Topological Data Analysis for Classification of AI Generated Faces

Abstract

The proliferation of AI-generated images raises significant concerns regarding their authenticity and potential misuse. In this paper, we propose a novel approach utilizing Topological Data Analysis (TDA) to discern the authenticity of AI-generated faces. Traditional methods for detecting fake images often rely on pixel-level analysis or statistical features, which can be easily circumvented by sophisticated AI algorithms. In contrast, TDA offers a robust framework for analyzing complex data structures, such as those found in facial images, by capturing essential topological features.

Our methodology involves the construction of a topological representation of facial images, where each image is treated as a point cloud embedded in a high-dimensional space. By leveraging techniques such as persistent homology, we extract topological signatures that encapsulate the underlying structure of real and fake faces. Specifically, we explore topological properties such as connected components and loops, which are indicative of the authenticity of facial features.

To validate the effectiveness of our approach, we conduct analysis on diverse datasets containing both real and AI-generated faces. Our results demonstrate that topological features extracted through TDA exhibit modest but significant discriminative power, enabling fairly accurate differentiation between authentic and synthetic faces.

In conclusion, this paper presents a pioneering application of TDA for determining the authenticity of AI-generated faces. By leveraging topological insights, our approach offers a promising solution to mitigate the proliferation of synthetic media and safeguard the integrity of digital content in an era dominated by AI-generated imagery.

1 Introduction

The modification of images to resemble other humans, commonly known as Deepfakes, is considered a potentially harmful source of disinformation in the media. Deepfakes, by definition, are images or videos of a human in which parts of their image are digitally altered by artificial intelligence (Vaccari and Chadwick, 2020). Typically, these focus on changing a person's facial expressions or words spoken. As a result, Deepfakes are notorious for spreading misinformation to other individuals through media platforms. This ability allows bad actors to disseminate false information to other individuals. As such, the detection of deepfakes has become a hotbed of computer science and statistical research.

Recent detection research has mainly focused on complex deep-learning architectures. The incredible number of models focus on variations of convolutional neural network structures, sequentially improving on similar models. As discussed in Gupta et al. (2023), the vast number of different models and unique ideas has been overwhelming over the last decade. Even further, new models occur on a near-continuous basis. For instance, the state space model, as stated in Gu and Dao (2023) has recently become the new popular machine learning model, with variations of this model coming out quickly in short succession (Zhu et al., 2024). The complexity and ever-changing research space cannot be overstated.

While most research into Deepfake detection focuses on the architectural ideas of the (typically neural network-based) decision-making model, there is a lack of research into the detection of Deepfakes focusing on lower-level methods or increasing the feature space. We introduce a new method of providing key features to a model using Topological Data Analysis (TDA) (Edelsbrunner et al., 2002). Recently, this method has been suggested as a potential avenue for breakthrough statistical learning research (Chazal and Michel, 2021). This method is used to extract features from a set of data based on the structure of the data as a point cloud. Images especially have a structure as a grid allowing us to extract topological features from the pictures. This feature extraction allows us to use less complex models like random forest classifiers and support vector machines for classification purposes.

We intend to apply features generated by the TDA method on many Deepfake examples into many statistical learning settings to examine how well these generated features can help a statistical model. We will show how this application can improve the overall precision and recall of Deepfake detection models.

2 Topological Data Analysis

2.1 Simplicial Complexes and Filtrations

The main structure involved in TDA is the abstract simplicial complex and the filtrations that arise from these complexes. A simplicial complex is a tuple $K := (X, E)$ of points and edges ($E \subseteq 2^X$) such that if $\sigma \in E, \varnothing \neq \gamma \subseteq \sigma$ then $\gamma \in E$.

TDA is concerned with taking structured data in a point cloud and creating a simplical complex from our data. This simplicial complex is generally created from Vietoris-Rips (VR) complexes which are defined as follows. The VR complex is the simplicial complex with X equal to your original data set and E equal to the set $VR_r(X) := \{\sigma \subseteq X | \forall x, y \in \sigma, d(x, y) < r\}$, where r is some parameter and d is the Euclidean metric.

From abstract simplicial complexes we can define a filtration. A filtration is a collection of simplicial complexes indexed by the real numbers, $\{K_t : t \in \mathbb{R}\}$. These filtrations are defined by

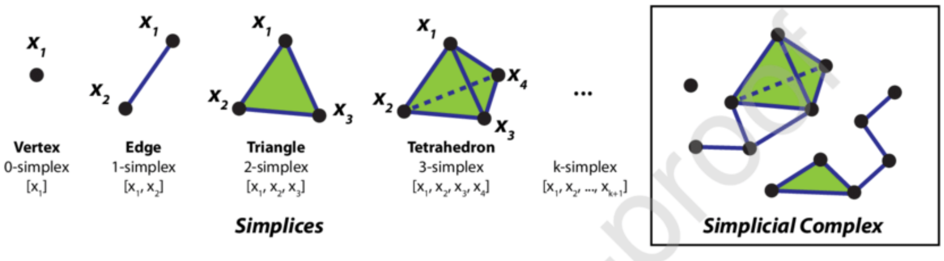


Figure 1: Examples of Simplicial Complexes (Zhang et al., 2020)

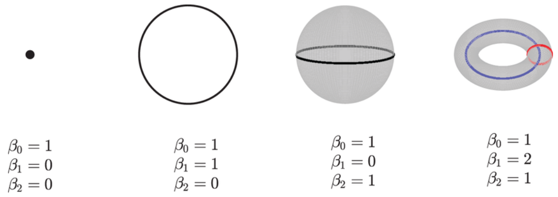


Figure 2: Examples of Betti numbers of degree 0,1,2 (Bai et al., 2024)

the property if $a < b$, then $K_a \subseteq K_b$. Concretely, our VR complexes give us a filtration indexed by the radius parameter. Essentially, filtrations can be thought of as sequences of increasing simplicial complexes. The importance of these constructions will be shown in the subsequent sections (Bubenik, 2022).

