CSE 847 Final Report

Discovering the underlying topological space of extracted features

Hamid Karimi
karimiha@msu.edu

Harrison LeFrois
lefroish@math.msu.edu

ACM Reference format:
Hamid Karimi and Harrison LeFrois. 2016. CSE 847 Final Report. In *Proceedings of The Best Conference, East Lansing, MI, 04/28/2017,* 10 pages.
DOI: 10.1145/nnnnnnn.nnnnnnn

1 PROBLEM DESCRIPTION

Manifold learning is a common approach to dimensionality reduction. The idea is to find an underlying space (a manifold) that actually describes the data in a lower dimension than the initial data representation. For our project, we have decided to examine what is happening in the hidden layers of neural networks. Apart from the simplest examples, the maps that a neural network is learning become complicated very quickly. We are particularly interested to see if it is possible to find an underlying topological space (potentially a manifold) that parametrizes our feature space for a layer in our network. In other words, what is the space that spans our selected features? Is it possible to relate the representation of our data at each layer to something geometric? Data analysis generally does not leverage the geometry and topology of the data [1], which is what we are aiming towards. It may not be possible to find an underlying manifold for our data, but parametrizing the feature space by a topological space can help us better understand the particular data set and what is happening in our network.

2 SURVEY AND RELATED WORK

Dimensionality reduction is a well-established and often used strategy to analyze high dimensional data and enables the user to visualize data that would otherwise be inaccessible. This is done by extracting features that best represent the data and mapping the data to a lower dimensional space based on these features, hopefully without too much loss of information.

Some common methods of dimensionality reduction and manifold learning are Principal Component Analysis (PCA), Linear Discriminant Analysis (LDA), Multidimensional Scaling (MDS), and ISOMAP. PCA, LDA, and MDS are linear approaches that are not very useful for non-linear structures. While ISOMAP is a non-linear approach, it fails to retrieve the underlying embedding in complicated structures. Nevertheless, deep neural networks have shown to be quite successful in extracting the underlying manifold of data feature space. For our project, we will utilize a Convolutional Neural Network (CNN) since the data we will be working with are images

This project is inspired by the work of Dr. José Perea in [2]. A motivating example in his work involved finding the underlying topological space of 7×7 pixel grey-scale images displaying a fixed-width line. Let $X \subset \mathbb{X}$ denote a sample of these images from the set $\mathbb{X}$ of all such images. Each image could be represented as a 7×7 matrix whose entries, consisting of -1 for black and 1 for white, sum to 0. By concatenating the columns of each matrix, we could view this data set X as a subset of $\mathbb{R}^{49}$. He noticed that despite being embedded in a 49 dimensional space, each image was governed by the distance of the line from the center and the angle of the line compared to the horizontal. Locally, $\mathbb{X}$ appears to be 2 dimensional. Using a combination of MDS and a strategy to discover the orientability of $\mathbb{X}$, Perea was able to conclude that X was actually sampled from the real projective plane $\mathbb{R}\mathbf{P}^2$. Each image in X was then able to be assigned a pair of coordinates in $\mathbb{R}\mathbf{P}^2$ by using an extension of PCA called Principal Projective Components. In this particular instance, the data was sampled from a manifold and the important features of each image, namely the angle and distance, provided some of the information needed to find the underlying topological space. The goal of this work, possibly in conjunction with algebro-topological tools, is to utilize machine learning to make similar discoveries.

3 TOPOLOGY BACKGROUND

The field of topology, an area of mathematics, is a way of studying shape. It differs from geometry in that topologists are concerned with the properties of a space that are invariant under continuous deformations as opposed to more rigid measurements and properties like angles. Intuitively, this means the properties that do not change when under stretching, shrinking, translation, etc. One is not allowed to cut, puncture, or glue the space. The typical example is that a coffee mug and donut are topologically equivalent. The defining property here is the hole in the middle of the donut which becomes the handle of the coffee mug. This provides an easy example, however, it is generally difficult to tell if two spaces are ”equivalent” (diffeomorphic, homeomorphic, homotopy equivalent, etc).

In topology, we assign invariants (such as numbers or algebraic objects like groups) to a space as a way to help distinguish different spaces from each other. Two spaces with differing invariants (e.g. Euler characteristic) are not equivalent, however just because two spaces share the same invariant does not mean they are necessarily equivalent. By construction, these invariants do not change when we make small deformations to our original space, for example denting a sphere. This makes topology particularly well-suited to applications in data analysis, where robustness to noise is desirable. However, there is not much that is topologically interesting or illuminating about a collection of points, the invariants we use are trivial (homology groups, homotopy groups, etc). It is only when we construct a more complex topological space that we are then able to use the tools from topology effectively to deduce properties of our data set such as shape.

The Best Conference, East Lansing, MI
2016. 978-x-xxxx-xxxx-x/YY/MM…$15.00
DOI: 10.1145/nnnnnnn.nnnnnnn

The topological tool we use to study our data sets in this project is called persistent homology [3]. Homology tells us about the number of "holes" in our space. For example, if X is a topological space, the zeroth homology group $H_0(X)$ gives the number of connected components, $H_1(X)$ gives the number of one-dimensional holes (like the hole in a donut), and $H_2(X)$ gives the number of holes, as in a cavity (like the interior of a beach ball). What persistent homology computes is how the homology groups of a space (in this case a simplicial complex constructed from our data) change as we vary a particular parameter. In order to compute the persistent homology of a data set, we need to first give the data a topological structure since the homology of a set of points tells us nothing. We do this by constructing what is called the Rips complex, which is done in the following way. Let D be our data set, $\epsilon > 0$, and $R_\epsilon(D)$ denote our Rips complex. Then we call $\{x_0, x_1, ..., x_k\} \subset D$ a set of vertices and say that this set spans a k-simplex $\sigma \in R_\epsilon(D)$ if the distance $d(x_i, x_j)$ is less than or equal to ϵ for every i and j. If $\epsilon < \epsilon'$, then we get an inclusion of the complexes $R_\epsilon(D) \longrightarrow R_{\epsilon'}(D)$. For more technical definitions and to see simplicial complexes in greater detail, please refer to [4].

