INCORPORATING TASK-AGNOSTIC INFORMATION IN TASK-BASED ACTIVE LEARNING USING A VARIATIONAL AUTOENCODER

by

CURTIS GODWIN

(Under the Direction of Shannon Quinn)

ABSTRACT

When building datasets for machine learning, it is often much easier and less expensive to collect data than to label it, resulting in large pools of unlabeled data. Active learning (AL) is a subfield of machine learning that focuses on choosing which unlabeled points to label next for a given dataset and task, with the core assumption that labeling certain points will result in higher performance models than other points. Standard AL approaches identify informative samples by querying a trained task model. Task-agnostic AL approaches ignore the task model and instead makes selections based on separately defined properties of the dataset. We seek to combine these approaches and measure the contribution of incorporating task-agnostic information into task-focused AL. We use a ResNet classifier as our task model and experiment across two AL utility functions with and without added information from a variational autoencoder (VAE).

INDEX WORDS: Active Learning, Deep Learning, Semi-Supervised Learning, Variational Autoencoder

INCORPORATING TASK-AGNOSTIC INFORMATION IN TASK-BASED ACTIVE LEARNING USING A VARIATIONAL AUTOENCODER

by

CURTIS GODWIN

B.S., University of Georgia, 2020

A Thesis Submitted to the Graduate Faculty of the
University of Georgia in Partial Fulfillment of the Requirements for the Degree.

MASTER OF SCIENCE

ATHENS, GEORGIA

2022

INCORPORATING TASK-AGNOSTIC INFORMATION IN TASK-BASED ACTIVE LEARNING USING A VARIATIONAL AUTOENCODER

by

CURTIS GODWIN

Major Professor: Shannon Quinn

Committee: Sheng Li
Tianming Liu

Electronic Version Approved:

Ron Walcott
Dean of the Graduate School
The University of Georgia
December 2022

ACKNOWLEDGMENTS

I want to thank Dr. Quinn and the cilia team for all the resources and support throughout the writing process. I am grateful to have found a research group with such an encouraging atmosphere and such understanding people. This would also not have been possible without the support of my family, who were always there for me.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

For many machine learning problems, labeling data is time consuming and expensive. Data is often gathered at a faster rate than it can be labeled, resulting in large pools of unlabeled data. The field of active learning (AL) provides methods for choosing which unlabeled points are best to label next.

In AL, the utility of unlabeled points is typically defined with respect to a task model that is trained on the labeled set. The points of highest utility are those which, if labeled and trained on, would result in the greatest reduction of the task model's average loss. Since this is a determination that cannot be made until a given point is actually labeled, various heuristics for informativeness have been proposed for comparing unlabeled points in this context.

The heuristics can be broadly divided into two approaches: task-focused and task-agnostic. Task-focused approaches have access to the task model, including its weights and outputs [1, 2, 3]. Task-agnostic approaches select points based on separate observations of the dataset [4, 5], only using the task model when measuring performance.

We propose a novel methodology for incorporating task-agnostic information into task-focused approaches. We start by training a variational autoencoder (VAE) to learn features for representing the dataset separately from the task model. We then incorporate these features into two relevant task-focused methods from the literature, in each case entangling the task model information with the additional task-agnostic VAE features.

CHAPTER 2

RELATED LITERATURE

§ 2.1 Active learning

In active learning (AL), we consider a partially labeled dataset $\mathcal{D} \subset \mathbb{R}^n$. We have a labeled subset $L \subset \mathcal{D}$ and an unlabeled subset $U \subset \mathcal{D}$ such that $L \cup U = \mathcal{D}$ and $L \cap U = \varnothing$. We also consider a supervised task model

$$\mathcal{T} : \mathbb{R}^n \to \{0, 1, ..., c\},$$

where c is the number of classes in $\mathcal{D}$.[1] AL is comprised of heuristics for selecting high priority unlabeled points $u \in U$ to be labeled.

