Vision, Modeling, and Visualization (2017)
M. Hullin, R. Klein, T. Schultz, A. Yao (Eds.)

Improved Image Classification using Topological Persistence

Tamal Dey[1,2] and Sayan Mandal[1,3] and William Varcho[1,4]

[1]Department of Computer Science and Engineering, The Ohio State University, Ohio, USA
{dey.8[2],mandal.25[3],varcho.5[4]}@osu.edu

Abstract

Image classification has been a topic of interest for many years. With the advent of Deep Learning, impressive progress has been made on the task, resulting in quite accurate classification. Our work focuses on improving modern image classification techniques by considering topological features as well. We show that incorporating this information allows our models to improve the accuracy, precision and recall on test data, thus providing evidence that topological signatures can be leveraged for enhancing some of the state-of-the art applications in computer vision.

1. Introduction

Image classification has been a buzzing topic of interest for researchers in both computer vision and machine learning for quite some time. Affordable high-quality cameras and online social media platforms have provided the internet with billions of images, most of which are raw and unclassified. Recent developments of new techniques for classification have shown very promising results, even on large datasets, such as ImageNet [DDS*09]. In this paper, we intend to investigate if topological signatures arising from topological persistence can provide additional information that could be used in the classification of images.

Use of topological signatures based on persistent homology [HH10] has been a topic of interest for the emerging area of topological data analysis. The prime advantages of using these topological signatures are their robustness and scale invariance: first, they are global and thus more resilient to local perturbations; second, the computations do not depend on a particular scale of the data. The idea of persistent homology has found applications in robotics [DL13], sensor networks [DSG07, GM05], and medical imaging [TMB14, CBK09], amongst others. In computer vision, it has been used for image segmentation [Kur15] and shape characterization [BEK10] as well. However, these works did not include machine leaning features for classification. We use topological persistence for providing some added signatures to the different classes of images on top of the traditional machine learning classifiers that use kernel methods or neural networks. We illustrate how these signatures can be used as additional features in the training process.

A main difficulty in using topological signatures for image classifications is that the current state-of-the-art techniques for computing such signatures for a large set of images do not scale up appropriately. Attempting to compute the persistent homology of images, which tend to live in high-dimensional spaces, is a road block in applying the topological data analysis techniques to image classifications. Several methods with associated software have been reported in the literature for computing persistent homology signatures. Among them, PHAT toolbox based on several efficient matrix reduction strategies of Bauer et al. [BKRW17] and GUDHI [The15] library based on compression techniques have been popular because of their space and time efficiencies. A recent software called SimBa [DSW16] has been shown to work faster, specially for data sets in dimensions higher than three, a case we are interested in. Even this method is not completely satisfactory for our purpose. For an image with dimension 200x200 pixels, persistence computation takes about 9.6s per image. Now imagine having to do this for several thousands of images which is the typical size of a database.

To work around this bottleneck, we take a different approach for computing a topological signature. In traditional persistence computations, a growing sequence of triangulations, simplicial complexes in general are involved. In our approach, we collapse an original complex, thereby reducing size instead of increasing it successively. This provides significant time improvements, which proves essential for computing the topological signatures for large training and test data sets. For the same 200x200 pixel image referenced earlier, our new method computes these signatures in only 1.18s.

Several pieces of novel work on image classification have been reported over the past few years. Firstly, there have been some experiments on the improvement of the Fisher kernel for image classification. The work by Perronnin et.al [PSM10] using normalisation and spatial pyramids has shown an improvement in accuracy. The work by Kobayashi [Kob14] uses the Dirichlet Fisher kernel for transforming the histogram feature vector. The primary use of Fisher-Vector for image classification was in [SPMV13]. Fisher-Vector has been used in Deep Neural Networks as well [SVZ13], by stacking it in multiple layers. Another more recent work by Perronin et al. have used Fisher-Vectors with convolutional Neural Networks, through the use of a hybrid architecture [PL15].

DOI: 10.2312/vmv.20171272

[OVS13] uses the same technique for action recognition in video frames, which is an advanced task of classification as well. Additional recent work on classification includes [YYGH09, WYY*10], which uses the Spatial Pyramid Matching approach based on the Bag of Features model, and has shown promising results. [DDJ*14] uses a label-relation graph for multiclass classification.

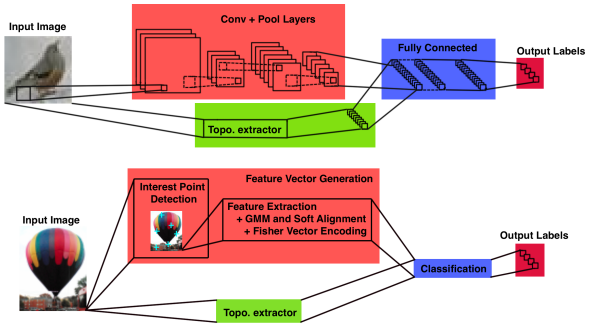


Figure 1: *Top: The topological features act as inputs to the fully connected part of our modified convolutional neural network. Bottom: Using modified Fisher Vector on SIFT along with topological features for training.*

For image classification with machine learning techniques, we investigate both feature vector based supervised classification and neural network based classification; see Figure 1 for a schematic diagram. For the first method, we use one of the state-of-the-art techniques using Fisher-Vector encoding [SPMV13] to generate the feature vectors. Classifications using Convolutional Neural Networks have been tested with another state-of-the-art model; AlexNet [KSH12]. Classification using Fisher-Vector has an accuracy of 59.7% on the Caltech-256 dataset using SIFT as feature for 60 or more training samples. We have reproduced the experiments with and without our topological features to get an improvement for the latter. This trend is also evident in modifying AlexNet (which has a precision of 83.2%) and evaluating on the CIFAR-10 image data set, where we found consistent improvements in model precision when including topological features.

