This work was carried out under the supervision of Professor Yoram Louzoun. Through the Department of Computer Science, Bar-Ilan University

Acknowledgments

First, I would like to thank my family for their help and infinite support during my studies. Mummy and Daddy, thanks for always having my back. Noam, thanks for the ongoing support, I could've not done it without you. I would also like to thank my supervisor, Prof. Yoram Louzoun, for his attention, support, patience, guidance and availability during the research especially during this trying year. I would like to express my gratitude to the Department of Computer Science for providing the resources necessary to complete this research. Finally, I would like to thank my colleagues for the lab, past and present, for the support and for many interesting discussions.

Contents

Abstract

Multiple Instance Learning (MIL) methods are typically supervised. However, a bag-to-bag metric is needed in many applications, including clustering, statistical tests, and dimension reduction. Such a metric should differentiate between bags, regardless of the sparsity or overlap between the instances of the bags. We propose SUMIT (Self sUpervised MIL dIsTance) as an instance-embedding-based distance that maximizes the distinction between bags. SUMIT is optimized using five criteria: self-similarity within a bag, quality of instance reconstruction, robustness to sampling depth, conservation of triangle inequality, and separation of instances to clusters. We show using current standard MIL datasets and a novel wiki-based set of wiki topics that the within bag-similarity loss is the most important for a bag-to-bag metric that best separates bags of similar classes. SUMIT bridges the gap between instance-level and bag-level approaches, by keeping the embedding of all instances but ensuring their proximity within a bag.

1 Introduction

Multiple Instance Learning (MIL) is weakly supervised learning where the training instances are not stand-alone examples, but instead are arranged in sets, called bags[2]. Labels are provided for the entire bag and not for the individual samples, further denoted as instances. To exemplify MIL, consider chapters in a book. The sentences can be considered as the individual instance, and the chapter is a bag. In the training set, we know that chapters belong to a category. The goal of MIL classification is to detect whether a chapter in the book belongs to a category. While a large number of algorithms were developed for MIL classification, there is no inherent method to define distances between bags. Note that many methods were proposed to define distances on sets (e.g. the energy distance[21], Wasserstein Distance[22] Hausdorff distance[28], etc). However, these distances do not consider the distribution of instances in each bag. We propose to combine self-supervised learning and metric learning to produce such a metric between bags. Using the book chapters example above- when considering the chapters as the bags, one would expect the distances between chapters in the same book to be smaller than the distance between chapters from other books. Also, the distances between chapters from books of the same genre or subject should be smaller than the distances between chapters from other genres or subjects.

MIL classification methods can be grouped into three categories: instance-space (IS), bag-space (BS), and embedded-space (ES) methods[31]. The categorization is based on how information from the bag is extracted. The Instance-Space (IS) paradigm is based on local, instance-level information, ignoring the characteristics of the whole bag. This is based on the assumption that the information lies in the instances. The Bag-Space (BS) paradigm is based on global, bag-level information, and each bag is treated as an entity. The Embedded-Space (ES) paradigm, like the BS paradigm, is based on global, bag-information. In contrast with the BS paradigm, which uses a non-vectorial representation of the bag, in the ES paradigm, each bag is mapped to a single feature vector.

We here propose a combination of the instance and bag level, by computing a distance on an embedding of each instance, requiring that the distance between instances of the same bag to be closer than instances between bags. We do not assume any label on the bag. As such, the metric presented here is an instance-level self-supervised distance at the bag level. This approach combines the advantages of instance and bag-level approaches.

We deviate from the mainly supervised approach used in MIL. A well-defined metric in the MIL setting has multiple benefits, including, distance-based machine learning (e.g. KNN[11]), the definition of density in the bag space using kernel density estimation - KDE[7], clustering using distance-based clustering (e.g. community detection[17] or dBscan [8], dimension reduction using distance conserving projection (e.g. MDS [16],UMAP [18], t-SNE[26] and the resulting visualization when projected to two or three dimensions, and distance based statistical tests[3].

2 Related work

While no method was explicitly developed for a self-supervised MIL metric definition, some supervised and unsupervised methods provide a distance metric between bags. In general, one can produce a distance from all ES and BS representation methods.

1. **ES (Embedding Space)** methods in MIL explicitly map entire bags into continuously embeddings[13, 30]. Instance embeddings can be combined using pooling[10] or attention[27] to compute bag-level metrics. The Euclidean (or any other) distance between the embedding of the bags is a metric.

2. **BS (Bag Similarity)** [28, 9, 34, 6, 24, 33, 12, 29] methods indirectly derive distances between bags by first computing distances or similarities between instances within bags, often using kernel functions or distance metrics like the Hausdorff distance.

3. **IS (Instance Space)** methods consider instances individually and aggregate their predictions or features to infer bag-level decisions. There are no bag-to-bag good Instance-based metrics. In the last decade, there has been much less interest in instance-based methods, since those suffer from the aggregation problem - how to combine the instances to represent a bag. Each combination method has limitations. Using the mean can lead to instances canceling one another, and using

each instance creates different size representations for each bag (note that the same problem may emerge in ES).

Other methods applied existing distances to specific problems (Appendix A). The main works that explicitly produced distances are:

1. Citation-KNN[28] proposed a modified Hausdorff Distance-based MIL approach. The Hausdorff distance is utilized to provide a metric function between subsets of a metric space. Specifically, two bags X,Y are in distance d if every instance of bag X is within distance d of at least one of the instances in bag Y:

$$d_{Hausdorff}(X_i, X_j) = max(h(X_i, X_j), h(X_j, X_i)); h(X_i, X_j) = \max_{x_i \in X_i} \min_{x_j \in X_j} ||x_i - x_j|| \quad (1)$$

2. Kernel Methods[9] define a kernel function as the sum of instance kernels:

$$d_{Kernel}(X_i, X_j) = K(X_i, X_j) := \sum_{x_i \in X_i, x_j \in X_j} k(x_i, x_j) \quad (2)$$

3. Graph Kernel[34] transforms the bags into undirected graphs implicitly or explicitly, and applies standard distance metrics of graphs such as edit-distance[23] and inexact matching distance[4]:

$$d_{\text{Graph}}(X_i, X_j) = D(G_{X_i}, G_{X_j}) \quad (3)$$

   ,where G_{X_i} is the graph produced from X_i.

4. MInd[6] represents each bag by a vector of its dissimilarities to other bags (e.g. Hausdorff distance) in the training set and treats these dissimilarities as a feature representation:

$$d_{MInd}(X_i, X_j) = d\left(F_{MInd}(X_i), F_{MInd}(X_j)\right), \quad F_{MInd}(X_i) = \left[d\left(B_i, B_1\right), \ldots, d\left(B_i, B_M\right)\right] \quad (4)$$

   where the bag B_i is compared to each of $B_j, 1 <= j <= M$ the M bags of a training set.

5. Contrastive Multiple Instance Learning[24]: Method initially trains a tile-wise encoder using SimCLR[5], from which subsets of tile-wise (instances) embeddings are extracted and fused via an attention-based multiple-instance learning framework to yield slide-level (bag) representations. The resulting set of intra-slide-level and inter-slide-level embeddings are attracted and repelled via contrastive loss, respectively.

6. Multi-instance clustering with applications to multi-instance prediction[33]: Each bag is represented by a k-dimensional feature vector, where the value of the i-th feature is the distance between the bag and the medoid (the center). The medoid is set in the minimum distance $medoid = \arg\min_{X_i \in G} \sum_{X_j \in G} d(X_i, X_j)$ to all bags in group G, where d is the distance to the medoid) of the i-th group. The distance between bags can be Min, Max, or Hausdorff of the instances.

3 Novelty

All current metrics use either a projection of the entire bag and as such lose most of the information on the bag, or use the original representation of each instance combined with some predefined distance between sets. This suffers from a high overlap between bags.

Here we propose a novel approach - use an energy distance via the embedding of each instance. The embedding of each instance is learned to ensure that the distance between instances of different bags is larger than the distance of instances in a bag. In addition, we optimize the embedding of each instance to ensure that the distance between bags adheres to the requirements of a distance metric (e.g. the triangle inequality.

Specifically, we do not optimize the distances between bags. Instead, we optimize the embedding of each instance to minimize five different losses:

1. **Reconstruction Loss.** We minimize the discrepancy between the original and the reconstructed features following an encoder-decoder model.
2. **Contrastive Loss** We minimize the distance between instances of the same bag compared to distances from different bags.
3. **Invariance Loss.** We ensure the distance is not sensitive to the number of instances sampled in each bag.
4. **Clustering.** We optimize the division of the encoding into distinct clusters.
5. **Triangle Loss.** We maximize the gap between the undirect distance from one instance to another through a third instance and the direct distance between said instances (adherence to the triangular inequality theorem).

To the best of our knowledge, this is the first time the embedding of each instance is optimized to produce an optimal distance between bags.

4 Methods

4.1 Data

We used two types of datasets1: benchmark MIL datasets[19][1][25] and a self-curated dataset from Wikipedia pages (Appendix C.1 for the list of topics). The wiki data is available in the github.

In the benchmark datasets, the MUSK1 and MUSK2 datasets represent molecules identified as musks or non-musks by human experts. The features of each instance represent the shape of the molecules, and the different instances are the different conformations of the molecules. The Fox, Tiger, and Elephant datasets are subsets of the Corel image retrieval dataset, separated into images presenting (or not) the associated animal. Each image is a bag, and the instances are created by dividing the image into segments, where each segment's features represent the segment's color, texture, and shape. If any image segment includes the animal, the bag is labeled positive. The Wiki dataset introduced here is a novel dataset developed for the current analysis. The bags are Wikipedia pages of cities in different countries, such that each country is a label of the city. The instances are the Bag Of Words (BOW) of each section on the page.

Data Sets				
Name	No. Features	No. Instances	No. Bags	Data Type
MUSK1	166	476	92	Tabular
MUSK2	166	6598	102	Tabular
FOX	230	1320	200	Tabular
TIGER	230	1220	200	Tabular
ELEPHANT	230	1391	200	Tabular
Wikipedia	1205	19568	1504	Text Sequences

Table 1: Data sets statistics- the number of different features (the instance dimension), the total number of instances (samples), and bags, and the type of data.

4.2 Model and experimental setup

We used an encoder-decoder model, which was trained using the losses below. Let $x \in \mathbb{R}^{d_{\text{input}}}$ represent the input data, where d_{input} is the input instances dimension. We used an encoder-decoder model with layers: $[d_{\text{input}}, 128, 64, 32, 64, 128, d_{\text{input}}]$, applying ReLU activations and Batch Normalization[14] between each layer. The encoding is $\tilde{x_i}$. The output of the decoder is $\hat{x_i}$.

The latent layer is normalized using Layer Normalization (LP). For the Wikipedia dataset, the encoder-decoder has layers: $[d_{\text{input}}, 16, 10, 16, d_{\text{input}}]$. The model was trained using 5-fold cross-validation with an 80-20 train-test split, over 2,000 epochs, using the Adadelta optimizer[32].