Algorithm 1: Measuring the performance of a given active learning heuristic

Input: training dataset D, task model $\mathcal{T}$, budget β, initial number of labels ξ, desired number of labels η, set selection heuristic $\mathcal{H}$

$L \leftarrow \xi$-sized random subset of D
$U \leftarrow D \backslash L$
$A \leftarrow \varnothing$
train $\mathcal{T}$ on L
while $|L| \leq \eta$ **do**
  $S \leftarrow \beta$-sized subset of U, selected using $\mathcal{H}$
  $L \leftarrow L \cup S$
  retrain or fine-tune $\mathcal{T}$ on L
  $a \leftarrow$ (validation accuracy of $\mathcal{T}$, $|L|$) `// save accuracy tuple`
  $A \leftarrow A \cup a$ `// record accuracies across the labeling process`
end
create a line graph plotting a_0 against a_1 for each $a \in A$

[1] We limit our analysis to classifiers as they are the most common type of task model in the AL literature.

Much of the early active learning (AL) literature is based on shallower, less computationally demanding networks since deeper architectures were not well-developed at the time. Settles [6] provides a review of these early methods, with the main branches being membership query synthesis [7], stream-based sampling [8], and pool-based sampling [9]. The latter method takes a holistic approach of ranking all available unlabeled points by some chosen heuristic $\mathcal{H}$ and choosing to label the points of highest ranking. This is the current default AL approach, as technological advancements have made it a less demanding task in terms of processing and memory.

The popularity of the pool-based method has led to a widely-used evaluation procedure, which we describe in Algorithm 1. This procedure trains a task model $\mathcal{T}$ on the initial labeled data, records its test accuracy, then uses $\mathcal{H}$ to label a set of unlabeled points. We then once again train $\mathcal{T}$ on the labeled data and record its accuracy. This is repeated until a desired number of labels is reached, and then the accuracies can be graphed against the number of available labels to demonstrate performance over the course of labeling. We can separately pass multiple heuristics through this evaluation algorithm to compare their performance based on the resulting accuracy graphs. This is utilized in many AL papers to show the efficacy of their methods in comparison to others [1, 2, 3, 10].

The prevailing approach to the pool-based method has been to choose unlabeled points for which the model is most uncertain, the assumption being that uncertain points will be the most informative [11]. A popular early method was to label the unlabeled points of highest Shannon entropy [12] under the task model, which is a measure of uncertainty between the classes of the data. This method is now more commonly used in combination with a representativeness measure [10] to encourage that very similar samples are not successively selected.

§ 2.1.1 Recent heuristics using deep features

For convolutional neural networks (CNNs) in image classification settings, the task model $\mathcal{T}$ can be decomposed into a feature-generating module

$$\mathcal{T}_f \colon \mathbb{R}^n \to \mathbb{R}^f,$$

which maps the input data vectors to the output of the final fully connected layer before classification, and a classification module

$$\mathcal{T}_c \colon \mathbb{R}^f \rightarrow \{0, 1, ..., c\},$$

where c is the number of classes.

(1) Core-set and MedAL

Recent deep learning-based AL methods have approached the notion of uncertainty in terms of the rich features generated by the learned model $\mathcal{T}$. Core-set [1] and MedAL [2] select unlabeled points that are the furthest from the labeled set in terms of L_2 distance between the learned features. For core-set, each point constructing the set S in step 6 of Algorithm 1 is chosen by

$$\mathbf{u}^* = \arg\max_{\mathbf{u} \in U} \min_{\boldsymbol{\ell} \in L} ||(\mathcal{T}_f(\mathbf{u}) - \mathcal{T}_f(\boldsymbol{\ell}))||^2,$$

where U is the unlabeled set and L is the labeled set. The analogous operation for MedAL is

$$\mathbf{u}^* = \arg\max_{\mathbf{u} \in U} \frac{1}{|L|} \sum_{i=1}^{|L|} ||\mathcal{T}_f(\mathbf{u}) - \mathcal{T}_f(\mathbf{L_i})||^2.$$

Note that after a point $\mathbf{u}^*$ is chosen, the selection of the next point assumes the previous $\mathbf{u}^*$ to be in the labeled set. This way we discourage choosing sets that are closely packed together, leading to sets that are more diverse in terms of their features. This effect is more pronounced in the core-set method since it takes the minimum distance whereas MedAL uses the average distance.