We compute the topological features for each image and append this data with the feature vector obtained from the traditional method to classify the images. While there may be improvements in classifiers and methods for augmenting features into them, we claim that even naively including the topological features adds additional relevant information about the image which can be utilized by the network in making more accurate classifications. Since traditional feature extractors rely on geometric and image processing properties, such as gradient, orientation of sub-pixels, statistical distribution of colors or the learned features found in Convolutional Neural Networks (CNNs), topological features are lost in this process. By reintroducing these features, we add additional relevant information which can be utilized by existing classification techniques. Our entire technique has been illustrated in a video which is available at https://youtu.be/hq4DYse2c-Y.

2. Topological signature via persistence

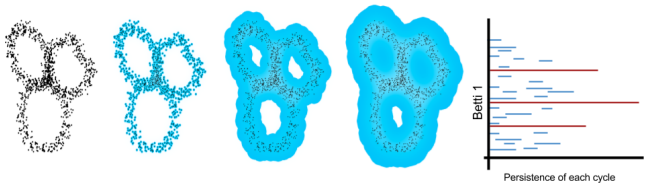
We use the theory of persistent homology to generate feature vectors for a point cloud. But, instead of the classical Rips complex filtrations, we use a hierarchy of collapsed simplicial complexes and a novel point selection strategy to save time. The resulting topological features are added to the features used for traditional image classification techniques. We test our method using two frameworks: one uses the feature vector based supervised learning and the other one uses convolutional neural network based learning.

Feature Vector based Supervised learning: Given an image, the Fisher Vector Encoding [SPMV13] describes a set of low level features by its deviation from the Gaussian Mixture Model (GMM). It is known to improve over the Bag of Visualization technique [GCB04] because it takes into account the higher order statistics instead of feature count. Essentially, it encodes the mean and covariance deviation vectors per component of the GMM and each element of the local feature descriptors together. We use SIFT features [Low99] as the low level feature and compute its divergence from the GMM using Fisher Vector. This serves as our original feature vector for each image before adding the topological feature vector. The quality of classification depends on the number of SIFT interest points found in the image, as greater number of such points would improve the feature description of the image.

Convolutional Neural Network based Learning: Next, we consider Convolutional Neural Networks [GWK*15] as a more advanced classification model. These neural networks have found success by sharing weights among nodes in the early layers of images, which can lead to faster training and convergence. As an image is forward-propagated through the network, these shared weights act as convolutional filter on the image, allowing prominent features to be detected in a translation independent manner. Because of the precision that CNNs have acheived in image classification, we believe that modifying them to include topological features is a good testimony to their utility in image classification.

In the following section, we briefly discuss the relevant concepts of topological data analysis, primarily persistent homology signatures, which are used as features for the learning frameworks. Since it is an involved topic we refer the reader to [HH10, ELZ02] for a more in-depth discussion. We start by giving an intuitive notion of persistence and then follow with key concepts used in our new algorithm for computing a topological signature.

Figure 2: *Tracing persistence of a point cloud in $\mathbb{R}^2$ with corresponding barcode*



Persistence of point cloud data

Consider a point cloud $\mathbb{P} \subset \mathbb{R}^2$, comprising of points sampled from some base shape having three loops as shown in Figure 2. We can

imagine starting to grow balls of increasing radius, r, around each point sample, and observe the behavior of the union of all these balls as r increases. At some radius r_1, we notice that the three holes in our original curve are accurately represented in the union of balls. However at a larger radius the two smaller loops become filled, and at an even bigger radius the largest loop fills in as well. Keeping track of which holes exist in this union of balls allows us to know how long the hole *'persists'*. For features that are more prominent we expect them to persist for longer periods of increasing r (e.g. the larger loop in Figure 2 persists longer than the two smaller ones). This is the basic intuition of topological persistence. A result of the topological persistence computation is a set of *birth-death* pairs of homology cycle classes (represented by cycles–loops here) that indicate when a class is born and when it dies. The birth and death points are a pair of real numbers indicating the time (corresponding to the radius value) at which these events occur. These pair of numbers are visualized by a set of points in $\mathbb{R}^2$ called the persistence diagram [HH10], or by a collection of line segments called the barcodes that stretch horizontally from the birth value to the death value as shown in Figure 5. These birth-death pairs can be considered as topological features (signatures) for the point cloud. For computational purposes, the growing sequence of union of balls is converted to a growing sequence of triangulations, simplicial complexes in general, called a *filtration*.

We carry out a similar process for images by transforming each pixel to a point $p \in \mathbb{R}^5$ by taking the RGB intensity values as well as its (x,y) coordinates. The theory of persistent homology tracks the birth and death of the cycles. The zero-dimensional homology, which accounts for number of connected components is trivial in this case. Hence, we ignore the persistence for zero-dimensional homology. For ease of computation we compute persistence only for one dimensional homology, which keeps track of the 1-cycles or loops in $\mathbb{R}^5$. This allows us to balance the necessity of getting relevant topological information against the increased computation time required for generating high-dimensional homological features.