From the inclusion map mentioned above, we get an induced map between the homology groups $H_i(R_\epsilon(D)) \longrightarrow H_i(R_{\epsilon'}(D))$, and this leads us to the persistent homology. As a technical side note, all of our homology groups have coefficients in a field, often $\mathbb{Z}/2\mathbb{Z}$ or $\mathbb{Z}/3\mathbb{Z}$. When viewed as a module over the coefficient ring, we see that our homology groups are actually now vector spaces with linear maps between them.

The persistent homology of our collection of complexes $\{R_\epsilon(D)\}_{\epsilon \geq 0}$ measures when a homological class (a connected piece in zeroth dimension, a loop in the first dimension, etc) is born and when it dies. The birth time is the ϵ for which the class first appears, and the death time is defined analogously. As an example, suppose that $D = \{(0,0), (0,1)\} \subset \mathbb{R}^2$. For each $\epsilon > 0$, we construct $R_\epsilon(D)$. Notice that $R_\epsilon(D)$ only changes when $\epsilon = 1$. For $\epsilon < 1$, we see that $R_\epsilon(D)$ has two vertices and no other simplices. When $\epsilon = 1$, we see that $d((0,0),(0,1)) = 1$, so $\{(0,0),(0,1)\}$ spans a 1-simplex, i.e. we draw an edge between those two points. Now there is only one connected component instead of two. So one of the zeroth homology classes no longer exists (it died!). The birth and death of homology classes can be represented by intervals, and these can be drawn as barcodes. Generally, the longer intervals in the barcode represent the topological features which persist longest, and are hence thought to represent the salient topological features and shape of our data. All of our homology computations are done using the Javaplex package for Matlab [5].

4 MOTIVATION

Large amounts of data are needed to train a neural network, so we need a large data set before doing anything else. We have decided to work with the CASIA-Webface data set. This data set contains close to 494,414 face images of 10575 subjects, which is larger than any idea for data we have had thus far. We will examine the feature space associated to this data set from a Convolutional Neural Network (CNN), the motivation for which is explained in this section.

The motivation for our idea of investigating the feature space of images of faces came from the persistent homology of images of a rubber duck.

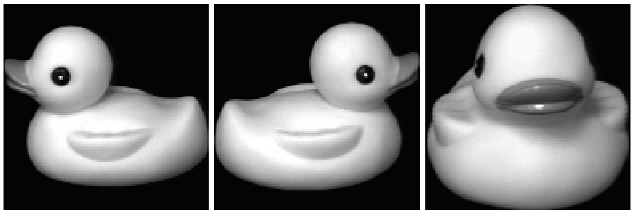


Figure 1: Rotations of a rubber duck

The Columbia Object Image Library (COIL) contains a collection, COIL-20, of gray-scale images of 20 everyday objects [6]. There are 72 images of each object, where each image differs from the next by rotating the object five degrees. Images can be analyzed by viewing them as matrices whose entries are pixel values (integers between 0 and 255). To analyze the images of the rubber duck, each 128 × 128 image is represented as a 128 × 128 matrix. We then take the first column of this matrix and concatenate each subsequent column to create a column vector in $\mathbb{R}^{16384}$. By repeating this process with each image, we get 72 vectors (or data points) in $\mathbb{R}^{16384}$ associated to each object. This point cloud can then be used to construct the Rips complex and compute the persistent homology.

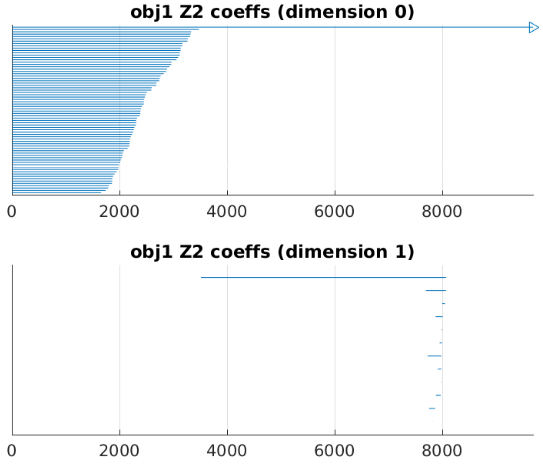


Figure 2: The barcode associated to the rubber duck images. The top is the zeroth homology and bottom is the first homology.

From the barcode shown, particularly the first homology, we see that there is one prominent 1-cycle or loop in our data. Despite living in a very high dimensional space, our data points roughly form a circle, as far as the topological perspective is concerned. Note that in topology, a square is viewed as identical to a circle because one can continuously deform the square to make it look like a circle (and vice versa). Continuous deformations can be thought of as wiggling and moving pieces of your space without tearing or cutting. Periodic behavior in data, i.e. starting at one point, changing locations in space then returning to the starting point, can be thought of as a loop. These loops then show up as a 1-cycle in the barcode associated to the data. When viewed from this perspective it is not surprising that we have a 1-cycle appearing in our data because we are starting at one angle of rotation, rotating the duck 360°, hence returning to our starting point.