4.3 Loss functions

We tested five loss functions1 aimed at optimizing the reconstruction, but also ensuring a high distance between bags.

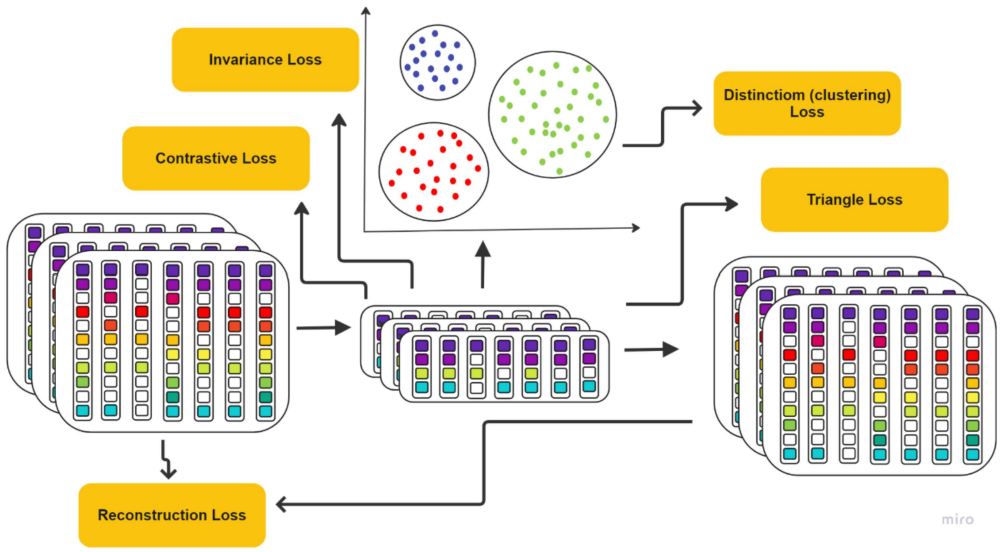

Figure 1: The five losses schema. The scheme shows the bags (rightmost large stack) that are made of the instances (rectangles inside squares). The instances are projected through the model to a latent space (middle stack) and then projected back to the original dimension (left-most stack). The different losses are derived from different layers of the model. **Reconstruction** from the difference between the original and reconstructed bags, **Distinction** from the latent space (through clustering, as illustrated in the three centroids on the middle-top axis), and then **Contrastive**, **Invariance** from the latent space at the bag level, and **Triangle** at the instance level.

- **Reconstruction Loss** The norm of the distance between the decoder output and the original input for each instance:

$$L_{Reconstruction} = \|x_i - \hat{x}_i\|_2 \tag{5}$$

- **Contrastive Loss.** Minimization of the distance between pairs of instances originating from the same bag and maximizing the distance between pairs from different bags.

$$L_{Contrastive} = -log\frac{\exp(\frac{sim(\tilde{x_i},\tilde{x_j})}{\tau})}{\sum_{k=1}^{2N}\exp(\frac{sim(\tilde{x_i},\tilde{x_k})}{\tau})}, where\ \tilde{x_i},\tilde{x_j} \in X_\alpha\ and\ \tilde{x_k} \notin X_\alpha \tag{6}$$

- **Invariance Loss**. The invariance loss ensures the distance is not sensitive to the sample depth,

defined as the number of instances sampled from each bag.

$$L_{Invariance} = -log\frac{\exp(\frac{sim(Agg(\tilde{X}_i),Agg(\tilde{X}_j))}{\tau})}{\sum_{k=1}^{2N}\exp(\frac{sim(Agg(\tilde{X}_i),Agg(\tilde{X}_k)}{\tau})}, \\ where\ \ Agg(\tilde{X}_i), Agg(\tilde{X}_j)\ are\ an\ aggreggation\ of\ the\ the\ same\ bag\ X_\alpha\ in \\ different\ depths\ and\ Agg(\tilde{X}_k)\ of\ other\ bag\ X_\beta. \tag{7}$$

- **Clustering Loss.** The clustering loss is tailored to maximize the clustering of the encoded instances, akin to the objectives pursued in the K-Means algorithm. Specifically, it aims to minimize the within-cluster variances, measured as squared Euclidean distances:

$$L_{Clustering} = \frac{SSE}{\sum_i \sum_j \|x_i - x_j\|} = \sum_{i=1}^{K} \sum_{x_j \in C_i} \frac{\|c_i - x_j\|^2}{\sum_i \sum_j \|x_i - x_j\|} \tag{8}$$

  The clustering is done using the method proposed by Xai beyond classification[20] where the data is encoded and then clustered, where the clustering is optimized every other epoch.

- **Triangle Loss.** The triangle loss is designed to optimize the adherence to the triangular inequality principle. The distances between points forming the vertices of a triangle are represented by the Euclidean distances between the corresponding instance projections. It is computed as the percentage of deviation from the triangular inequality across all possible triples of data points. It is defined as follows:

$$L_{Triangular} = p(d(x_i, x_j) > d(x_i, x_k) + d(x_k, x_j)) \tag{9}$$

  This loss tries to spread the points in wide angles.

4.4 Energy Distance

We use the energy distance[21] for the bag-to-bag distance. Each bag is represented as a set of its instances projection in $\mathbb{R}^n$. For two bags X. Y, it is:

$$D(X, Y) = 2\mathbb{E}[\|X - Y\|] - \mathbb{E}[\|X - X'\|] - \mathbb{E}[\|Y - Y'\|]. \tag{10}$$

4.4.1 Kernel Density Estimation

To ensure that the projection maximizes the similarity of instances within the same bag, we use a kernel-density estimate (KDE) (Eq. 11). KDE is a non-parametric method used to estimate the probability density function of a random variable in a metric space. As seen in the schematic figure 2, it smooths the data points using a kernel function, providing a non-parametric continuous estimate of the underlying distribution. Here we use a class-dependent KDE. For each instance of class A, we compute the density of instances of either class A (self) or class B (other), using a Gaussian kernel with a radius h.

$$\hat{f}(X) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{d(X.X_i}{h}\right); K(d) \qquad = \frac{1}{\sqrt{2\pi}}\exp\left(-\frac{d^2}{2}\right) \tag{11}$$

h was defined as $2 * \sigma(X)$ - the standard deviation of all the distances between the bags. We define the quality of the embedding using the ratio of the self to other densities:

$$S = \frac{Average_X KDE(Self, X)}{Average_X KDE(Other, X)} \tag{12}$$

5 Results

We propose here to learn an embedding of the instances, and then apply an energy distance to the bags. The embedding is learned using an encoder-decoder architecture with five different losses:

1. Reconstruction: error between the original input features and the reconstructed features produced by a mode
2. Contrastive: minimize distance for pairs from the same bag, maximize distance for pairs from different bags.
3. Invariance: maintains consistent distances between the same pair of bags across different sampling depths.
4. Clustering: Minimizes the bag's distance to assigned clusters, aiming at creating a more dense representation.
5. Triangle: optimizes adherence to the triangular inequality by minimizing deviations in Euclidean distances between points forming a triangle.

To evaluate the quality of the distance, we compared the density of bags from class A or B around bags from class A (as estimated by a KDE). A good projection should maximize the ratio of the self (density of A around A) to other (density of B around A) densities.

To illustrate that, we generated toy data of bags from two labels, where the instances of bags with one label are far from the instances of bags with the second label. Following the method above, the bag-to-bag distance was calculated using energy distance, and the bag projections were visualized using multi-dimensional scaling (MDS). (Fig. 2 A). The higher KDE for bags from the same label (A, A) vs bags of different labels (A, B) captures the separation of the bags as seen in (Fig. 2 b). We propose the same approach to estimate the quality of the distance on real-world datasets.

5.1 Limitation of instance-based distances

When we applied the same approach to real-world datasets (using pre-defined projections of the instances), a poor separation was noticed between bags of similar labels and those of different labels, as can be observed by the low densities, as well as similar density distributions between the self and other densities (Fig. 2 d). One can see that for example the MDS projection based on the bag distance (computed similarly to the toy model) of the Elephant dataset (Fig. 2 d), reveals a lack of visual distinctiveness among labels, with the red and blue points overlapping. This observation is further supported by the low ratio between the KDE values for the distances based on instances across all datasets (Fig. 2 b).

5.2 An autoencoder that learns the instance distribution does not improve the bag distance metric.

If indeed instances are closer within a bag than between bags, an embedding of the instances learned via an auto-encoder (AE) could in theory ensure a more uniform distribution of the instances within bags, and as such improve the within-to-between bag density. We trained an AE using only a reconstruction loss (i.e. only ensuring that the AE reproduces the input). However, in general, the AE only decreases the within-to-between class density (Fig 3C). The reconstruction loss decreases to very low values, suggesting a good reconstruction of the instance values. However, using the encoded latent representation to compute the bag-based distance does not improve the within-to-between class density ratio.

5.3 The within-to-between bag density ratio can be improved using other losses

Reconstructing the original instance space does not improve the within-to-between class density ratio. We thus tested alternative losses. Those include Contrastive, Invariance, and Triangle losses. When

