Preserving Hidden Hierarchical Structure: Poincaré Distance for Enhanced Genomic Sequence Analysis

Sarwan Ali[1], Haris Mansoor[2], Prakash Chourasia[1], Imdad Ulla Khan[2], and Murray Pattersn[1]

[1] Georgia State University, Atlanta GA 30303, USA
[2] Lahore University of Management Sciences, Lahore Punjab 54792, Pakistan
sali85@student.gsu.edu, 16060061@lums.edu.pk, pchourasia1@student.gsu.edu, imdad.khan@lums.edu.pk, mpatterson30@gsu.edu

Abstract. The analysis of large volumes of molecular (genomic, proteomic, etc.) sequences has become a significant research field, especially after the recent coronavirus pandemic. Although it has proven beneficial to sequence analysis, machine learning (ML) is not without its difficulties, particularly when the feature space becomes highly dimensional. While most ML models operate with the conventional Euclidean distance, the hidden hierarchical structure present among a set of phylogenetically related sequences is difficult to represent in Euclidean space without losing much information or requiring many dimensions. Since such hierarchical structure can be informative to analysis tasks such as clustering and classification, we propose two measures for generating a distance matrix from a set of sequences based on distance in the Poincaré disk model of hyperbolic geometry, or the *Poincaré distance*, for short. Such a distance measure can allow to embedding of even a fully resolved phylogenetic tree in just two dimensions with minimal distortion to any hierarchical structure. Our first approach is based purely on the classical Poincaré distance, while the other approach modifies this distance by combining the Euclidean norms and the dot product between the sequence representations. A thorough analysis of both measures demonstrates its superiority in a variety of genomic and proteomic sequence classification tasks in terms of efficiency, accuracy, predictive performance, and the capacity to capture significant sequence correlations. These approaches perform better than existing state-of-the-art methods across the majority of evaluation metrics.

Keywords: Poincaré Distance, Kernel Matrix, molecular sequences

1 Introduction

In the field of bioinformatics, the analysis of molecular (genomic, proteomic, etc.) sequences is a fundamental task that plays a crucial role in understanding their structure, function, and evolution [1,2]. On the other hand, the advancements in

machine learning and artificial intelligence has led to the emergence of learning representations for symbolic data [3], such as text [4], graphs [5], and multi-relational data [6]. Use of machine learning (ML) models has become popular due to their ease of use, generalizability, and efficient performance. However, since most of the ML models work with Euclidean distance in general, it is difficult to capture hierarchical information such as phylogenetic relationships simply with pairwise Euclidean distances between molecular sequence representations.

Recent methods for designing embeddings for molecular sequences include one-hot encoding (OHE) [7] and k-mers spectrum representations [8]. OHE embeddings are highly-dimensional and lack order information [9], while the k-mers spectrum representation preserves sequence order but overlooks structural details [10]. Euclidean distance in either the OHE or the k-mers spectrum representation fails to consider hierarchical information, impacting downstream classification tasks. Innovative approaches, like kernel matrices, attempt to address this by projecting data into a high-dimensional space [11], yet might not fully capture all nonlinear relationships [11]. Thus, there remains a need for more effective methods that comprehensively capture sequence similarity and other relationships between sequences. To overcome these drawbacks and enhance sequence analysis potential, this research proposes a new distance measure based on distance in the Poincaré disk model of hyperbolic geometry, or the *Poincaré distance*, for short [12]. Hyperbolic geometry has the unique property of preserving both visible and hidden hierarchical structure in the data. By leveraging this property, the Poincaré distance aims to provide a more accurate representation of genomic sequence similarity compared to the conventional Euclidean distance, capturing the complex relationships and dependencies present in the data.

In this study, we compute a kernel matrix for molecular sequences using the Poincaré distance, which captures nonlinear, hierarchical relationships and is more robust to outliers compared to traditional dot product-based kernel matrices.

Furthermore, we propose a modified version of the Poincaré distance, denoted as M-Poincaré, which combines Euclidean norms and the dot product between the molecular sequence representations. This modification aims to enhance the representation of sequence similarities by incorporating both hyperbolic and Euclidean geometry. This strikes a balance between the simplicity of computing the representation and learning from it, with the ability to capture a more complex (hierarchical) structure and what can be learned from this. The suggested Poincaré and M-Poincaré distances hold great promise and have real-world applications in disciplines such as medicine, biotechnology, and information retrieval. Offering better genomic sequence similarity representation and enhancing tasks like sequence classification, functional annotation, and evolutionary analysis.

Overall, we propose a novel distance-based kernel matrix using Poincaré distance, its evaluation and validation on a diverse set of genomic and proteomic sequence data sets, and the practical implications in real-world applications. In summary, the contributions are as follows:

1. **Novel Distance Measure:** This paper proposes two novel distance measures based on the Poincaré disk model of hyperbolic geometry for molecular sequence analysis. Unlike conventional Euclidean distance, the proposed distance measure takes advantage of the unique properties of the hyperbolic space to capture the complex relationships and dependencies within molecular sequence data. By utilizing hyperbolic geometry, which is known for its ability to represent hierarchical (tree-like) structure, the proposed distance measure offers a more accurate representation of similarity for sets of phylogenetically related sequences.
2. **Kernel Matrix Computation:** Using the Poincaré distance matrix, we design a kernel matrix using the Gaussian kernel, which shows superior performance compared to the k-mers spectrum-based kernel computation method from the literature.
3. **Evaluation and Validation:** This paper extensively evaluates the effectiveness of the proposed distance measures on three sets of genomic (or proteomic) sequences sequences to cover a broad range of analysis cases. We compare the results for the Poincaré kernel and M-Poincaré kernel with several recently proposed baselines and show the superiority of the proposed distance measures in terms of accuracy, efficiency, and the ability to capture meaningful sequence relationships.
4. **Real-World Applications:** The contributions of this paper extends beyond theoretical advancements and has practical implications in real-world applications. Accurate sequence (in the general sense, *i.e.*, beyond just molecular sequences) comparison is crucial in various domains, such as genomics, evolutionary biology, and recommendation systems.

The rest of the paper is organized as follows: Section 2 contains the literature review for the proposed method. This paper's contribution is discussed in Section 3. The dataset statistics and experimental setup detail is given in Section 4. The results for proposed and baseline models are reported in Section 5. Finally, the conclusion of this study is given in Section 6.

2 Related Work

Traditional distance measures, such as Euclidean distance [13], Hamming distance [14], and cosine similarity [15], have been widely used in molecular sequence analysis. These measures are based on the assumption of sequences being represented as vectors in an Euclidean space [16]. While they have proven to be effective in certain applications, they suffer from limitations for sequences with complex structures and dependencies [16]. For example, Euclidean distance fails to capture the hierarchical relationships and tree-like structures that are prevalent in many sequence datasets in a natural way, requiring many more dimensions to capture it. Moreover, these measures often struggle to handle high-dimensional sequence data and may produce suboptimal results [17].