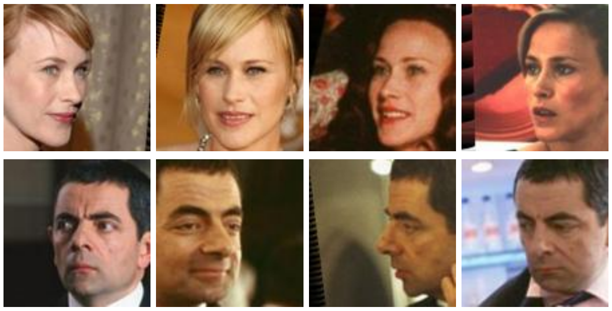


Figure 3: Examples from the CASIA-Webface data set

Our original goal was to look at the feature space from a hidden layer of a CNN to see if we could find an underlying topological space that would parametrize our feature space. In this instance, our data roughly forms a circle in high dimensional space, so one would hope that the feature space would be parametrized by a circle as well. However, for each of the twenty objects we only have 72 data points. This is not remotely enough data to train a network. It was then brought to our attention that there is a large database of faces called CASIA-Webface [7]. Since many of these photos show different angles of the same person as seen in Figure 3 (i.e. rotations of the person!), analyzing the feature space of this data space for different network architectures may provide some insights about the data set as a whole. We hope that these techniques and subsequent analysis will shed more light on how neural networks work.

5 DATA SETS

As a rule of thumb, the more training samples we have at our disposal, the better performance we can achieve. In addition, the more diversity and variability that exist in the samples, the more we can generalize the trained model. CASIA-Webface [7] enjoys both of these properties. It consists of 494,414 images of 10,575 subjects. The images are crawled out of profiles of celebrities from the IMDB website [1]. This data set is the largest publicly available face data set and the second largest data set after Facebook's private face data set, SFC [8]. CASIA-Webface is our **training set** for this project. In other words, we train our CNN network on the entire CASIA-webface data set.

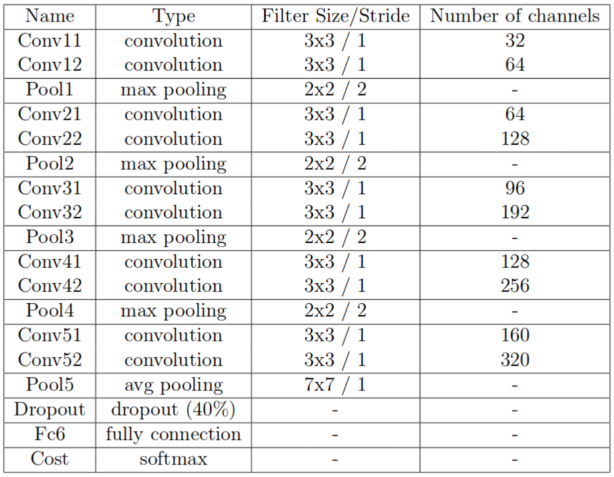
Face data sets used to be collected in controlled environments wherein images were of high quality and had the least distortion. Facial recognition algorithms on these data sets have already achieved excellent results. However, images in real-world situations are not that perfect. In order to have better performance in more practical scenarios, facial recognition has moved toward working on images generated in an uncontrolled manner. One of the popular unconstrained facial recognition benchmarks called LFW (Labeled Faces in the Wild) [9] was released in 2007. It consists of 5749 subjects crawled from Internet. We will use this data set for extracting features for our topological study.

6 PREPROCESSING

The first thing we need to do is align the images. We aligned the images utilizing the recently published method called Multitask Cascaded Convolutional Networks (MTCCN) [10]. We cropped the training images to 110×110 and the evaluation images to 160×160.

7 CONVOLUTIONAL NEURAL NETWORK

Convolutional Neural Networks (CNN) have shown to be successful in facial recognition and verification. Thanks to deep networks and larger data sets, we can achieve incredibly high accuracy in facial recognition [7] [11]. We use a deep CNN to train the model. Our base implementation for this project is from [11] [2], where the authors have implemented a basic training model as well as several deep CNN architectures. We implemented the CNN architecture in Fig 4 in that base code with some modifications. This simple model has been shown to achieve good accuracy on the CASIA-Webface data set [7]. The images are cropped to 100×100 and converted to grayscale. Images are read from disk in batches of 100 images. Each image is flipped along the width dimension (i.e., left to right). By flipping the images we double the size of our data set, which is crucial in preventing overfitting. In addition, by flipping we try contribute more in forming cyclic structure of the image samples.



Name	Type	Filter Size/Stride	Number of channels
Conv11	convolution	3x3 / 1	32
Conv12	convolution	3x3 / 1	64
Pool1	max pooling	2x2 / 2	-
Conv21	convolution	3x3 / 1	64
Conv22	convolution	3x3 / 1	128
Pool2	max pooling	2x2 / 2	-
Conv31	convolution	3x3 / 1	96
Conv32	convolution	3x3 / 1	192
Pool3	max pooling	2x2 / 2	-
Conv41	convolution	3x3 / 1	128
Conv42	convolution	3x3 / 1	256
Pool4	max pooling	2x2 / 2	-
Conv51	convolution	3x3 / 1	160
Conv52	convolution	3x3 / 1	320
Pool5	avg pooling	7x7 / 1	-
Dropout	dropout (40%)	-	-
Fc6	fully connection	-	-
Cost	softmax	-	-

Figure 4: CASIA CNN architecture [7]

8 PROPOSED LOSS FUNCTION

In addition to examining the embeddings of these images into the feature space and looking for an underlying topological space, we

[1] http://www.imdb.com

[2] Codes available from https://github.com/davidsandberg/facenet

Parameter	Value	Remarks
Simulator	Tensorflow 1.0.1	We installed our own version on HPCC
Number of GPUs	2	
Number of Nodes	1	
cores per node	10	
Entire simulation	10 hours	
Simulation epochs	136	
Alignment protocol	MTCNN [10]	
Input Image size	100 × 100 × 1	
Random crop	applied	
Flipping	applied	all images are flipped
Batch size	100 images	
Epoch size	1000 batches	
L2 parameter on CNN weights	0	due to augmentation less chance of overfitting
L2 parameter on fully connected layer	0.0005	
Feature size	120	last layer of CNN
Dropout	40%	applied on feature vector
Batch normalization	applied	On CNN and fully connected layer
Loss function	softmax	
Optimizer	ADAM	
Learning rate	dynamic in epochs	0:30 0.01 30:100 0.001 100:136 0.0001

