# A Universal Non-Parametric Approach For Improved Molecular Sequence Analysis

Sarwan Ali, Tamkanat E Ali,
Prakash Chourasia, and
Murray Patterson

Georgia State University
May 9, 2024

# Table of Contents

# Introduction

- Classification of molecular sequences offers profound insights into their structural, functional, and evolutionary characteristics.
- Molecular Sequence classification plays an indispensable role in advancing the fields like gene annotation, drug discovery, and evolutionary biology.
- Traditional Methods for the classification of molecular sequences include Neural Networks, language models, Feature Embedding, and Kernel functions.
- Neural Network models have been most widely used but they often require a substantial number of parameters, long training times, and demand huge data to work, making them computationally expensive and resource-demanding.

# Introduction (Existing Methods Problem)

- The Neural Networks are highly inefficient and entail high computation costs in dealing with high-dimensional biological datasets.
- The requirement of large-scale training data proves to be a challenge as it may not be readily available for certain biological datasets, particularly in low-resource or rare species scenarios making it incapable of delivering optimal accuracy.
- There is a need for low-dimensional embedding for molecular sequence analyses for which sequence data compression could prove to be useful.

# Motivation

- The availability of large-scale training data can be a significant bottleneck in developing machine-learning models for molecular sequence analysis.
- As the number of parameters in a neural network increases, the model's interpretability often decreases.
- Techniques like compression can help mitigate the challenges associated with large parameter counts while retaining model performance.

# Our (High Level) Idea

- We propose compression-based classification method that offers a novel approach to molecular sequence classification, addressing limitations present in traditional methods.
- The GZIP classification method is comprised of three core components:
  - Uses the compression-based models including Gzip and Bz2.
  - Distance Matrix computation - a non-symmetric Distance matrix using Normalized Compression Distance (NCD)
  - Convert the distance matrix into a kernel matrix and use kernel PCA to get low-dimensional embedding.

# Advantages of the GZIP Classification Method

Achieving goals:

- The proposed method reduces the parameter requirements and makes them more lightweight and accessible.
- It can efficiently handle low-resource biological datasets where labeled data is scarce or limited.

Other Benefits:

- Enhanced Accuracy: The GZIP classification method achieves superior accuracy compared to traditional methods.
- Improved Efficiency: The GZIP classification method functions more efficiently, enabling classification with low-resource data.
- Greater Scalability: The GZIP classification method provides low dimensional representation making it scalable.

# Methodology

We use basic compression algorithms like Gzip and Bz2, with Normalized Compression Distance (NCD) algorithm.

- Step 1: Concatenate the sequences.
- Step 2: Encode the individual and concatenated sequences (using UTF-8 encoding).
  - UTF-8 $->$ Variable-width character encoding (characters can be encoded using 1 to 4 bytes, depending on the character's code point in the Unicode standard) compatible with ASCII
- Step 3: Compress the encoded sequences using Gzip and Bz2.
- Step 4: Generate Distance Matrix by computing Normalized Compression Distances between each pair of molecular sequences.
- Step 5: Generate a kernel matrix using a Gaussian kernel and distance metric from step 4.
- Step 6: Employ kernel Principal Component Analysis to get the vector representations for the corresponding molecular sequence.

# Compression Method (GZIP)

- Gzip Compressor: Gzip uses very few bits to represent information, which is based on the lossless compression algorithm LZ77 (Lempel-Ziv 77 compression algorithm) [1] and dynamic Huffman algorithm [2].
  - LZ77 Compression: LZ77 is a dictionary-based compression algorithm that replaces repeated occurrences of data with references to previously occurring data. It identifies and encodes duplicate data sequences, reducing redundancy and achieving compression.
  - Dynamic Huffman Algorithm: Gzip uses a dynamic Huffman coding technique where the Huffman codes are generated dynamically based on the frequency of symbols encountered during compression. This adaptive approach optimizes the encoding process by adjusting code lengths as needed during compression.

# Compression Method (Bz2)

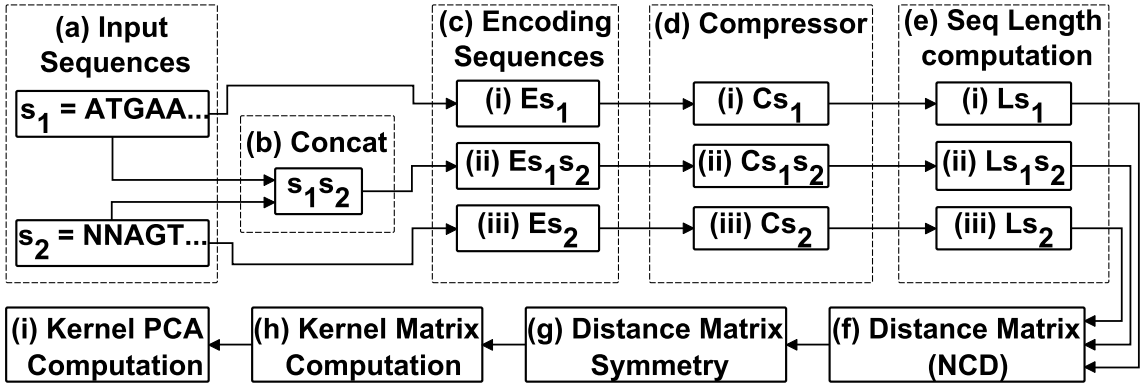
- Bz2 Compressor: Bz2 compressor is a general-purpose lossless compressor, based on the Burrows-Wheeler transform (BWT) and Huffman coding [3].
  - Burrows-Wheeler Transform (BWT): BWT reorganizes the input data to group similar characters together, making it easier for Huffman coding to encode repetitive patterns efficiently.
  - Huffman Coding: Huffman coding is used to encode the transformed data, assigning shorter codes to more frequent characters or patterns and longer codes to less frequent ones. This results in efficient compression by reducing the overall bit representation of the data.