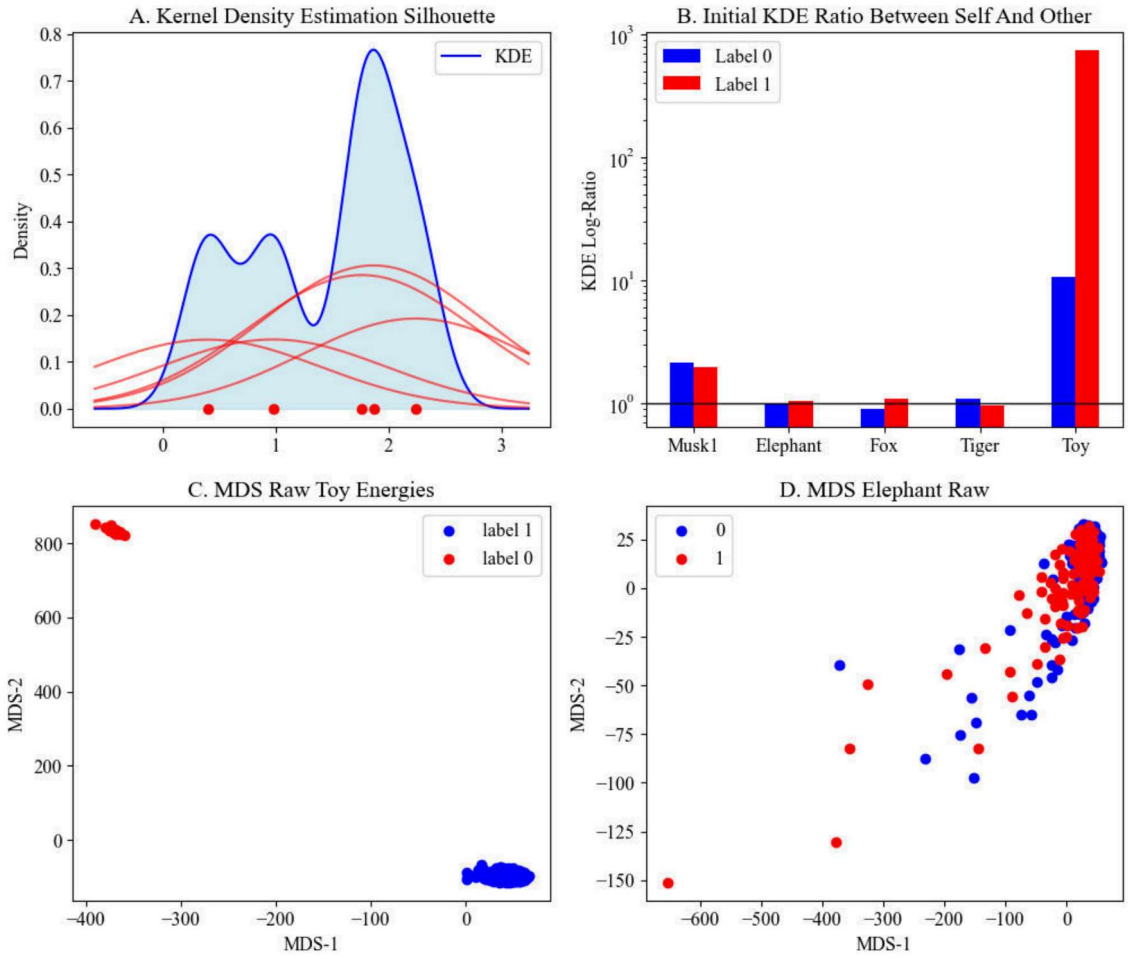

Figure 2: **(a)** KDE schematic showing smoothing of data points. **(c)** MDS visualization for toy data and raw Elephant data **(d)**. Axes represent MDS dimensions. Blue and red denote labels 0 and 1. Label 0 of toy data in range [0, 2] and label 1 in range [10,15]. Toy data shows clear label separation, while instance-based distance on Elephant lacks distinct separation. This correlates with the high KDE values for Toy data and low for Elephant as seen **(b)** in bar plot for KDE of both labels.

training with one of the losses at a time, each of the four losses converges rapidly (See musk data results in Fig. 3 a. The results are similar for other datasets). The ratio between the density of each of the labels improves and grows over time for the contrastive and invariance losses, but not for the reconstruction or triangle loss (See Fig. 3b for Musk - results are similar for other datasets). Indeed, following convergence, both Invariance and contrastive losses increase the within to between class densities, in contrast with the reconstruction and triangle that reduce the ratio (See average for training and validation in Fig. 3c, and distribution over 5 folds in Fig. 3d).

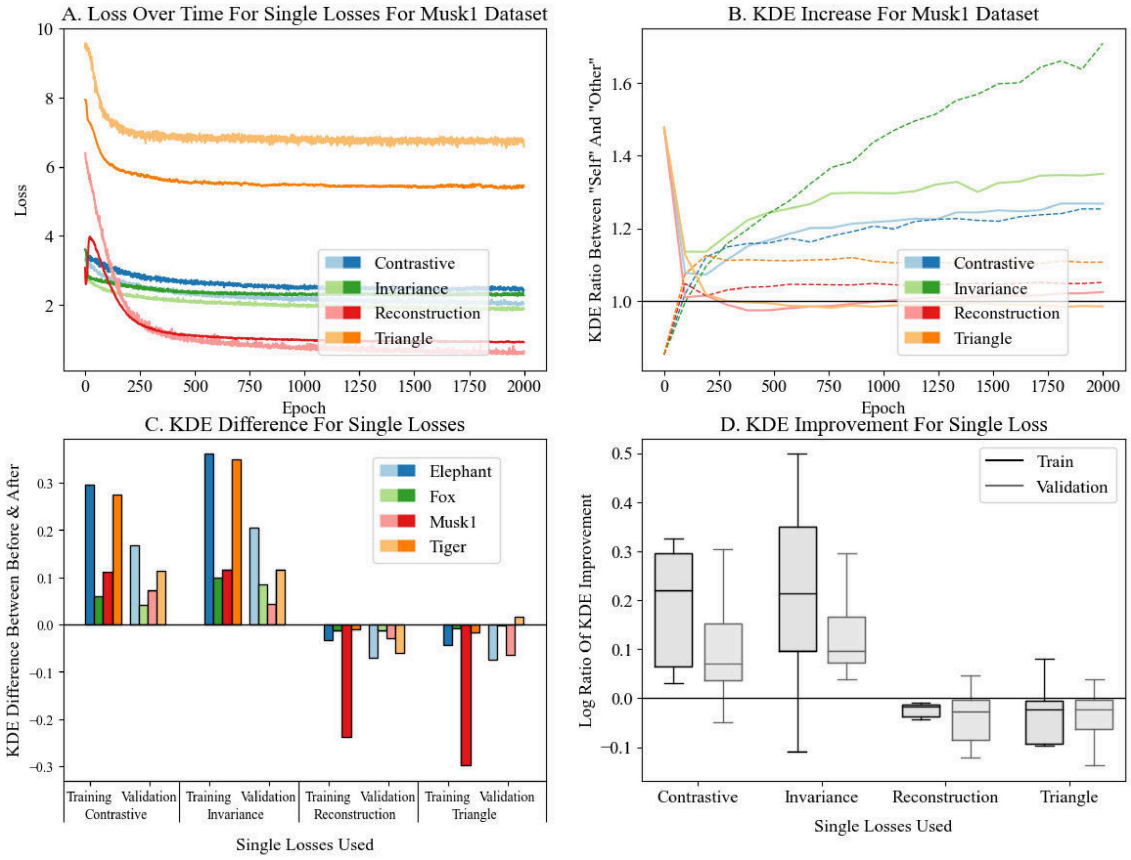

Figure 3: The four losses decay when training with a single loss at a time for the musk data **(a)**, each color accounting for a single loss, with the darker colored line for the training loss and the lighter for validation loss. An example of the KDE decay with the Musk data when training with a single loss at a time **(b)**, with the different colors for indicating the losses, where the light and dark trend lines exhibit the two different labels. The beginning-to-end ratio of KDE improvement $(KDEend - KDEbeginning)/KDEend$ for each dataset for each loss **(c)**, with the different colors indicating the losses, and the light and dark bars representing the two labels. Summarizing of the KDE improvement for each loss on all datasets **(d)**, with the two shades of gray denoting training and validation.

5.4 Multi-class classification using novel Wikipedia dataset

To test that the conclusions above can be extended to multiple classes, we developed a new MIL dataset composed of the BOW of Wikipedia cities descriptions from different countries (See Fig. 4 d for the number of cities per country). The class of each bag was its country. We computed the density of the bag own country bags vs other countries at the end of training. (Fig. 4b for average relative improvement per country in within to between class density in the validation set, and 4c. for the distribution). Again, the contrastive and invariance losses induce the best improvement consistently over all cities. the other losses have practically no effect.

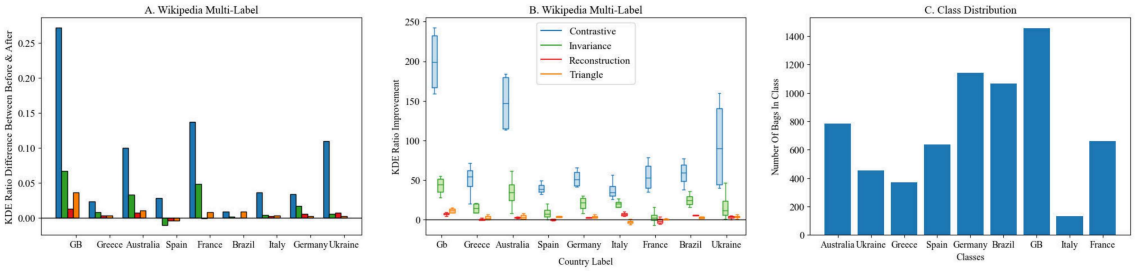

Figure 4: The beginning-to-end KDE improves $(KDEend - KDEbeginning)/KDEend$ for each of the four losses **(a)**. Each color accounts for a single loss. Distribution of the KDE improvement percentage for each loss for each label across all folds **(b)**, with the four colors corresponding to the four losses. The labels in the exhibited fold are almost balanced **(c)**, causing one label to improve more than others.

5.5 Inclusion of clustering loss and combination of losses

A simple solution to increase the self-density is to produce a projection that would better cluster. We added a clustering loss that mimics a k-means clustering (see methods). For each loss, (reconstruction, contrastive, invariance, and triangle) we trained the model with the loss fixed to 1 and clustering loss with weights $0, 0.01, 0.1, 1, 10$. The clustering loss slightly improves the within-to-between class density (Fig 5a for wiki data and b for all other datasets). However, the difference is non-significant (ANOVA on all methods and clustering loss weight). Note that the clustering is unstable, leading to large fluctuations in the loss and KDE. However, the clustering loss may be important for the representation, since a more dense cluster of bags aids with a clear distinction between the bags.

Given the importance of the contrastive and invariance loss and the failure of the reconstruction loss, we tested whether combining the losses can actually improve the within-to-between class density. Adding a reconstruction term to the loss consistently decreases the ratio. In contrast, the ratio of the contrastive to invariance loss is inconsequential, and both losses seem to be equivalent (Fig. 5c and d).