Table 1: Simulation settings

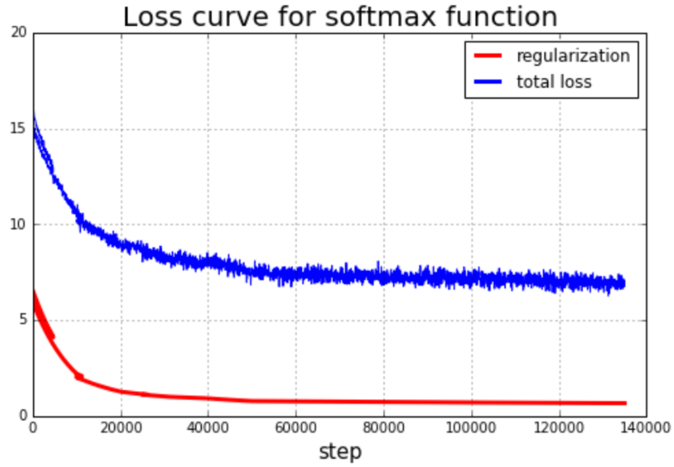


Figure 5: Loss curve of softmax

also wish to expand upon the success of this CNN by proposing a new loss function.

8.1 Related works

What we are basically interested in this project is a function f which embeds our data X in d-dimensional Euclidean space. As far as clustering performance is concerned, we are interested in an embedding in which samples from the same class are closer to each other while samples from different classes are far apart. To this end, some authors have defined a loss function for their deep network to achieve this aforementioned property (i.e., less inter-cluster similarity and less intra-cluster variability). In other words, $d(f(x_1), f(x_2))$ should be less than $d(f(x_1), f(x_3))$ where x_1 and x_2 belong to the same class while x_3 class is different and d is a distance function such Euclidean or cosine. Authors in [11] have proposed triple loss :

$$\sum_{i=1}^{n}[\| f(x_i^a) - f(x_i^p) \|_2^2 - \| f(x_i^a) - f(x_i^n) \|_2^2 + \alpha]_+ \tag{1}$$

x_i^a is an image of a person (anchor image), x_i^p is another image of the same person (positive), x_i^p is an image of different person (negative image) and α is a margin that is enforced between positive and negative pairs.

In [12] the authors have proposed the following loss function.

$$\min_P \lambda \frac{\sum_M d(a_i, b_j)}{|M|} + (1-\lambda)\frac{\sum_V [1 + d(a_i, b_j) - d(a_i, c_k)]_+}{|V|} \tag{2}$$

Where P is the learned mapping, y is class of a sample, $M = \{(i,j)|y_{a_i} = y_{b_j}\}$, and $V = \{(i,j,k)|y_{a_i} = y_{b_j}, y_{a_i} \neq y_{c_k}\}$. As noted in [12], the first term in Eq. 2 is a pull term that is minimized when points that belong to the same cluster are close, while the second term is a push term that is minimized when the points that belong to different classes are farther than points that belong to the same class. These two are controlled by the parameter $0 \leqslant \lambda \leqslant 1$.

Authors in [13] have proposed a center loss as follows:

$$\sum_{i=1}^{N} \| x_i - c_{y_i} \|_2^2 \tag{3}$$

where x is a sample and c_{y_i} is the centroid of y_i th class.

8.2 Motivation for new loss function

The objective of each clustering algorithm is to make elements with the same label closer together and those with differing labels farther apart. Therefore, the target embedding of the data should also follow a similar criteria i.e. similarly labelled data should be closer than differently labelled data. Previous loss functions somewhat follow these criteria but not fully. Triplet loss function in Eq 1 does not consider centroids of clusters while they could be helpful in making the clusters disjoint. The loss function proposed in 3 has employed a centroid of a cluster but not its relation to the other cluster centroids or samples. In Eq 2 distances among the centroids of different clusters are handled implicitly by pushing dissimilar instances away. This pushing is arbitrary and has no direction of convergence. If we incorporate distance from the centroid (of own cluster) to be one of the objectives, then it tries to push dissimilar objects to the centroid of its own cluster. Moreover, we also want distances between cluster centroids to have margin more than 1. Fig 6 shows the graphical representation of different components of the proposed loss function.

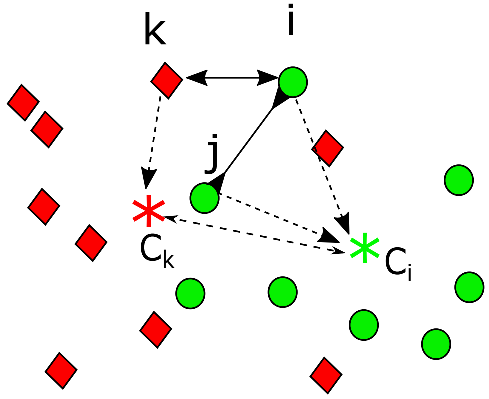


Figure 6: Graphical representation of loss function in the embedded space

The effectiveness of our new loss function has been shown for a toy example below. Data points are following a Swiss-roll distribution shown in Fig 7a, which contains 4 different classes. Fig 7b shows the initial embedding. Fig 7c shows the embedding achieved by applying Eq 2 loss function after training for 150 epochs . Fig 7d shows the embedding of the proposed loss function after 150 epochs.