Recent research has explored the use of hyperbolic geometry in sequence analysis [16,12]. Hyperbolic space offers unique properties that make it well-

suited for capturing hierarchical and tree-like structures [16]. Several studies have demonstrated the effectiveness of hyperbolic representations in tasks such as text classification [18], network analysis [19], and knowledge graph embeddings [20]. Graph-based approaches in sequence analysis capture complex relationships by representing sequences as nodes, with edges [21] indicating similarity (k-nearest neighbor or kNN graphs) or interactions [22], such as protein-protein binding. Despite their potential, these methods can be computationally intensive and may struggle with large-scale datasets and capturing hierarchical information efficiently. Embedding-based methods project sequences into low-dimensional vector spaces, reflecting similarity between molecular sequences [23,24,25]. Techniques like k-mers spectrum [9,8] have shown reasonable performance in various tasks but can be computationally expensive and may fail to preserve hierarchical and structural relationships effectively.

A common approach to computing sequence similarity is generating a kernel or gram matrix [9,11]. This can be computationally expensive, but some methods improve efficiency by using the dot product of sequence spectra [11]. These matrices [9] are used in classifiers like SVMs or through kernel PCA [26]. However, they often miss hierarchical and structural information in molecular sequences. Models like Protein Bert [27] and SeqVec [28] aim to address this for proteins but struggle to generalize across different functions and perform poorly on nucleotide sequences due to varying biological properties.

3 Proposed Approach

Given an abstract space $\mathscr{X}$ (e.g., proteins, etc.), function : $\mathscr{X} \times \mathscr{X} \rightarrow \mathbb{R}$ is called a kernel function. Kernel functions are used to quantify the similarity between a pair of objects $x, y \in \mathscr{X}$. We propose a kernel function using two distance methods, namely Poincaré distance and Modified Poincaré (or M-Poincaré) distance in hyperbolic space. Hyperbolic geometry is a non-Euclidean geometry that studies spaces of constant negative curvature. Geodesics and distances are also curved in hyperbolic geometry, which are generalizations of straight lines in Euclidean geometry Figure 1 (b). The underlying hyperbolic geometry allows us to learn similarity better than Euclidean space. Figure 1 (a) shows a visualization of a particular type of tiling (isosceles) in hyperbolic geometry. In hyperbolic isosceles tiling, the tiles used are hyperbolic isosceles triangles: they have two equal sides and angles. These triangles are created by cutting a hyperbolic plane into congruent triangles, each having two sides of the same length.

We divide our method into three steps, where we first compute the pairwise distance between the data points. The second step involves computing the kernel values using the Gaussian kernel. In the end, we use kernel PCA to get the final embeddings, which are then used for underlying supervised analysis.

(a) (b)

Fig. 1: (a) Hyperboloid isosceles tiling, each triangle has two equal sides and equal angles, we can observe negative curvature and exponential growth. (b) As the curvature (-1/K) of the Poincare disk decreases, the distance between points x and y connected by geodesics (shortest paths) on the disk increases.

3.1 Distance Computation

We will now describe both of these distance methods in detail. The Poincaré distance $d(x, y)$ between a pair x, y of vectors is defined as follows:

$$d(x, y) = arcosh\left(1 + 2\frac{\|x - y\|^2}{(1 - \|x\|^2)(1 - \|y\|^2)}\right) \tag{1}$$

The notation $\|\cdot\|$ denotes the Euclidean norm, which is the square root of the sum of squared differences between corresponding elements, $\text{arcosh}(z)$ is the inverse hyperbolic cosine function, a.k.a the area hyperbolic cosine function. It is the inverse function of the hyperbolic cosine function $\cosh(z)$ and returns the value whose hyperbolic cosine is z. Therefore, the Poincaré distance between vectors x and y is calculated by taking the Euclidean distance between them, scaling it using the terms $(1 - \|x\|^2)(1 - \|y\|^2)$, multiplying by 2, adding 1, and then taking the inverse hyperbolic cosine of the result.

We propose a distance function (a modified form of Poincaré distance) that combines elements of Euclidean norms and dot products. Our distance function calculates the distance between two vectors x and y. It is a measure of dissimilarity, defined as:

$$d'(x, y) = arcosh\left(\frac{1 + \frac{2\|x-y\|^2}{(1-\|x\|^2)(1-\|y\|^2)}}{1 - \frac{(x \cdot y)^2}{\|x\|^2\|y\|^2}}\right) \tag{2}$$

where $d'(x, y)$ represents the modified Poincaré distance between vectors x and y , and $x \cdot y$ represents the dot product of vectors x and y.

The difference of modified Poincaré distance formula d' in Equation 2 from the original Poincaré distance formula d in Equation 1 is in the denominator of the fraction. In the standard Poincaré distance formula, the denominator is defined as the product of the norms of the vectors x and y. In the modified formula (Equation 2), the denominator is defined as the dot product of the vectors x and y divided by the product of their norms. This modification in the denominator allows for the inclusion of the dot product term, which can capture additional information about the similarity or dissimilarity between the

vectors x and y based on their orientations. Hence, in addition to magnitude, it incorporates the orientation information of the vectors.

The M-Poincaré distance represents a significant advancement in sequence analysis by seamlessly integrating the strengths of hyperbolic and Euclidean geometries. By combining hyperbolic geometry's hierarchical structure preservation with Euclidean geometry's compatibility with high-dimensional data, M-Poincaré offers a comprehensive representation of sequence relationships. It unites hyperbolic divergence and Euclidean norms, effectively capturing both global and local features in molecular sequences, addressing the limitations of methods relying on a single geometry. This innovative hybrid approach provides a nuanced perspective on evolutionary dynamics, making M-Poincaré a superior choice for sequence analysis that bridges existing gaps in current methodologies.

3.2 Theoretical Properties

A kernel function typically satisfies the following two properties: non-negativity, and symmetry. A kernel with these properties will loosely have the interpretation as a similarity quantification. Both $d(x, y)$ and $d'(x, y)$ represent a similarity metric on input space.

- **Non-Negativity:** $d(x, y) > 0,\ d(x, y)' > 0$
- **Symmetry:** $d(x, y) = d(y, x),\ d'(x, y) = d'(y, x)$

Remark 1. Note that the difference between $d(x, y)$ and $d'(x, y)$ is that the modified Poincaré distance has an inner product in the denominator which accounts for the extra cosine similarity measure.

Input Embedding Representation In this study, we utilize the k-mers spectrum [9] as the input to the Poincaré and modified Poincaré distance functions. The k-mers spectrum is a representation of sequences that captures the frequency of occurrence of all possible subsequences of length k, where $k > 0$. Formally, given an alphabet Σ (unique amino acid characters), the length of the k-mers spectrum is equal to $|\Sigma|^k$ (where $\Sigma = 20$ for protein sequences and $\Sigma = 4$ for DNA sequences.). The k-mers spectrum provides a compact representation of sequences by encoding the information about the frequencies of all possible k-mers. Offers several advantages over one-hot encoding (OHE) [7], which suffers from a high-dimensional and sparse representation, especially for large alphabets or long sequences. The k-mers spectrum also retains important positional and contextual information within the sequence. Unlike one-hot encoding, which treats each element independently (hence does not preserve the order of amino acids/nucleotides within molecular sequences), the k-mers spectrum considers the occurrence of subsequences and their frequencies. This enables the detection of patterns, motifs, and dependencies within the sequence, which are crucial for various sequence analysis tasks.

Remark 2. We also computed results using the one-hot encoding-based representation of molecular sequences as input. However, the results were not encouraging

compared to the k-mers-based spectrum (due to the reasons discussed above) for the proposed method. Hence we did not include those results in this paper.

