function D (the L_2 metric), the remaining component to be defined in a metric learning system is the loss function for training f. In practice, we will be using triplet loss ([SKP15]), one of the most common metric learning loss functions.

Methodology

We look to discover the potential of applying components of the VAE methodology to DML systems. We test this through presenting incremental modifications to the basic DML architecture. Each modified architecture corresponds to an improvement proposal about how a specific part of the VAE training regime and loss function may be adapted to assist the performance of a DML method for a semi-supervised dataset.

The general method we will take for creating modified DML models involves extending the training regimen to two phases, a supervised and unsupervised phase. In the supervised phase the modified DML model behaves identically to the base DML model, training on the same metric loss function. In the unsupervised phase, the DML model will train against an unsupervised loss inspired by the VAE. This may require extra steps to be added to the DML architecture. In the pseudocode, s refers to boolean variable representing if the current phase is supervised. α is a hyperparameter which modulates the impact of the unsupervised on total loss for the DML autoencoder.

Algorithm 1: Base DML Training Routine

Input: Dataset dataset D, encoder network f, metric loss function m, learning rate γ, weights θ
Result: updated weights θ
for *batch* x, y *in* D **do**
  $z = f_\theta(x)$;
  $c = m(z, y)$;
  Compute $\frac{dc}{d\theta}$ with backpropogation;
  $\theta = \theta - \gamma \frac{dc}{d\theta}$;
end

Improvement Proposal 1

We first look to evaluate the improvement proposal that adding a reconstruction loss to a DML system can improve the quality of clustering in the latent representations on a semi-supervised dataset. Reconstruction loss in and of itself enforces a similar semantic mapping onto the latent space as a metric loss, but can be computed without labelled data. In theory, we believe that the added constraint that the latent vector must be reconstructed to approximate the original output will train the spatial positioning to reflect semantic information. Following this reasoning, observations which share similar semantic information, specifically observations of the same class (even if not labelled as such), should intuitively be positioned nearby within the latent space. To test if this intuition occurs in practice, we evaluate if a DML model with an autoencoder structure and reconstruction loss (described in further detail below) will perform better than a plain DML model in terms of clustering quality. This will be especially evident for semi-supervised datasets in which the amount of labelled data is not feasible for solely supervised DML.

Given a semi-supervised dataset, we assume a standard DML system will use only the labelled data and train given a metric loss L_{metric} (see Algorithm 1). Our modified model DML Autoencoder will extend the DML model's training regime by adding a decoder network which takes the latent point z as input and produces an output $\hat{x}$. The unsupervised loss L_U is equal to the reconstruction loss.

Improvement Proposal 2

Say we are aware that a dataset has n classes. It may be useful to encourage that there are n clusters in the latent space of a DML model. This can be enforced by using a prior distribution containing n many Gaussians. As we wish to measure only the affect of inducing bias on the representation without adding any complexity to the model, the prior distribution will not be learnable (unlike VAE with VampPrior). By testing whether the classes of points in the latent space are organized along the prior components we can test whether bias can be induced using a prior to constrain the latent space of a DML. By testing whether clustering improves performance, we can evaluate whether this inductive bias is helpful.

Given a fully supervised dataset, we assume a standard DML system will use only the labelled data and train given a metric loss L_{metric}. Our modified model will extend the DML system's training regime by setting the unsupervised loss to a KL divergence term that measures the difference between posterior distributions and a prior distribution. It should also be noted that, like the VAE encoder, we will map the input not to a latent point but to a latent distribution. The latent point is stochastically sampled from the latent distribution during training. Mapping the input to a distribution instead of a point will allow us to calculate the KL divergence.

In practice, we will be evaluating a DML model with a unit prior and a DML model with a mixture of gaussians (GMM) prior. The latter model constructs the prior as a mixture of n gaussians – each the vertice of the unit (i.e. each side is 2 units long) hypercube in the latent space. The logvar of each component is set equal to one. Constructing the prior in this way is beneficial in that it is ensured that each component is evenly spaced within the latent space, but is limiting in that there must be exactly 2^d components in the GMM prior. Thus, to test, we will test a dataset with 10 classes on the latent space dimensionality of 4, such that there are $2^4 = 16$ gaussian components in the GMM prior. Though the number of prior components is greater than the number of classes, the latent mapping may still exhibit the pattern of classes forming clusters around the prior components as the extra components may be made redundant.