2.1. Topological signature by successive collapses

Traditionally, topological persistence for a point cloud is computed via a particular filtration called the *Vietoris-Rips* (VR) filtration. Given a point set, the VR-complex with an input parameter α connects two points if they have distance less than α and then add any simplex (triangles, tetrahedra etc.) whose all edges have been selected. So, for different values of α, we get different mappings of the point set to simplicial complexes. As the value of α increases, a sequence of nested simplicial complexes is generated through which we track the persistent homology classes represented by cycles. For our case, this essentially determines how long classes of different cycles *'persist'* as α changes, thereby generating a signature for each image represented by the point cloud.

The problem with VR-complex, however, is that, since we connect every pair of points within distance α to form a higher dimensional simplex, there is a steep rise in size, which is more so for points living in a high dimension such as $\mathbb{R}^5$. Hence, we sparsify the point cloud by taking a δ-sparse, δ-net subsample on it, which means that for each point in the initial cloud, we have a point at a distance δ in the subsample and that no two points are more than δ distance close to each other in the sparser cloud. We then build a complex by joining certain number of nearest neighbours in this subset point cloud and bringing in higher dimensional simplices (only up to triangles for one dimensional homology cycles) satisfying certain conditions as described in [DFW13]. The complex thus built is called the graph induced complex ([DFW13]). This initial complex is reduced in size by successive edge collapses that collapses vertices as well; see Figure 4 and also Section C. In effect, it generates a sequence of simplicial complexes where successive complexes do not nest, but are connected with simplicial maps (these are maps that extend vertex maps to simplices, see [Mun84] for details). Algorithms for computing persistence under simplicial maps have been presented in [DFW12], and the authors have announced a software (Simpers) for it, which we use for our purpose. In order to choose a subsample very fast that respects the local density, we use the Morton Ordering [LSP*12] of the point cloud; see Section B. The Morton Ordering provides a total ordering on points $\in \mathbb{Z}^d$ where points close in the ordering are usually close in space, thus respecting spatial density(see Figure 3). Our data is sparsified by removing every n^{th} point from the current Morton ordering, and then repeating the process until there are less than n points remaining for a chosen n. Note that there are other algorithms which can be used for this purpose, such as implementing a k-means clustering with n-clusters and choosing the center of each cluster for removal. However, the Morton Ordering is very fast as it is based on bit operations, hence we inculcate it in our algorithm.

Figure 3: *Visualization of a Morton Ordering of point cloud data. Points with similar hue are close in the total ordering.*

It should be noted that in most cases, datasets often have real-valued data instead of integer values required for the Morton Ordering. To overcome this, we apply a basic scaling to the data as a preprocessing step, and then consider the closest integer point $\in \mathbb{Z}^d$ when determining the ordering. Finally, we compute the persistence of this sequence of collapse operations (simplicial maps) connecting successive complexes using the software Simpers [DFW12]. The details (along with pseudocode for our algorithm) are given in Section A.

To illustrate the speed-up we gain by our collapse based persistence computation with Morton Ordering, we report its running time on several datasets ranging from 3D data as geometric meshes to high dimensional data embedded in dimension as high as 150-dim (Table 1). The time taken by the algorithm when run on a random sample image from the Caltech-256 and MNIST datasets has also been included. We compare the speed of computation with SimBa [DSW16]. Since the authors of [DSW16] already showed that it generates results faster than existing techniques, beating its speed indicates the superiority of our approach. For this comparison, we only compute persistence up to the one-dimensional ho-

Table 1: *Comparison of time required to compute one-dimensional homology against SimBa*

Data	#points	Dim	SimBa	Our-Algo
Kitten	90120	3	35.72	19.05
PrCl-s	15000	18	94.13	28.17
PrCl-l	15000	25	254.37	47.12
Surv-s	252996	3	469.40	165.28
Surv-l	252996	150	1696.59	294.6
Caltech-256	10786	5	8.38	2.27
MNIST	2786	5	1.86	0.56

mology. While comparing our technique with SimBa , we retain the default parameter values suggested in the software manual.

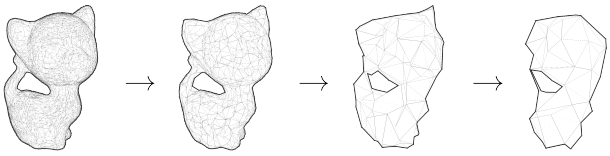


Figure 4: *A visual example of the sequence generated by subsampling a point set via the Morton Ordering*

We test the speed of our method for computing topological signature on several datasets having dimensions much larger than three in Table 1. The **PrCl-s** and **PrCl-l** are the datasets containing Primary Circles formed from natural images [AC09]. In PrCl-s, each point corresponds to a 5×5 image patch whereas in PrCl-l, they are of size 7×7. We run our algorithm on the **Adult** data [Koh96] obtained from the UCI Machine Learning Repository. This is a 14-dim classification data used for determining whether the average salary in the US exceeds 50k. We also experimented on the **Surviving protein** dataset [HPR*14]. This includes 252996 protein conformations, each being considered as a point in $\mathbb{R}^{150}$. We generate a scaled down version of this dataset as well by reducing the dimension to $\mathbb{R}^3$ using PCA, and testing on it.

As is evident from Table 1, our algorithm performs much faster, especially in high dimensions. Since we avoid simplex insertions of the classical inclusion based persistence computations, we are able to yield a significant speed-up.

3. Feature Vector Generation

Now we describe the method we use to incorporate the topological signatures as features for image classification.