2.2 Persistent Homology

Persistent homology is the main feature extraction method of TDA. This method is concerned with how long certain topological features, the homology groups, persist over time in the filtrations. This persistence tells us key features about the topological structure of our data. Informally, the homology groups are a measure of different features involving the holes present in data. They are characterized by dimensions, where H_0 is the homology group of degree 0 and measures the number of connected components of our simplicial complex, H_1 is the homology group of degree 1 and measures number of 1 dimensional holes of our simplicial complex, etc. The dimension of these groups are the Betti numbers and they give us the number of connected components, β_0, and the number of 1 dimension holes, β_1 (Examples of Betti numbers are given in Figure 2). For the purposes of TDA, only H_0 and H_1 are usually considered for computational complexity purposes. We will have these same constraints for this project.

With the constructions of homology groups and filtrations of simplicial complexes, we can now define the features in question for TDA. Given a filtration, we can find the homology groups at each simplicial complex of our filtration. Namely, if we consider at some real number r, then we have homology groups $H_0(K_r)$ and $H_1(K_r)$. Then we consider how long each connected component or 1 dimensional hole persists as we increase the filtration parameter. This gives rise to the first construction of TDA: the persistence diagram. The persistence diagram is a set of pairs of birth and death times for each connected component or 1 dimensional hole (corresponding to the homology groups). Further analysis can be done on these persistence diagrams to create different features like persistence entropy, Wasserstein amplitude, and persistence landscapes which we will leave in Appendix A (Bubenik, 2022). The main advantage and statistical significance we get from TDA is the stability of these features, namely small perturbations in the data yield small perturbations of the features. This feature allows a robust usage for statistical analysis as noise and other data imperfections do not significantly impact the result of calculations in TDA (Cohen-Steiner et al., 2005).

2.3 Cubical Persistence

While we could use simplicial complexes for our images, a far more natural alternative is to consider the cubical complex, which takes into account the pixel structure of our image. These simplices take the forms of squares, cubes, and their higher dimensional analogs. We can define persistent homology in a similar way for cubical complexes which allows us to use these constructions for analyzing our images (Garin and Tauzin, 2019).

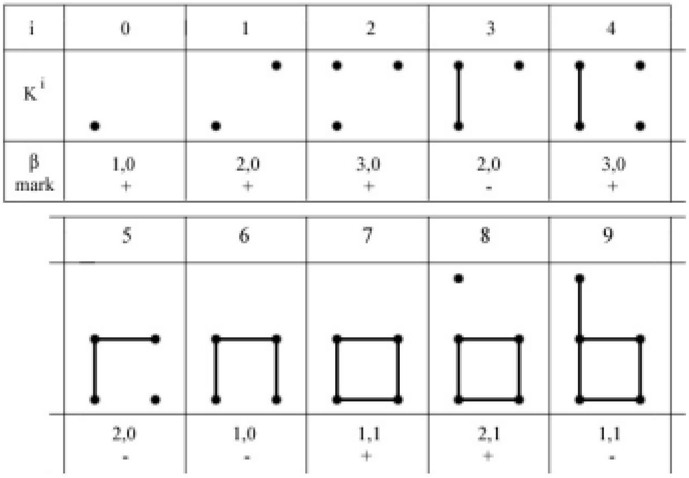


Figure 3: Example of a Filtered Cubical Complex

2.4 Sublevel Persistent Homology

A further tool of persistent homology we will use is sublevel persistent homology. A common technique for image analysis is the use of spectral analysis through 2D discrete Fourier transform (2DDFT) as AI image generation tends to lead to more periodic distribution of the amplitudes of the DFT (Corvi et al., 2022). This structure to analyze, namely in the case of our analysis of images of a function, $f : \mathbb{R}^2 \to \mathbb{R}$ to sublevel set homology. Sublevel homology is concerned with visualizing the shape of real valued functions. The sublevel set of a function $f : \mathbb{R}^n \to \mathbb{R}$ at a height r is the set $S_r := \{x \in \mathbb{R}^n : f(x) \leq r\}$. We can observe that this collection of sets $\{S_r\}$ is a filtration indexed by r, allowing us to observe persistent homological features over varying values of r. Sublevel set persistent homology is a higher dimensional equivalent to similar methods of analyzing real valued functions in disconnectivity graphs, but a discussion of these is beyond the scope of this paper. For further information on sublevel set homology and disconnectivity graphs, refer to Mirth et al. (2021).

3 Constructions from Persistence Diagrams

In this section, we will give a brief overview of the construction of the main topological features we used in the paper. For all of these, we will let our persistence diagram be denoted $PD := \{(b_i, d_i) : i \in \{1, \dots, N\}\}$, where N is the number of points on our persistence diagram and b_i and d_i are the birth and death time of feature i.

3.1 Wasserstein Amplitudes, Persistence Entropy, and Persistence Landscapes

The three constructions given in this subsection are commonly used constructions in the literature. The Wasserstein amlitude and persistence entropy were features used for cubical persistence, but the persistence landscape was used for both models.