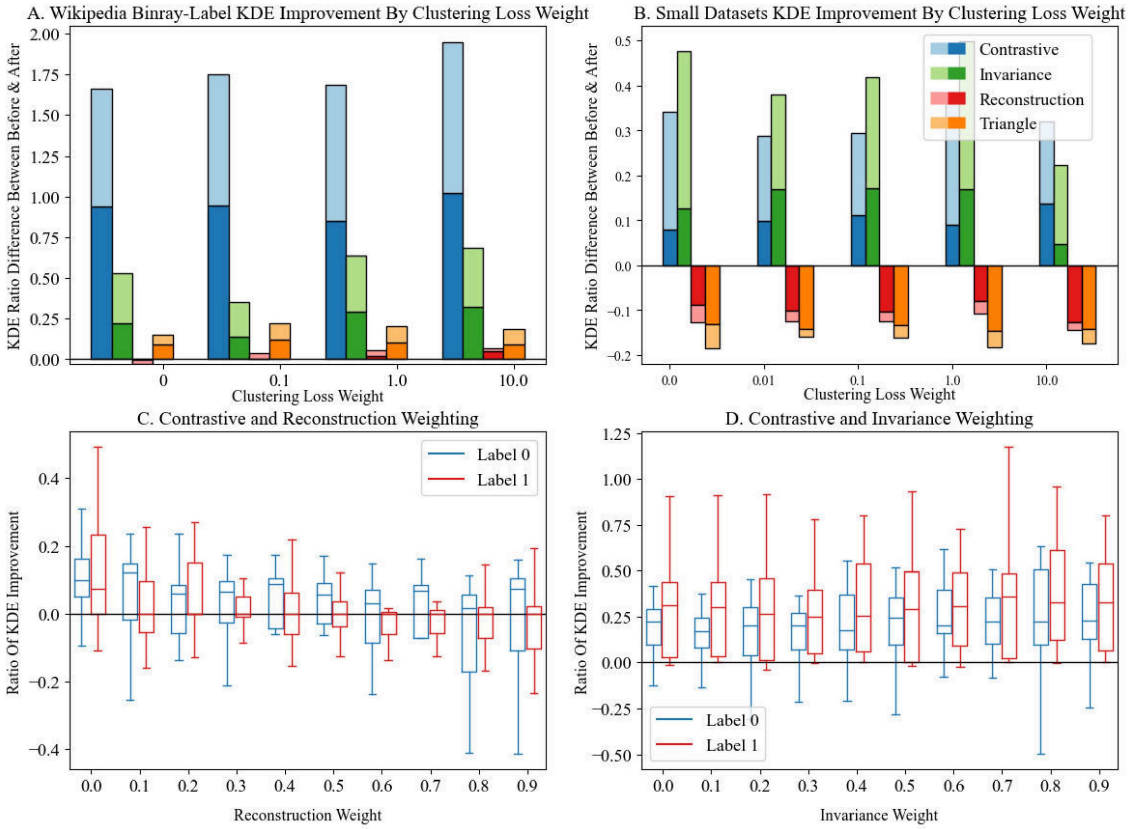

Figure 5: Clustering loss does not improve nor deteriorate. The models trained with clustering loss weighted with every other loss, for every weight of clustering loss while fixing the other loss weight to 1 for Wikipedia **(a)** and all other datasets **(b)**. The darker shade is for label 1, and the lighter is for label 0. The percentage of improvement $(KDEend - KDEbegining)/KDEend$ of the KDE before vs after training exhibits no distinct impact.Balancing between contrastive loss and other important losses reconstruction **(c)** and invariance **(d)** depends on the dataset.

5.6 Classification Using Learned Metric

Finally, we tested whether the self-supervised loss also improved the classification accuracy. We used a simple KNN setup, where we defined the distance between each pair of bags using the learned embedding and the energy distance between bags. We then computed the classification accuracy. Following the self-supervised training. The test bags are classified using the majority vote of the 9 nearest neighbors for all datasets. We then computed the accuracy and increase in accuracy (vs the original instances values) of the KNN classification, either as a binary (average of each class vs all others), or multiclass (Fig. 6). As was the case for all other measures and datasets, in all cases, the contrastive and invariance loss improve the accuracy, while the triangle and reconstruction reduce the accuracy. The clustering loss has a limited effect.

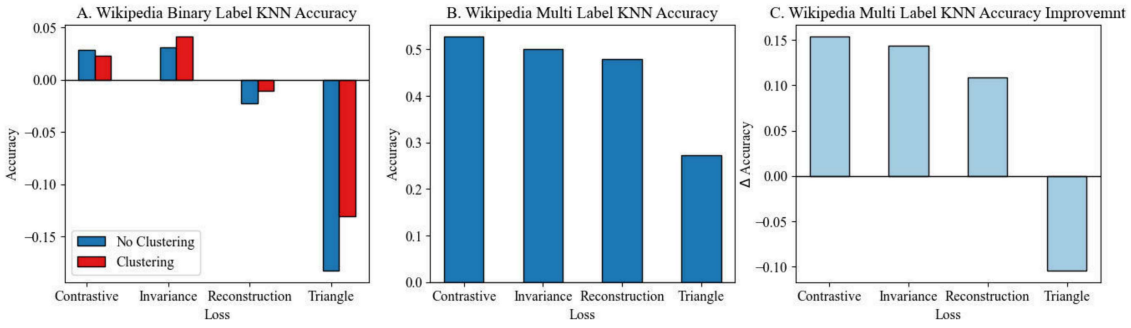

Figure 6: KNN classification accuracy. **(a)** for Wikipedia dataset for the binary label, with clustering loss and without. **(b)** KNN classification accuracy on the multi-label Wikipedia setup. **c** KNN classification accuracy improvement compared to KNN accuracy on original data

6 Discussion

While MIL problems were extensively studied in supervised contexts, there are very limited solutions for distances between bags. We have here presented SUMIT a self-supervised approach to embed each instance to optimize the distance between bags. We used five types of distances: A reconstruction distance focused on reproducing the instance values as precisely as possible, a triangle distance that maintains the geometry of the projection, a clustering distance that ensures that the projection induces distinct clusters, and two bag-level self-supervised distances that minimize the distance within each bag. The first is a contrastive loss that compares the distance between and within bags, and the second distance ensures that the distance between bags is not sensitive to the number of instances sampled from each bag. Following the projection, we used an energy distance on the instances to define a bag distance.

We tested this distance definition by measuring the density of bags of different labels around bags of the same labels in standard datasets as well as a novel wiki-based dataset scrapped for the current task from the wiki pages of different cities. We have shown that while the instance-based distances (Triangles and reconstruction) do not increase, and sometimes decrease the density of bags of a given label around bags of the same label, the self-supervised distances increase this density. Thus, for the sake of an optimal distance, maintaining the bag information is more important than the precision of the instance reconstruction from the projection.

Adding the clustering distance to the bag level losses further improves the density of same-level bags, probably by ensuring more dense and separated projections.

SUMIT bridges the gap between bag-embedding methods where the entire bag is projected into a single vector in $\mathbb{R}^n$, and instance-based methods, where the focus is on the instances. Moreover, the combination of the clustering loss increases the typical self-density, but also the contrast between bags of different labels.

To the best of our knowledge, SUMIT is the first effort explicitly targeted at developing an instance-based distance between bags that is optimized at the bag level. Many choices in this analysis are arbitrary and can be changed. Those include among others the model used for the projection (currently a two-three-layer fully connected neural network), and the precise definition of the loss for each of the components (e.g. changing different types of self-supervised loss). However, from our experience, the precise details of either the encoder or the decoder, or the precise shape of the loss do not affect the qualitative results presented here.

Appendix

A Related Work and Applied Work

Table 2 provides a summary of related work and applied work in the domain of MIL. The papers selected are ones that either create or use a distance notion.

Paper	**Method**	**Distance**
Citation-KNN	Hausdorff Distance	Hausdorff Distance
Multi-Instance Learning by Treating Instances As Non-I.I.D. Samples	MI-Graph	Graph Distance
Multiple Instance Learning with Bag Dissimilarities - MIND	Bag represented by vector of its dissimilarities to other bags which is the feature vector	Dissimilarity, not Distance
Contrastive Multiple Instance Learning: An Unsupervised Framework for Learning Slide-Level Representations of Whole Slide Histopathology Images without Labels	SIMCLR for instances, fuzing together with attention	Can be derived from encoding
Multi-instance clustering with applications to multi-instance prediction	Bag representations as a vector of distances to i medoids.	Hausdorff distance
Unsupervised Multiple-Instance Learning for Functional Profiling of Genomic Data	agglomerative or partition clustering and MIL's citationkNN	Measurement based of maximum pairs of instances with minimum MI
Unsupervised Multiple-Instance Learning For Instance Search	Finding Instance That Matches Bag Label	NA
Applied Work		
Deep learning of feature representation with multiple instance learning for medical image analysis	Extracting features then classifying with them	NA
Dual attention multiple instances learning with an unsupervised complementary loss for COVID-19 screening	Contrastive loss is applied at the instance level to encode the similarity of features from the same patient against representative pooled patient features.	NA

Table 2: Table summarizing related work and applied work in MIL that use a distance

B Data Sets

The benchmark MIL datasets additional statistics are presented in Table 3 and Figure 7. We can see that the number of instances per bag can have a big variance, especially in the Musk1 data. This exhibits a general issue of the variance in the "depth" of bags, the number of instances per bag. The larger bags can pose problems when trying to learn an MIL problem, as well as the variance of the bags. In the BS paradigm, when we have a large number of instances in the bag we miss a lot of information[2], ES setting, a large number of instances can average out the bag representation compared to the smaller bags. In the instance-space setting, large bags can also have a disproportionate influence. Taking these impacts into consideration, we require our model to learn the invariance loss as well, i.e. be invariant to the depth of the bags.

Database	**Number of bags**	**Avergae Number of instances class per bag**	**Bags per class (pos./neg.)**	**Number of dimensions**	**Classes**
Musk1	92	5.17	47/45	166	2
Elephant	200	6.10	100/100	230	2
Fox	200	6.60	100/100	230	2
Tiger	200	6.96	100/100	230	2

Table 3: Data sets statistics, Additional Information

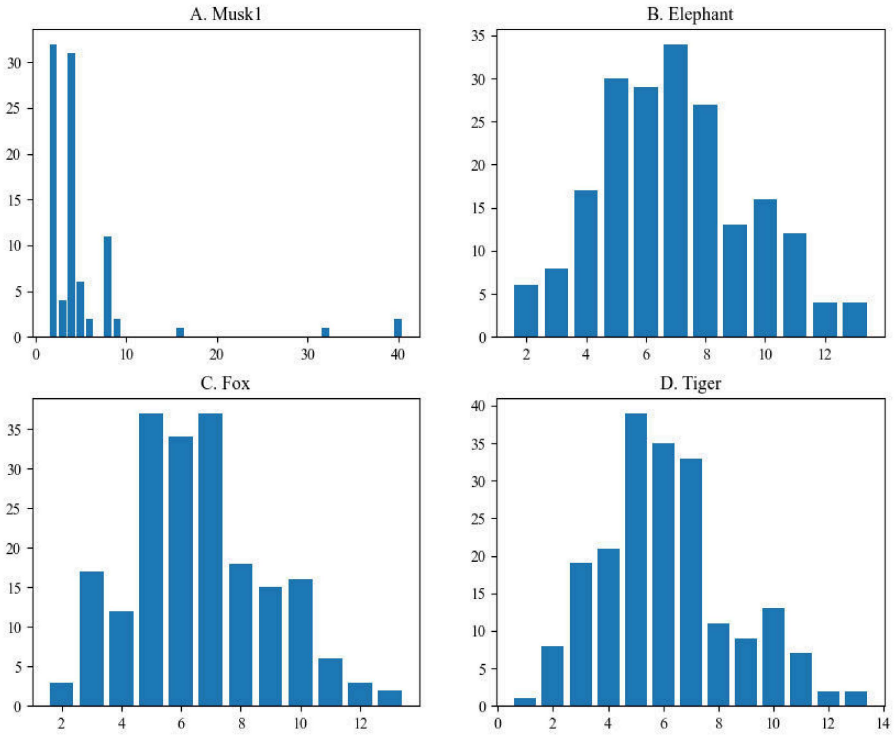

Figure 7: **Histogram Of Number of Instances per Bag**. Musk1 **A**, Elephant **B**, Fox **C**, Tiger **D**

C Wikipedia

C.1 Wikipedia Topics

The Wikipedia dataset consists of 10 country-specific labels. For each country, we extracted pages for its cities, treating each city as a bag and the sections of its page as instances. The sections are represented as a bag of words, with sentences cleaned of stop words and other irrelevant words. For the binary form of the Wikipedia data, GB was counted as label 1, and all other countries where 0.