# Justification of Employing the Kernel Matrix

Kernel matrix from the NCD-based distance matrix offers several benefits:

- **Nonlinearity:** Enable the capture of intricate nonlinear relationships.
- **Capturing Complex Relationships:** Assignment of high similarity values to similar sequences and lower values to different ones enables the representation of complex patterns and structures in the data, surpassing the limitations of linear methods.
- **Theoretical Properties of Gaussian Kernel:** Leverages several underlying theoretical properties including Reproducing Kernel Hilbert Space (RKHS) [4], Universal Approximation property [5] and Mercer's Theorem [6]. The smoothness, continuity, and sensitivity to variations further enhance its ability to capture local relationships.
- **Flexibility:** Kernel matrix can be effectively employed with various ML algorithms that exploit the expressive power of kernel matrices.
- **Enhanced Performance:** The capturing of complex relationships in the data enhances the classification of the sequences.

# Workflow




- Concatenation allows for a more comprehensive representation of the information contained in both sequences.
- This combined sequence captures not only the individual characteristics of sequences but also potential relationships or similarities between them.
- NCD $->$ Normalizes the difference in compressed lengths
- Normalization ensures that the distance metric is not biased by the absolute size of the sequences or their compression lengths alone but rather considers the relative compression gains (hence alignment free)

# Algorithm

**Algorithm** Distance matrix computation with Gzip

**Input:** `Set of sequences(S)`
**Output:** `Distance Matrix(D)`

**for** $s_1$ `in S` **do**
  $Es_1 \leftarrow encoded \;\; s_1$
  $Cs_1 \leftarrow Gzip \;\; compressed \;\; Es_1$
  $Ls_1 \leftarrow length \;\; of \;\; Cs_1$
  $D\_local \leftarrow [\,]$
  **for** $s_2$ `in S` **do**
    $Es_2 \leftarrow encoded \;\; s_2$
    $Cs_2 \leftarrow Gzip \;\; compressed \;\; Es_2$
    $Ls_2 \leftarrow length \;\; of \;\; Cs_2$
    $s_1 s_2 \leftarrow Concatenate(s_1, s_2)$
    $Es_1 s_2 \leftarrow encoded \;\; s_1 s_2$
    $Cs_1 s_2 \leftarrow Gzip \;\; compressed \;\; Es_1 s_2$
    $Ls_1 s_2 \leftarrow length \;\; of \;\; Cs_1 s_2$
    $$\text{NCD} \leftarrow \frac{Ls_1 s_2 - Min(Ls_1, Ls_2)}{Max(Ls_1, Ls_2)}$$
    $D\_local.append(NCD)$
  **end for**
  $D.append(D\_local)$
**end for**
return $D$

# Human DNA Dataset

| Name | Seq. | Classes | Sequence Statistics | | | Reference | Description |
|---|---|---|---|---|---|---|---|
| | | | Max | Min | Mean | | |
| Human DNA | 4380 | 7 | 18921 | 5 | 1263.59 | [7] | Unaligned nucleotide sequences to classify gene family to which humans belong |

- The human DNA Dataset comprised of 4380 unaligned Human DNA nucleotide sequences (having ACGT nucleotides) [7].
- The class label (total 7 unique labels) comprised of the gene family to which a human belongs, i.e., G Protein Coupled, Tyrosine Kinase, Tyrosine Phosphatase, Synthetase, Synthase, Ion Channel, and Transcription Factor.