8.3 Formulation

We represent the embedded space to be $\mathbf{F} = (f_1, \ldots, f_p)$. We now have two formulations of the loss function : Loss 1 or $L^1_{i,j,k}$ (previous) and Loss 2 or $L^2_{i,j,k}$ (Current). We select a triplet (i, j, k) in such a way that instances i and j have the same label (if labels exist) and i and k have different labels. We select the j-th instance to be the furthest and the k-th to be the nearest instance from i. In the case of more than two clusters, we select the different cluster to be random. These loss functions are formulated as follows. Here super script denotes instance i, j or k and subscript p denotes the p-th dimension of the output embedding.

$$\begin{aligned}
L^1_{i,j,k} =& \frac{1}{2}(\lambda \|f^i - f^j\|_2^2 + (1-\lambda)\max(1 + \|f^i - f^j\|_2^2 \\
&- \|f^i - f^k\|_2^2, 0)) \\
L^2_{i,j,k} =& \frac{1}{2}(\|f^i - f^j\|_2^2 + \max\left(1 + \|f^i - f^j\|_2^2\right. \\
&\left.- \|f^i - f^k\|_2^2, 0\right) + \|f^i - C^i\|_2^2 + \|f^j - C^i\|_2^2 + \\
&\|f^k - C^k\|_2^2 + \max\left(1 - \|C^i - C^k\|_2^2, 0\right))
\end{aligned} \tag{4}$$

C^i denotes the centroid of the elements (in embedded space) with the same label as the i-th instance. Assume that C^i is the set of those elements with the same label. The centroid is calculated by the equation below.

Note that the i-th and j-th instance have the same centroid C^i but the k-th instance has a different centroid C^k. The derivative with respect to f^i_p, f^j_p and f^k_p for the output neuron p is calculated below. The derivative $\frac{\partial L}{\partial f_p}$ is the sum of the three partial derivatives for each element in a triplet.

$$C^i_p = \frac{1}{|C^i|} \sum_{s \in C^i} f^s_p, \; \forall p \in \{1, \ldots, p\} \tag{5}$$

We define τ and ρ as below

$$\tau = \begin{cases} 1, & \text{if } \left(1 + \|f^i - f^j\|_2^2 - \|f^i - f^k\|_2^2, 0\right) > 0 \\ 0, & \text{otherwise} \end{cases}$$

$$\rho = \begin{cases} 1, & \text{if } \left(1 - \|C^i - C^k\|_2^2, 0\right) > 0 \\ 0, & \text{otherwise} \end{cases}$$

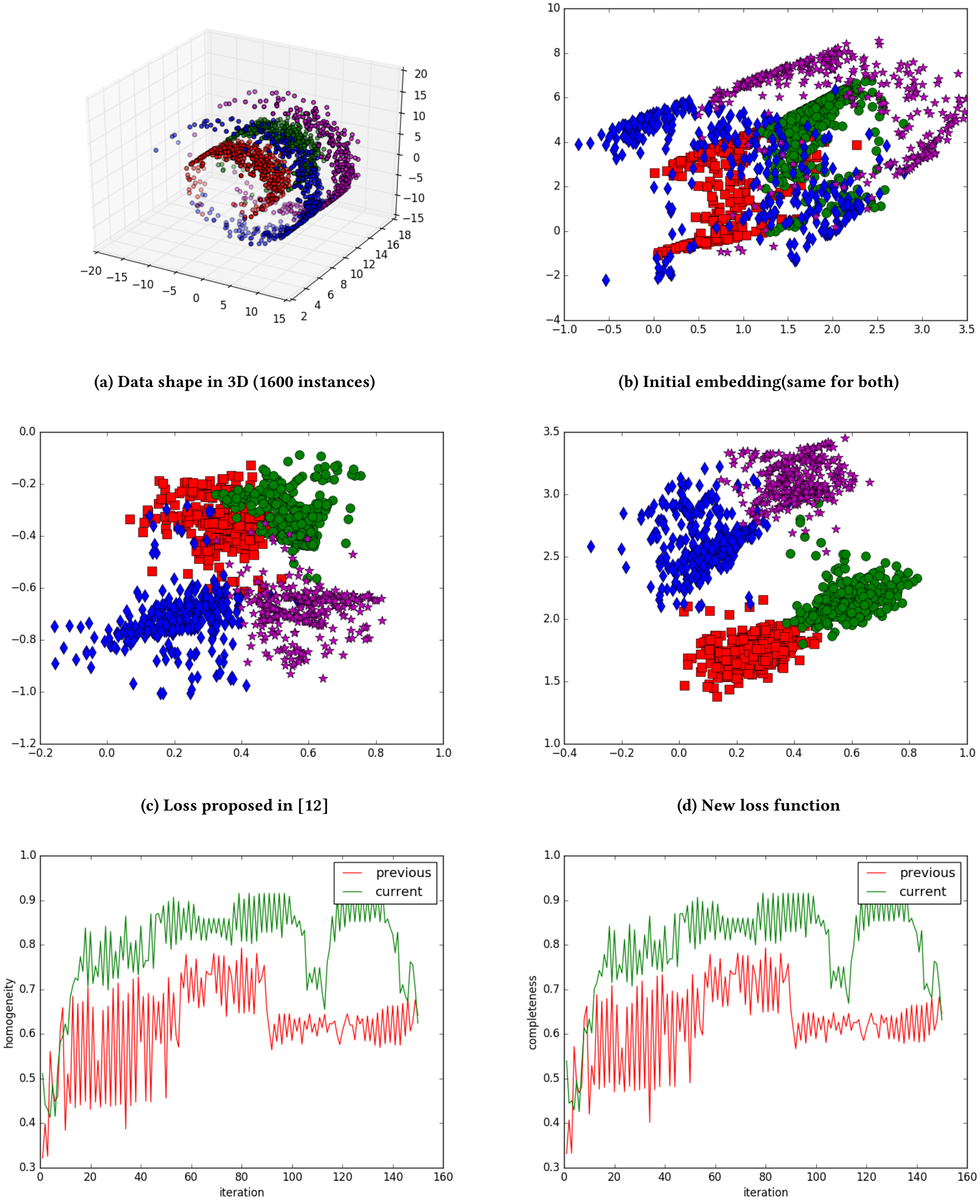
$$\begin{aligned}
\frac{\partial L}{\partial f_p} =& \frac{\partial L}{\partial f^i_p} + \frac{\partial L}{\partial f^j_p} + \frac{\partial L}{\partial f^k_p} \\
\frac{\partial L}{\partial f^i_p} =& (f^i - f^j) + (f^k - f^j)\tau + \left(f^i - C^i\right)\left(1 - \frac{1}{|C^i|}\right) \\
&- \frac{(f^j - C^i)}{|C^i|} + \frac{(C^k - C^i)}{|C^i|}\rho \\
\frac{\partial L}{\partial f^j_p} =& (f^j - f^i) + (f^j - f^i)\tau - \frac{(f^i - C^i)}{|C^i|} + \\
&\left(f^j - C^i\right)\left(1 - \frac{1}{|C^i|}\right) + \frac{(C^k - C^i)}{|C^i|}\rho \\
\frac{\partial L}{\partial f^k_p} =& (f^i - f^k)\tau - \frac{(f^k - C^k)}{|C^k|} + \frac{(C^i - C^k)}{|C^k|}\rho
\end{aligned} \tag{6}$$