3.3 Kernel Matrix From Poincaré Distance

Since Equation 1 and 2 are used to design the distance matrix between each pair of sequences, we cannot use them as kernel matrix. Generating a kernel matrix from the Poincaré-based distance matrix offers technical justifications such as capturing nonlinear relationships, flexibility with different ML algorithms, preserving essential information, and potential performance enhancements [12]. Detailed benefits of designing kernel matrix from the Poincaré distance function are given in Section 3.6. To construct the kernel matrix based on the Poincaré distance or M-Poincaré distance, we employ a Gaussian kernel. The Gaussian kernel, also known as the Radial Basis Function (RBF) kernel [29], is a popular choice for capturing complex relationships and nonlinear patterns in data. These kernel matrices are then used to design the embeddings using kernel PCA [26].

The kernel matrix is constructed by applying the Gaussian kernel function to the pairwise distances between sequences using either the Poincaré distance or the M-Poincaré distance. The Gaussian kernel is defined as:

$$K(x, y) = \exp\left(-\frac{d(x, y)^2}{2\sigma^2}\right) \tag{3}$$

where $d(x, y)$ represents either the Poincaré distance or the M-Poincaré distance between vectors x and y, and σ is a user-defined width of the kernel.

The Gaussian kernel assigns a similarity value between 0 and 1 to each pair of sequences based on their distance. The closer the sequences are, the higher the similarity value. The exponential term in Equation 3 ensures that the similarity decays exponentially with increasing distance. By applying the Gaussian kernel to the pairwise distances, we obtain a symmetric positive semi-definite kernel matrix. This matrix captures the similarity structure among the sequences, with higher values indicating similarity and lower values indicating dissimilarity. The pseudocode for computing the Poincaré kernel is given in Algorithm 1 and Figure 2 shows the flow chart. The algorithm takes a set of molecular sequences as input and computes k-mers spectrum-based numerical embeddings (line 1). It then iterates for all pairs of embeddings (upper triangle only due to symmetry property), computes the Poincaré distance using Equation 1 (line 8), and applies Gaussian kernel to get the final kernel value (line 9). The same process is performed for M-Poincaré by replacing Equation 1 (line 8) with Equation 2.

3.4 Kernel PCA [26]

Kernel PCA creates low-dimensional embeddings of sequences by leveraging the kernel matrix generated from distances like Poincaré or M-Poincaré. It extends PCA by utilizing the kernel matrix's eigenvectors to capture nonlinear patterns and structures. By applying kernel PCA, hierarchical or tree-like structures within data and complex nonlinear patterns can be accurately captured

Algorithm 1 Poincaré Kernel Matrix

Input: Set of molecular sequences S
Output: Poincaré kernel matrix K
embed ← KMERSSPECTRUM(S) ▷ Using method from [9]
K ← np.zeros($|S|$, $|S|$)
Initialize kernel matrix K with zeros
for i in range(len(embed)) **do** ▷ for all embeddings i
  for j in range(len(embed)) **do** ▷ for all embeddings j
    if $i \leq j$ **then**
      Set $\sigma_{\text{val}} = 1$
      dist ← POINCAREDIST(embed[i], embed[j]) ▷ Eq. 1
      kVal = np.exp(-$\frac{power(dist,2)}{2 \times power(\sigma_{\text{val}},2)}$) ▷ Gaussian Kernel
      Set $K[i,j] = kVal$
      Set $K[j,i] = K[i,j]$
    end if
  end for
end for
Return K

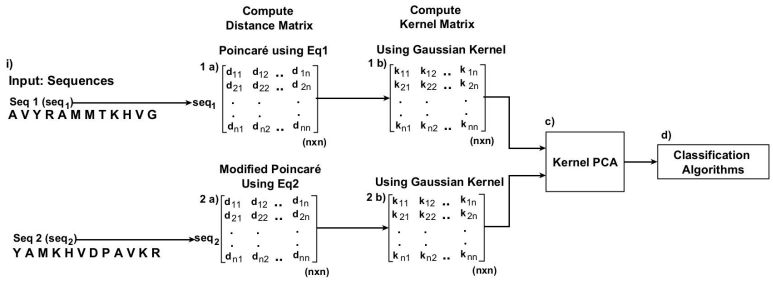


Fig. 2: Poincaré and Modified Poincaré Distance-based classification Workflow.

and embedded. Incorporating hyperbolic geometry in distance calculations and the Gaussian kernel enhances the expressiveness and informativeness of these embeddings, benefiting tasks such as clustering and classification. A detailed justification of using Kernel Matrix is given in Section 3.6. Further detailed discussion on how the Poincaré-based distance preserves hierarchical information is given in Section 3.5.

3.5 Discussion On How the Poincaré-based distance preserves hierarchical information

The utilization of the Poincaré-based distance, rooted in hyperbolic geometry, provides a distinctive perspective for capturing hierarchical connections within molecular sequences. This preservation of hierarchy stands out especially when compared to the traditional Euclidean distance.

Hyperbolic Geometry's Hierarchical Preservation: Hyperbolic geometry functions within a space that deviates from Euclidean norms by not adhering to the parallel postulate. This departure from Euclidean principles grants hyperbolic space unique qualities that are particularly adept at capturing hierarchies. Notably, volume increases exponentially in hyperbolic space as one moves away

from a given point, resulting in a “concentration effect." This mirrors the way molecular sequences evolve, with new species or variations branching out on the evolutionary tree. The Poincaré distance leverages this concentration effect, making it a powerful tool for representing evolutionary hierarchies. As sequences diverge over time, the Poincaré distance amplifies their differences, effectively representing their hierarchical relationships.

Limitations of Euclidean Distance in Capturing Hierarchy: In contrast, the Euclidean distance operates in a flat Euclidean space, where distances grow linearly which imposes limitations on capturing hierarchies. For instance, as species evolve and diverge, the linear distance growth in Euclidean space may inadequately capture the extent of divergence. This results in a loss of hierarchical information, as linear scaling does not align with the exponential branching characteristic of biological evolution. This leads to a struggle to encapsulate the intricate branching and divergence inherent in molecular sequences.

Illustrative Scenario: Imagine species A, B, and C evolving from a common ancestor. In hyperbolic space, the Poincaré distance between A and B, and between B and C, will be noticeably smaller compared to the combined distances between A and C in a pairwise Euclidean comparison. This mirrors the hierarchical relationship, indicating that A and B share a more recent common ancestor than A and C. The Poincaré-based distance inherently preserves this evolutionary tree structure, accurately reflecting the dynamics of hierarchy.