The drawback of the decision to set the GMM components' means to the coordinates of the unit hypercube's vertices is that the manifold of the chosen dataset may not necessarily exist in 4 dimensions. Choosing gaussian components from a d-dimensional hypersphere in the latent space $\mathscr{R}^d$ would solve this issue, but there does not appear to be a solution for choosing n evenly spaced points spanning d dimensions on a d-dimensional hypersphere. KL Divergence is calculated with a monte carlo approximation for the GMM and analytically with the unit prior.

Improvement Proposal 3

The third improvement proposal we look to evaluate is that given a semi-supervised dataset, optimizing a DML model jointly with a VAE on the VAE's latent space will produce superior clustering than the DML model individually. The intuition behind this approach is that DML methods can learn from only supervised data and VAE methods can learn from only unsupervised data; the proposed methodology will optimize both tasks simultaneously to learn from both supervised and unsupervised data.

The MetricVAE implementation we create jointly optimizes the VAE task and DML task on the VAE latent space. The unsupervised loss is set to the VAE loss. The implementation uses the VAE with VampPrior model instead of the vanilla VAE.

Results

Experimental Configuration

Each set of experiments shares a similar hyperparameter search space. Below we describe the hyperparameters that are included in the search space of each experiment and the evaluation method.

Learning Rate (lr): Through informal experimentation, we have found that the learning rate of 0.001 causes the models to converge consistently (relative to 0.005 and 0.0005). The learning rate is thus set to 0.001 in each experiment.

Algorithm 2: DML Autoencoder Training Routine

Input: Dataset dataset D, encoder network f, decoder network g, metric loss function m, learning rate γ, coefficient α, weights θ
Result: updated weight θ
for *batch* x, y *in* D **do**
  $z = f_\theta(x)$;
  $\hat{x} = g_\theta(z)$;
  if s **then**
    $c = (1-\alpha) * m(z, y) + \alpha * MSE(\hat{x}, x)$;
  else
    $c = \alpha * MSE(\hat{x}, x)$;
  end
  Compute $\frac{dc}{d\theta}$ with backpropogation;
  $\theta = \theta - \gamma\frac{dc}{d\theta}$;
end

Algorithm 3: DML with Prior Training Routine

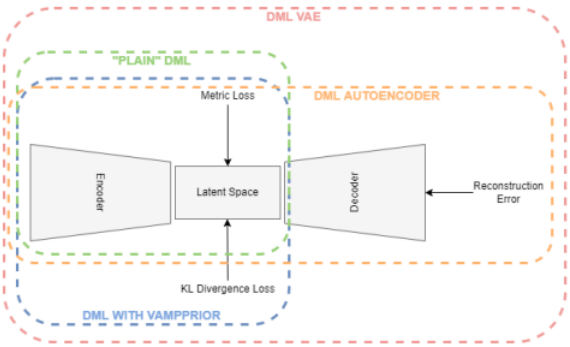
Input: Dataset dataset D, encoder network f, metric loss function m, learning rate γ, coefficient α, prior distribution mean and log-variance μ_p, σ_p, weights θ
Result: updated weights θ
for *batch* x, y *in* D **do**
  $\mu, \sigma = f_\theta(x)$;
  $z \sim N(\mu, \sigma)$;
  if s **then**
    $c = (1-\alpha) * m(z, y) + \alpha * KL(z, \mu, \sigma, \mu_p, \sigma_p)$;
  else
    $c = \alpha * KL(z, \mu, \sigma, \mu_p, \sigma_p)$;
  end
  Compute $\frac{dc}{d\theta}$ with backpropogation;
  $\theta = \theta - \gamma\frac{dc}{d\theta}$;
end