3.1. Persistence as Feature Vector

Given an image I, we use the function $f : I \to \mathbb{R}^5$ mapping the RGB intensity of the pixel with coordinates $(x,y) \in I$ to a point

$$(\frac{r-\mu_r}{\sigma_r}, \frac{g-\mu_g}{\sigma_g}, \frac{b-\mu_b}{\sigma_b}, x-\bar{x}, y-\bar{y})$$

in the point cloud $P \in \mathbb{R}^5$. (Here μ_i and σ_i refer to the mean and standard deviation of the intensity channel i which can be red, green and blue respectively. Similarly, $\bar{x},\bar{y}$ are the corresponding mean.). The color of images which varies from $0-255$ and the size of images typically 200×200 depending on the dataset are essentially normalised using this process. We apply this mapping to all pixels in the input image in order to obtain an initial point set P on which the algorithm in section A operates. This algorithm computes the barcodes denoting the birth-death value for each cycle (as described in section 2). Typically, cycles with short barcode length correspond to intermediate/insignificant cycles or noise. So, to find the cycles which persist longer, we sort the barcodes wrt their lengths and find the largest difference in length between two consecutive barcodes. The (death-birth) value for every barcode above this gap is taken as our feature vector. Therefore, if there are *'m'* barcodes above the widest gap of the barcodes for an image, l_i denoting length of the i_{th} barcode, we take the length of the top *'m'* barcodes $(l_1, l_2, ..., l_m)$ as our feature vector. This m-length vector is added to the feature vector obtained from the traditional machine learning approach and used for training and testing. The barcode of a sample image from Caltech-256 is given in Figure 7 with the bottom 6 lines in blue forming $(l_1, l_2, ..., l_6)$. We note that, one may compute the feature vectors from our topological signatures using the methods proposed in [Bub15] [KHN*15]. We adopt the simple approach as described here because of the consideration of speed and simplicity.

3.2. Choosing Parameters

Next, we discuss how we choose the different parameters to generate persistence for the images. We need to tune two parameters, the first being the value k which are the number of nearest neighbors each node connected to for the δ-sparse, δ-sample complex we build (Section A Step 1). The second is the parameter n, where we choose the n^{th} point from V^i and collapse to its nearest neighbor for the simplicial mapping $f^i : \mathbb{C}^{i-1} \to \mathbb{C}^i$; see Section A. For choosing these parameters, we do an initial unsupervised clustering of the images based on the t-distributed stochastic neighbor embedding or t-SNE [vdMH08]. This technique provides visualizations of high-dimensional point clouds via a nonlinear embedding which attempts to place close points in the high dimensional space to close places in either $\mathbb{R}^2$ or $\mathbb{R}^3$. We take a subset of images from each of our datasets and generate the persistent signatures as described above. These signatures are embedded in $\mathbb{R}^2$ using tSNE. The effects are evident in Figure 6a, where we experiment on the MNIST digit dataset. MNIST digits clustered based on the computed barcodes, shows that digits with similar one-dimensional homology features are close together. Specifically, the digits 0-9 can be partitioned into 3 equivalence classes based on the number of holes in each digit, where

$$[0] = \{1,2,3,5\}, [1] = \{0,4,6,9\}, [2] = \{8\}$$

and this is reflected in 6a. We provide a bad clustering example on the CIFAR-10 dataset with values $k = 8$, we can see that t-SNE produced a noisy result, where not a lot of distinct clusters are formed. Tuning the parameter values of k to be $k = 15$, more distinct visible clusters were formed on the CIFAR-10 dataset as evident in 6c. We found this to indicate that we should modify

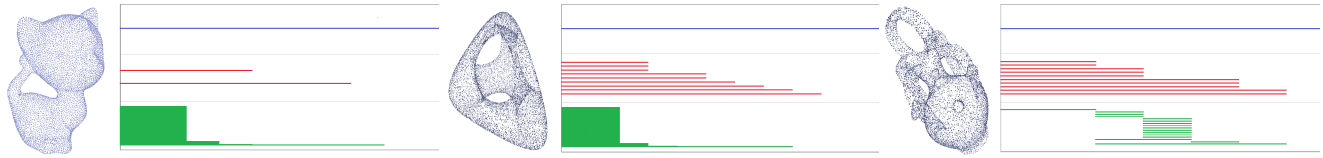
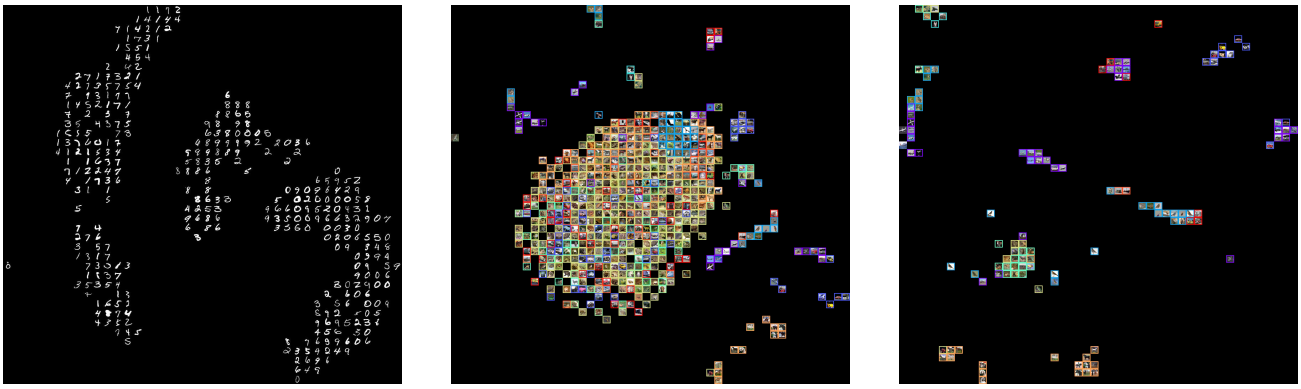