3.6 Justification of Using Kernel Matrix

Generating a kernel matrix from the Poincaré-based distance matrix offers several technical justifications and benefits:

1. **Nonlinearity:** The Poincaré distance captures the underlying hyperbolic geometry, which is a non-Euclidean space. By constructing a kernel matrix based on the Poincaré distance, we can implicitly map the data into a high-dimensional feature space, allowing for the capture of complex nonlinear relationships. This is especially beneficial when dealing with hierarchical or tree-like structures, which can be present in molecular sequences.
2. **Capturing Complex Relationships:** The Gaussian kernel used in generating the kernel matrix allows for the capture of complex relationships between sequences. It assigns higher similarity values to similar sequences and lower values to dissimilar ones. This enables the representation of intricate patterns and structures in the data, which may not be captured by linear methods.
3. **Theoretical Properties of Gaussian Kernel:** Since we are using the popular Gaussian kernel, all the underlying theoretical properties including (1) Reproducing Kernel Hilbert Space “RKHS" [30] (allows the use of kernel methods, such as support vector machines, for efficient and effective learning from data), (2) Universal Approximation [31] (can approximate any continuous function arbitrarily well in the RKHS, making it a powerful tool for

modeling complex relationships and capturing nonlinear patterns in data), (3) Mercer's Theorem [32] (guarantees that the kernel matrix is positive semi-definite), (4) Smoothness and Continuity [33] (assigns higher similarity values to points that are closer together and lower values to points that are farther apart, allowing the Gaussian kernel to capture local relationships and gradually decrease the influence of distant data points), and (5) Sensitivity to Variations [34] (capture fine-grained changes in the data distribution and assign different similarity values accordingly).

4. **Flexibility and Generalization:** The kernel matrix derived from the Poincaré-based distance matrix can be used with various machine learning algorithms that operate on kernel matrices. This provides flexibility and allows for the application of a wide range of techniques, such as kernel PCA and kernel SVM, etc. By utilizing these algorithms, we can leverage the expressive power of the kernel matrix to address different tasks, including dimensionality reduction and classification.
5. **Preserving Nonlinear Information:** Kernel PCA, performed on the kernel matrix, captures the essential nonlinear information in the data. By projecting the data onto the principal components, we retain the discriminative information while reducing the dimensionality. This can be useful when visualizing the data or when dealing with high-dimensional datasets.

4 Experimental Evaluation

This section describes the experimental setup, dataset used, and evaluation metrics. All experiments are conducted using Python on a system equipped with a Core i5 processor running at 2.4 GHz, 32 GB of memory, and the Windows 10 OS. Code can be accessed online for reproducibility [3]

Dataset Statistics We use 3 real-world molecular sequence data comprised of both proteins and nucleotide sequences as follows:

Spike7k: The spike protein sequences of the SARS-CoV-2 virus total 7000 sequences are distributed among a total of 22 coronavirus Lineage [35].

Human DNA: A total of 4380 Unaligned nucleotide sequences to classify among 7 gene families to which humans belong [36].

Coronavirus Host: The spike protein sequences belonging to various clades of the Coronaviridae family are accompanied by the infected host label total of 21 labels e.g. Humans, Bats, Chickens, etc. A total of 5558 sequences are includedViPR [37], GISAID [35].

Baseline Methods We carefully selected several well-known baselines from the literature comprised of different embedding generation categories including feature engineering (i.e. PWM2Vec [23]), traditional kernel matrix generation with kernel PCA (i.e. String Kernel [11]), neural networks (i.e. WDGRL [24]), pre-trained language model (i.e. SeqVec [28]), and pre-trained transformer (i.e. Protein BERT [27]) for protein sequences.

[3] Available in the published version

Evaluation Metrics To evaluate different models, we use average accuracy, precision, recall, F1 (weighted), F1 (macro), Receiver Operator Characteristic Curve (ROC) Area Under the Curve (AUC), and training runtime. For the metrics designed for the binary classification problem, we use the one-vs-rest approach for multi-class classification. Supervised analysis is performed using different linear and non-linear classifiers including Support Vector Machine (SVM), Naive Bayes (NB), Multi-Layer Perceptron (MLP), K-Nearest Neighbors (KNN), Random Forest (RF), Logistic Regression (LR), and Decision Tree (DT).

5 Results And Discussion

The classification results (averaged over 5 runs) for the proposed and baseline methods are given in Table 1 for Spike7k data. We can observe that the proposed Poincaré and M-Poincaré based methods consistently outperform the baseline models in terms of predictive performance. Across most evaluation metrics, these methods achieve higher scores compared to other embedding methods.

The average classification results for the Human DNA (nucleotides) dataset are reported in Table 2. Here we can observe that the proposed Poincaré kernel outperforms all other methods in terms of average accuracy, recall, weighted F1, macro F1, and ROC-AUC. For precision, the M-Poincaré performs the best. An important behavior to note here is that the pretrained Protein Bert performed very poorly on this dataset compared to its performance on Spike7k and coronavirus host datasets. The reason behind this behavior is the fact that the Protein Bert model is specifically trained on the protein sequence data. Since Human DNA consists of nucleotide sequences, this method failed to generalize in this case. On the other hand, the proposed Poincaré kernel and M-Poincaré kernel-based methods show the highest performance compared to the baselines.

The average classification results for the coronavirus host (spike proteins) dataset are reported in Table 3. Here we can observe that the proposed Poincaré kernel outperforms all other methods in terms of average accuracy, precision, recall, weighted F1, and ROC-AUC. For macro F1, the baseline PWM2Vec method performs the best. Overall the proposed Poincaré kernel and M-Poincaré kernel outperform all baselines for both proteins and nucleotide datasets in terms of average classification results. We also compared the results for Euclidean vs. Hyperbolic embeddings (Poincaré and M-Poincaré) in Section 5.1.

5.1 Euclidean vs. Hyperbolic Embedding Results

To analyze if the proposed embedding method from hyperbolic geometry (i.e. Poincaré and M-Poincaré), we generate the embeddings using Euclidean (i.e. L2) distance between the input vectors as an alternative to the Poincaré and M-Poincaré distances from Equation (1) and Equation (2), respectively, from the main paper. The remaining steps of this Euclidean-distance-based representation remain the same as discussed in Algorithm 1 in the main paper.