(2) Loss prediction

Another recent method [3] trains a regression network to predict the loss of the task model, then takes the heuristic $\mathcal{H}$ to select the unlabeled points of highest predicted loss. To implement this, the loss prediction network $\mathcal{P}$ is attached to a ResNet task model $\mathcal{T}$ and is trained jointly with $\mathcal{T}$. The inputs to $\mathcal{P}$ are the features output by the ResNet's four residual blocks. These features are mapped into the same dimensionality via a fully connected layer and then concatenated to form a representation $\mathbf{c}$. An additional fully connected layer then maps $\mathbf{c}$ into a single value constituting the loss prediction:

$$\mathbf{u}^* = \text{argmax}_{\mathbf{u} \in U} \mathcal{P}(u).$$

When attempting to train a network to directly predict $\mathcal{T}$'s loss during training, the ground truth losses naturally decrease as $\mathcal{T}$ is optimized, resulting in a moving objective. The authors of [3] find that a more stable ground truth is the inequality between the losses of given pairs of points. In this case, $\mathcal{P}$ is trained on pairs of labeled points, so that $\mathcal{P}$ is penalized for producing predicted loss pairs that exhibit a different inequality than the corresponding true loss pair.

More specifically, for each batch of labeled data $L_{batch} \subset L$ that is propagated through $\mathcal{T}$ during training, the batch of true losses is computed and split randomly into a batch of pairs P_{batch}. The loss prediction network produces a corresponding batch of predicted loss pairs, denoted $\widetilde{P}_{batch}$. The following pair loss is then computed given each $p \in P_{batch}$ and its corresponding $\tilde{p} \in \widetilde{P}_{batch}$:

$$\mathcal{L}_{pair}(p, \tilde{p}) = \max(0, -\mathcal{I}(p) \cdot (\tilde{p}^{(1)} - \tilde{p}^{(2)}) + \xi),$$

where $\mathcal{I}$ is the following indicator function for pair inequality:

$$\mathcal{I}(p) = \begin{cases} 1, & p^{(1)} > p^{(2)} \\ -1, & p^{(1)} \leq p^{(2)} \end{cases}.$$

§ 2.2 Variational Autoencoders

Variational autoencoders (VAEs) [13] are an unsupervised method for modeling data using Bayesian posterior inference. We begin with the Bayesian assumption that the data is well-modeled by some distribution, commonly a multivariate Gaussian. We also assume that this data distribution can be inferred reasonably well by a lower dimensional random variable, also modeled by a multivariate Gaussian.

The inference process then consists of an encoding into the lower dimensional latent variable, followed by a decoding back into the data dimension. We parametrize both the encoder and the decoder as neural networks, jointly optimizing their parameters with the following loss function [14]:

$$\mathcal{L}_{\theta,\phi}(\mathbf{x}) = \log p_\theta(\mathbf{x}|\mathbf{z}) + [\log p_\theta(\mathbf{z}) - \log q_\phi(\mathbf{z}|\mathbf{x})],$$

where θ and ϕ are the parameters of the encoder and the decoder, respectively. The first term is the reconstruction error, penalizing the parameters for producing poor reconstructions of the input data. The second term is the regularization error, encouraging the encoding to resemble a pre-selected prior distribution, commonly a unit Gaussian prior.

The encoder of a well-optimized VAE can be used to generate latent encodings with rich features which are sufficient to approximately reconstruct the data. The features also have some geometric consistency, in the sense that the encoder is encouraged to generate encodings in the pattern of a Gaussian distribution.

§ 2.3 t-SNE Dimensionality Reduction and Visualization

Many methods have been developed for visualizing high-dimensional data, which can be informative in machine learning where many important objects are high dimensional and difficult to analyze. In our research, we utilize the popular dimensionality reduction technique t-SNE.