9 RESULTS

For points of comparison, we have calculated the persistent homology for images with their normal features (i.e. in pixel space) and we also calculated the persistent homology of the embedded images that are the output of the CNN. The CNN produces an embedding in $\mathbb{R}^{128}$, which is of significantly lower dimension than the original ambient space of $\mathbb{R}^{12100}$.

From these examples of the resulting barcodes in 8, we see that the embeddings essentially preserves the general shape, at least from what we can tell in the persistent homology. We do not get the generators in the first homology (i.e. loops) that we saw in rotations of the rubber duck. This may be because there are not any photos of the backs of peoples' heads, and so the photos we do have are not enough to make "a full rotation" in the sense of the rubber duck example. In essence, what we are seeing is the result of there being two degrees of freedom in the movement of the person's head: up and down, left and right. This would give a shape, looking somewhat like a cross, that is contractible and hence would have trivial homology apart from H_0.

Our proposed loss function has shown promising results when tested on the Swiss-roll data. As can be seen in figure 7, under our new loss function we have greater separation of clusters. We also



(a) Data shape in 3D (1600 instances)

(b) Initial embedding(same for both)

(c) Loss proposed in [12]

(d) New loss function

(e) Homogeneity on test data)

(f) Completeness on test data

Figure 7: Performance of two embedding loss functions in Swiss-roll data

find that our new loss function provides greater homogeneity and completeness in our data as compared to the previous loss function. Despite these successes on the Swiss-roll data, utilizing this CNN construction with our improved loss function did not provide a topologically meaningful embedding, at least when viewed through the lens of persistent homology.

10 CONCLUSIONS

What we originally set out to do was to analyze the structure of the feature space for the CASIA-Webface data set, whose features were extracted utilizing our CNN architecture. From our analysis of the original data and the embedded data, we find that the persistent homology indicates a structure with no loops and hence trivial first homology. This was contrary to what we originally hypothesized, that we would find some kind of loop structure since there may have been photos of faces from enough angles to create that cyclic behavior in the pixel space. Calculating higher dimensional persistent homology was not possible, given the extremely high dimensional ambient space and the number of data points. When any attempt was made to calculate the second homology, we quickly ran out of memory except for on the smallest cases (less than 20 data points) - even when run on HPCC. The reason for this is that the computer needs to keep track of many more simplices as you increase the dimension.

We did find that our proposed loss function does provide measurable improvements over the original when tested on the Swiss-roll data. These are promising results and they inspire us to do more testing to make sure that this new loss function works well on larger and more complicated data sets (discussed further below).

11 FUTURE WORK

Our experience during this project has inspired both of us to continue research in similar directions. It is worth further exploration to see the effects that a new loss function can have on the quality of the embeddings that one can attain from a CNN. Based on our results with the new loss function on the Swiss-roll data, we would like to test the new versus the old loss function on larger data sets (possibly CASIA) to compare performance.