Country	**Number Of City Pages**
GB	5340
Australia	814
Ukraine	413
France	1149
Greece	347
Spain	481
Italy	754
Germany	987
Brazil	170

Table 4: Country Values Table

C.2 Wikipedia Preprocessing

The raw Wikipedia pages are processed into a bag-of-words representation as follows. First, all stop words are removed using the NLTK package[15]. Next, the words are filtered by their frequency, such that words with a count under a minimum threshold of 0.01 and over a maximum threshold of 0.9 are filtered out. The reason for filtering out very frequent words is to mostly ignore parts of speech that do not convey information about the topic of the page, such as prepositions. The words that appear under a certain frequency are filtered out since the notion is that if a word appears only a single number of times in the data, one cannot infer from it anything. Note that if a different representation of the words was chosen, i.e. some word embedding, then also words which appear one time in our data can be inferred from and therefore would not be dropped.

D Notations

We will use the following notations throughout the appendix:

1. A bag α of m instances will be marked $X_\alpha = x_1, x_2, ..., x_m$, where each x_i is an instance of the bag.
2. Each instance j is represented by a feature vector x_j in a d-dimensional feature space (i.e., $x_j \in \mathcal{R}^d$).
3. An encoding of an instance will be marked as $\tilde{x_i} \in R^e$ in an e-dimensional encoding space, and for a bag X_i with n instances the encoding will be marked $\tilde{X_i} \in R^{eXn}$.
4. A decoding of an instance will be marked as $\hat{x_i} \in R^d$ in the original d-dimension feature space, and for a bag X_i with n instances the decoding will be marked $\hat{X_i} \in R^{dXn}$.

E Clustering

The clustering loss cannot be applied alone. This has to do with the way the clustering is performed, where the latent space of the encoder-decoder is projected to a linear layer in the size of the number of desired clusters. If the latent layer is an arbitrary projection of the original data, obviously there is no significance for the clustering which is based on it. (The approach chosen is based off of Interpretable neural clustering[20] and further explanations for this approach are provided there). The clustering loss does not significantly improve the KDE, but may be important for the representation, since a more dense cluster of bags aids with a clear distinction between the bags. The clustering loss may slightly help with classification. To visually examine that hypothesis, the latent layer of the encoder-decoder trained with clustering loss is projected to a two-dimensional space using PCA, and then a bag's cluster is indicated by color, and labeled by the marker (Fig. 8). As can be seen in (Fig. 8), distinct clusters are formed, and the clusters also mostly have the same label, as shown by the different clusters indicated by colors having the same markers. This is a visual illustration, so clustering efficiency was determined by comparing mutual information between the clusters and the labels. Note that other reduction methods can be used for vizualization.

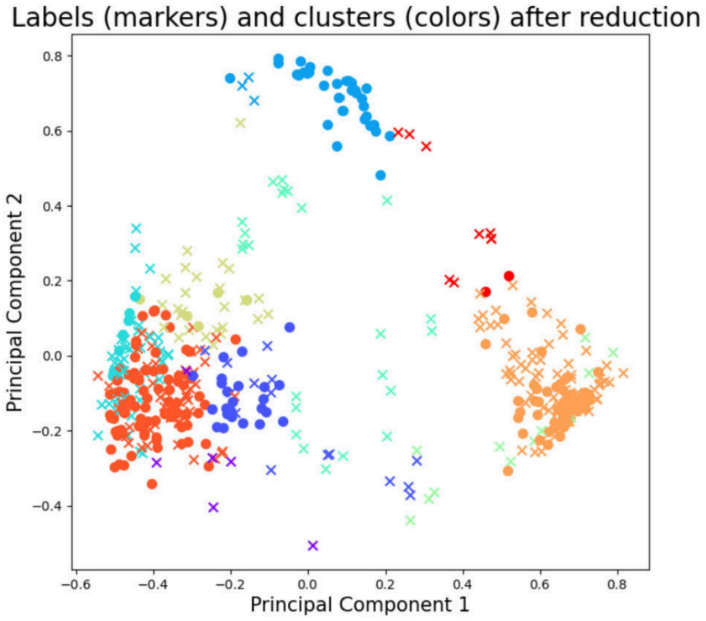

Figure 8: **Clusters Visualization After PCA Reduction** Each bag is represented as a dot in the 2-dimensional space, the color indicates the cluster and the marker indicates the label. The model is trained for 300 epochs, with the learning rate set to 0.1 with the Adelta optimizer. The ratio between the reconstruction loss and the clustering loss is 100, and the clustering is divided into 10 clusters.

F Triangle Computation

The triangle loss is defined as:

$$L_{Triangular} = p(d(x_i, x_j) > d(x_i, x_k) + d(x_k, x_j)) \tag{13}$$

Computing this in a triple for loop for each $i, j, k \in [0, N]$ for n data points takes a long time. Here is the more efficient computation done in the code and an explanation for how it is equivalent: We will start with what we want for a single $[i, j]$, then can generalize it for all i, j. In a nxn adjacency matrix where $mat[i, j] = d[i, j]$, i.e. the distance between points i and j, we would like to maximize, $mat[i, j] < mat[i, k] + mat[k, j]$ for every point $k, k \in [0, N]$. This can be written as maximizing:

$$\frac{mat[i, k] + mat[k, j]}{mat[i, j]} \tag{14}$$

We can apply an exponent:

$$e^{\frac{mat[i,k]+mat[k,j]}{mat[i,j]}} = e^{mat[i,k]+mat[k,j]} - e^{mat[i,j]} = e^{mat[i,k]} * e^{mat[k,j]} - e^{mat[i,j]} \tag{15}$$

This really should be written as sum of every point k:

$$\sum_{k=0}^{N} e^{mat[i,k]} * e^{mat[k,j]} - e^{mat[i,j]} = \sum_{k=0}^{N} e^{mat[i,k]} * e^{mat[k,j]} - N * e^{mat[i,j]} \tag{16}$$

So, we want to maximize the sum from every point k to points i and j:

$$\sum_{i=0}^{N}\sum_{j=0}^{N}\sum_{k=0}^{N} e^{mat[i,k]} * e^{mat[k,j]} - N * e^{mat[i,j]} \tag{17}$$

Since losses are learned such that they are minimized in training, we can say that we want to minimize:

$$-(\sum_{i=0}^{N}\sum_{j=0}^{N}\sum_{k=0}^{N} e^{mat[i,k]} * e^{mat[k,j]} - N * e^{mat[i,j]}) \tag{18}$$

Figure 9 illustrates the computation. If we look at multiplying a matrix (AxB=C), specifically at multiplying row 1 with column 2: when we perform exponentiation on the matrix and subsequently multiply it by itself, the resulting matrix will contain entries that represent the sum of all possible paths of length two from node i to node j, passing through an intermediary node k. Specifically, this applies to the cell corresponding to the coordinates j. Next, we can normalize the resulting matrix C by dividing each element by the exponentiated values of the original matrix. Our objective then becomes maximizing the sum of this normalized matrix. And then we would wish to maximize the sum of this final matrix. The original matrix is normalized to be on the same scale as all other losses. The exp matrix multiplication is divided by the size of the matrix for normalizing. Also, the sum is divided by the size of the matrix for normalizing.

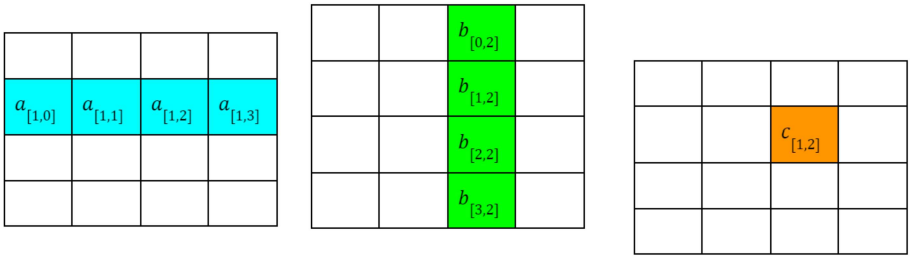

Figure 9: **Triangular Loss Computation**. The blue row in matrix a, and the green column in matrix b, correspond to all paths with a start in 1 and end in 2, so start and end in the orange cell in matrix c.

G Single Losses

G.1 Losses Convergence

Each of the Reconstruction, Contrastive, Invariance, and Triangle losses converge when applied as a stand-alone loss. For each loss, changing the parameters can optimize convergence, as shown in each of the figures below. All of the figures are on the Elephant dataset, although the loss behaves the same for all datasets.