The general strategy in t-SNE is to take a given high dimensional set $X \subset \mathbb{R}^n$, randomly initialize a corresponding lower dimensional set $Y \subset \mathbb{R}^l$ of the same cardinality, and iteratively update Y to have a similar data distribution as X in terms of conditional probability. Recall the multivariate Gaussian distribution function

$$f(\mathbf{x}) = \frac{1}{\sqrt{(2\pi)^n |\Sigma|}} \exp(-\frac{1}{2}(\mathbf{x}-\mu)^T \Sigma^{-1}(\mathbf{x}-\mu))$$

which we use to model both distributions. The authors note that the conditional probability $p_{j|i}$ between given points $\mathbf{x}_i$ and $\mathbf{x}_j$ can be specified in this context as

$$p_{j|i} = \frac{\exp(-||\mathbf{x}_i - \mathbf{x}_j||^2/2\sigma_i^2)}{\sum_{k \neq i} \exp(-||\mathbf{x}_i - \mathbf{x}_k||^2/2\sigma_i^2)}$$

where σ is a variance hyperparameter.

We begin by initializing each $\mathbf{y} \in Y$ as randomly sampled Gaussian noise in the desired dimensionality. The conditional probability $q_{j|i}$ in the lower dimensionality is similar to $p_{j|i}$, except that the authors keep the variance constant at $\frac{1}{\sqrt{2}}$ for simplicity:

$$q_{j|i} = \frac{\exp(-||\mathbf{y}_i - \mathbf{y}_j||^2)}{\sum_{k \neq i} \exp(-||\mathbf{y}_i - \mathbf{y}_k||^2)}$$

If $p_{j|i}$ and $q_{j|i}$ are equal, then Y has a similar distribution as X. Thus we can modify Y to more closely resemble X by encouraging $p_{j|i}$ and $q_{j|i}$ to be closer. To achieve this, we define a cost function C as the KL divergence between $p_{j|i}$ and $q_{j|i}$:

$$C = \mathbf{KL}(P||Q) = \sum_i \sum_j p_{i|j} \log \frac{p_{i|j}}{q_{i|j}}.$$

We can then calculate the gradient of C with respect to $y_i \in Y$:

$$\frac{\delta C}{\delta y_i} = 2 \sum_j (p_{j|i} - q_{j|i} + p_{i|j} - q_{i|j})(\mathbf{y_i} - \mathbf{y_j}).$$

We can now specify the update step as

$$\mathcal{Y}^{(t)} = \mathcal{Y}^{(t-1)} + \lambda \frac{\delta C}{\delta y_i} + \alpha(t)(\mathcal{Y}^{(t-1)} - \mathcal{Y}^{(t-2)}),$$

where λ and $\alpha(t)$ are hyperparameters corresponding to learning rate and momentum, respectively.

With the update step, we can now run t-SNE to generate a lower dimensional representation of our data with given number of components, learning rate, and additional hyperparameters on the training process. For our purposes, we use the default hyperparameters in the `scikit-learn` Python package.

CHAPTER 3

METHODOLOGY

We observe that both core-set and loss prediction utilize reasoning about a representation in a vector space. In particular, the core-set method relies on distances between feature vectors modeled by the task model $\mathcal{T}$. Loss prediction relies on a fully connected layer mapping from a feature space to a single value, producing different predictions depending on the values of the relevant feature vector. Both core-set and loss prediction then produce evaluation heuristics $\mathcal{H}$ indicating the uncertainty of all unlabeled points. The most uncertain points are then assumed to be the most effective points to label next.

In each of these methods, the final heuristic $\mathcal{H}$ is derived from the task model, which is trained only on the labeled points at a given timestep in the labeling procedure. To address this limitation, we suggest that these methods may be improved by incorporating information learned over the entire dataset. For this purpose we employ a variational autoencoder (VAE), which can be trained without labels and which can produce useful representations of image datasets. These additional unsupervised features will constitute a new perspective on the data, which may improve the active learning process.

We implement this by first training a VAE model $\mathcal{V}$ on the given dataset. $\mathcal{V}$ can then be used as a function returning the VAE features for any given datapoint. We append these additional features to the relevant vector spaces using vector concatenation, an operation we denote with the symbol $\frown$. The modified point selection operation in core-set then becomes

$$\mathbf{u}^* = \arg\max_{\mathbf{u} \in U} \min_{\boldsymbol{\ell} \in L} ||[(\mathcal{T}_f(\mathbf{u}) \frown \alpha\mathcal{V}(\mathbf{u})] - [\mathcal{T}_f(\boldsymbol{\ell}) \frown \alpha\mathcal{V}(\boldsymbol{\ell})])||^2,$$

where α is a hyperparameter that scales the influence of the VAE features in computing the vector distance. To similarly modify the loss prediction method, we concatenate the VAE features to the final ResNet feature concatenation $\mathbf{c}$ before the loss prediction, so that the extra information is factored into the training of the prediction network $\mathcal{P}$.