Algorithm 4: Monte-Carlo KL Divergence Algorithm

Input: Latent variable z, approximate posterior distribution mean and variance μ,σ, prior distribution mean and log-variance μ_p, σ_p
Result: KL Divergence between distributions q and p
$P(z|\mu, \sigma) = -0.5 * \frac{(z-\mu)^2}{\exp\sigma}$;
$Q(z|\mu_p, \sigma_p) = -0.5 * \frac{(z-\mu_p)^2}{\exp\sigma_p}$;
return $Q(z|\mu_p, \sigma_p) - P(z|\mu, \sigma)$

Algorithm 5: DML VAE Training Routine

Input: Dataset dataset D, encoder network f, decoder network g, metric loss function m, learning rate γ, coefficient α, coefficient β, prior distribution mean and log-variance μ_p, σ_p, weights θ
Result: updated weights θ
for *batch* x, y *in* D **do**
  $\mu, \sigma = f_\theta(x)$;
  $z \sim N(\mu, \sigma)$;
  $\hat{x} = g_\theta(z)$;
  $c_{vae} = MSE(x, x_{hat}) + \beta * KL(z, \mu, \sigma, \mu_p)$;
  if s **then**
    $c = (1-\alpha) * m(z, y) + \alpha * c_{vae}$;
  else
    $c = \alpha * c_{vae}$;
  end
  Compute $\frac{dc}{d\theta}$ with backpropogation;
  $\theta = \theta - \gamma\frac{dc}{d\theta}$;
end



Latent Space Dimensionality (lsdim): Latent space dimensionality refers to the dimensionality of the vector output of the encoder of a DML network or the dimensionality of the posterior distribution of a VAE (also the dimensionality of the latent space). When the latent space dimensionality is 2, we see the added benefit of creating plots of the latent representations (though we can accomplish this through using dimensionality reduction methods like tSNE for higher dimensionalities as well). Example values for this hyperparameter used in experiments are 2, 4, and 10.

Alpha: Alpha (α) is a hyperapameter which refers to the balance between the unsupervised and supervised losses of some of the modified DML models. More details about the role of α in the model implementations are discussed in the methodology section of the model. Potential values for alpha are each between 0 (exclusive) and 1 (inclusive). We do not include 0 in this set as if α is set to 0, the model is equivalent to the fully supervised plain DML model because the supervised loss would not be included. If α is set to 1, then the model would train on only the unsupervised loss; for instance if the DML Autoencoder had α set to 1, then the model would be equivalent to an autoencoder.

Partial Labels Percentage (pl%): The partial labels percentage hyperparameter refers to the percentage of the dataset that is labelled and thus the size of the partion of the dataset that can be used for labelled training. Of course, each of the datasets we use is fully labelled, so a partially labelled datset can be trivially constructed by ignoring some of the labels. As the sizes of the dataset vary, each percentage can refer to a different number of labelled samples. Values for the partial label percentage we use across experiments include 0.01, 0.1, and 10 (with each value referring to the percentage).

Datasets: Two datasets are used for evaluating the models. The first dataset is MNIST ([LC10]), a very popular dataset in machine learning containing greyscale images of handwritten digits. The second dataset we use is the organ OrganAMNIST dataset from MedMNIST v2 ([YSW+21]). This dataset contains 2D slices from computed tomography images from the Liver Tumor Segmentation Benchmark – the labels correspond to the classification of 11 different body organs. The decision to use a second dataset was motivated because as the improvement proposals are tested over more datasets, the results supporting the improvement proposals become more generalizable. The decision to use the OrganAMNIST dataset specifically is motivated in part due to the Quinn Research Group working on similar tasks for biomedical imaging ([ZRS+20]). It is also motivated in part because OrganAMNIST is a more difficult dataset, at least for the classfication task, as the leading accuracy for MNIST is .9991 ([ALP+20]) while the leading accuracy for OrganAMNIST is .951 ([YSW+21]). The MNIST and OrganAMNIST datasets are similar in dimensionality (1 x 28 x 28), number of samples (60,000 and 58,850, respectively) and in that they are both greyscale.