Figure 5: *Point cloud and persistence diagram of (a) Kitten (b) Genus (c) Botijo. The blue, red and green lines belong to the zero, one and two dimensional homology respectively.*



(a) *Good clustering on MNIST* **(b)** *Bad clustering on CIFAR-10* **(c)** *Better clustering on CIFAR-10*

Figure 6: *T-SNE was used as an aide in picking parameters for the computation of barcodes.*

Figure 7: *One dimensional homology for an image in Caltech-256. The bars in blue are above the widest gap and chosen as feature vector*

the aforementioned parameters. We repeat this experiment for k = $\{15,17,19,21,23,25\}^{th}$ nearest neighbors and value of n ranging from 8 to 20. Finally, we choose those values of k and n for which tSNE produces the most distinct clustering of images, namely k=17 and n=15.

4. Results

We have primarily focused on two classification frameworks. The first one is the traditional machine learning approach where we generate a feature set for each image; thereby training and testing on an optimal classifier. The second one is the convolutional neural network approach where we modify the final layers of the classifier to accommodate our new features. We have worked on several image datasets chiefly the CIFAR-10, the Caltech-256 and the MNIST hand written dataset. All our codes are freely available on our website: https://web.cse.ohio-state.edu/∼dey.8/imagePers/ImagePers.html .

Feature Vector based Supervised Learning

For classification, the number of features extracted for each image is generally quite large and in addition, can vary depending on the image. Because of this, the images are transformed into a fixed sized image representation using global image descriptor known as the Fisher Vector encoding [SPMV13]. We have assumed 16 Gaussian functions to form the GMM used as the generative model. We first compute the Hessian-Affine [MS02] extractor to generate interest points, thereafter calculate the SIFT [Low99] features of these interest points. If there are *'l'* interest points, this process generates a $128 \times l$ dimensional vector. This vector is transformed to a feature vector of length 4096×1 using the Fisher Vector encoding. Finally, we train an SVM model using the feature vector generated from each image and use it to classify the images.

We first tried to classify images from the CIFAR-10 dataset [Low09]. The dataset contains 6 batches of 1000 images with a total of 6000 images for each of the 10 output classes. Each individual image is extremely tiny with dimension 32x32. Since these images are so low in resolution, the number of interest points extracted is very small, and thus insufficient to characterize each image. There are ten classes of images in CIFAR-10, giving a baseline of 0.1 for precision and recall. For each class we trained on 4000 images and tested on the other 2000. We present the average result over all the 10 classes in Table 2.

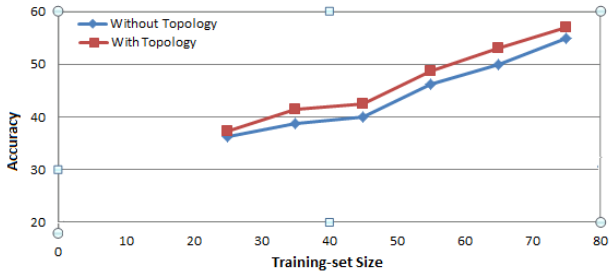
Next, we compute the persistence of each image in $\mathbb{R}^5$. The value of *'m'* as discussed in Section 3.1, computed as an average over all images, is 10. Hence we have appended the longest 10 barcodes to the signature described above, giving vector of length 5006×1. The

Dataset	#classes	Method	P - T	P + T	R - T	R + T
CIFAR-10	10	SIFT + Fisher Vector Encoding	23.63	28.24	23.56	31.16
CIFAR-10	10	AlexNet	83.2	83.8	98.25	98.15
Caltech-256	20	SIFT + Fisher Vector Encoding	57.50	59.50	51.50	64.00
MNIST	10	Deep MNIST SoftMax Regression	98.15	98.46	99.57	99.48

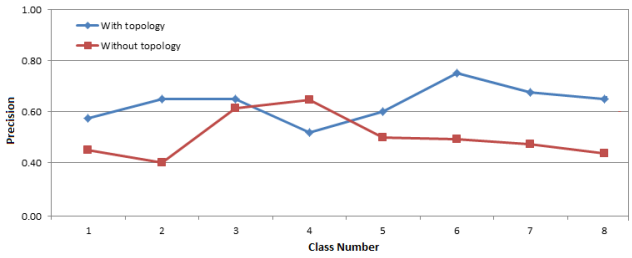
Table 2: *Precision(P) and Recall(R) for different Methods with and without using Topological features. (P-T and R-T) indicates Precision and Recall Without Topology whereas (P+T and R+T) indicates Precision and Recall With Topology*

precision and recall for this case increased significantly as noted in the Table 2.