3.1.1 Wasserstein Amplitude

The Wasserstein amplitude is a single number that summarizes how much a given persistence diagram differs from the empty diagram. This can be calculated with any L^p norm, but usually is considered under the $p = 1, 2$ norm (Garin and Tauzin, 2019). Thus our Wasserstein amplitude is defined as

$$A_W = \frac{\sqrt{2}}{2}(\sum_{i=1}^{N}(d_i - b_i)^p)^{\frac{1}{p}}$$

3.1.2 Persistence Entropy

The persistence entropy is a way of measuring the amount of information contained in a persistence diagram. This feature extends the Shannon entropy to the diagrams, using the sample proportion of lengths of individual persistence bars, viz.

$$PE(PD) = \sum_{i=1}^{N} \frac{d_i - b_i}{L(PD)} \log(\frac{d_i - b_i}{L(PD)}),$$

where $L(PD) = \sum_{i=1}^{N}(d_i - b_i)$ is the total length of all persistence in our image (Atienza et al., 2019).

3.1.3 Persistence Landscapes

A persistence landscape is a way of containing all the information about the persistence diagram into a function $\lambda : \mathbb{N} \times \mathbb{R} \to \mathbb{R}$. It is defined as follows: for $a < b$, we define $f_{(a,b)}(t) = \max(0, \min(a + t, b - t))$, and we define $\lambda(k, t) = \text{kmax}\{f_{(b_i,d_i)}(t)\}_{i\in\{1,...,N\}}$, where kmax is the kth largest element. The use of persistence diagrams comes from their desirable statistical properties:

(1) The mapping of persistence diagrams to persistence landscapes is invertible.

(2) Persistence landscapes are stochastically stable, viz. close diagrams yield close persistence landscapes.

(3) We can compute means and other desired statistical quantities with persistence landscapes far easier than with persistence diagrams.

Information regarding these properties and more about persistence landscapes can be found in Bubenik (2020)

3.2 Kernel Methods

All of these computations are considered under a representation of persistence diagrams as vectors in a reproducing kernel Hilbert space (RKHS), a discussion of which is outside the scope of this paper but can be found in resources such as Paulsen and Raghupathi (2016). We are concerned with generating a matrix representing our given kernel, K, in matrix form, namely $K_{ij} = d_K(D_i, D_j)$, where $D_i, D_j \in PD$ are diagrams and d_K is the "distance" generated by our kernel. As $|PD| < \infty$, this process gives us a feature of a matrix for us to use classification methods and other learning methods on. Furthermore, discussion of these kernels involves an assumed background in measure theory, a complete treatment of which can be found in Rudin (1986)

3.2.1 Sliced Wasserstein Kernel

The sliced Wasserstein kernel is a modification of the 1-Wasserstein distance, which is a measure of distance between two finite measures on the real line. Given 2 non-negative measures μ, ν such that $\mu(\mathbb{R}) = \nu(\mathbb{R}) < \infty$, we have that the 1-Wasserstein distance $\mathcal{W}$ is

$$\mathcal{W}(\mu, \nu) = \inf_{P \in \Pi(\mu,\nu)} \int\int_{\mathbb{R}\times\mathbb{R}} |x - y| P(dx, dy),$$

where $\Pi(\mu, \nu)$ is the set of all $\mathbb{R}^2$ measures with marginals μ, ν, namely $\pi \in \Pi(\mu, \nu)$ if for all sets $X \subseteq \mathbb{R}$, we have that

$$(1) \int_X \int_{\mathbb{R}} \pi(x, y) dx dy = \mu(X)$$

$$(2) \int_X \int_{\mathbb{R}} \pi(x, y) dy dx = \nu(X).$$

Our Sliced Wasserstein distance is defined as follows. Let S_1 be the unit circle in the plane. Given a unit vector, $\theta \in \mathbb{R}$, define $L_\theta = \{r\theta | r \in \mathbb{R}\}$, and let $\pi_\theta : \mathbb{R}^2 \twoheadrightarrow L_\theta$ be an orthogonal projection. If $D_1, D_2 \in PD$, then let $\mu_1^\theta := \sum_{p \in D_1} \delta_{\pi_\theta(p)}$ and $\mu_{1\delta}^\theta = \sum_{p \in D_1} \delta_{\pi_{\theta(\pi_\Delta(p))}}$, where δ_a is the point mass measure on $\mathbb{R}$. We can define μ_2^θ and μ_2^θ similarly, given that π_Δ is the orthogonal projection onto the diagonal. Observe that $\mu_\cdot^\cdot$ are all measures as they are the sum of point mass measures. The Sliced Wasserstein distance is defined as:

$$SW(D_1, D_2) := \frac{1}{2\pi} \int_{S_1} \mathcal{W}(\mu_1^\theta + \mu_{2\Delta}^\theta, \mu_2^\theta, \mu_{1\Delta}^\theta) d\theta.$$

We can thus define the Sliced Wasserstein Kernel as $k_{SW} := \exp(\frac{-SW(D_1,D_2)}{2\sigma^2})$, with σ acting as a hyperparameter. A further discussion of this definition and results about this kernel can be found in Carrière et al. (2017).