CHAPTER 4

EXPERIMENTS

In order to measure the efficacy of the newly proposed methods, we generate accuracy graphs using Algorithm 1, freezing all settings except the selection heuristic $\mathcal{H}$. We then compare the performance of the core-set and loss prediction heuristics with their VAE-augmented counterparts.

We use ResNet-18 pretrained on ImageNet as the task model, using the SGD optimizer with learning rate 0.001 and momentum 0.9. We train on the MNIST [15] and ChestMNIST [16] datasets. ChestMNIST consists of 112,120 chest X-ray images resized to 28x28 and is one of several benchmark medical image datasets introduced in [16].

For both datasets we experiment on randomly selected subsets, using 25000 points for MNIST and 30000 points for ChestMNIST. In both cases we begin with 3000 initial labels and label 3000 points per active learning step. We opt to retrain the task model after each labeling step instead of fine-tuning.

We use a similar training strategy as in [2], training the task model until >99% train accuracy before selecting new points to label. This ensures that the ResNet is similarly well fit to the labeled data at each labeling iteration. This is implemented by training for 10 epochs on the initial training set and increasing the training epochs by 5 after each labeling iteration.

The VAEs used for the experiments are trained for 20 epochs using an Adam optimizer with learning rate 0.001 and weight decay 0.005. The VAE encoder architecture consists of four convolutional downsampling filters and two linear layers to learn the low dimensional mean and log variance. The decoder consists of an upsampling convolution and four size-preserving convolutions to learn the reconstruction.

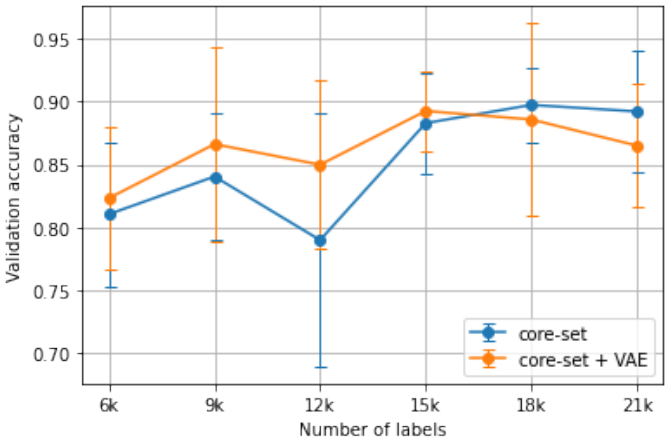

Figure 4.1: Average results of Algorithm 1 on MNIST over 5 runs using the core-set heuristic versus the VAE-augmented core-set heuristic.

Experiments were run five times, each with a separate set of randomly chosen initial labels, with the displayed results showing the average validation accuracies across all runs. Figures 4.1 and 4.3 show the core-set results, while Figures 4.2 and 4.4 show the loss prediction results. Within each 5-run batch, shared random seeds were used to ensure that the task models being compared were supplied with the same initial set of labels.

With four NVIDIA 2080 GPUs, the total runtime for the MNIST experiments was 5113s for core-set and 4955s for loss prediction; for ChestMNIST, the total runtime was 7085s for core-set and 7209s for loss prediction.

To investigate the qualitative difference between the VAE and non-VAE approaches, we performed an additional experiment to visualize an example of core-set selection. We first train the ResNet-18 with the same hyperparameter settings on 1000 initial labels from the ChestMNIST dataset, then randomly choose 1556 (5%) of the unlabeled points from which to select 100 points to label. These smaller sizes were chosen to promote visual clarity in the output graphs. We then use t-SNE [17] dimensionality reduction to compare the ResNet features of the labeled set, the unlabeled set, and the points chosen to be labeled by core-set. The results are displayed in Figures 4.5 and 4.6.