The second dataset that we use is the Caltech-256 [GHP06]. The number of images in each category varies within a range between 80 and 827, the median being 100. Since the dataset is skewed, we use a subset of the dataset taking the first 20 classes having 100 or more images as a measure for image classification. We also fix the number of training and test images as 75 and 20 respectively, for each class to maintain consistency. We use the same technique as before, computing precision and recall for features with and without the persistence values. In this case, for the 20 classes, the precision and recall have improved significantly as well. The average accuracy for all the 20 datasets using SIFT with the Fisher Vector Encoding comes out to be 53.27%. However if we use the signature of our persistent homology, the accuracy increases to 56.74%. There is an increase in the average precision and recall for each class as well, as listed in Table 2. We also plot the accuracy varying the training set size from 25 to 75. The accuracy has a considerable increase using topological features in each case(see Figure 8a). Figure 8b plots the precision of a subset of eight classes on the dataset, and shows that the fluctuations in precision across different classes varies a lot for the fisher vector method, generating a result of 0% on two occasions, whereas the output using topology is reasonable for all the classes.
Two things are worth noting at this point. Fist, our algorithm runs faster than the state-of-the-art software SimBa used for generating topological features from point cloud data. In this regard, we have provided a quantitative comparison (Table 1). Second, we do not include the two dimensional homology features to save computational time. Therefore, we show the result that we would have obtained after including these topological features. The following table shows the accuracy, precision and recall of running SimBa and our algorithm (with 2D features) on 5 classes of the Caltech-256 dataset and on CIFAR-10. Note that, since we have taken a subset of the entire dataset, better accuracy on this subset does not necessarily mean better overall performance. Table 3 illustrates the similarity between the results obtained by considering SimBa and our algorithm with and without the two dimensional homological features. Interestingly, in some cases, considering only one dimensional features provides better accuracy.



(a) *Performance versus training size*



(b) *Fluctuations in precision across different class*

Figure 8: *Comparison of accuracy and precision with and without topological features*

	SimBa			Our Algo			Our Algo +β2		
	Acc	Pr	Re	Acc	Pr	Re	Acc	Pr	Re
I	54.6	61.9	63.9	58.9	65.6	64.8	59.2	60.1	64.8
II	19.6	21.2	33.0	21.3	28.2	31.4	22.4	28.2	31.2

Table 3: *Qualitative comparison of our algorithm with SimBa with and without 2-dimensional homological features. Acc: Accuracy, Pr: Precision, Re: Recall. I- 5 classes of CalTech256, II - CIFAR-10*

CNN and Deep Learning based Training

The second framework in our experimentation was based on the Convolutional and Deep Neural Network models. For these models we started by experimenting with the MNIST handwritten dataset [LBBH98]. We implement a straightforward Deep MNIST softmax regression network [SB98]. In a nutshell, the network comprises of two sets of convolutional layers followed by pooling and rectified linear activations, which is then input to a fully connected layer from which we determine the final output probabilities. After training, this model has a precision of 98.16%. However, including the topological features in the second to last layer of the fully connected network, we get a further improvement of 0.36% over the previous reported result. While this may not seem significant improvement, getting a slight improvement on a model which is already so accurate is encouraging. This trend in improvement continues for another dataset that we tried, namely the CIFAR-10

which we discussed earlier. While the SIFT feature vector is not a very good method to classify these tiny images, Deep Neural Networks have proven to be quite effective in such cases. A classic, successful model for this dataset is AlexNet [KSH12], which we modify slightly for our purpose. AlexNet starts with multiple convolutions, followed by normalization and the pooling layers and finally two fully connected layers with ReLu activation function; see [KSH12] for more details. Training the classifier with 50,000 iterations with a batch size of 64, we obtained a precision of 83.2%. On top of that, we added the topological features to the fully connected layer in the last stage of the model to get a 0.6% increase in precision. The details of all the results are included in Table 2.

5. Conclusion

In our work, we have accumulated ample evidence that topological features provide additional information for the classification of images. This is not surprising as most techniques rely on geometric, textural, or gradient based features for classification that do not necessarily capture topological features. However, computational time for generating the topological signatures remains to be the main bottleneck. Even though our faster technique has made progress in this direction, further research in needed to incorporate topological features for image classification for very large databases.

Appendix A: Algorithm for fast computation of topological signature

To compute the topological signature for an initial point cloud $P \subset \mathbb{R}^n$, we follow the procedure below:

1. Create a Nearest Neighbor graph on P by creating an edge between each point and its k-nearest neighbors.
2. Create a δ-sparse, δ-net subsample on P to form V^0
3. Build a graph induced complex $\mathbb{C}^0$ on V^0 using the algorithm described in ([DFW13])
4. Undergo a sequence of subsampling of the initial point set $V^0 \supset V^1 \supset ... \supset V^k$ based on the Morton Ordering discussed in B. (For every $V^i \supset V^{i+1}$, we remove every n^{th} point from the i^{th} sample to form V^{i+1}).
5. Generate a sequence of vertex maps $f^i : V^i \to V^{i+1}$, as defined in section C. This in turn generates a sequence of collapsed complexes: $\mathbb{C}^0, \mathbb{C}^1, ..., \mathbb{C}^n$. Each vertex map induces a simplicial map $f^i : \mathbb{C}^{i-1} \to \mathbb{C}^i$ that associate simplices in $\mathbb{C}^{i-1}$ to simplices in $\mathbb{C}^i$
6. Compute the persistence for the simplicial maps in the sequence $\mathbb{C}^0 \xrightarrow{f_1} \mathbb{C}^1 \xrightarrow{f_2} ... \xrightarrow{f_k} \mathbb{C}^k$ to generate the topological signature of the point set P.

Thus given a sequence of simplicial maps, we can compute persistence by sequence of collapse operations (induced by the maps) on our initial complex (which is described in section C).