Embeddings	Algo.	Acc. ↑	Prec. ↑	Recall ↑	F1 (Weig.) ↑	F1 (Macro) ↑	ROC AUC ↑	Train Time (sec.) ↓
PWM2Vec	SVM	0.818	0.820	0.818	0.810	0.606	0.807	22.710
	NB	0.610	0.667	0.610	0.607	0.218	0.631	1.456
	MLP	0.812	0.792	0.812	0.794	0.530	0.770	35.197
	KNN	0.767	0.790	0.767	0.760	0.565	0.773	1.033
	RF	0.824	0.843	0.824	0.813	0.616	0.803	8.290
	LR	0.822	0.813	0.822	0.811	0.605	0.802	471.659
	DT	0.803	0.800	0.803	0.795	0.581	0.791	4.100
String Kernel	SVM	0.845	0.833	0.846	0.821	0.631	0.812	7.350
	NB	0.753	0.821	0.755	0.774	0.602	0.825	0.178
	MLP	0.831	0.829	0.838	0.823	0.624	0.818	12.652
	KNN	0.829	0.822	0.827	0.827	0.623	0.791	0.326
	RF	0.847	0.844	0.841	0.835	0.666	0.824	1.464
	LR	0.845	0.843	0.843	0.826	0.628	0.812	1.869
	DT	0.822	0.829	0.824	0.829	0.631	0.826	0.243
WDGRL	SVM	0.792	0.769	0.792	0.772	0.455	0.736	0.335
	NB	0.724	0.755	0.724	0.726	0.434	0.727	0.018
	MLP	0.799	0.779	0.799	0.784	0.505	0.755	7.348
	KNN	0.800	0.799	0.800	0.792	0.546	0.766	0.094
	RF	0.796	0.793	0.796	0.789	0.560	0.776	0.393
	LR	0.752	0.693	0.752	0.716	0.262	0.648	0.091
	DT	0.790	0.799	0.790	0.788	0.557	0.768	**0.009**
Auto-Encoder	SVM	0.699	0.720	0.699	0.678	0.243	0.627	4018.028
	NB	0.490	0.533	0.490	0.481	0.123	0.620	24.6372
	MLP	0.663	0.633	0.663	0.632	0.161	0.589	87.4913
	KNN	0.782	0.791	0.782	0.776	0.535	0.761	24.5597
	RF	0.814	0.803	0.814	0.802	0.593	0.793	46.583
	LR	0.761	0.755	0.761	0.735	0.408	0.705	11769.02
	DT	0.803	0.792	0.803	0.792	0.546	0.779	102.185
SeqVec	SVM	0.796	0.768	0.796	0.770	0.479	0.747	1.0996
	NB	0.686	0.703	0.686	0.686	0.351	0.694	0.0146
	MLP	0.796	0.771	0.796	0.771	0.510	0.762	13.172
	KNN	0.790	0.787	0.790	0.786	0.561	0.768	0.6463
	RF	0.793	0.788	0.793	0.786	0.557	0.769	1.8241
	LR	0.785	0.763	0.785	0.761	0.459	0.740	1.7535
	DT	0.757	0.756	0.757	0.755	0.521	0.760	0.1308
Protein Bert	_	0.836	0.828	0.836	0.814	0.570	0.792	14163.52
Poincaré (ours)	SVM	0.484	0.235	0.484	0.316	0.030	0.500	5.789
	NB	0.215	0.663	0.215	0.213	0.357	0.703	0.149
	MLP	0.740	0.734	0.740	0.731	0.526	0.760	18.037
	KNN	0.808	0.812	0.808	0.805	0.630	0.803	0.512
	RF	0.798	0.794	0.798	0.780	0.629	0.789	10.054
	LR	0.484	0.235	0.484	0.316	0.030	0.500	3.828
	DT	0.804	0.803	0.804	0.799	0.623	**0.833**	1.581
M-Poincaré (ours)	SVM	0.605	0.457	0.605	0.490	0.090	0.532	7.592
	NB	0.187	0.442	0.187	0.225	0.223	0.645	0.440
	MLP	0.713	0.724	0.713	0.707	0.478	0.738	5.155
	KNN	0.816	0.821	0.816	0.811	0.620	0.798	0.225
	RF	**0.851**	**0.849**	**0.851**	**0.840**	**0.669**	0.796	5.622
	LR	0.476	0.248	0.476	0.326	0.029	0.498	3.647
	DT	0.787	0.797	0.787	0.786	0.584	0.777	1.517

Table 1: Classification results (averaged over 5 runs) on **Spike7k** dataset for different evaluation metrics. The best values are shown in bold.

Embeddings	Algo.	Acc. ↑	Prec. ↑	Recall ↑	F1 (Weig.) ↑	F1 (Macro) ↑	ROC AUC ↑	Train Time (sec.) ↓
PWM2Vec	SVM	0.302	0.241	0.302	0.165	0.091	0.505	10011.3
	NB	0.084	0.442	0.084	0.063	0.066	0.511	4.565
	MLP	0.310	0.350	0.310	0.175	0.107	0.510	320.555
	KNN	0.121	0.337	0.121	0.093	0.077	0.509	2.193
	RF	0.309	0.332	0.309	0.181	0.110	0.510	65.250
	LR	0.304	0.257	0.304	0.167	0.094	0.506	23.651
	DT	0.306	0.284	0.306	0.181	0.111	0.509	1.861
String Kernel	SVM	0.618	0.617	0.618	0.613	0.588	0.753	39.791
	NB	0.338	0.452	0.338	0.347	0.333	0.617	0.276
	MLP	0.597	0.595	0.597	0.593	0.549	0.737	331.068
	KNN	0.645	0.657	0.645	0.646	0.612	0.774	1.274
	RF	0.731	0.776	0.731	0.729	0.723	0.808	12.673
	LR	0.571	0.570	0.571	0.558	0.532	0.716	2.995
	DT	0.630	0.631	0.630	0.630	0.598	0.767	2.682
WDGRL	SVM	0.318	0.101	0.318	0.154	0.069	0.500	0.751
	NB	0.232	0.214	0.232	0.196	0.138	0.517	**0.004**
	MLP	0.326	0.286	0.326	0.263	0.186	0.535	8.613
	KNN	0.317	0.317	0.317	0.315	0.266	0.574	0.092
	RF	0.453	0.501	0.453	0.430	0.389	0.625	1.124
	LR	0.323	0.279	0.323	0.177	0.095	0.507	0.041
	DT	0.368	0.372	0.368	0.369	0.328	0.610	0.047
Auto-Encoder	SVM	0.621	0.638	0.621	0.624	0.593	0.769	22.230
	NB	0.260	0.426	0.260	0.247	0.268	0.583	0.287
	MLP	0.621	0.624	0.621	0.620	0.578	0.756	111.809
	KNN	0.565	0.577	0.565	0.568	0.547	0.732	1.208
	RF	0.689	0.738	0.689	0.683	0.668	0.774	20.131
	LR	0.692	0.700	0.692	0.693	0.672	0.799	58.369
	DT	0.543	0.546	0.543	0.543	0.515	0.718	10.616
SeqVec	SVM	0.656	0.661	0.656	0.652	0.611	0.791	0.891
	NB	0.324	0.445	0.312	0.295	0.282	0.624	0.036
	MLP	0.657	0.633	0.653	0.646	0.616	0.783	12.432
	KNN	0.592	0.606	0.592	0.591	0.552	0.717	0.571
	RF	0.713	0.724	0.701	0.702	0.693	0.752	2.164
	LR	0.725	0.715	0.726	0.725	0.685	0.784	1.209
	DT	0.586	0.553	0.585	0.577	0.557	0.736	0.24
Protein Bert	_	0.542	0.580	0.542	0.514	0.447	0.675	58681.57
Poincaré Kernel (ours)	SVM	0.307	0.094	0.307	0.144	0.067	0.500	10.709
	NB	0.149	0.345	0.149	0.114	0.114	0.522	0.086
	MLP	0.660	0.660	0.660	0.659	0.616	0.779	28.152
	KNN	0.647	0.660	0.647	0.650	0.611	0.774	0.540
	RF	**0.764**	0.792	**0.764**	**0.762**	**0.756**	**0.832**	12.927
	LR	0.307	0.094	0.307	0.144	0.067	0.500	2.009
	DT	0.608	0.617	0.608	0.611	0.574	0.758	4.373
M-Poincaré Kernel (ours)	SVM	0.353	0.471	0.353	0.223	0.136	0.525	12.650
	NB	0.309	0.434	0.309	0.306	0.295	0.596	0.131
	MLP	0.677	0.684	0.677	0.678	0.656	0.804	22.044
	KNN	0.714	0.728	0.714	0.716	0.685	0.827	0.515
	RF	0.743	**0.817**	0.743	0.745	0.747	0.812	20.750
	LR	0.374	0.482	0.374	0.272	0.186	0.541	3.366
	DT	0.585	0.590	0.585	0.586	0.558	0.746	9.763

Table 2: Classification results (averaged over 5 runs) on **Human DNA** dataset for different evaluation metrics. The best values are shown in bold.