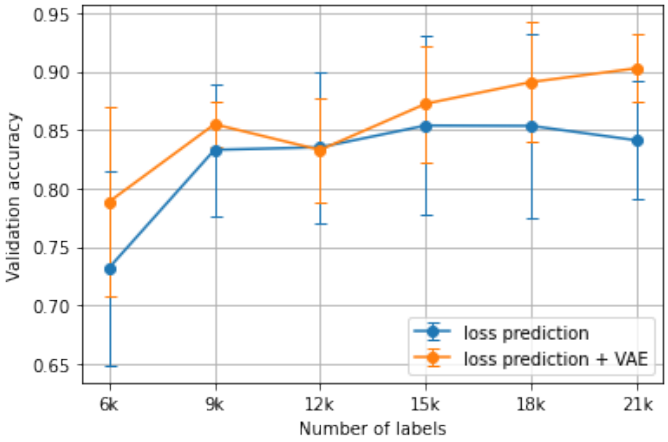

Figure 4.2: Average results of Algorithm 1 on MNIST over 5 runs using the loss prediction heuristic versus the VAE-augmented loss prediction heuristic.

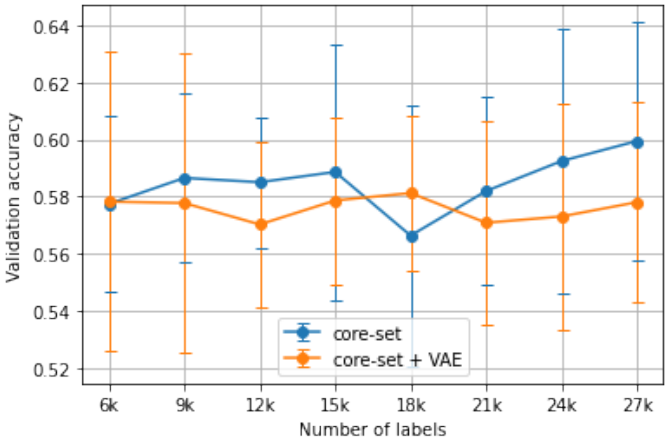

Figure 4.3: Average results of Algorithm 1 on ChestMNIST over 5 runs using the core-set heuristic versus the VAE-augmented core-set heuristic.

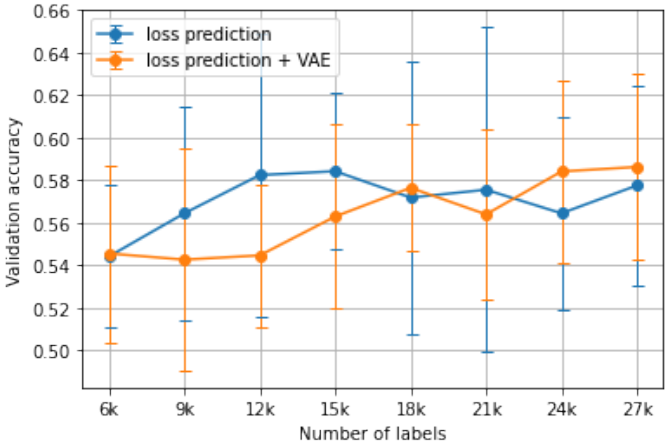

Figure 4.4: Average results of Algorithm 1 on ChestMNIST over 5 runs using the loss prediction heuristic versus the VAE-augmented loss prediction heuristic.