Appendix B: Subsampling by Morton Coding

Now we will discuss our method for generating a sequence of batch collapses, given an initial simplicial complex $\mathbb{C}$. To do this, we first create a total ordering on our point set V^0. This ordering is explicitly defined by the Morton Ordering mapping $M : \mathbb{Z}^N \mapsto \mathbb{Z}$ such that

$$M(p) = \bigvee_{b=0}^{B} \bigvee_{i=0}^{N} x_2^{i,b} \ll N(b+1) - (i+1)$$

where $x_2^{i,b}$ denotes the b^{th} bit value of the binary representation of the i^{th} component of x, '$\vee$' denotes the binary or operation, and '$\ll$' denotes the left binary shift operation. This mapping is merely a bit interleaving of the different components of p. Applying M to every $p \in V^0$ yields a total ordering on our initial point set.

We can exploit the knowledge that points with similar Morton encoding are found in similar areas, to generate a new subset $V^1 \subset V^0$ that respects the underlying density of the initial point set.

First choose a value k such that $1 < k \leq \|V^0\|$. V^{i+1} can then be defined as

$$V^{i+1} = \{x_j \mid x_j \in V^i, j \not\equiv 0 \bmod k\}$$

Where x_j is the j^{th} vertex in the Morton Ordering of V^i. Following this approach, the process can be repeated to create a sequence of subsets

$$V^0 \supset V^1 \supset ... \supset V^n, \|V^n\| \leq k$$

Appendix C: Vertex Map and Collapses

This sequence of subsets of V^i allows us to define a simplicial map between any two adjacent subsets V^i and V^{i+1} by the following map.

$$f^i(p) = \begin{cases} p & \text{if } p \in V^{i+1} \\ \inf\limits_{v \in V^{i+1}} d(p,v) & \text{otherwise} \end{cases}$$

Essentially, each vertex in V^i is either chosen for the subsampling or mapped to its nearest neighbor in V^{i+1} .

This map on the vertices, then induces a map on all higher-order simplices of $\mathbb{C}$. More formally these maps are collapses of the simplicial complex $\mathbb{C}$.

$$\mathbb{C}^0 \xrightarrow{f_1} \mathbb{C}^1 \xrightarrow{f_2} ... \xrightarrow{f_k} \mathbb{C}^k$$

Given a sequence of simplicial maps $f_1, f_2, ... f_n$ between an initial simplicial complex $\mathbb{C}^0$ and a resulting simplicial complex $\mathbb{C}^n$, Authors in [DFW12] describes an annotation-based technique to generate the persistence of such a sequence. The authors use a set of elementary inclusions (not needed in our case) and collapses to break down the the simplicial maps into their fundamental elements. Using this, they derive the persistence of the simplicial maps. For our purposes we utilize this annotation based algorithm on the sequence of maps f^i described above.

References

[AC09] ADAMS H., CARLSSON G.: On the nonlinear statistics of range image patches. *SIAM Journal on Imaging Sciences* 2, 1 (2009), 110–117. 4

[BEK10] BENDICH P., EDELSBRUNNER H., KERBER M.: Computing robustness and persistence for images. *IEEE Transactions on Visualization and Computer Graphics 16*, 6 (Nov 2010), 1251–1260. 1

[BKRW17] BAUER U., KERBER M., REININGHAUS J., WAGNER H.: Phat - persistent homology algorithms toolbox. *J. Symb. Comput. 78*, C (Jan. 2017), 76–90. 1

[Bub15] BUBENIK P.: Statistical topological data analysis using persistence landscapes. *Journal of Machine Learning Research 16*, 1 (2015), 77–102. 4

[CBK09] CHUNG M., BUBENIK P., KIM P.: Persistence diagrams of cortical surface data. In *Information processing in medical imaging* (2009), Springer, pp. 386–397. 1

[DDJ*14] DENG J., DING N., JIA Y., FROME A., MURPHY K., BENGIO S., LI Y., NEVEN H., ADAM H.: *Large-Scale Object Classification Using Label Relation Graphs*. Springer International Publishing, 2014, pp. 48–64. 2

[DDS*09] DENG J., DONG W., SOCHER R., LI L.-J., LI K., FEI-FEI L.: ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09* (2009), vol. 115, pp. 211–252. 1

[DFW12] DEY T. K., FAN F., WANG Y.: Computing topological persistence for simplicial maps. *CoRR abs/1208.5018* (2012). 3, 7

[DFW13] DEY T. K., FAN F., WANG Y.: Graph induced complex on point data. In *Proceedings of the Twenty-ninth Annual Symposium on Computational Geometry* (New York, NY, USA, 2013), SoCG '13, ACM, pp. 107–116. 3, 7

[DL13] DIRAFZOON A., LOBATON E.: Topological mapping of unknown environments using an unlocalized robotic swarm. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems* (Nov 2013), pp. 5545–5551. 1

[DSG07] DE SILVA V., GHRIST R.: Coverage in sensor networks via persistent homology. *Algebraic & Geometric Topology 7*, 1 (2007), 339–358. 1

[DSW16] DEY T. K., SHI D., WANG Y.: Simba: An efficient tool for approximating rips-filtration persistence via simplicial batch-collapse. In *European Symposium on Algorithms* (2016), vol. 57. 1, 3

[ELZ02] EDELSBRUNNER H., LETSCHER D., ZOMORODIAN A.: Topological persistence and simplification. *Discrete Comput. Geom. 28*, 4 (Nov. 2002), 511–533. 2

[GCB04] GABRIELLA CSURKA CHRISTOPHER R. DANCE L. F. J. W., BRAY C.: Visual categorization with bags of keypoints. *ECCV SLCV Workshop* (2004), 1–22. 2