3.2.2 Weighted Gaussian Kernel

With the previous discussion of kernels, we can define the Persistence Weighted Gaussian Kernel(PWGK) fairly easily. First, consider the persistence diagram as being represented in some RKHS. Then, let us define the Gaussian kernel $k_G(x, y) = \exp(\frac{-||x-y||^2}{2\sigma^2})$ for $\sigma > 0$ and $|| \cdot ||$ being the norm associated with our RKHS, and define an arctangent weight function as $w_{arc}(x) = \arctan(C\text{pers}(x)^p)$ for $C, p > 0$, where $\text{pers}(x)$ is $d_i - b_i$ for some $i = 1, \ldots, n$. This is the PWGK and can be used to generate the Gram matrix corresponding to the set of diagrams as defined in our initial discussion. Care must be taken to recognize that the domain of the weight and kernel function are representations of the diagrams in a RKHS. A further discussion of this defintion and properties of this kernel can be found in Kusano et al. (2016).

3.2.3 Persistence Image

The persistence image is a way of vectorizing our persistence diagram while maintaining a strong connection with the original diagram. To construct this feature, a few prerequisite constructions are needed. Define the normalized symmetric gaussian as follows,

$$g_u(x, y) = \frac{1}{2\pi\sigma^2} \exp(-[(x - \mu_x)^2 + (y - \mu_y)^2]]/2\sigma^2),$$

where u, σ are hyperparameters. Further, fix some non negative weight function $f : \mathbb{R}^2 \to \mathbb{R}$ such that $f = 0$ along the horizontal axis, and f is both continuous and piecewise differentiable.

We define the persistence surface $\rho_D : \mathbb{R}^2 \to \mathbb{R}$ for some $D \in PD$ as $\rho_B(z) = \sum_{u \in T(D)} f(u) g_u(z)$, where $T(D) = \{(x, y - x) : (x, y) \in D\}$. Finally, we can define the persistence image as

$$I(\rho_D)_p = \int \int_p \rho_B dy dx,$$

where this image is a collection of pixels. A further discussion of this definition and the properties of the image based on our selection of weighting function and other considerations can be found in Adams et al. (2016).

4 Methods

4.1 Collection and Treatment of Data

For this paper, our data was taken from the Michigan State deepfake dataset (Stehouwer et al., 2019). This dataset consists of around 300000 images with around 250000 of those being fake images. The fake images consisted of images generated by FFHQ, CelebA, FaceForensics++, Deepfakes, FaceSwap, Face2Face, FaceAPP, StarGAN, PGGAN, and StyleGAN. This dataset was selected as it consisted of images which are highly pertinent for AI detection: faces. Furthermore, it consisted of a variety of methods for generating fake images, which should help to increases the generalizability of our algorithm. All of our images were greyscaled for better analysis by our TDA algorithms. After applying the topological pipeline, the images were split into an 80/20 testing/training split for evaluation of our features and model. Note for training purposes, a real image was encoded as a 1 and a fake image as a 0.

4.2 Preprocessing for Cubical Persistence

In this problem, we will consider $\mathcal{P}$ to be our pixel space and our data to be in the form of a collection of N greyscaled images $\{I_n\}_{n=1}^N$, namely a collection of functions with $I_n : \mathcal{P} \to \mathbb{R}$, where $I_n(p)$ is the greyscale value of pixel p. In order to be able to create a filtration for our data to perform TDA, we have to treat our data in a specific way. Our data was first binarized based on a threshold of $t \in (0, 1)$; namely, for each greyscaled image I, our binarized image is a function $\mathcal{B} : I \to \{0, 1\}$ where

$$B(i) = \begin{cases} 1 & i \geq t \\ 0 & i < t \end{cases}.$$

Thus we have binarized images in the form $\{\mathcal{B}(I_n)\}$. To build a filtration on top of this, we can use a variety of filtrations. The filtrations we used for this analysis are height filtration, radial filtration, density filtration, and dilation filtration. A height filtration is a function $\mathcal{H} : I \to \mathbb{R}$. This function is defined by choosing a unit vector $u \in \mathbb{R}^{dim(I)}$ and defining

$$\mathcal{H}(i) = \begin{cases} \langle i, u \rangle & \mathcal{B}(i) = 1 \\ \max\{\langle p, u \rangle : \mathcal{B}(p) = 1\} & \mathcal{B} = 0 \end{cases},$$

where $\langle \cdot, \cdot \rangle$ is an inner product on our pixel space. A radial filtration is a function $\mathcal{R} : I \to \mathbb{R}$ indexed by a center pixel $c \in I$. We define the filtration as

$$\mathcal{R}(i) = \begin{cases} ||c - i|| & \mathcal{B}(i) = 1 \\ \max_{j \in I}\{||c - j||\} & \mathcal{B}(i) = 0 \end{cases},$$

where $|| \cdot ||$ is a norm on our pixel space, in our case the euclidean norm. A density filtration is a measure of the number of positive binary pixels in a given neighborhood of a pixel. Namely it is a function $\mathcal{D} : I \to \mathbb{R}$ indexed by a parameter r of our neighborhood size by

$$\mathcal{D}_r(i) = |\{p \in I : \mathcal{B}(p) = 1 \text{ and } ||p - i|| \leq r\}|,$$

where $|\cdot|$ is set cardinaliyt, and $||\cdot||$ is the norm on our pixel space, in our case the taxicab norm. The final filtration that we used was the dilation filtration. The dilation filtration is defined as $\mathcal{D} : I \to \mathbb{R}$ such that

$$D(i) = \min\{||i - p|| : \mathcal{B}(p) = 1\},$$

where $|| \cdot ||$ is the euclidean norm in this case.