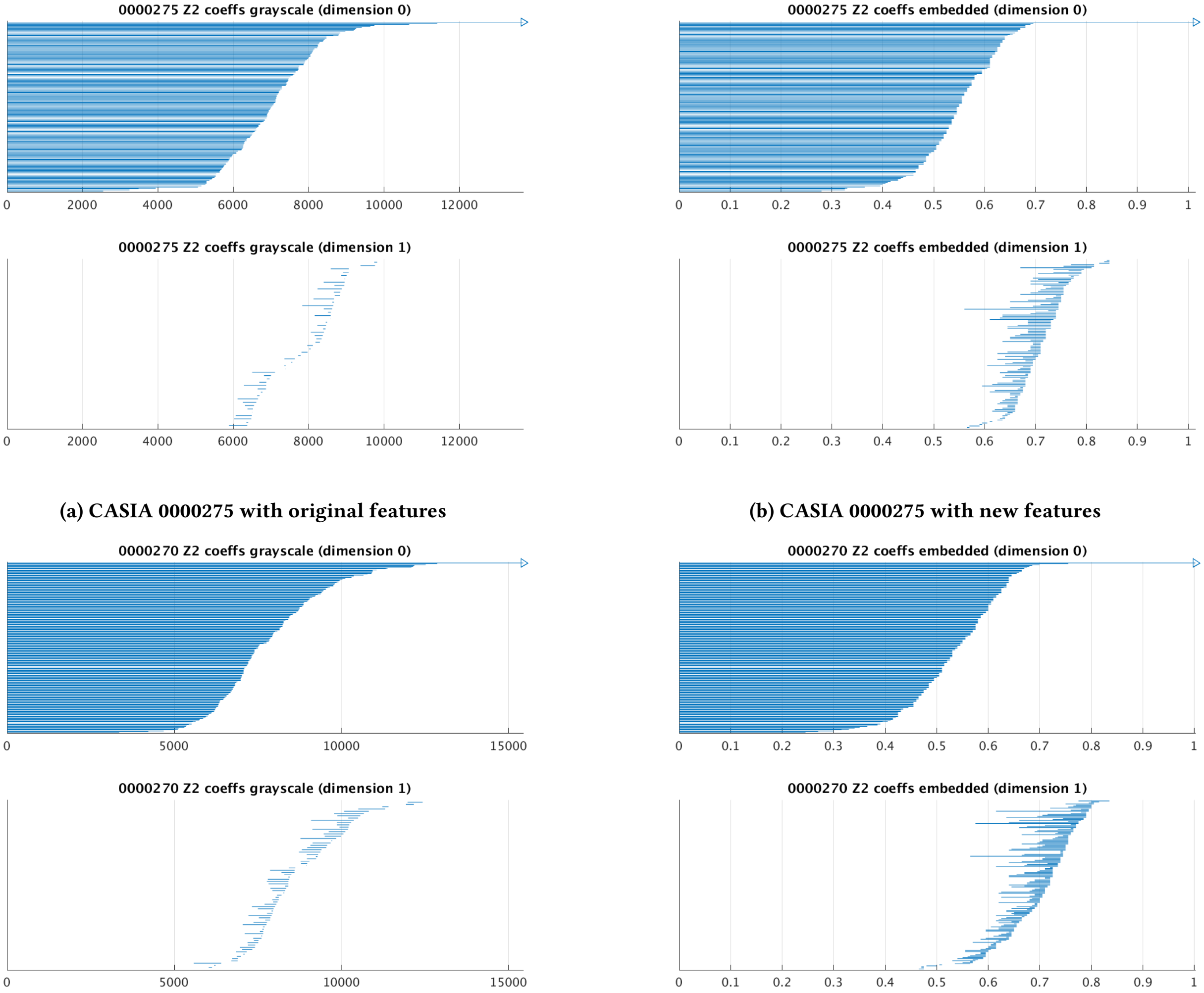
On the topology side, we believe that though the persistent homology yielded no interesting structure for the CASIA-Webface data set or the LFW data set, there may still be nontrivial structure underlying our feature space. One could apply the Mapper algorithm [14], which has been effective at detecting dendrite type structure (looks kind of like a starfish) [15]. The homology of a dendrite structure is trivial, so persistent homology would not be an effective tool at detecting this type of shape. We hypothesize that this is a viable shape of the data because of the plane of movement of the peoples' faces in the photos. Most of looking to the left/right, up/down, or straight ahead. So this up/down left/right motion would give some kind of shape like a cross, which would have trivial homology but who's shape would be detectable by the Mapper algorithm.

Another possible direction that future research could take would be to consider other data sets such as MNIST, natural images, speech data, etc. This project was exploratory in nature, we did not know what to expect. So it is possible that the CASIA-Webface and LFW data sets were not the best ones to start with as we begin an investigation to understand neural networks and look for structure in the feature space.

The final idea that we had to expand upon this work is to consider images in scale space rather than pixel space. In scale space, one takes an image and convolves it with a Gaussian distribution of varying widths [16]. The convolution with a Gaussian blurs the image to a certain extent (depending on the width), giving us a stack of images rather than just one. It is possible that there would be some interesting and useful topological structure underlying the feature space in this different scenario (scale space vs. pixel space) [17]. One could hope, for example, that a neural network would choose features that correspond to edges in this scale space, which can then possibly be parametrized and given coordinates on some underlying manifold as in [2] utilizing the projective coordinates if appropriate.

12 CONTRIBUTIONS

Both Hamid and Harrison contributed equally to the writing of the proposal, intermediate report, final report, and final presentation. Hamid contributed the CNN, new loss function, and the idea of using the CASIA-Webface data set. Harrison did the topology computations and contributed the topology expertise to analyze our results.