1. Reconstruction: error between the original input features and the reconstructed features produced by a mode

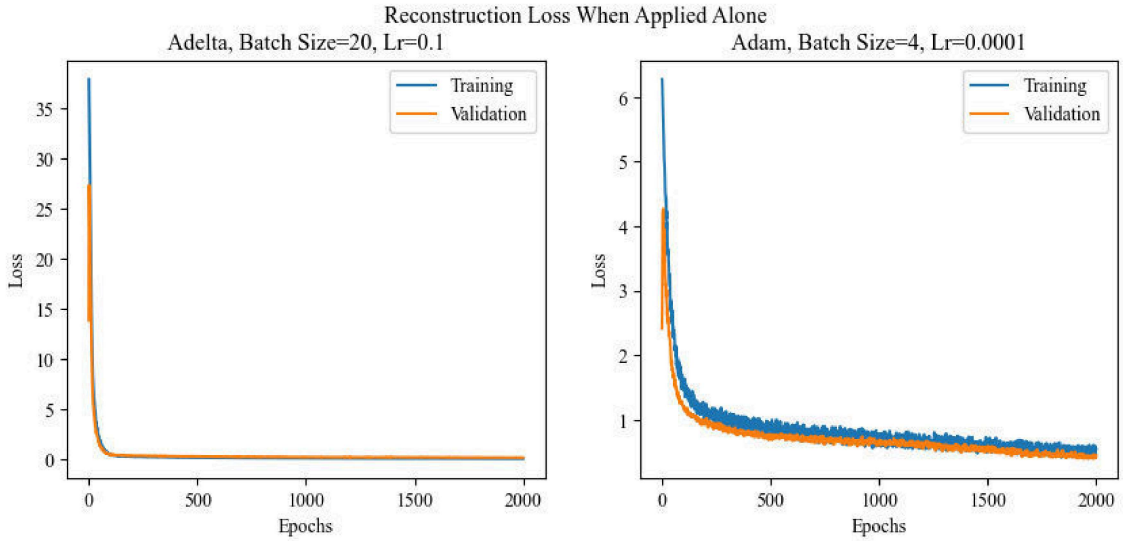

Figure 10: Reconstruction Loss Convergence

2. Contrastive: minimize distance for pairs from the same bag, maximize distance for pairs from different bags.

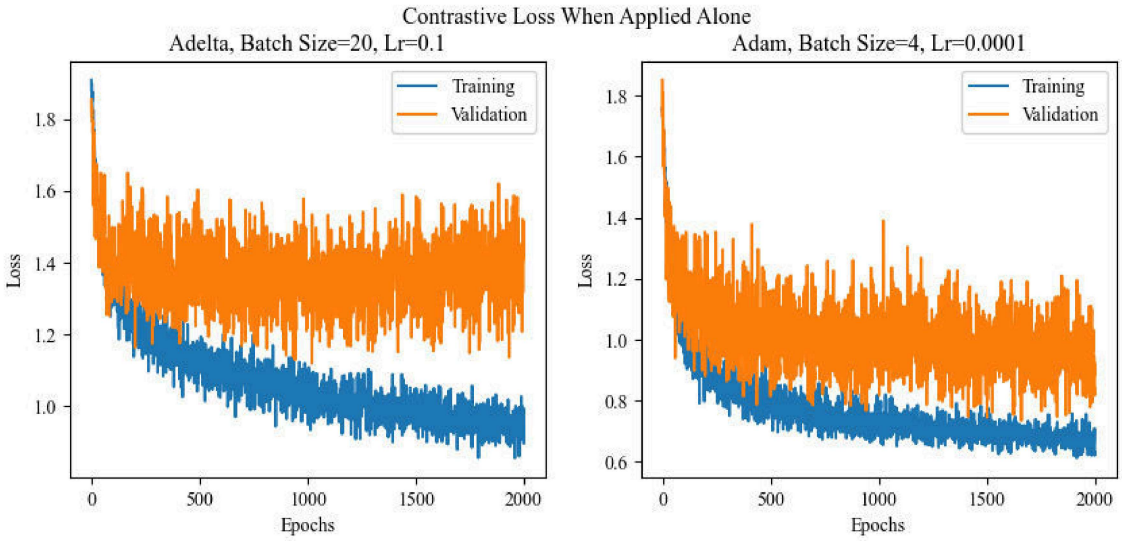

Figure 11: Contrastive Loss Convergence

3. Invariance: maintains consistent distances between the same pair of bags across different sampling depths.

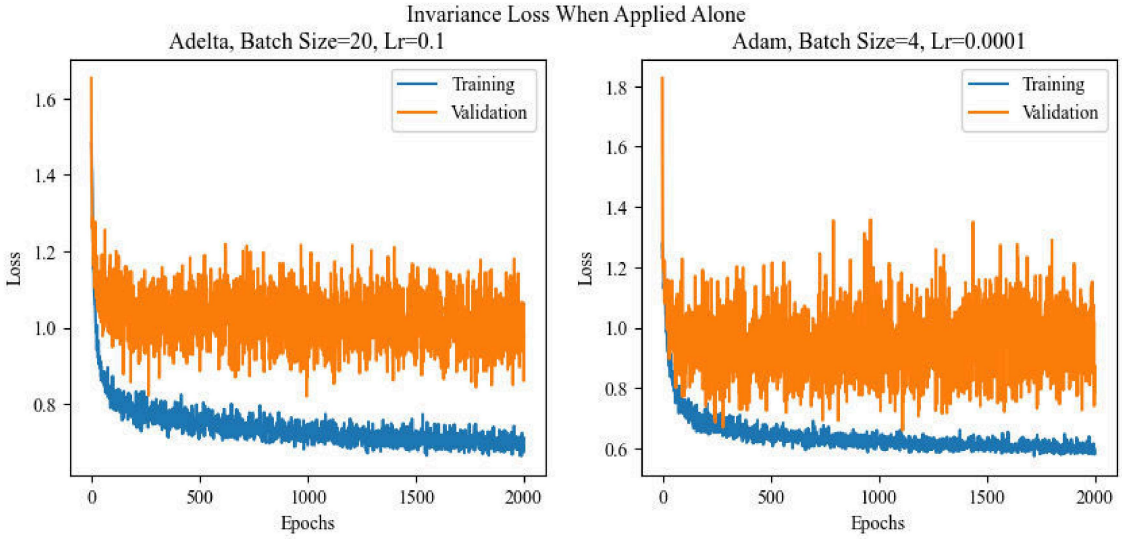

Figure 12: Invariance Loss Convergence

4. Triangle: optimizes adherence to the triangular inequality by minimizing deviations in Euclidean distances between points forming a triangle.

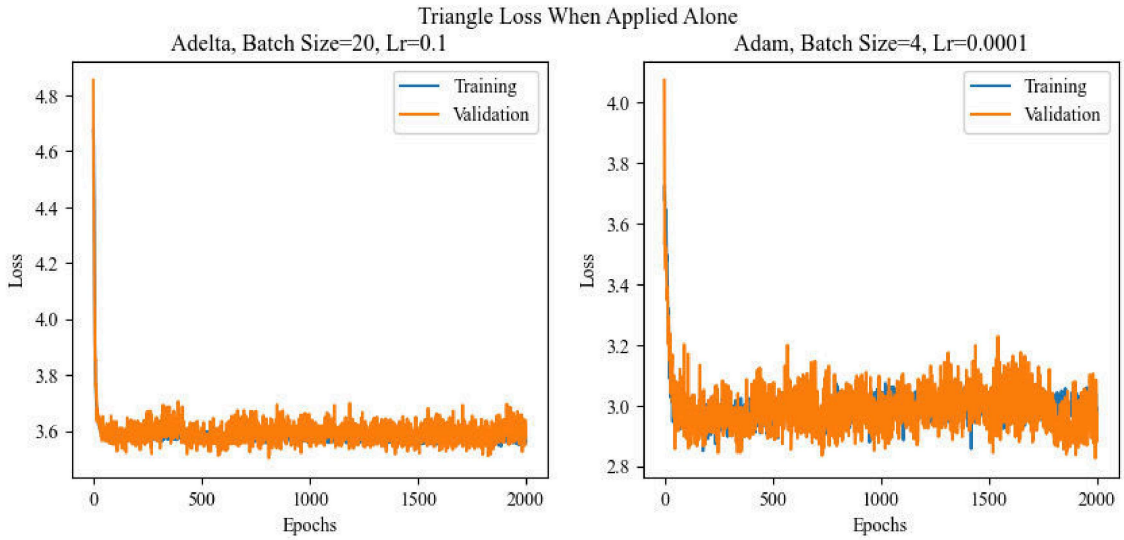

Figure 13: Triangle Loss Convergence

G.2 Losses Projection

After training with each of the Reconstruction, Contrastive, Invariance, and Triangle losses, the data is projected to the latent space, and the bag-to-bag distance is computed via energy distance and then reduced to a two-dimensional space using UMAP. For each bag, a cluster is assigned and the mutual information between the bag label and the assigned cluster is computed. One can see how the mutual information between the clusters and the labels for each bag is greater for the contrastive and invariance loss for each benchmark datasets. Umap projection after training with reconstruction and clustering is presented in Figure 14. The mutual information is relatively low compare to the other losses.

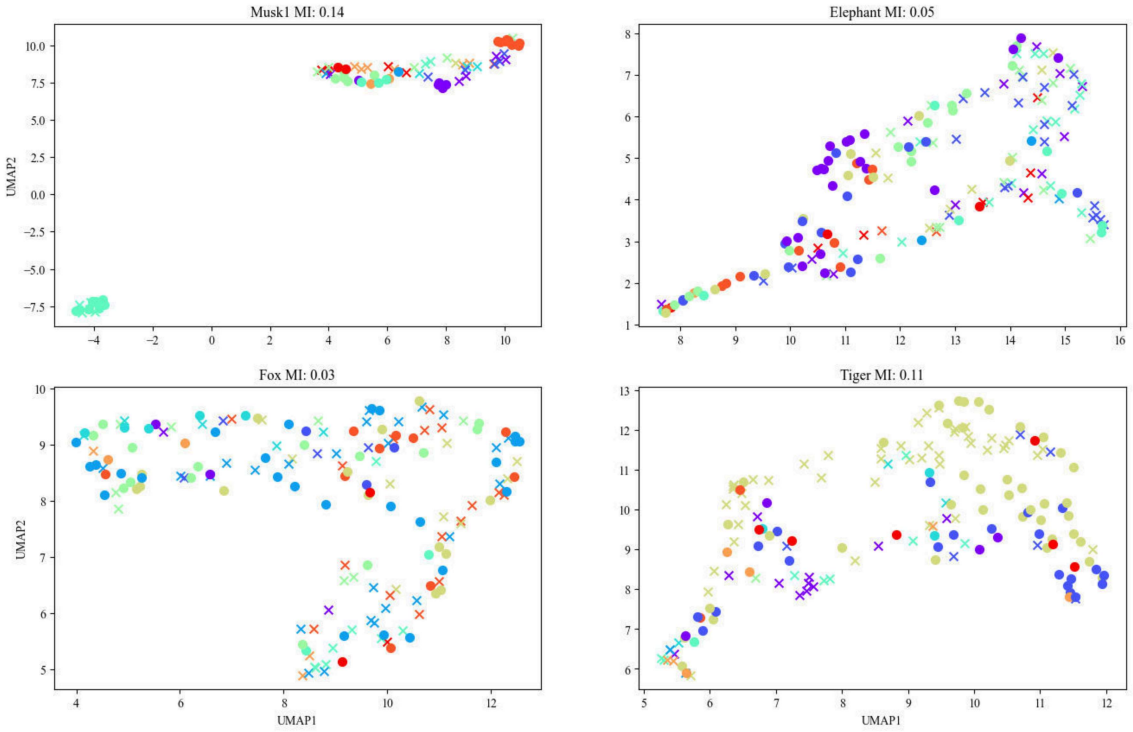

Figure 14: **Projection After Training With Reconstruction Loss** colors indicate clusters, markers indicate labels.

Umap projection after training with contrastive and clustering is presented in Figure 15. The mutual information is relatively higher than in reconstruction.