[GHP06] GRIFFIN G., HOLUB A., PERONA P.: Caltech-256 object category dataset. 6

[GM05] GHRIST R., MUHAMMAD A.: Coverage and hole-detection in sensor networks via homology. In *IPSN 2005. Fourth International Symposium on Information Processing in Sensor Networks, 2005.* (April 2005), pp. 254–260. 1

[GWK*15] GU J., WANG Z., KUEN J., MA L., SHAHROUDY A., SHUAI B., LIU T., WANG X., WANG G.: Recent advances in convolutional neural networks. *CoRR abs/1512.07108* (2015). 2

[HH10] HERBERT E., HARER J.: Computational topology: an introduction. 1, 2, 3

[HPR*14] HARVEY W., PARK I.-H., RÃIJBEL O., PASCUCCI V., BREMER P.-T., LI C., WANG Y.: A collaborative visual analytics suite for protein folding research. *Journal of Molecular Graphics and Modelling 53* (2014), 59 – 71. 4

[KHN*15] KWITT R., HUBER S., NIETHAMMER M., LIN W., BAUER U.: Statistical topological data analysis - a kernel perspective. In *Advances in Neural Information Processing Systems 28*, Cortes C., Lawrence N. D., Lee D. D., Sugiyama M., Garnett R., (Eds.). Curran Associates, Inc., 2015, pp. 3070–3078. 4

[Kob14] KOBAYASHI T.: Dirichlet-based histogram feature transform for image classification. In *CVPR14* (June 2014), pp. 3278–3285. 1

[Koh96] KOHAVI R.: Scaling up the accuracy of naive-bayes classifiers: A decision-tree hybrid. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining* (1996), KDD'96, AAAI Press, pp. 202–207. 4

[KSH12] KRIZHEVSKY A., SUTSKEVER I., HINTON G. E.: Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems* (2012), pp. 1097–1105. 2, 7

[Kur15] KURLIN V.: A fast persistence-based segmentation of noisy 2D clouds with provable guarantees. 3–12. 1

[LBBH98] LECUN Y., BOTTOU L., BENGIO Y., HAFFNER. P.: Gradient-based learning applied to document recognition. *Proceedings of the IEEE 86*, 11 (November 1998), 2278–2324. 6

[Low99] LOWE D. G.: Object recognition from local scale-invariant features. In *Proceedings of the Seventh IEEE International Conference on Computer Vision* (1999), vol. 2, pp. 1150–1157 vol.2. 2, 5

[Low09] LOWE D.: *Learning Multiple Layers of Features from Tiny Images*. P, April 2009. 5

[LSP*12] LI S., SIMONS L., PAKARAVOOR J. B., ABBASINEJAD F., OWENS J. D., AMENTA N.: kann on the gpu with shifted sorting. In *Proceedings of the Fourth ACM SIGGRAPH / Eurographics Conference on High-Performance Graphics* (Aire-la-Ville, Switzerland, Switzerland, 2012), EGGH-HPG'12, Eurographics Association, pp. 39–47. 3

[MS02] MIKOLAJCZYK K., SCHMID C.: An affine invariant interest point detector. 5

[Mun84] MUNKRES J. R.: *Elements of Algebraic Topology*, 1 ed. Perseus,Cambridge, Massachusetts, 1984, ch. 1. 3

[OVS13] ONEATA D., VERBEEK J., SCHMID C.: Action and event recognition with fisher vectors on a compact feature set. 1817–1824. 2

[PL15] PERRONNIN F., LARLUS D.: Fisher vectors meet neural networks: A hybrid classification architecture. pp. 3743–3752. 1

[PSM10] PERRONNIN F., SÁNCHEZ J., MENSINK T.: *Improving the Fisher Kernel for Large-Scale Image Classification*. Springer Berlin Heidelberg, Berlin, Heidelberg, September 2010, pp. 143–156. 1

[SB98] SUTTON R. S., BARTO A.: *Reinforcement Learning: An Introduction*. The MIT Press,Cambridge, Massachusetts, 1998. 6

[SPMV13] SANCHEZ J., PERRONNIN F., MENSINK T., VERBEEK J.: Image classification with the fisher vector: Theory and practice. *International Journal of Computer Vision,Springer* (May 2013), 222–245. 1, 2, 5

[SVZ13] SIMONYAN K., VEDALDI A., ZISSERMAN A.: Deep fisher networks for large-scale image classification. In *Advances in Neural Information Processing Systems 26*, Burges C. J. C., Bottou L., Welling M., Ghahramani Z., Weinberger K. Q., (Eds.). Curran Associates, Inc., 2013, pp. 163–171. 1

[The15] THE GUDHI PROJECT: *GUDHI User and Reference Manual*. GUDHI Editorial Board, 2015. 1

[TMB14] TURNER K., MUKHERJEE S., BOYER D. M.: Persistent homology transform for modeling shapes and surfaces. *Information and Inference* (2014), iau011. 1

[vdMH08] VAN DER MAATEN L., HINTON G.: Visualizing data using t-SNE, November 2008. 4

[WYY*10] WANG J., YANG J., YU K., LV F., HUANG T., GONG Y.: Locality-constrained linear coding for image classification. In *CVPR10* (June 2010), pp. 3360–3367. 2

[YYGH09] YANG J., YU K., GONG Y., HUANG T.: Linear spatial pyramid matching using sparse coding for image classification. In *CVPR09* (June 2009), pp. 1794–1801. 2