Evaluation: We evaluate the results by running each model on a test partition of data. We then take the latent points Z generated by the model and the corresponding labels Y. Three classifiers (sklearn's implementation of RandomForest, MLP, and kNN) each output predicted labels $\hat{Y}$ for the latent points. In most of the charts shown, however, we only include the kNN classification output due to space constraints and the lack of meaningful difference between the output for each classifier. We finally measure the quality of the predicted labels $\hat{Y}$ using the Adjusted Mutual Information Score (AMI) ([**?**]) and accuracy (which is still helpful but is also easier to interpret in some cases). This scoring metric is common in research that looks to evaluate clustering performance ([ZG21]) ([EKGB16]). We will be using sklearn's implementation of AMI ([PVG+11]). The performance of a classifier on the latent points intuitively can be used as a measure of quality of clustering.

Improvement Proposal 1 Results: Benefits of Reconstruction Loss

In evaluating the first improvement proposal, we compare the performance of the plain DML model to the DML Autoencoder model. We do so by comparing the performance of the plain DML system and the DML Autoencoder across a search space containing the lsdim, alpha, and pl% hyperparameters and both datasets.

In Table 1 and Table 2, we observe that for relatively small amounts of labelled samples (the partial labels percentages of 0.01 and 0.1 correspond to 6 and 60 labelled samples respectively), the DML Autoencoder severely outperforms the DML model. However, when the number of labelled samples increases (the partial labels percentage of 10 correspond to 6000 labelled samples respectively), the DML model significantly outperforms the DML Autoencoder. This trend is not too surprising, as when there is sufficient data to train unsupervised methods and insufficient data to train supervised method, as is the case for the 0.01 and 0.1 partial label percentages, the unsupervised method will likely perform better.

The data looks to show that adding a reconstruction loss to a DML system can improve the quality of clustering in the latent representations on a semi-supervised dataset when there are small amounts (roughly less than 100 samples) of labelled data and a sufficient quantity of unlabelled data. But an important caveat is that it is not convincing that the DML Autoencoder effectively combined the unsupervised and supervised losses to create a superior model, as a plain autoencoder (i.e. the DML Autoencoder

Fig. 1: *Sample images from the MNIST (left) and OrganAMNIST of MedMNIST (right) datasets*

with $\alpha = 1$) outperforms the DML for the partial labels percentage of or less than 0.1% and underperforms the DML for the partial labels percentage of 10%.

Improvement Proposal 2 Results: Incorporating Inductive Bias with a Prior

In evaluating the second improvement proposal, we compare the performance of the plain DML model to the DML with a unit prior and a DML with a GMM prior. The DML prior with the GMM prior will have 2^2 = 4 gaussian components when lsdim = 2 and 2^4 = 16 components when lsdim = 4. Our broad intention is to see if changing the shape (specifically the number of components) of the prior can induce bias by affecting the pattern of embeddings. We hypothesize that when the GMM prior contains n components and n is slightly greater than or equal to the number of classes, each class will cluster around one of the prior components. We will test this for the GMM prior with 16 components (lsdim = 4) as both the MNIST and MedMNIST datasets have 10 classes. We are unable to set the number of GMM components to 10 as our GMM sampling method only allows for the number of components to equal a power of 2. Bseline models include a plain DML and a DML with a unit prior (the distribution N(0, 1)).

In Table 3, it is very evident that across both datasets, the DML models with any prior distribution all devolve to the null model (i.e. the classifier is no better than random selection). From the visualizations of the latent embeddings, we see that the embedded data for the DML models with priors appears completely random. In the case of the GMM prior, it also does not appear to take on the shape of the prior or reflect the number of components in the prior. This may be due to the training routine of the DML models. As the KL divergence loss, which can be said to "fit" the embeddings to the prior, trains on alternating epochs with the supervised DML loss, it is possible that the two losses are not balanced correctly during the training process. From the discussed results, it is fair to state that adding a prior distribution to a DML model through training the model on the KL divergence between the prior and approximated posterior distributions on alternating epochs does is not an effective way to induce bias in the latent space.

Improvement Proposal 3 Results: Jointly Optimizing DML with VAE

To evaluate the third improvement proposal, we compare the performance of DMLs to MetricVAEs (defined in the previous chapter) across several metric losses. We run experiments for triplet loss, supervised loss, and center loss DML and MetricVAE models. To evaluate the improvement proposal, we will assess whether the model performance improves for the MetricVAE over the DML for the same metric loss and other hyper parameters.