The results comparison for Euclidean vs. Hyperbolic representations are reported in Table 4 for Spike7k, Human DNA, and Covid Host dataset. We can observe that for all the datasets, the Poincaré and M-Poincaré-based representations significantly outperform the L2-based Euclidean representations for the supervised analysis. This behavior highlights that there is indeed a hidden hierarchical structure in the molecular sequences, which vanishes when we represent the sequences in Euclidean space. However, those structures are better preserved in the case of hyperbolic geometry.

An interesting observation can be observed in the case of the Human DNA dataset (in Table 4), which comprises nucleotide sequences. Since the nucleotide sequences provide direct information about the DNA composition of organisms, they are well-suited for studying evolutionary relationships at the genetic level, including mutations, genetic diversity, and population genetics. Therefore, nucleotide sequences can be used to trace evolutionary events such as gene duplications, insertions, deletions, and recombination. Based on these analyses,

preserving the hierarchical information becomes more important when dealing with the nucleotide sequences. However, we can observe in Table 4 that Euclidean distance-based embeddings show significantly lower results compared to hyperbolic distance-based embeddings. This behavior highlights the fact that if a certain type of data contains more (and important) hierarchical information, preserving that information is more important, and ignoring such information can reduce the performance of the generated embeddings.

Compared to the nucleotide sequences, the protein sequences might not capture certain types of genetic changes (e.g., silent mutations) that nucleotide sequences do. Therefore, the hidden hierarchical information in the case of protein sequences might not be as important compared to the nucleotide sequences. Due to these reasons, we can observe in Table 4 that although the Euclidean-distance-based embeddings (for Spike7k, and Coronavirus Host protein sequence datasets) show lower predictive accuracies compared to the hyperbolic distance-based representations, the difference in the results is not too significant compared to the difference observed in the case of Human DNA (nucleotide sequence) dataset.

5.2 Statistical Significance

Since the computation of the results involves splitting the data randomly into training and testing sets, the statistical significance of the results is needed. We used the student t-test and observed the p-values using averages (reported in the above results table) and standard deviations (SD) of 5 runs. Note that the SD values (of 5 runs with different random splits) for all metrics are very small i.e. < 0.002 in the majority of the cases. Therefore, we noted that p-values were < 0.05 in the majority of the cases (because SD values are very low), confirming the statistical significance of the results.

5.3 Inter-Class Embedding Interaction

We use heat maps to analyze further if the proposed kernel can identify different classes better. We took the average of the similarity values to compute a single value for each pair of classes and then computed the pairwise cosine similarity of different class's embeddings with one another. The heat map is further normalized between [0-1] to the identity pattern. The heatmaps for the k-mers spectrum-based embedding [8] and its comparison with the proposed M-Poincaré Kernel-based embeddings are reported in Figure 3, 4 and 5. For the k-mers spectrum-based heatmap, the embeddings for different labels have high similarity among them, which means that it is difficult to distinguish between different classes due to high pairwise similarities among their vectors. On the contrary, it is distinguishable for the proposed M-Poincaré Kernel-based embeddings. This essentially means that the embeddings from similar classes are highly similar to each other, indicating that the proposed methodology can accurately identify similar classes and different classes.

Embeddings	Algo.	Acc. ↑	Prec. ↑	Recall ↑	F1 (Weig.) ↑	F1 (Macro) ↑	ROC AUC ↑	Train Time (sec.) ↓
PWM2Vec	SVM	0.799	0.806	0.799	0.801	0.648	0.859	44.793
	NB	0.381	0.584	0.381	0.358	0.400	0.683	2.494
	MLP	0.782	0.792	0.782	0.778	0.693	0.848	21.191
	KNN	0.786	0.782	0.786	0.779	0.679	0.838	12.933
	RF	0.836	0.839	0.836	0.828	**0.739**	0.862	7.690
	LR	0.809	0.815	0.809	0.800	0.728	0.852	274.917
	DT	0.801	0.802	0.801	0.797	0.633	0.829	4.537
String Kernel	SVM	0.601	0.673	0.601	0.602	0.325	0.624	5.198
	NB	0.230	0.665	0.230	0.295	0.162	0.625	0.131
	MLP	0.647	0.696	0.647	0.641	0.302	0.628	42.322
	KNN	0.613	0.623	0.613	0.612	0.310	0.629	0.434
	RF	0.668	0.692	0.668	0.663	0.360	0.658	4.541
	LR	0.554	0.724	0.554	0.505	0.193	0.568	5.096
	DT	0.646	0.674	0.646	0.643	0.345	0.653	1.561
WDGRL	SVM	0.329	0.108	0.329	0.163	0.029	0.500	2.859
	NB	0.004	0.095	0.004	0.007	0.002	0.496	**0.008**
	MLP	0.328	0.136	0.328	0.170	0.032	0.499	5.905
	KNN	0.235	0.198	0.235	0.211	0.058	0.499	0.081
	RF	0.261	0.196	0.261	0.216	0.051	0.499	1.288
	LR	0.332	0.149	0.332	0.177	0.034	0.500	0.365
	DT	0.237	0.202	0.237	0.211	0.054	0.498	0.026
Auto-Encoder	SVM	0.602	0.588	0.602	0.590	0.519	0.759	2575.955
	NB	0.261	0.520	0.261	0.303	0.294	0.673	21.7474
	MLP	0.486	0.459	0.486	0.458	0.216	0.594	29.93393
	KNN	0.763	0.764	0.763	0.755	0.547	0.784	18.51143
	RF	0.800	0.796	0.800	0.791	0.648	0.815	57.90582
	LR	0.717	0.750	0.717	0.702	0.564	0.812	11072.67
	DT	0.772	0.767	0.772	0.765	0.571	0.808	121.3628
SeqVec	SVM	0.711	0.745	0.711	0.698	0.497	0.747	0.751
	NB	0.503	0.636	0.503	0.554	0.413	0.648	0.012
	MLP	0.718	0.748	0.718	0.708	0.407	0.706	10.191
	KNN	0.815	0.806	0.815	0.809	0.588	0.800	0.418
	RF	0.833	0.824	0.833	0.828	0.678	0.839	1.753
	LR	0.673	0.683	0.673	0.654	0.332	0.660	1.177
	DT	0.778	0.786	0.778	0.781	0.618	0.825	0.160
Protein Bert	_	0.799	0.806	0.799	0.789	0.715	0.841	15742.95
Poincaré (ours)	SVM	0.334	0.115	0.334	0.169	0.056	0.510	35.848
	NB	0.594	0.694	0.594	0.579	0.461	0.749	0.729
	MLP	0.752	0.750	0.752	0.744	0.463	0.733	62.323
	KNN	0.793	0.789	0.793	0.789	0.645	0.815	1.543
	RF	**0.844**	**0.847**	**0.844**	**0.836**	0.687	**0.868**	35.333
	LR	0.333	0.111	0.333	0.167	0.028	0.500	18.710
	DT	0.795	0.794	0.795	0.791	0.546	0.786	10.869
M-Poincaré (ours)	SVM	0.332	0.195	0.332	0.175	0.032	0.501	25.668
	NB	0.450	0.524	0.450	0.424	0.296	0.642	0.351
	MLP	0.607	0.599	0.607	0.598	0.268	0.627	40.005
	KNN	0.678	0.710	0.678	0.684	0.353	0.686	0.912
	RF	0.788	0.792	0.788	0.778	0.480	0.714	19.283
	LR	0.346	0.242	0.346	0.212	0.043	0.504	10.930
	DT	0.733	0.734	0.733	0.730	0.417	0.721	6.422

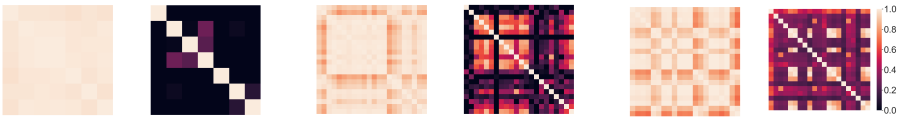
Table 3: Classification results (averaged over 5 runs) for different evaluation metrics for **Coronavirus Host Dataset**. The best values are in bold.