4.3 Processing for Sublevel Set Homology

Similarly to cubical persistence, our images were first greyscaled to values between 0 and 1. After greyscaling we applied the 2D Discrete Fourier Transform (2D DFT) to the greyscaled images. Namely given an image $I : \mathbb{N}^2 \to [0, 1]$ with dimensions $N \times N$, the 2DDFT is the function

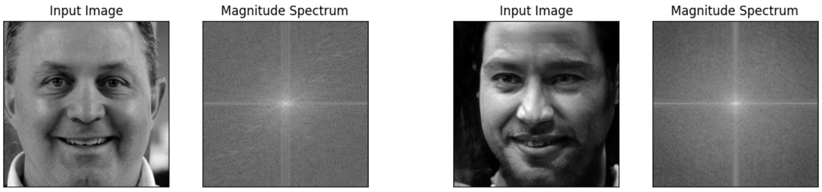
$$F(x, y) = \sum_{n=0}^{N-1} \sum_{m=0}^{N-1} I(n, m) e^{-i2\pi(\frac{xm}{N} + \frac{yn}{N})}.$$

For every $x, y \in \mathbb{N}$ this gives us a complex number $F(x, y)$, of which we will only consider the amplitude $|F(x, y)|$. This amplitude of the fourier transform gives us a measure of how rapidly the intensities of the pixels vary in a given region. Furthermore, the amplitude spectrum of the fourier transform, namely the graph of the fourier transform function, is such that real images usually have a more noisy amplitude spectrum and fake images have a more periodic and smooth amplitude spectrum (An example of this can be seen in figure 4). Notice that this gives us a real valued function and thus we can use sublevel set homology to analyze the information of the fourier transform function.

4.4 Features from Persistent Homology

4.4.1 Cubical Homology

In order to train a machine learning model on our data, we need to use these filtrations to create topological features that can be used for the supervised classification of the AI generated images. For this paper, we focused on the features of H_0 persistence landscapes, H_1 persistence



(a) Real Image

(b) Fake Image

Figure 4: Magnitude Spectrum of Face Images

landscapes, Wasserstein amplitudes, and persistence entropy. For the computation of these features, we used the python package Giotto-TDA created in the paper by Tauzin et al. (2021). To create our features, we began with the training set of the images, and applied the three different types of filtrations mentioned above. For height filtration, we chose vectors in the form $\{(\{0, \pm 1\}, \{0, \pm 1\})\} \setminus \{(0, 0)\}$ namely those in the form $(0, 1)$, $(1, 1)$, $(-1, 1)$, etc. For radial filtration we chose 8 pixels in every direction corresponding to the vectors in our height filtration and the middle pixel. For density filtration, we chose $r = 10, 20.40$. For dilation filtration we chose the same pixels as we used for radial filtration. We then used cubical persistence to generate the persistence diagrams. In order to standardize our persistence diagrams, each landscape was scaled to a standard size and points within 0.05 of the diagonal were removed from the diagram in order to aid in computation time and reduce noise. Using these persistence diagrams, we computed persistence landscapes, Wasserstein amplitudes, and persistence entropies. This gave us a total of 116 different feature sets (corresponding to each type of filtration and topological information) to use for classification of our images. Furthermore, we use the discretized persistence landscape to be able to represent it as a value in euclidean space, thus increasing our feature set further.

In order to classify our images as real or fake we used a random forest classifier with the default number of 100 trees and a support vector classifier with a radial kernel. We first use a subset of our features in order to reduce any issues with collinearity in our classifier. Thus, we need to investigate this correlational structure to decide how to mitigate this issue. Given a lower sample size of images available for our current analysis, we use 10-fold cross validation for estimation of our training error for our feature set.

4.4.2 Sublevel Set Homology

For our sublevel set homology, for computation sake, we only considered the diagrams of degree 1 in our feature space. For this classification task, we took an approach of a grid search of a variety of models to determine the best fit. Our grid search included a support vector classifier (SVC) using a sliced Wasserstein kernel, an SVC using a persistence weighted Gaussian kernel, an SVC with the persistence image, a random forest classifier (RFC) with a persistence landscape, and a K-nearest neighbor (KNN) with the Bottleneck distance (more on these constructions can be found in appendix A). Our model was fit with 3 folds of cross validation using a grid search method to find the optimal model out of this parameter space.

5 Results

Due to computational and time constraints, we performed our analysis of the cubical persistence methodology on 2500 images containing 500 real images, 500 StarGAN generated images, 500

StyleGAN, and 1000 PGGAN images. As the sublevel set homology was more computationally expensive, we were only able to run this algorithm with around 200 images for a cursory survey of the abilities of this method, with 50 real images and a similar split of StarGAN and PGGAN images.