Like the previous improvement proposal, the proposed MetricVAE model does not perform better than the null model. As with improvement proposal 2, it is possible this is because the training routine of alternating between supervised loss (in this case, metric loss) and unsupervised (in this case, VAE loss) is not optimal for training the model.

We have trained a seperate combined VAE and DML model which trains on both the unsupervised and supervised loss each epoch instead of alternating between the two each epoch. In the results for this model, we see that an alpha value of over zero (i.e. incorporating both the supervised metric loss into the overall MVAE loss function) can help improve performance especially among lower dimensionalities. Given our analysis of the data, we see that incorporating the DML loss to the VAE is potentially helpful, but only when training the unsupervised and supervised losses jointly. Even in that case, it is unclear whether the MVAE performs better than the corresponding DML model even if it does perform better than the corresponding VAE model.

Conclusion

Conclusion

In this work, we have set out to determine how DML can be extended for semi-supervised datasets by borrowing components of the variational autoencoder. We have formalized this approach through defining three specific improvement proposals. To evaluate each improvement proposal, we have created several variations of the DML model, such as the DML Autoencoder, DML with Unit/GMM Prior, and MVAE. We then tested the performance of the models across several semi-supervised partitions of two datasets, along with other configurations of hyperparameters.

We have determined from the analysis of our results, there is too much dissenting data to clearly accept any three of the improvement proposals. For improvement proposal 1, while the DML Autoencoder outperforms the DML for semisupervised datasets with small amounts of labelled data, it's peformance is not consistently much better than that of a plain autoencoder which uses no labelled data. For improvement proposal 2, each of the DML models with an added prior performed extremely poorly, near or at the level of the null model. For improvement proposal 3, we see the same extremely poor performance from the MVAE models.

From the results in improvement proposals 1 and 3, we find that there may be potential in incorporating the autoencoder and VAE loss terms into DML systems. However, we were unable to show that any of these improvement proposals would consistently outperform the both the DML and fully unsupervised architectures in semisupervised settings. We also found that the training routine used for the improvement proposals, in which the loss function would alternate between supervised and unsupervised losses each epoch, was not effective. This is especially evident in comparing the two combined VAE DML models for improvement proposal 3.

pl%	lsdim	knn acc	knn MI
0.01	2	**0.2618**	**0.1124**
	10	0.2610	0.1073
0.1	2	0.4256	0.2904
	10	**0.6556**	**0.4983**
10	2	0.7476	0.6200
	10	**0.9502**	**0.8838**

pl%	alpha	lsdim	knn acc	knn MI
0.01	0.25	2	0.3957	0.2815
		10	**0.8351**	**0.7172**
	0.5	2	0.3358	0.1932
		10	0.6688	0.5218
	0.75	2	0.3112	0.1835
		10	0.8323	0.7102
	1	2	0.4166	0.2923
		10	0.8319	0.7089
0.1	0.25	2	0.4349	0.3339
		10	0.8718	0.7616
	0.5	2	0.4033	0.3150
		10	**0.8798**	**0.7725**
	0.75	2	0.3812	0.2771
		10	0.7820	0.6399
	1	2	0.4202	0.3022
		10	0.8589	0.7463
10	0.25	2	0.1028	0
		10	0.8478	0.7258
	0.5	2	0.3465	0.2189
		10	0.7925	0.6606
	0.75	2	0.3536	0.2175
		10	0.8497	0.7373
	1	2	0.5137	0.3793
		10	**0.8570**	**0.7407**