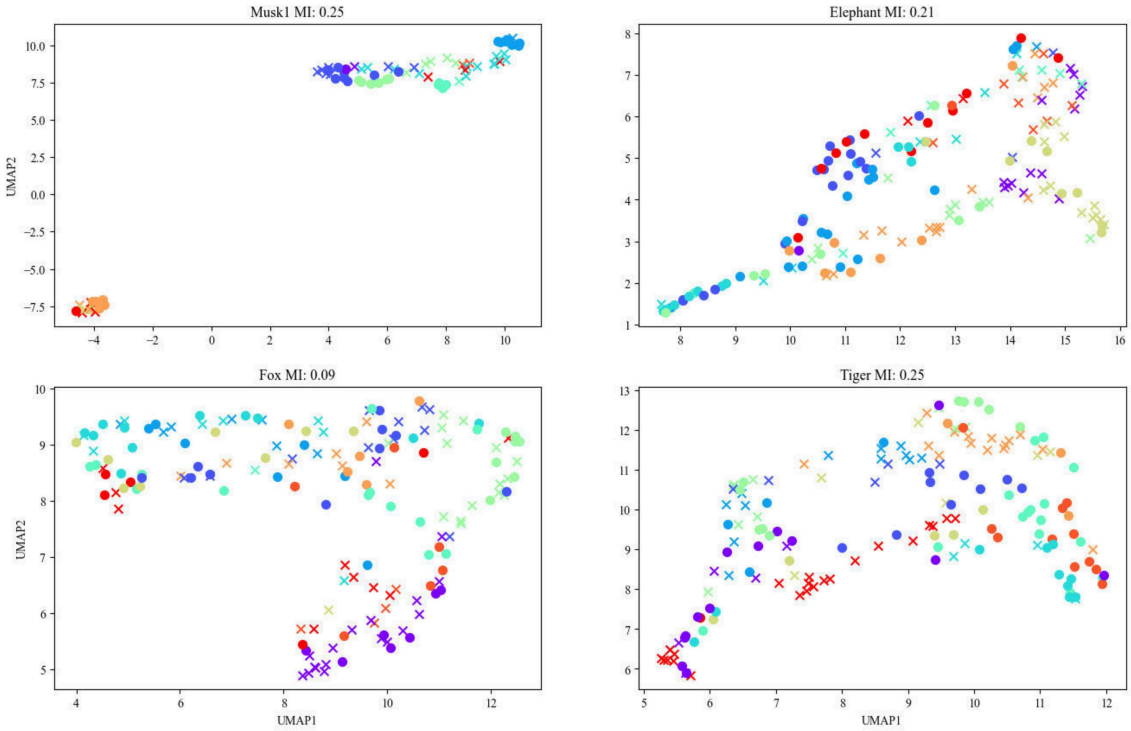

Figure 15: **Projection After Training With Contrastive Loss** colors indicate clusters, markers indicate labels.

Umap projection after training with invariance and clustering is presented in Figure 16. The mutual information is relatively higher than reconstruction and relatively close to contrastive.

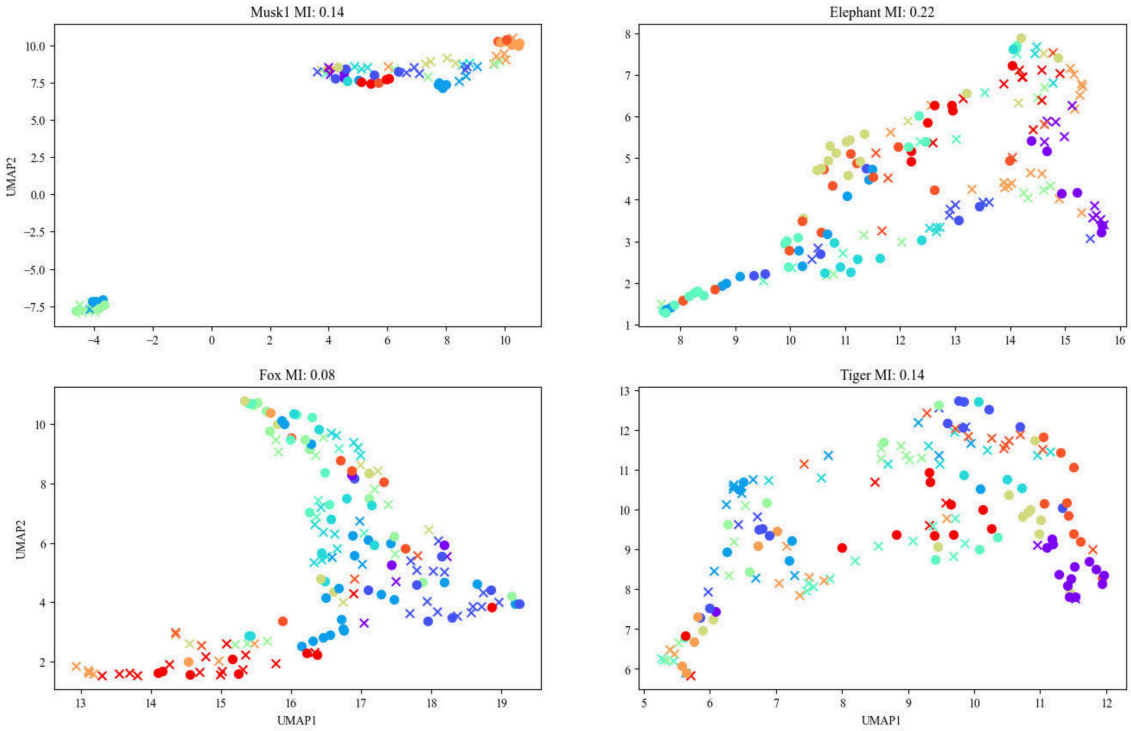

Figure 16: **Projection After Training With Invariance Loss** colors indicate clusters, markers indicate labels.

Umap projection after training with triangle and clustering is presented in Figure 14. The mutual information is relatively low when compared to the other losses.

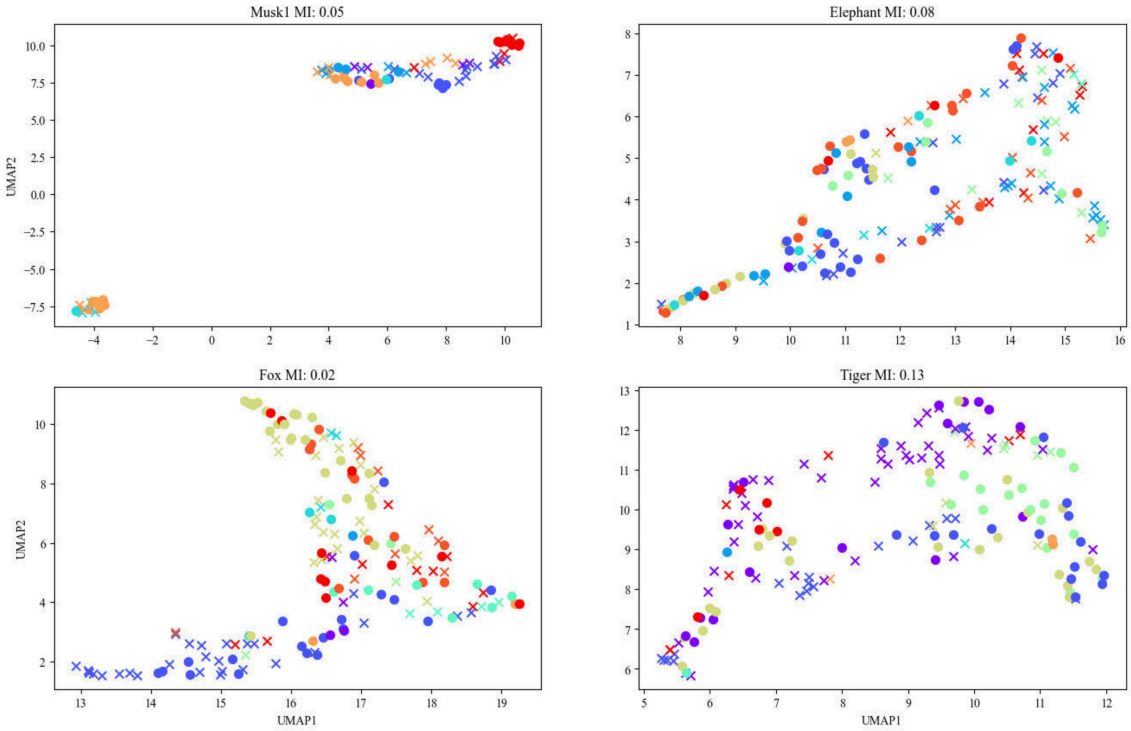

Figure 17: **Projection After Training With Triangle Loss** colors indicate clusters, markers indicate labels.

H Combining Losses

After ensuring each loss converges on its own, a combination of the losses is required. Linear combinations for each of the two leading losses - contrastive and invariance were tested, and also for the contrastive and reconstruction, since reconstruction loss is the main component of the basic encoder-decoder setup (Figure 18).

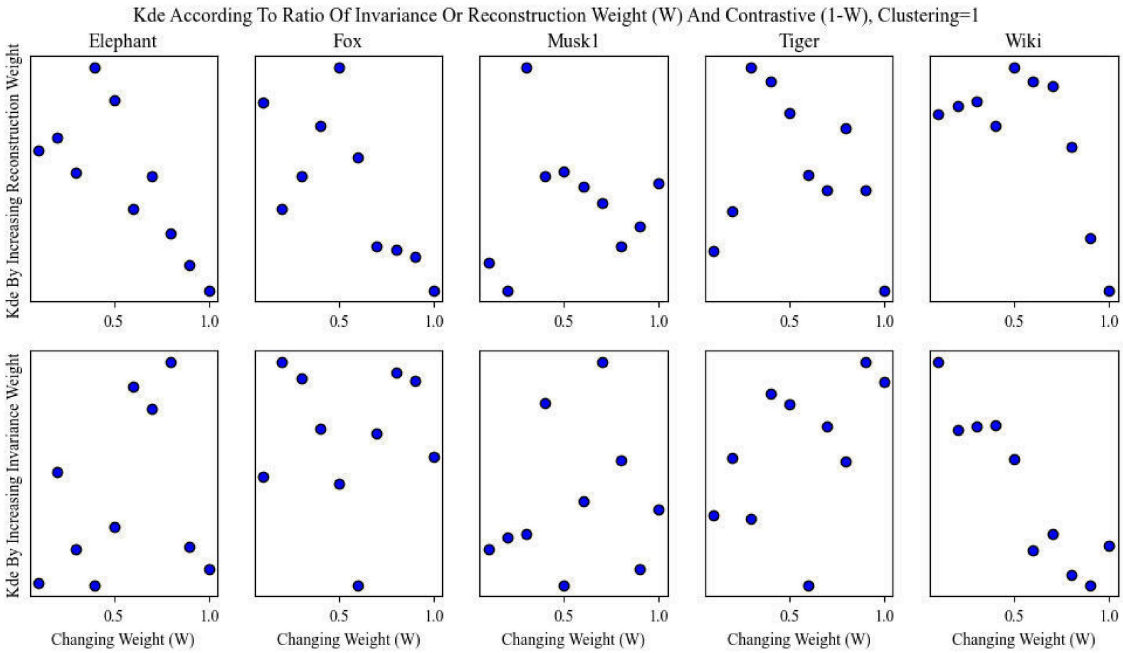

Figure 18: **Contrastive Loss combined with Invariance And Reconstruction Loss** Need new image here, place holder

The invariance loss and the contrastive loss when combined, i.e. setting the weight of invariance to 0.5 and weight of contrastive to 0.5, the two losses converge together, and learning one loss does not contradict learning the other as can be seen in Figure 19