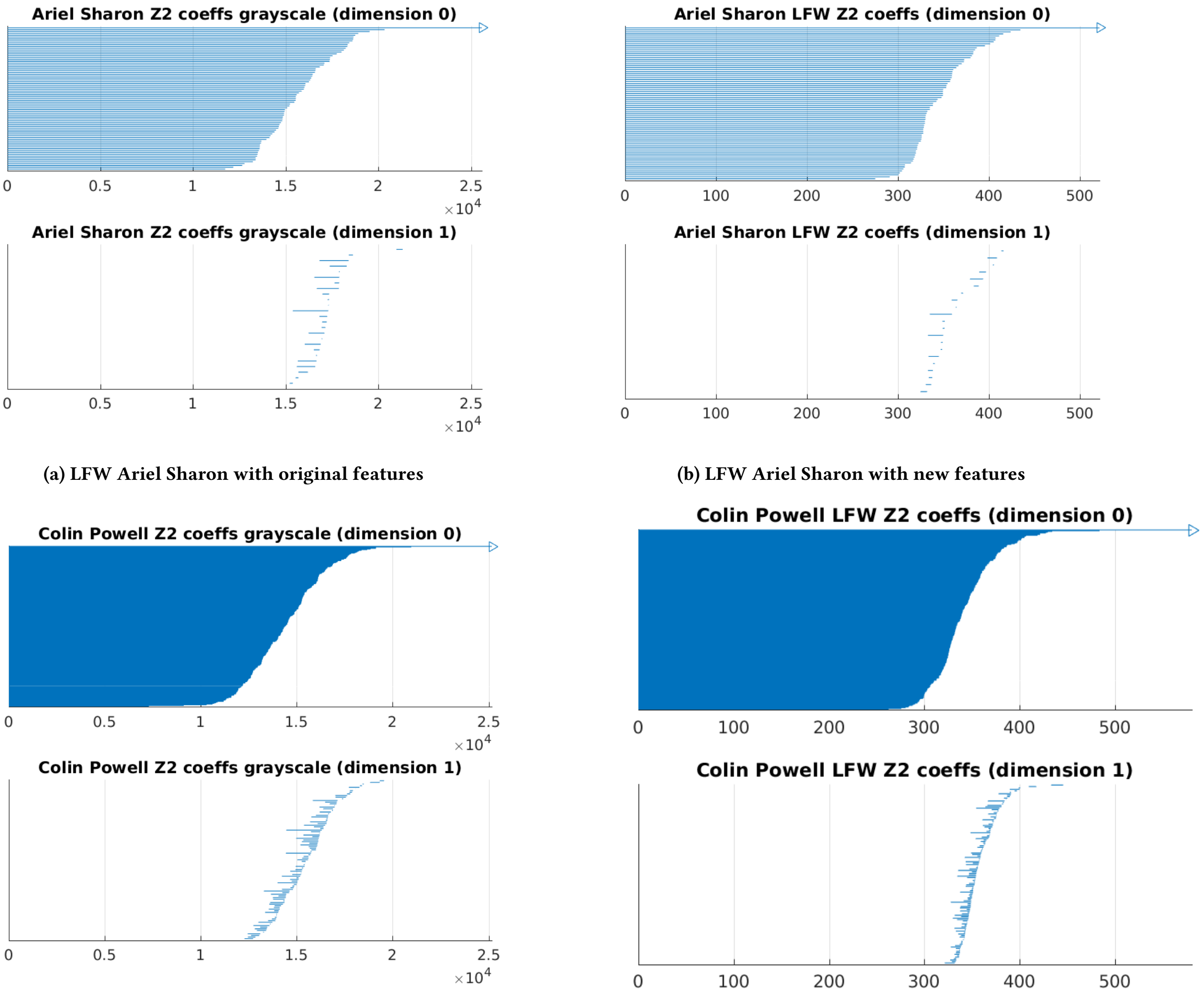
(a) CASIA 0000275 with original features

(b) CASIA 0000275 with new features

(c) CASIA 0000270 with original features

(d) CASIA 0000270 with new features

Figure 8: Example barcodes from CASIA data set



(a) LFW Ariel Sharon with original features

(b) LFW Ariel Sharon with new features

(c) LFW Colin Powell with original features

(d) LFW Colin Powell with new features

Figure 9: Example barcodes from LFW data set

REFERENCES

[1] Woong Bae, Jae Jun Yoo, and Jong Chul Ye. Beyond deep residual learning for image restoration: Persistent homology-guided manifold simplification. *CoRR*, abs/1611.06345, 2016. URL http://arxiv.org/abs/1611.06345.

[2] J. A. Perea. Multi-Scale Projective Coordinates via Persistent Cohomology of Sparse Filtrations. *ArXiv e-prints*, December 2016. URL http://adsabs.harvard.edu/abs/2016arXiv161202861P.

[3] Afra Zomorodian and Gunnar Carlsson. Computing persistent homology. *Discrete Comput. Geom.*, 33(2):249–274, February 2005. ISSN 0179-5376. doi: 10.1007/s00454-004-1146-y. URL http://dx.doi.org/10.1007/s00454-004-1146-y.

[4] James R. Munkres. *Elements of Algebraic Topology*. Addison-Wesley, 1993.

[5] Andrew Tausz, Mikael Vejdemo-Johansson, and Henry Adams. JavaPlex: A research software package for persistent (co)homology. In Han Hong and Chee Yap, editors, *Proceedings of ICMS 2014*, Lecture Notes in Computer Science 8592, pages 129–136, 2014. Software available at http://appliedtopology.github.io/javaplex/.

[6] S.A. Nene, S.K. Nayar, and H. Murase. Columbia Object Image Library (COIL-20), 1996. URL http://www.cs.columbia.edu/CAVE/software/softlib/coil-20.php.

[7] Dong Yi, Zhen Lei, Shengcai Liao, and Stan Z. Li. Learning face representation from scratch. *CoRR*, abs/1411.7923, 2014. URL http://arxiv.org/abs/1411.7923.

[8] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. Deepface: Closing the gap to human-level performance in face verification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1701–1708, 2014.

[9] Gary B Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical report.

[10] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao. Joint face detection and alignment using multitask cascaded convolutional networks. *IEEE Signal Processing Letters*, 23 (10):1499–1503, Oct 2016. ISSN 1070-9908. doi: 10.1109/LSP.2016.2603342.

[11] 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.

[12] S. Siena, V. N. Boddeti, and B. V. K. V. Kumar. Maximum-margin coupled mappings for cross-domain matching. In *2013 IEEE Sixth International Conference on Biometrics: Theory, Applications and Systems (BTAS)*, pages 1–8, Sept 2013. doi: 10.1109/BTAS.2013.6712686.

[13] Yandong Wen, Kaipeng Zhang, Zhifeng Li, and Yu Qiao. A discriminative feature learning approach for deep face recognition. In *European Conference on Computer Vision*, pages 499–515. Springer, 2016.

[14] Gurjeet Singh, Facundo Mémoli, and Gunnar E Carlsson. Topological methods for the analysis of high dimensional data sets and 3d object recognition. 2007.

[15] Monica Nicolau, Arnold J Levine, and Gunnar Carlsson. Topology based data analysis identifies a subgroup of breast cancers with a unique mutational profile and excellent survival. *Proceedings of the National Academy of Sciences*, 108(17): 7265–7270, 2011.

[16] Jan J Koenderink. The structure of images. *Biological cybernetics*, 50(5):363–370, 1984.

[17] Kim S Pedersen and Ann B Lee. Toward a full probability model of edges in natural images. In *European Conference on Computer Vision*, pages 328–342. Springer, 2002.