Fig. 2: *Table 1: Comparison of the DML (left) and DML Autoencoder (right) models for the MNIST dataset. Bolded values indicate best performance for each partial labels percentage partition (pl%).*

pl%	lsdim	knn acc	knn MI
0.01	2	0.3844	0.2637
	10	**0.6997**	**0.5529**
0.1	2	0.4323	0.3165
	10	**0.8309**	**0.7129**
10	2	0.4616	0.3614
	10	**0.8487**	**0.7339**

pl%	alpha	lsdim	knn acc	knn MI
0.01	0.25	2	0.3683	0.2685
		10	0.8060	0.6792
	0.5	2	0.3388	0.2146
		10	**0.8378**	**0.7207**
	0.75	2	0.3419	0.2183
		10	0.7568	0.5987
	1	2	0.3844	0.2637
		10	0.6997	0.5529
0.1	0.25	2	0.3717	0.2496
		10	0.8514	0.7381
	0.5	2	0.4314	0.2918
		10	0.8553	0.7419
	0.75	2	0.3896	0.2502
		10	**0.8597**	**0.7454**
	1	2	0.4323	0.3165
		10	0.8309	0.7129
10	0.25	2	0.4316	0.3354
		10	0.8560	0.7394
	0.5	2	0.3824	0.2404
		10	**0.8692**	**0.7589**
	0.75	2	0.5118	0.3891
		10	0.8250	0.7078
	1	2	0.4616	0.3614
		10	0.8487	0.7339

Fig. 3: *Table 2: Comparison of the DML (left) and DML Autoencoder (right) models for the MEDMNIST dataset..*

Future Work

In the future, it would be worthwhile to evaluate these improvement proposals using a different training routine. We have stated previously that perhaps the extremely poor performance of the DML with a prior and MVAE models may be due to alternating on training against a supervised and unsupervised loss. Further research could look to develop or compare several different training routines. One alternative would be alternating between losses at each batch instead of each epoch. Another alternative, specifically for the MVAE, may be first training DML on labelled data, training a GMM on it's outputs, and then using the GMM as the prior distribution for the VAE.

Another potentially interesting avenue for future study is in investigating a fourth improvement proposal: the ability to define a Riemannian metric on the latent space. Previous research has shown a Riemannian metric can be computed on the latent space of the VAE by computing the pull-back metric of the VAE's decoder function ([AHS20]). Through the Riemannian metric we could calculate metric losses such as triplet loss with a geodesic instead of euclidean distance. The geodesic distance may be a more accurate representation of similarity in the latent space than euclidean distance as it accounts for the structure of the input data.

REFERENCES

[AHS20] Georgios Arvanitidis, Søren Hauberg, and Bernhard Schölkopf. Geometrically enriched latent spaces. *arXiv preprint arXiv:2008.00565*, 2020. doi:10.48550/arXiv.2008.00565.

[ALP+20] Sanghyeon An, Min Jun Lee, Sanglee Park, Heerin Yang, and Jungmin So. An ensemble of simple convolutional neural network models for MNIST digit recognition. *CoRR*, abs/2008.10400, 2020. URL: https://arxiv.org/abs/2008.10400, arXiv:2008.10400, doi:10.48550/arXiv.2008.10400.

[ARJM18] Ali Arshad, Saman Riaz, Licheng Jiao, and Aparna Murthy. Semi-supervised deep fuzzy c-mean clustering for software fault prediction. *IEEE Access*, 6:25675–25685, 2018. doi:10.1109/ACCESS.2018.2835304.

[BBM04] Mikhail Bilenko, Sugato Basu, and Raymond J Mooney. Integrating constraints and metric learning in semi-supervised clustering. In *Proceedings of the twenty-first international conference on Machine learning*, page 11, 2004. doi:10.1145/1015330.1015360.

[BS09] Mahdieh Soleymani Baghshah and Saeed Bagheri Shouraki. Semi-supervised metric learning using pairwise constraints. In *Twenty-First International Joint Conference on Artificial Intelligence*, 2009.