# Baseline Models

| Method | Category | Description | Source |
|---|---|---|---|
| PWM2Vec | Feature Engineering | This method takes a biological sequence as input and generates fixed-length numerical embeddings. | [8] |
| String Kernel | Kernel Matrix | This approach designs an $n \times n$ kernel matrix that can be used with kernel classifiers or kernel PCA to obtain feature vectors based on principal components. | [9] |
| WDGRL<br>AutoEncoder | Neural Network (NN) | This method takes the one-hot representation of a biological sequence as input and generates embeddings using a neural network by minimizing a loss function. | [10]<br>[11] |
| SeqVec | Pretrained Language Model | This method takes biological sequences as input and fine-tunes the weights based on a pre-trained model to obtain the final embeddings. | [12] |
| ProteinBERT | Pretrained Transformer | This is a pre-trained protein sequence model that utilizes the Transformer/Bert architecture to classify the given biological sequences. | [13] |

# Results

| 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 |
| Autoencoder | 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 |

# Results

| Embeddings | Algo. | Acc. ↑ | Prec. ↑ | Recall ↑ | F1 (Weig.) ↑ | F1 (Macro) ↑ | ROC AUC ↑ | Train Time (sec.) ↓ |
|---|---|---|---|---|---|---|---|---|
| Gzip (ours) | SVM | 0.692 | 0.844 | 0.692 | 0.699 | 0.692 | 0.771 | 2.492 |
| | NB | 0.464 | 0.582 | 0.464 | 0.478 | 0.472 | 0.704 | 0.038 |
| | MLP | **0.831** | 0.833 | **0.831** | **0.830** | **0.813** | **0.890** | 7.546 |
| | KNN | 0.773 | 0.792 | 0.773 | 0.776 | 0.768 | 0.856 | 0.193 |
| | RF | 0.810 | **0.858** | 0.810 | 0.812 | 0.811 | 0.858 | 6.539 |
| | LR | 0.621 | 0.822 | 0.621 | 0.616 | 0.581 | 0.712 | 0.912 |
| | DT | 0.648 | 0.651 | 0.648 | 0.648 | 0.624 | 0.780 | 2.590 |
| Bz2 (ours) | SVM | 0.545 | 0.769 | 0.545 | 0.524 | 0.501 | 0.669 | 2.856 |
| | NB | 0.403 | 0.577 | 0.403 | 0.411 | 0.410 | 0.653 | 0.034 |
| | MLP | 0.696 | 0.702 | 0.696 | 0.698 | 0.670 | 0.809 | 7.601 |
| | KNN | 0.697 | 0.715 | 0.697 | 0.699 | 0.677 | 0.813 | 0.215 |
| | RF | 0.720 | 0.804 | 0.720 | 0.722 | 0.721 | 0.798 | 6.000 |
| | LR | 0.488 | 0.721 | 0.488 | 0.449 | 0.401 | 0.626 | 0.899 |
| | DT | 0.574 | 0.577 | 0.574 | 0.574 | 0.547 | 0.735 | 2.290 |

# Conclusion and Future Work

## Conclusion

- We propose a lightweight and efficient compression-based method presenting a promising new direction for molecular sequence classification.
- Combining the simplicity of the compression methods with a powerful distance computation algorithm, our method achieves SOTA.
- We Offer a more accessible and computationally efficient solution, especially in low-resource scenarios.

## Future Work

- Future research aims include applying this method in other bioinformatics domains and investigating ways to further optimize and tailor the approach for specific biological datasets.

# Thank You


J. Ziv and A. Lempel, "A universal algorithm for sequential data compression," *IEEE Transactions on information theory*, vol. 23, no. 3, pp. 337–343, 1977.


A. S. Shah and M. A. J. Sethi, "The improvised gzip, a technique for real time lossless data compression," *EAI Endorsed Transactions on CASA*, vol. 6, no. 17, 6 2019.


B. Carpentieri, "Compression of next-generation sequencing data and of dna digital files," *Algorithms*, vol. 13, no. 6, 2020.


J.-W. Xu *et al.*, "An explicit construction of a reproducing gaussian kernel hilbert space," in *Conference on Acoustics Speech and Signal Processing Proceedings*, vol. 5, 2006, pp. V–V.


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.


H. Q. Minh, P. Niyogi, and Y. Yao, "Mercer's theorem, feature maps, and smoothing," in *Conference on Learning Theory*, 2006, pp. 154–168.


Human DNA, https://www.kaggle.com/code/nageshsingh/demystify-dna-sequencing-with-machine-learning/data, 2022, [Online; accessed 10-October-2022].


S. Ali, B. Bello, P. Chourasia, R. T. Punathil, Y. Zhou, and M. Patterson, "Pwm2vec: An efficient embedding approach for viral host specification from coronavirus spike sequences," *MDPI Biology*, 2022.


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.


J. Shen, Y. Qu, W. Zhang, and Y. Yu, "Wasserstein distance guided representation learning for domain adaptation," in *AAAI conference on artificial intelligence*, 2018.


J. Xie, R. Girshick, and A. Farhadi, "Unsupervised deep embedding for clustering analysis," in *International conference on machine learning*, 2016, pp. 478–487.


M. Heinzinger, A. Elnaggar, Y. Wang, C. Dallago, D. Nechaev, F. Matthes, and B. Rost, "Modeling aspects of the language of life through transfer-learning protein sequences," *BMC bioinformatics*, vol. 20, no. 1, pp. 1–17, 2019.


N. Brandes *et al.*, "Proteinbert: A universal deep-learning model of protein sequence and func." *Bioinformatics*, vol. 38, no. 8, 2022.