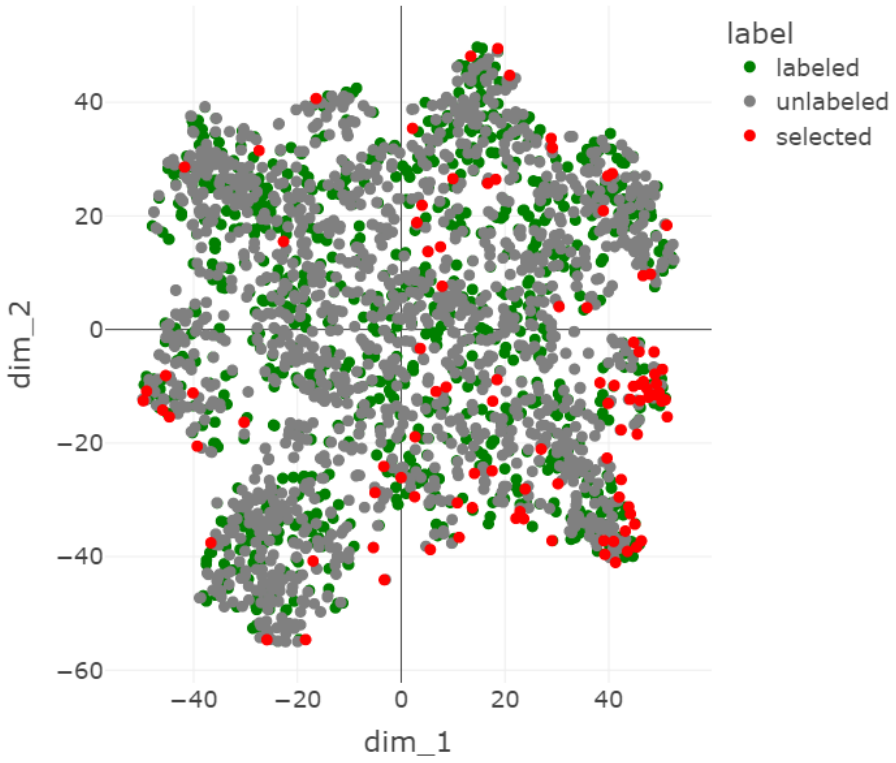

Figure 4.5: A t-SNE visualization of the ChestMNIST points chosen by core-set.

Overall, the VAE-augmented active learning heuristics did not exhibit a significant performance difference when compared with their counterparts. The only case of a significant p-value (<0.05) occurred during loss prediction on the MNIST dataset at 21000 labels.

The t-SNE visualizations in Figures 4.5 and 4.6 show some of the influence that the VAE features have on the core-set selection process. In 4.5, the selected points tend to be more spread out, while in 4.6 they cluster at one edge. This appears to mirror the transformation of the rest of the data, which is more spread out without the VAE features, but becomes condensed in the center when they are introduced, approaching the shape of a Gaussian distribution.

It seems that with the added VAE features, the selected points are further out of distribution in the latent space. This makes sense because points tend to be more sparse at the tails of a Guassian distribution and core-set prioritizes points that are well-isolated from other points.

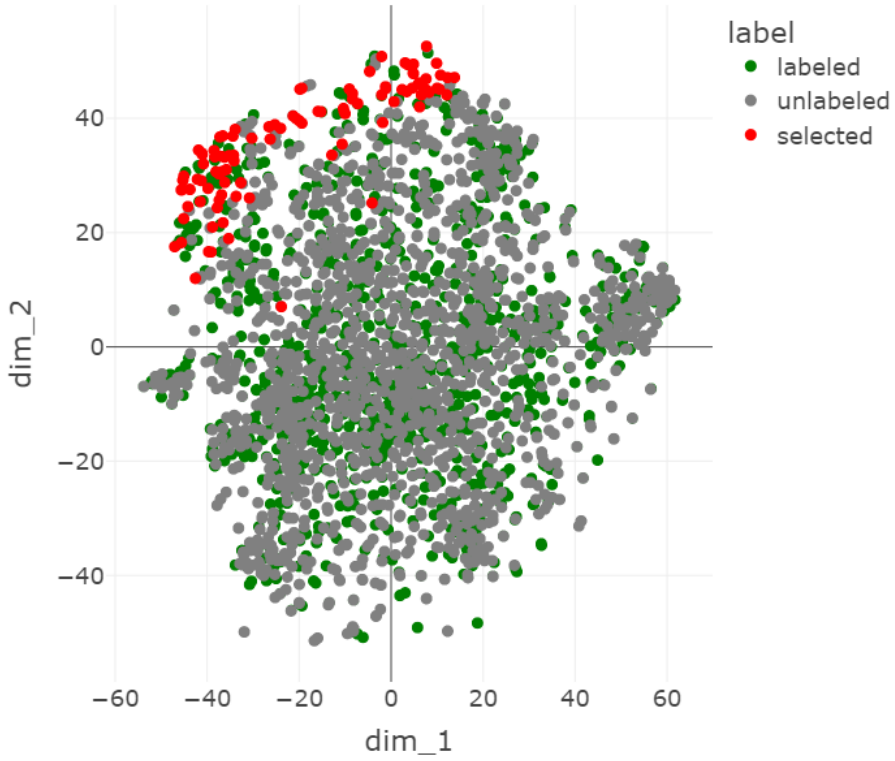

Figure 4.6: A t-SNE visualization of the ChestMNIST points chosen by core-set when the ResNet features are augmented with VAE features.