[DCGO19] Sara Dahmani, Vincent Colotte, Valérian Girard, and Slim Ouni. Conditional variational auto-encoder for text-driven expressive audiovisual speech synthesis. In *INTERSPEECH 2019-20th Annual Conference of the International Speech Communication Association*, 2019. doi:10.21437/interspeech.2019-2848.

pl%	lsdim	knn acc	knn MI
0.01	2	0.3941	0.2651
	4	0.3791	0.2262
0.10	2	0.4286	0.3135
	4	0.2723	0.1519
10	2	0.4625	0.3653
	4	0.9319	0.8490

pl%	alpha	lsdim	knn acc	knn MI
0.01	0.33	2	0.0962	-0.0008
		4	0.0905	0.0005
	0.66	2	0.1007	0
		4	0.0998	0
	1	2	0.1061	-0.0006
		4	0.1088	0.0013
0.10	0.33	2	0.1064	0.0002
		4	0.1031	0
	0.66	2	0.1061	-0.0011
		4	0.1016	-0.0006
	1	2	0.0959	-0.0005
		4	0.1058	-0.0005
10	0.33	2	0.0950	-0.003
		4	0.1034	0
	0.66	2	0.1043	0
		4	0.1088	0.0007
	1	2	0.0995	-0.0013
		4	0.1055	0.0010

knn acc	knn MI
0.1016	-0.0009
0.0983	0.0005
0.1043	-0.0006
0.1004	-0.0002
0.0935	0.0017
0.1079	-0.0002
0.1079	-0.0005
0.1019	-0.0001
0.1085	0.0012
0.1049	-0.0009
0.0971	0.0007
0.0974	0.0006
0.1052	0.0005
0.0971	-0.0004
0.0965	0.0009
0.09628	-0.0001
0.1112	0
0.1073	0.0012

Fig. 4: *Table 3: Comparison of the DML model (left) and the DML with prior models with a unit gaussian prior (center) and GMM prior (right) models for the MNIST dataset.*

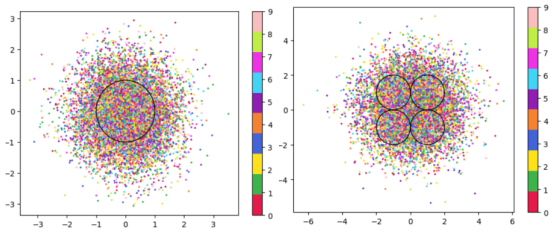


Fig. 5: *Comparison of latent spaces for DML with unit prior (left) and DML with GMM prior containing 4 components (right) for lsdim = 2 on OrganAMNIST dataset. The gaussian components are shown as black with the raidus equal to variance (1). There appears to be no evidence of the distinct gaussian components in the latent space on the right. It does appear that the unit prior may regularize the magnitude of the latent vectors*

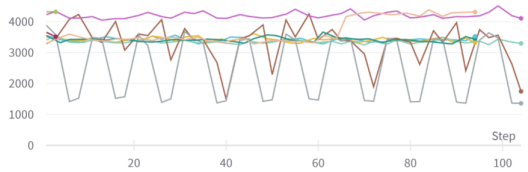


Fig. 6: *Graph of reconstruction loss (componenet of unsupervised loss) of MVAE across epochs. The unsupervised loss does not converge despite being trained on each epoch.*

Architecture Parameters		MVAE Clasif. Accuracy (%)			DML Clasif. Accuracy (%)		
lsdim	gamma	linear	rf	kNN	linear	rf	kNN
2	0	57.8	65.2	64.6	95.0	94.5	95.0
	2	**75.0**	69.8	70.6			
	5	68.8	67.9	66.9			
	10	51.6	**75.4**	**74.2**			
5	0	81.3	83.9	84.5	98.1	97.9	97.9
	2	85.9	88.5	88.4			
	5	82.8	88.0	88.5			
	10	**89.1**	**92.0**	**92.4**			
10	0	85.9	92.1	92.7	98.5	97.7	98.0
	2	87.4	91.8	91.3			
	5	**98.4**	**92.8**	**93.4**			
	10	93.8	92.5	91.9			

Fig. 7: *Table 4: Experiments performed on MVAE architecture across fully labelled MNIST dataset that trains on objective function $L = LU + \gamma * LS$ on fully supervised dataset. The best results for the classification accuracy on the MVAE embeddings in a given latent-dimensionality are bolded.*

[DHS21] Ujjal Kr Dutta, Mehrtash Harandi, and Chellu Chandra Sekhar. Semi-supervised metric learning: A deep resurrection. 2021. doi:10.48550/arXiv.2105.05061.