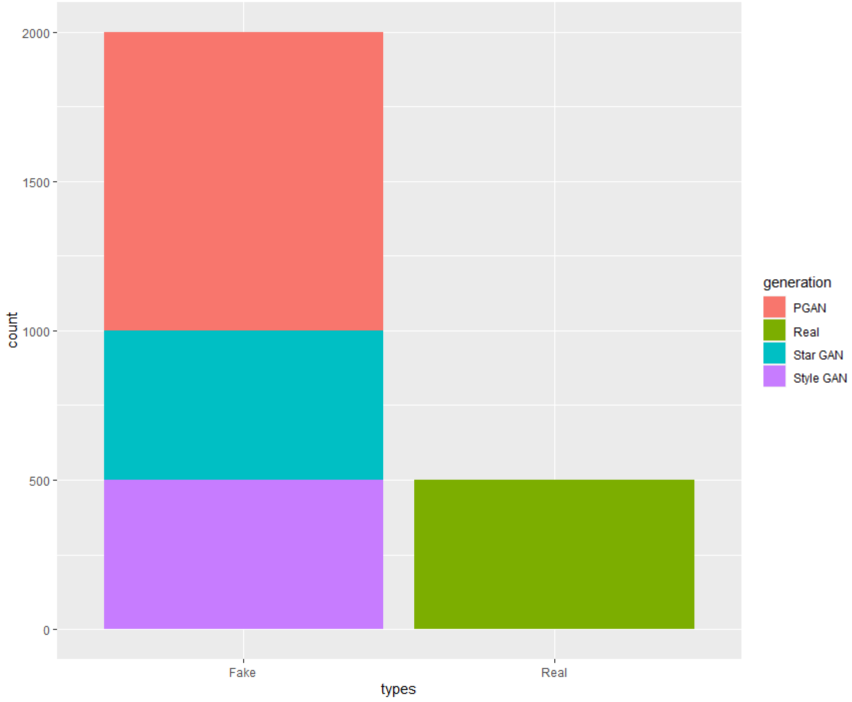


Figure 5: Visualization of Data Split for Cubical Persistence

5.1 Visualization of Homology

To visualize the differences between the homology of the real and fake images, we will show the 2D Principle Component Analysis (PCA) of the persistence landscapes of the images to detect if there is any low dimensional linear separability of the data.

5.1.1 Cubical Homology

From our PCA, there does not appear to be low dimensional separability of the real and fake images from our topological features. This suggests that the SVC may do a better job at classifying this data because the presence of the real images contained in a larger cloud of fake images may yield a spherical containment in the true high dimensional structure of our data.

Further, we can see the correlational structure of our data from the correlation heatmap which elucidates the presence of high correlation between features as we expected. The values on our heatmap indicate that to help reduce the multicolinearity present, we should remove instances where the correlation between 2 features is greater than 0.9. To implement this, we only remove one of these features in order to maintain as much information as possible.

5.1.2 Sublevel Set Homology

For this, we will consider the persistence landscapes for the degree 1 homology groups as this was selected as the best performing feature from our grid search. Furthermore, since we are

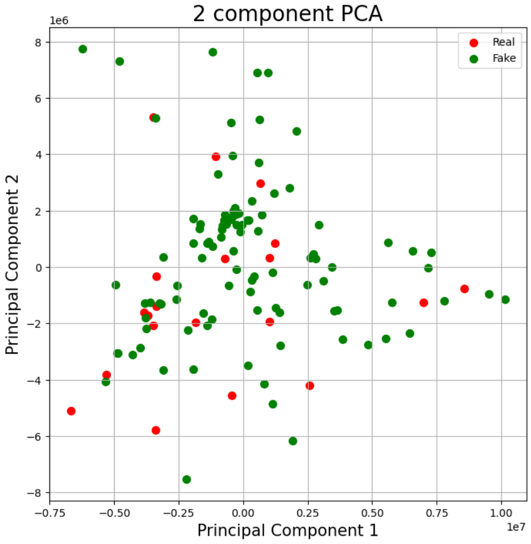


Figure 6: 2 Component PCA for Cubical Persistence

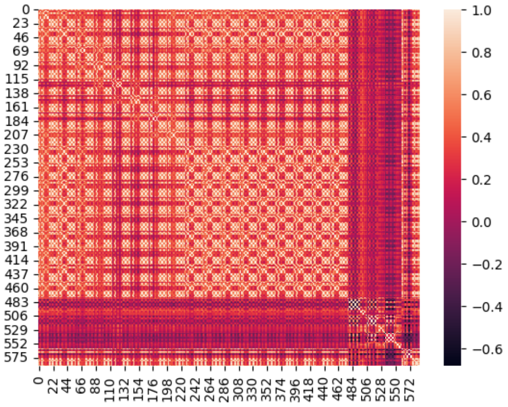


Figure 7: Correlation Heat Map of Cubical Persistence Features

only considering one type of persistence landscape, that of degree 1, we can also display the difference in average persistence landscapes of real and fake images.

From Figure 8, we can see that there appears to be a significant difference in the persistence landscapes of real images and fake images. This can be seen by the large negative spike, greatly differing from zero. This gives us confidence to the usage of sublevel set homology for analyzing these images as there appears to be a significant difference between the topological features of real and fake images.

Furthermore, we can see from our PCA there appears to be a linear separation between real and fake images in 2 principal components. For reference, these two principle components contain around 80% of the variation in our data.

These exploratory results give us confidence in our application of this methodology, as it appears to have created a significant feature space that can distinguish between fake and real images. This will be further shown from our results of sublevel set homology.