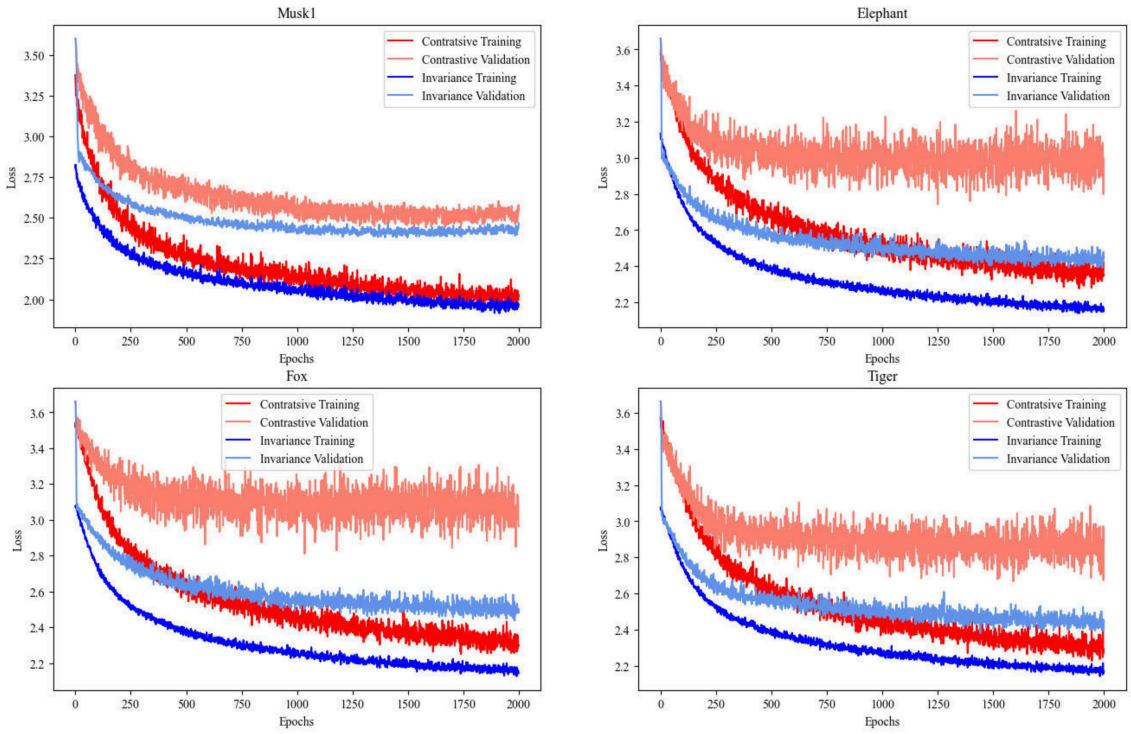

Figure 19: **Convergence of Contrastive Loss combined equally with Invariance**

The reconstruction loss and the contrastive loss when combined, i.e. setting the weight of reconstruction to 0.5 and the weight of contrastive to 0.5, the two losses converge together, and learning one completely contradicts learning the other as can be seen in Figure 20. It seems that the model learns better the reconstruction loss over the contrastive loss, which complies with the finding that the contrastive loss is more important as can be seen in the lower KDE (Figure 18).

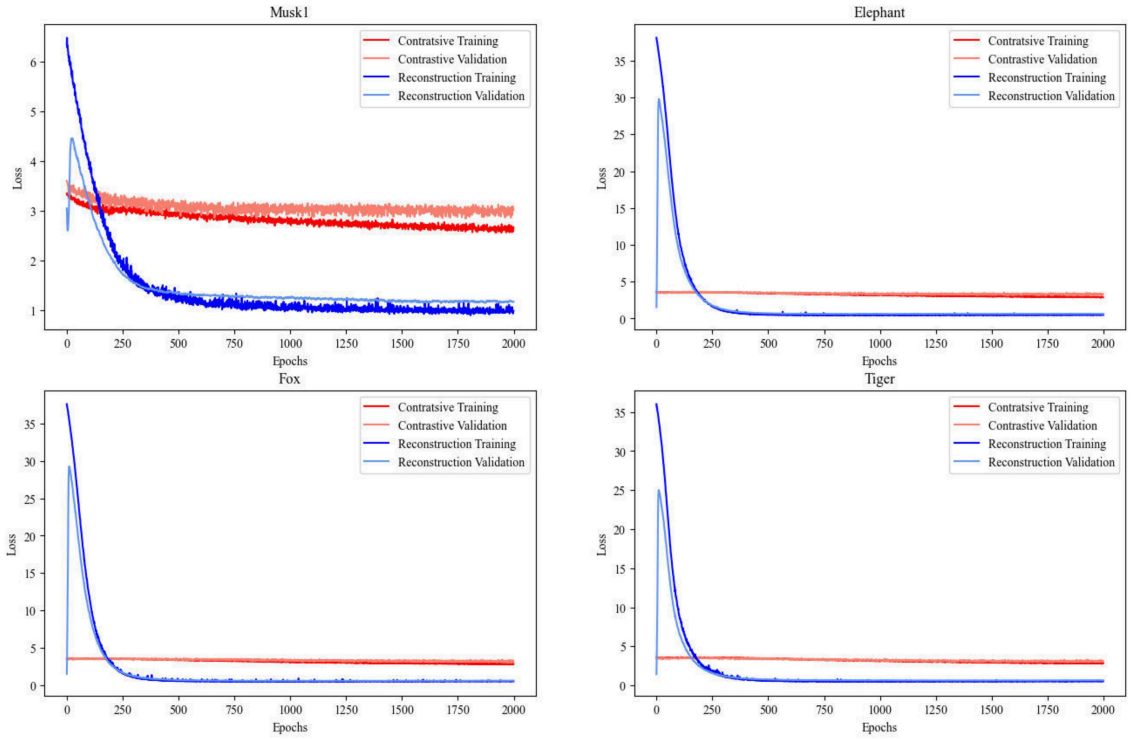

Figure 20: **Convergence of Contrastive Loss combined equally with Reconstruction**

References

[1] 20 Newsgroups corpus popularly used in text categorization. Used in "Z.-H. Zhou, Y.-Y. Sun, and Y.-F. Li. Multi-instance learning by treating instances as non-i.i.d. samples". Text Data for Multi-Instance Learning, 2009.

[2] Jaume Amores. Multiple instance classification: Review, taxonomy and comparative study. *Artificial Intelligence*, 201:81–105, 2013.

[3] Marti J Anderson. Distance-based tests for homogeneity of multivariate dispersions. *Biometrics*, 62(1):245–253, 2006.

[4] H Bunke and G Allermann. Inexact graph matching for structural pattern recognition. *Pattern Recognition Letters*, 1(4):245–253, 1983.

[5] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations, 2020.

[6] Veronika Cheplygina, David M.J. Tax, and Marco Loog. Multiple instance learning with bag dissimilarities. *Pattern Recognition*, 48(1):264–275, 2015.

[7] Luc Devroye and Gábor Lugosi. *The Kernel Density Estimate*, pages 79–97. Springer New York, New York, NY, 2001.

[8] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, volume 96, pages 226–231, 1996.

[9] Thomas Gartner, Peter Flach, Adam Kowalczyk, and Alex Smola. Multi-instance kernels. *Proceedings of 19th International Conference on Machine Learning*, 11 2003.

[10] Hossein Gholamalinezhad and Hossein Khosravi. Pooling methods in deep neural networks, a review. *arXiv preprint arXiv:2009.07485*, 2020.

[11] Gongde Guo, Hui Wang, David Bell, Yaxin Bi, and Kieran Greer. Knn model-based approach in classification. In Robert Meersman, Zahir Tari, and Douglas C. Schmidt, editors, *On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE*, pages 986–996, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg.

[12] Corneliu Henegar, Karine Clément, and Jean-Daniel Zucker. Unsupervised multiple-instance learning for functional profiling of genomic data. In Johannes Fürnkranz, Tobias Scheffer, and Myra Spiliopoulou, editors, *Machine Learning: ECML 2006*, pages 186–197, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[13] Shiluo Huang, Zheng Liu, Wei Jin, and Ying Mu. Bag dissimilarity regularized multi-instance learning. *Pattern Recognition*, 126:108583, 2022.

[14] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift, 2015.

[15] Edward Loper and Steven Bird. Nltk: The natural language toolkit, 2002.

[16] Robert MacCallum. *Multidimensional Scaling*, pages 421–445. Springer US, Boston, MA, 1988.

[17] Fragkiskos D Malliaros and Michalis Vazirgiannis. Clustering and community detection in directed networks: A survey. *Physics reports*, 533(4):95–142, 2013.

[18] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

[19] MIL data sets used in our 2002 NIPS by Columbia University. MIL datasets, 2002.

[20] Xi Peng, Yunfan Li, Ivor W. Tsang, Hongyuan Zhu, Jiancheng Lv, and Joey Tianyi Zhou. Xai beyond classification: Interpretable neural clustering. *Journal of Machine Learning Research*, 23(6):1–28, 2022.

[21] Maria L Rizzo and Gábor J Székely. Energy distance. *wiley interdisciplinary reviews: Computational statistics*, 8(1):27–38, 2016.

[22] Ludger Rüschendorf. The wasserstein distance and approximation theorems. *Probability Theory and Related Fields*, 70(1):117–129, 1985.

[23] Alberto Sanfeliu and King-Sun Fu. A distance measure between attributed relational graphs for pattern recognition. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-13(3):353–362, 1983.

[24] Thomas E. Tavolara, Metin N. Gurcan, and M. Khalid Khan Niazi. Contrastive multiple instance learning: An unsupervised framework for learning slide-level representations of whole slide histopathology images without labels. *Cancers*, 14(23), 2022.

[25] The data set has been used in: Z.-H. Zhou, K. Jiang, and M. Li. Multi-instance learning based web mining. Data for Multi-Instance Learning Based Web Index Recommendation, 2005.

[26] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

[27] A Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.

[28] Jun Wang and Jean-Daniel Zucker. Solving multiple-instance problem: A lazy learning approach. 2000.

[29] Zhenzhen Wang and Junsong Yuan. Unsupervised multiple-instance learning for instance search. In *2018 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6. IEEE, 2018.

[30] Muhammad Waqas, Syed Umaid Ahmed, Muhammad Atif Tahir, Jia Wu, and Rizwan Qureshi. Exploring multiple instance learning (mil): A brief survey. *Expert Systems with Applications*, 250:123893, 2024.

[31] Danyi Xiong, Ze Zhang, Tao Wang, and Xinlei Wang. A comparative study of multiple instance learning methods for cancer detection using t-cell receptor sequences. *Computational and Structural Biotechnology Journal*, 19:3255–3268, 2021.

[32] Matthew Zeiler. Adadelta: An adaptive learning rate method. 1212, 12 2012.

[33] Min-Ling Zhang and Zhi-Hua Zhou. Multi-instance clustering with applications to multi-instance prediction. *Applied intelligence*, 31:47–68, 2009.

[34] Zhi-Hua Zhou, Yu-Yin Sun, and Yu-Feng Li. Multi-instance learning by treating instances as non-i.i.d. samples. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, page 1249–1256, New York, NY, USA, 2009. Association for Computing Machinery.