Dataset	Embeddings	Algo.	Acc. ↑	Prec. ↑	Recall ↑	F1 (Weig.) ↑	F1 (Macro) ↑	ROC AUC ↑	Train Time (sec.) ↓
Spike7K	Euclidean Distance	SVM	0.702	0.718	0.702	0.657	0.484	0.694	25.998
		NB	0.257	0.731	0.257	0.327	0.301	0.693	0.484
		MLP	0.741	0.719	0.741	0.713	0.521	0.737	13.700
		KNN	0.741	0.746	0.741	0.737	0.554	0.753	3.437
		RF	0.769	0.743	0.769	0.742	0.565	0.759	4.471
		LR	0.674	0.671	0.674	0.618	0.416	0.662	5.791
		DT	0.767	0.751	0.767	0.747	0.575	0.768	1.273
	Poincaré (ours)	SVM	0.484	0.235	0.484	0.316	0.030	0.500	5.789
		NB	0.215	0.663	0.215	0.213	0.357	0.703	**0.149**
		MLP	0.740	0.734	0.740	0.731	0.526	0.760	18.037
		KNN	0.808	0.812	0.808	0.805	0.630	0.803	0.512
		RF	0.798	0.794	0.798	0.780	0.629	0.789	10.054
		LR	0.484	0.235	0.484	0.316	0.030	0.500	3.828
		DT	0.804	0.803	0.804	0.799	0.623	**0.833**	1.581
	M-Poincaré (ours)	SVM	0.605	0.457	0.605	0.490	0.090	0.532	7.592
		NB	0.187	0.442	0.187	0.225	0.223	0.645	0.440
		MLP	0.713	0.724	0.713	0.707	0.478	0.738	5.155
		KNN	0.816	0.821	0.816	0.811	0.620	0.798	0.225
		RF	**0.851**	**0.849**	**0.851**	**0.840**	**0.669**	0.796	5.622
		LR	0.476	0.248	0.476	0.326	0.029	0.498	3.647
		DT	0.787	0.797	0.787	0.786	0.584	0.777	1.517
Human DNA	Euclidean Distance	SVM	0.461	0.806	0.461	0.415	0.382	0.611	16.106
		NB	0.180	0.664	0.180	0.211	0.212	0.568	0.163
		MLP	0.457	0.674	0.457	0.413	0.376	0.612	7.440
		KNN	0.368	0.376	0.368	0.365	0.322	0.607	3.734
		RF	0.474	0.689	0.474	0.438	0.407	0.625	10.083
		LR	0.412	0.789	0.412	0.336	0.266	0.568	0.954
		DT	0.468	0.629	0.468	0.434	0.401	0.625	4.034
	Poincaré Kernel (ours)	SVM	0.307	0.094	0.307	0.144	0.067	0.500	10.709
		NB	0.149	0.345	0.149	0.114	0.114	0.522	**0.086**
		MLP	0.660	0.660	0.660	0.659	0.616	0.779	28.152
		KNN	0.647	0.660	0.647	0.650	0.611	0.774	0.540
		RF	**0.764**	0.792	**0.764**	**0.762**	**0.756**	**0.832**	12.927
		LR	0.307	0.094	0.307	0.144	0.067	0.500	2.009
		DT	0.608	0.617	0.608	0.611	0.574	0.758	4.373
	M-Poincaré Kernel (ours)	SVM	0.353	0.471	0.353	0.223	0.136	0.525	12.650
		NB	0.309	0.434	0.309	0.306	0.295	0.596	0.131
		MLP	0.677	0.684	0.677	0.678	0.656	0.804	22.044
		KNN	0.714	0.728	0.714	0.716	0.685	0.827	0.515
		RF	0.743	**0.817**	0.743	0.745	0.747	0.812	20.750
		LR	0.374	0.482	0.374	0.272	0.186	0.541	3.366
		DT	0.585	0.590	0.585	0.586	0.558	0.746	9.763
COVID Host	Euclidean Distance	SVM	0.567	0.738	0.567	0.525	0.348	0.631	21.795
		NB	0.253	0.649	0.253	0.319	0.193	0.653	0.475
		MLP	0.630	0.699	0.630	0.623	0.328	0.632	12.525
		KNN	0.723	0.747	0.723	0.727	0.376	0.680	4.265
		RF	0.650	0.678	0.650	0.644	0.400	0.668	5.166
		LR	0.540	0.719	0.540	0.484	0.177	0.562	4.432
		DT	0.644	0.676	0.644	0.641	0.403	0.672	1.337
	Poincaré (ours)	SVM	0.334	0.115	0.334	0.169	0.056	0.510	35.848
		NB	0.594	0.694	0.594	0.579	0.461	0.749	0.729
		MLP	0.752	0.750	0.752	0.744	0.463	0.733	62.323
		KNN	0.793	0.789	0.793	0.789	0.645	0.815	1.543
		RF	**0.844**	**0.847**	**0.844**	**0.836**	**0.687**	**0.868**	35.333
		LR	0.333	0.111	0.333	0.167	0.028	0.500	18.710
		DT	0.795	0.794	0.795	0.791	0.546	0.786	10.869
	M-Poincaré (ours)	SVM	0.332	0.195	0.332	0.175	0.032	0.501	25.668
		NB	0.450	0.524	0.450	0.424	0.296	0.642	**0.351**
		MLP	0.607	0.599	0.607	0.598	0.268	0.627	40.005
		KNN	0.678	0.710	0.678	0.684	0.353	0.686	0.912
		RF	0.788	0.792	0.788	0.778	0.480	0.714	19.283
		LR	0.346	0.242	0.346	0.212	0.043	0.504	10.930
		DT	0.733	0.734	0.733	0.730	0.417	0.721	6.422

Table 4: Euclidean vs. Hyperbolic geometry-based embeddings classification results. The best values for respective datasets are shown in bold.