[EKGB16] Scott Emmons, Stephen Kobourov, Mike Gallant, and Katy Börner. Analysis of network clustering algorithms and cluster quality metrics at scale. *PloS one*, 11(7):e0159161, 2016. doi:10.1371/journal.pone.0159161.

[GTM+21] Antoine Grosnit, Rasul Tutunov, Alexandre Max Maraval, Ryan-Rhys Griffiths, Alexander I Cowen-Rivers, Lin Yang, Lin Zhu, Wenlong Lyu, Zhitang Chen, Jun Wang, et al. High-dimensional bayesian optimisation with variational autoencoders and deep metric learning. *arXiv preprint arXiv:2106.03609*, 2021. doi:10.48550/arXiv.2106.03609.

[KCJ20] Ajinkya Kulkarni, Vincent Colotte, and Denis Jouvet. Deep variational metric learning for transfer of expressivity in multi-speaker text to speech. In *International Conference on Statistical Language and Speech Processing*, pages 157–168. Springer, 2020. doi:10.1007/978-3-030-59430-5_13.

[LC10] Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010. URL: http://yann.lecun.com/exdb/mnist/ [cited 2016-01-14 14:24:11].

[LDD+18] Xudong Lin, Yueqi Duan, Qiyuan Dong, Jiwen Lu, and Jie Zhou. Deep variational metric learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 689–704, 2018. doi:10.1007/978-3-030-01267-0_42.

[LYZ+19] Xiaocui Li, Hongzhi Yin, Ke Zhou, Hongxu Chen, Shazia Sadiq, and Xiaofang Zhou. Semi-supervised clustering with deep metric learning. In *International Conference on Database Systems for Advanced Applications*, pages 383–386. Springer, 2019. doi:10.1007/978-3-030-18590-9_50.

[PVG+11] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[RHD+19] Yazhou Ren, Kangrong Hu, Xinyi Dai, Lili Pan, Steven CH Hoi, and Zenglin Xu. Semi-supervised deep embedded clustering. *Neurocomputing*, 325:121–130, 2019. doi:10.1016/j.neucom.2018.10.016.

[SKP15] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015. doi:10.1109/cvpr.2015.7298682.

[SLY15] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. *Advances in neural information processing systems*, 28:3483–3491, 2015.

[Soh16] Kihyuk Sohn. Improved deep metric learning with multi-class n-pair loss objective. In *Advances in neural information processing systems*, pages 1857–1865, 2016.

[WFZ20] Sanyou Wu, Xingdong Feng, and Fan Zhou. Metric learning by similarity network for deep semi-supervised learning. In *Developments of Artificial Intelligence Technologies in Computation and Robotics: Proceedings of the 14th International FLINS Conference (FLINS 2020)*, pages 995–1002. World Scientific, 2020. doi:10.1142/9789811223334_0120.

[WYF13] Qianying Wang, Pong C Yuen, and Guocan Feng. Semi-supervised metric learning via topology preserving multiple semi-supervised assumptions. *Pattern Recognition*, 46(9):2576–2587, 2013. doi:10.1016/j.patcog.2013.02.015.

[YSW+21] Jiancheng Yang, Rui Shi, Donglai Wei, Zequan Liu, Lin Zhao, Bilian Ke, Hanspeter Pfister, and Bingbing Ni. Medmnist v2: A large-scale lightweight benchmark for 2d and 3d biomedical image classification. *arXiv preprint arXiv:2110.14795*, 2021. doi:10.48550/arXiv.2110.14795.

[ZG21] Zhen Zhu and Yuan Gao. Finding cross-border collaborative centres in biopharma patent networks: A clustering comparison approach based on adjusted mutual information. In *International Conference on Complex Networks and Their Applications*, pages 62–72. Springer, 2021. doi:10.1007/978-3-030-93409-5_6.

[ZRS+20] Meekail Zain, Sonia Rao, Nathan Safir, Quinn Wyner, Isabella Humphrey, Alexa Eldridge, Chenxiao Li, BahaaEddin AlAila, and Shannon P. Quinn. Towards an unsupervised spatiotemporal representation of cilia video using a modular generative pipeline. 2020. doi:10.25080/majora-342d178e-017.