One reason for the lack of performance improvement may be the homogeneous nature of the VAE, where the optimization goal is reconstruction rather than classification. Thus points may be chosen by core-set on the basis of their ability to reconstruct, which does constitute a new perspective on their uncertainty, but one which is not necessarily relevant to classification. This may be improved by using a multimodal prior in the VAE, which may do a better job of modeling relevant differences between points.

CHAPTER 5

CONCLUSION

Our original intuition was that additional unsupervised information may improve established active learning methods, especially when using a modern unsupervised representation method such as a VAE. The experimental results did not indicate this hypothesis, but additional investigation of the VAE features showed a notable change in the task model latent space. Though this did not result in superior point selections in our case, it is of interest whether different approaches to latent space augmentation in active learning may fare better.

Future work may explore the use of class-conditional VAEs in a similar application, since a VAE that can utilize the available class labels may produce more relevant representations, and it could be retrained along with the task model after each labeling iteration.

BIBLIOGRAPHY

[1] Ozan Sener and Silvio Savarese. "Active Learning for Convolutional Neural Networks: A Core-Set Approach". In: *International Conference on Learning Representations*. 2018.

[2] Asim Smailagic et al. "Medal: Accurate and robust deep active learning for medical image analysis". In: *2018 17th IEEE international conference on machine learning and applications (ICMLA)*. IEEE. 2018, pp. 481–488.

[3] Donggeun Yoo and In So Kweon. "Learning loss for active learning". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019, pp. 93–102.

[4] Samarth Sinha, Sayna Ebrahimi, and Trevor Darrell. "Variational adversarial active learning". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 5972–5981.

[5] Changsheng Li et al. "On deep unsupervised active learning". In: *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*. 2021, pp. 2626–2632.

[6] Burr Settles. "Active learning literature survey". In: (2009).

[7] Dana Angluin. "Queries and concept learning". In: *Machine learning* 2.4 (1988), pp. 319–342.

[8] Les Atlas, David Cohn, and Richard Ladner. "Training connectionist networks with queries and selective sampling". In: *Advances in neural information processing systems* 2 (1989).

[9] David D Lewis and William A Gale. "A sequential algorithm for training text classifiers". In: *SIGIR'94*. Springer. 1994, pp. 3–12.

[10] Keze Wang et al. "Cost-effective active learning for deep image classification". In: *IEEE Transactions on Circuits and Systems for Video Technology* 27.12 (2016), pp. 2591–2600.

[11] Samuel Budd, Emma C Robinson, and Bernhard Kainz. "A survey on active learning and human-in-the-loop deep learning for medical image analysis". In: *Medical Image Analysis* 71 (2021), p. 102062.

[12] Claude Elwood Shannon. "A mathematical theory of communication". In: *The Bell system technical journal* 27.3 (1948), pp. 379–423.

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

[14] Diederik P Kingma and Max Welling. "An introduction to variational autoencoders". In: *arXiv preprint arXiv:1906.02691* (2019).

[15] Li Deng. "The mnist database of handwritten digit images for machine learning research". In: *IEEE Signal Processing Magazine* 29.6 (2012), pp. 141–142. DOI: 10.1109/MSP.2012.2211477.

[16] Jiancheng Yang, Rui Shi, and Bingbing Ni. "MedMNIST Classification Decathlon: A Lightweight AutoML Benchmark for Medical Image Analysis". In: *2021 IEEE 18th International Symposium on Biomedical Imaging (ISBI)*. 2021, pp. 191–195. DOI: 10.1109/ISBI48211.2021.9434062.

[17] Laurens Van der Maaten and Geoffrey Hinton. "Visualizing data using t-SNE." In: *Journal of machine learning research* 9.11 (2008).