6 Conclusion

In this study, we have addressed the limitations of traditional Euclidean-based distance measurements discussed the concept of hyperbolic geometry, and proposed the use of Poincaré distance as a more effective and meaningful measure. By leveraging the unique properties of hyperbolic space, the Poincaré distance preserves the hierarchical structures present in molecular sequences. Furthermore, we introduced a modified version of the Poincaré distance, known as M-Poincaré, which combines Euclidean norms and the dot product between sequence representations. Our proposed distance measure shows superior performance in various evaluation metrics compared to existing methods. The practical



(a) k-mers (b) Poincaré (a) k-mers (b) Poincaré (a) k-mers (b) Poincaré

Fig. 3: **Human DNA**. Fig. 4: **Spike7K**. Fig. 5: **COVID Host**.
Heatmap for classes in different datasets.

implications of our research extend to domains such as medicine, biotechnology, and information retrieval, where accurate sequence comparison is crucial. Future research can explore the application of these methods in other domains along with interpretability studies.

References

1. E. Koonin and M. Y. Galperin, "Sequence—evolution—function: computational approaches in comparative genomics," 2002.
2. H. Nakagawa and M. Fujita, "Whole genome sequencing analysis for cancer genomics and precision medicine," *Cancer science*, vol. 109, no. 3, pp. 513–522, 2018.
3. R. C. Tillquist, "Low-dimensional embeddings for symbolic data science," Ph.D. dissertation, University of Colorado at Boulder, 2020.
4. A. K. Bhattacharjya and H. Ancin, "Data embedding in text for a copier system," in *International Conference on Image Processing*, vol. 2, 1999, pp. 245–249.
5. H. Cai, V. W. Zheng, and K. C.-C. Chang, "A comprehensive survey of graph embedding: Problems, techniques, and applications," *IEEE transactions on knowledge and data engineering*, vol. 30, no. 9, pp. 1616–1637, 2018.
6. P. Pezeshkpour, L. Chen, and S. Singh, "Embedding multimodal relational data for knowledge base completion," *arXiv preprint arXiv:1809.01341*, 2018.
7. K. Kuzmin, A. E. Adeniyi *et al.*, "Machine learning methods accurately predict host specificity of coronaviruses based on spike sequences alone," *Biochemical and Biophysical Research Communications*, vol. 533, no. 3, pp. 553–558, 2020.
8. S. Ali and M. Patterson, "Spike2vec: An efficient and scalable embedding approach for covid-19 spike sequences," in *IEEE Big Data (Big Data)*, 2021, pp. 1533–1540.
9. S. Ali, B. Sahoo *et al.*, "A k-mer based approach for SARS-CoV-2 variant identification," in *ISBRA*, 2021, pp. 153–164.
10. M. N. Asim, M. I. Malik, A. Dengel, and S. Ahmed, "K-mer neural embedding performance analysis using amino acid codons," in *IJCNN*, 2020, pp. 1–8.
11. S. Ali, B. Sahoo, M. A. Khan, A. Zelikovsky, I. U. Khan, and M. Patterson, "Efficient approximate kernel based spike sequence classification," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 2022.
12. M. Nickel and D. Kiela, "Poincaré embeddings for learning hierarchical representations," *Advances in neural information processing systems*, vol. 30, 2017.
13. S. Y. Ho and L. S. Jermiin, "Tracing the decay of the historical signal in biological sequence data," *Systematic biology*, vol. 53, no. 4, pp. 623–637, 2004.

14. T. Chappell, S. Geva *et al.*, "K-means clustering of biological sequences," in *22nd Australasian Document Computing Symposium*, 2017, pp. 1–4.
15. M. Deshpande and G. Karypis, "Evaluation of techniques for classifying biological sequences," in *PAKDD*, 2002, pp. 417–431.
16. G. Corso, Z. Ying, M. Pándy, P. Veličković, J. Leskovec, and P. Liò, "Neural distance embeddings for biological sequences," *NeurIPS*, pp. 18 539–18 551, 2021.
17. J. Liu and W. Wang, "Op-cluster: Clustering by tendency in high dimensional space," in *International conference on data mining*. IEEE, 2003, pp. 187–194.
18. B. Chen, X. Huang, L. Xiao, Z. Cai, and L. Jing, "Hyperbolic interaction model for hierarchical multi-label classification," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, no. 05, 2020, pp. 7496–7503.
19. J. Park, J. Cho, H. J. Chang, and J. Y. Choi, "Unsupervised hyperbolic representation learning via message passing auto-encoders," in *CVPR*, 2021, pp. 5516–5526.
20. F. Sala, C. De Sa *et al.*, "Representation tradeoffs for hyperbolic embeddings," in *International conference on machine learning*. PMLR, 2018, pp. 4460–4469.
21. C.-J. Tsai, B. Ma, and R. Nussinov, "Protein–protein interaction networks: how can a hub protein bind so many different partners?" *Trends in biochemical sciences*, vol. 34, no. 12, pp. 594–600, 2009.
22. S. Ali, "Evaluating covid-19 sequence data using nearest-neighbors based network model," *arXiv preprint arXiv:2211.10546*, 2022.
23. S. Ali, B. Bello *et al.*, "PWM2Vec: An efficient embedding approach for viral host specification from coronavirus spike sequences," *Biology*, vol. 11, no. 3, p. 418, 2022.
24. J. Shen, Y. Qu *et al.*, "Wasserstein distance guided representation learning for domain adaptation," in *AAAI conference on artificial intelligence*, 2018.
25. J. Xie, R. Girshick, and A. Farhadi, "Unsupervised deep embedding for clustering analysis," in *International conference on machine learning*, 2016, pp. 478–487.
26. H. Hoffmann, "Kernel PCA for novelty detection," *Pattern recognition*, vol. 40, no. 3, pp. 863–874, 2007.
27. N. Brandes, D. Ofer, Y. Peleg *et al.*, "Proteinbert: A universal deep-learning model of protein sequence and func." *Bioinformatics*, vol. 38, no. 8, 2022.
28. M. Heinzinger *et al.*, "Modeling aspects of the language of life through transfer-learning protein sequences," *BMC bioinformatics*, vol. 20, no. 1, pp. 1–17, 2019.
29. K. M. Tao, "A closer look at the radial basis function networks," in *Asilomar Conference on Signals, Systems and Computers*, 1993, pp. 401–405.
30. J.-W. Xu *et al.*, "An explicit construction of a reproducing gaussian kernel hilbert space," in *International Conf. on Acoustics Speech and Signal Processing*, 2006.
31. B. Hammer and K. Gersmann, "A note on the universal approximation capability of support vector machines," *neural processing letters*, vol. 17, pp. 43–53, 2003.
32. H. Q. Minh, P. Niyogi, and Y. Yao, "Mercer's theorem, feature maps, and smoothing," in *COLT*, 2006, pp. 154–168.
33. Y. He and Y. Zheng, "Short-term power load probability density forecasting based on yeo-johnson transformation quantile regression and gaussian kernel function," *Energy*, vol. 154, pp. 143–156, 2018.
34. Y. Bengio, O. Delalleau *et al.*, "The curse of highly variable functions for local kernel machines," *Advances in neural information processing systems*, vol. 18, 2005.
35. GISAID, https://www.gisaid.org/, 2022, [Online; accessed 17-Dec-2023].
36. Human DNA, in *https://www.kaggle.com/code/nageshsingh/demystify-dna-sequencing-with-machine-learning/data*, [Accessed Oct-2022].
37. B. E. Pickett, *et al.*, "Vipr: an open bioinformatics database and analysis resource for virology research," *Nucleic acids research*, vol. 40, no. D1, pp. D593–D598, 2012.