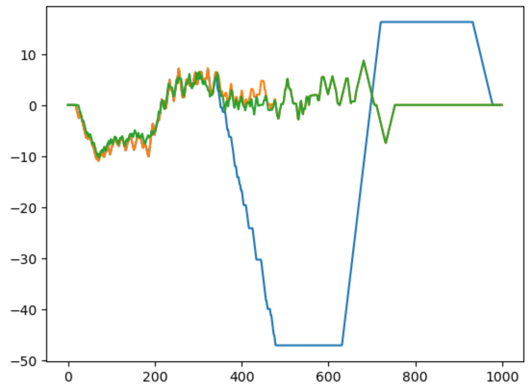


Figure 8: Difference in Average Persistence Landscapes of Degree 1

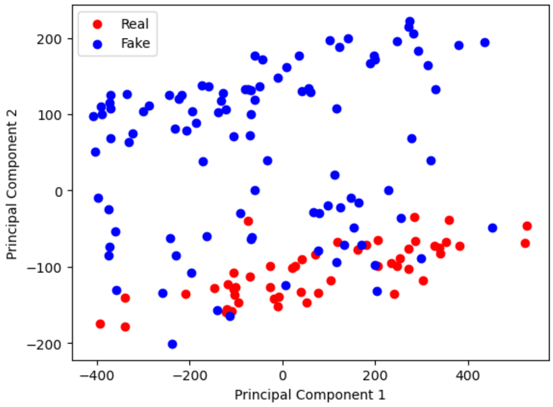


Figure 9: PCA of Sublevel Set Persistence Landscapes

5.2 Cubical Persistence Results

To perform our classification, we used a random forest classifier (RFC) and a support vector machine (SVM). The RFC had 100 trees in the forest, and the SVM used a radial kernel for the decision boundary. To evaluate our feature set, we used 10-fold cross validation to estimate the testing error. Further, we evaluate our other metrics, the confusion matrix and reciever operating characteristic (ROC) curve on a test set of 75 images with the same proportion split as our training data.

As we can see from our results, the removal of the high correlation features didn't appear to improve our testing accuracy at all. This suggests that the fear of multicollinearity impacting the predictions of our RFC and SVC were unfounded. Furthermore, we can see for our RFC, it does slightly better than a random guess at classifying the images, based on the receiver operating characteristic (ROC) curve and its corresponding area under curve (AUC) of 0.883, but significantly worse than the sublevel set approach which we will see in the next section.

5.3 Sublevel Set Homology Results

After performing our grid search, we found that the parameter that produced the best accuracy on 3-fold cross validation of our training set was the random forest classifier with 100 trees trained on the persistence landscape of homology degree 1. We further scored it on a testing set of the remaining images, including around 1350 images with about 450 real images and around 900 fake images. This resulted in a classification accuracy of 87.1%.

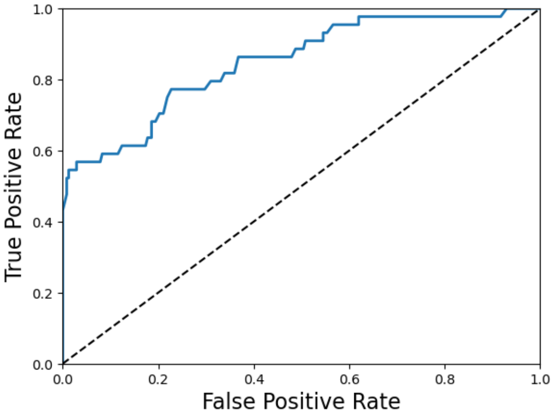


Figure 10: Receiver Operating Characteristic (ROC) Curve for Cubical Features

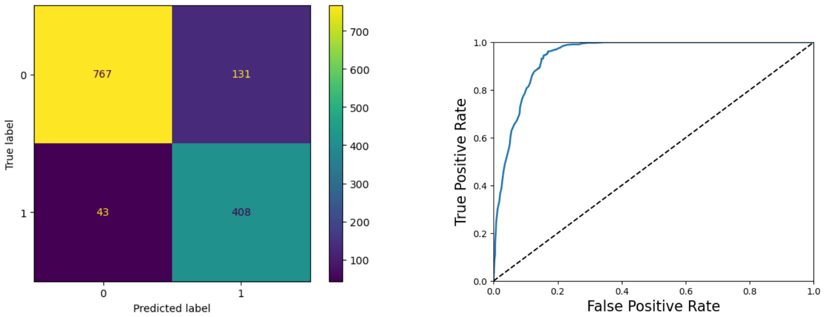
Model	**Accuracy**
Random Forest	≈ 81.7%
Support Vector Machine	≈ 83.3%
RFC with Correlation Removal	≈ 81%
SVM with Correlation Removal	≈ 83.3%

Figure 11: Example of a Filtered Cubical Complex

To evaluate how our model did correctly classifying positive and negative cases, we used a confusion matrix and a receiver operating characteristic curve (ROC). From our confusion matrix, we can see that we achieved a true positive rate of 90.5% and a true negative rate of 85.4% with an F-1 score of 82.4%. This reveals that our model did a slightly better job finding the images that were real, rather than correctly identifying which images were fake. This could likely be changed with an adjustment to the cutoff of our decision function to get a likely more important result of better accuracy at determining which images are fake. Our ROC curve shows that the model does a fairly good job at all levels of false positive rate (FPR) and an area under curve (AUC) of 0.943 indicates that our model did a far better job than a random guess of whether the image was real or fake.

6 Discussion and Conclusion

This research elucidates a different paradigm for analyzing the veracity of images on the internet. Overall, the classification results from the features generated from different topological methods performed worse than their complex architecture neural network models, but had the advantage of returning relatively positive results for a small sample size. Large neural network models need vast amounts of data, and with this necessity makes it very easy for such a classification model to fall out of date. The topological methods displayed in this paper, however, have the advantage of being able to detect distinguishing features from these images in a significantly smaller sample size. Theoretically, larger sample sizes in training the models based on topological features would provide better accuracy and robustness. Although our results are relatively modest in comparison to the current literature, comparisons with current state of the art deep learning architecture is unfeasible in this paper given the vastly different scales of the analysis. Furthermore, a vast improvement to our results could be achieved with more state of



(a) Confusion Matrix for Sublevel RFC

(b) ROC for Sublevel RFC

Figure 12: Metrics for Sublevel Classification

the art Gaussian denoising present in the literature. This would allow a stronger signal to be present in the images, excluding potential noise present which would aid in our classification. Overall, the accuracy of our models are lower than those present in the current research environment, but also include vast room for growth and ease of use which should inspire future work in field of topological image analysis.

A Citations and References

References

Adams, H., Chepushtanova, S., Emerson, T., Hanson, E., Kirby, M., Motta, F., Neville, R., Peterson, C., Shipman, P. and Ziegelmeier, L. (2016) Persistence images: A stable vector representation of persistent homology.

Atienza, N., Escudero, L. M., Jiménez, M. J. and Soriano-Trigueros, M. (2019) Persistent entropy: a scale-invariant topological statistic for analyzing cell arrangements. *ArXiv*, **abs/1902.06467**. URL: `https://api.semanticscholar.org/CorpusID:62841475`.

Bai, X., Yu, C. and Zhai, J. (2024) Topological data analysis of the firings of a network of stochastic spiking neurons. *Frontiers in Neural Circuits*, **17**.

Bubenik, P. (2020) *The Persistence Landscape and Some of Its Properties*, 97–117. Springer International Publishing. URL: `http://dx.doi.org/10.1007/978-3-030-43408-3`$_4$.

— (2022) Topological data analysis and persistence theory.

Carrière, M., Cuturi, M. and Oudot, S. (2017) Sliced Wasserstein kernel for persistence diagrams. In *Proceedings of the 34th International Conference on Machine Learning* (eds. D. Precup and Y. W. Teh), vol. 70 of *Proceedings of Machine Learning Research*, 664–673. PMLR. URL: `https://proceedings.mlr.press/v70/carriere17a.html`.

Chazal, F. and Michel, B. (2021) An introduction to topological data analysis: Fundamental and practical aspects for data scientists. *Frontiers in Artificial Intelligence*, **4**. URL: `http://dx.doi.org/10.3389/frai.2021.667963`.

Cohen-Steiner, D., Edelsbrunner, H. and Harer, J. (2005) Stability of persistence diagrams. vol. 37, 263–271.

Corvi, R., Cozzolino, D., Zingarini, G., Poggi, G., Nagano, K. and Verdoliva, L. (2022) On the detection of synthetic images generated by diffusion models.

Edelsbrunner, Letscher and Zomorodian (2002) Topological persistence and simplification. *Discrete amp; Computational Geometry*, **28**, 511–533. URL: `http://dx.doi.org/10.1007/s00454-002-2885-2`.

Garin, A. and Tauzin, G. (2019) A topological "reading" lesson: Classification of MNIST using TDA. *CoRR*, **abs/1910.08345**. URL: `http://arxiv.org/abs/1910.08345`.

Gu, A. and Dao, T. (2023) Mamba: Linear-time sequence modeling with selective state spaces.

Gupta, G., Raja, K., Gupta, M., Jan, T., Whiteside, S. T. and Prasad, M. (2023) A comprehensive review of deepfake detection using advanced machine learning and fusion methods. *Electronics*, **13**, 95. URL: `http://dx.doi.org/10.3390/electronics13010095`.

Kusano, G., Fukumizu, K. and Hiraoka, Y. (2016) Persistence weighted gaussian kernel for topological data analysis.

Mirth, J., Zhai, Y., Bush, J., Alvarado, E. G., Jordan, H., Heim, M., Krishnamoorthy, B., Pflaum, M., Clark, A., Z, Y. and Adams, H. (2021) Representations of energy landscapes by sublevelset persistent homology: An example with n-alkanes. *The Journal of Chemical Physics*, **154**. URL: `http://dx.doi.org/10.1063/5.0036747`.

Paulsen, V. I. and Raghupathi, M. (2016) *An Introduction to the Theory of Reproducing Kernel Hilbert Spaces*. Cambridge Studies in Advanced Mathematics. Cambridge University Press.

Rudin, W. (1986) *Real and Complex Analysis*. McGraw-Hill Science/Engineering/Math. URL: `http://www.amazon.com/exec/obidos/redirect?tag=citeulike07-20&path=ASIN/0070542341`

Stehouwer, J., Dang, H., Liu, F., Liu, X. and Jain, A. K. (2019) On the detection of digital face manipulation. *CoRR*, **abs/1910.01717**. URL: `http://arxiv.org/abs/1910.01717`.

Tauzin, G., Lupo, U., Tunstall, L., Pérez, J. B., Caorsi, M., Reise, W., Medina-Mardones, A., Dassatti, A. and Hess, K. (2021) giotto-tda: A topological data analysis toolkit for machine learning and data exploration.

Vaccari, C. and Chadwick, A. (2020) Deepfakes and disinformation: Exploring the impact of synthetic political video on deception, uncertainty, and trust in news. *Social Media + Society*, **6**, 205630512090340. URL: `http://dx.doi.org/10.1177/2056305120903408`.

Zhang, M., Kalies, W., Kelso, S. and Tognoli, E. (2020) Topological portraits of multiscale coordination dynamics. *Journal of Neuroscience Methods*, **339**, 108672.

Zhu, L., Liao, B., Zhang, Q., Wang, X., Liu, W. and Wang, X. (2024) Vision mamba: Efficient visual representation learning with bidirectional state space model.