PAKDD 2023

BioSequence2Vec: Efficient Embedding Generation For Biological Sequences

SARWAN ALI (sali85@student.gsu.edu)

➔ 3rd Year Ph.D. Student at Georgia State University, Atlanta, GA

Table of Content

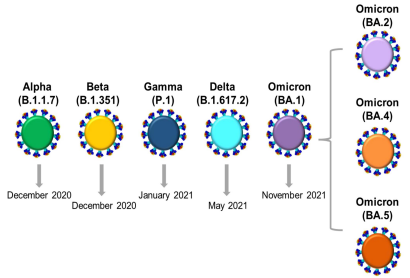
- Motivation
- Real-World Applications
- Challenges
- Previous Work
- Kernel Method
- Proposed Approach
- Dataset Statistics
- Results
- Conclusion

Motivation (Sequence Analysis)

- In-depth studies of alterations in the protein sequence to classify and predict amino acid changes in SARS-CoV-2 are crucial in
  - Understanding the immune invasion and host-to-host transmission properties of SARS-CoV-2 and its variants
  - Knowledge of mutations and variants will help identify transmission patterns of each variant that will help devise appropriate public health interventions to prevent rapid spread
  - This will also help in vaccine design and efficacy
- Understanding biological sequence classification can unravel the mysteries of genetic information and its functional implications
- Provides insights into the evolutionary relationships between organisms, helping us understand the origins and diversity of life on Earth
- Can contribute to advancements in personalized medicine, as it helps identify genetic variants associated with diseases and predict patient responses to treatments

Real World Applications

- Genomic surveillance: Tracking the spread of pathogens in terms of genomic content
- Real time identification of new and rapidly emerging coronavirus variants
- Track the spread of known coronavirus variants in new municipalities, regions, countries and continents



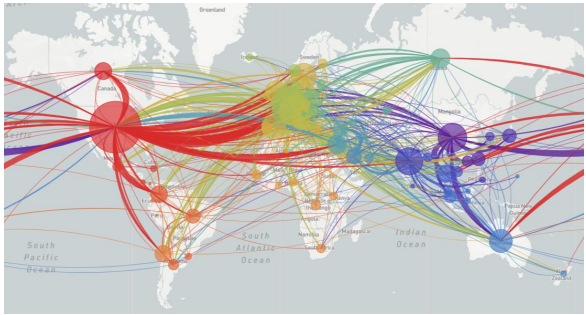


Image Source: https://www.hudsonalpha.org/genomic-surveillance-using-the-genome-to-track-and-monitor-viruses/
Image Source: https://www.mdpi.com/2075-1729/13/2/304
Image Source: https://genome.ucsc.edu/covid19.html

Challenges

- Mutations happen disproportionately in the different region of the genome
- Since new variants (for coronavirus) are emerging, not much information is available about these variants
- Generating fixed-length feature vectors from variable length sequences
- High dimensionality of generated embeddings (e.g. One Hot Encoding)
    a. Kernel-based methods are proven to be useful alternative
    b. Challenges:
    1) The computation time
    2) The memory usage (storing an n x n matrix)
    3) The usage of kernel matrices limited to kernel-based ML methods (difficult to generalize on non-kernel classifiers)

Kernel Method

- A method that allows us to apply linear classifiers to non-linear problems by mapping non-linear data into a higher-dimensional space
- Kernel-based methods (e.g., SVM) are proven useful for several machine learning (ML) tasks such as sequence classification
- There are three challenges involved with kernel methods in general
    1. Kernel computation (requires exponential complexity to compute dot product)
    2. scalability (storing n x n matrix in memory is not possible when n, the number of data points, is too large)
    3. The usage of kernel matrices limited to kernel-based ML methods (difficult to generalize on non-kernel classifiers)
- The computational complexity problem can be solved using approximate methods
- The scalability issue remains for the typical kernel methods in general
- For non-kernel classifiers, we can use kernel PCA (could result in loss of information or computationally expensive)

Research Problem

- How can we design a fixed length low dimensional representation of protein sequences that can enable us to apply sophisticated classification models on the protein sequences
  - Should use effectiveness of Kernel-based methods
  - Should avoid drawbacks of Kernel-based methods

Previous Work

- Some efforts have been done to perform classification of SARS-CoV-2 spike sequences
- However, those methods are not generalizable to disproportionality of mutations
- Although they were successful in getting high predictive accuracy, they usually proposed computationally expensive and/or high dimensional embedding representation
- Their generalizability is also not well explored on different types of biological sequences

Our Contribution

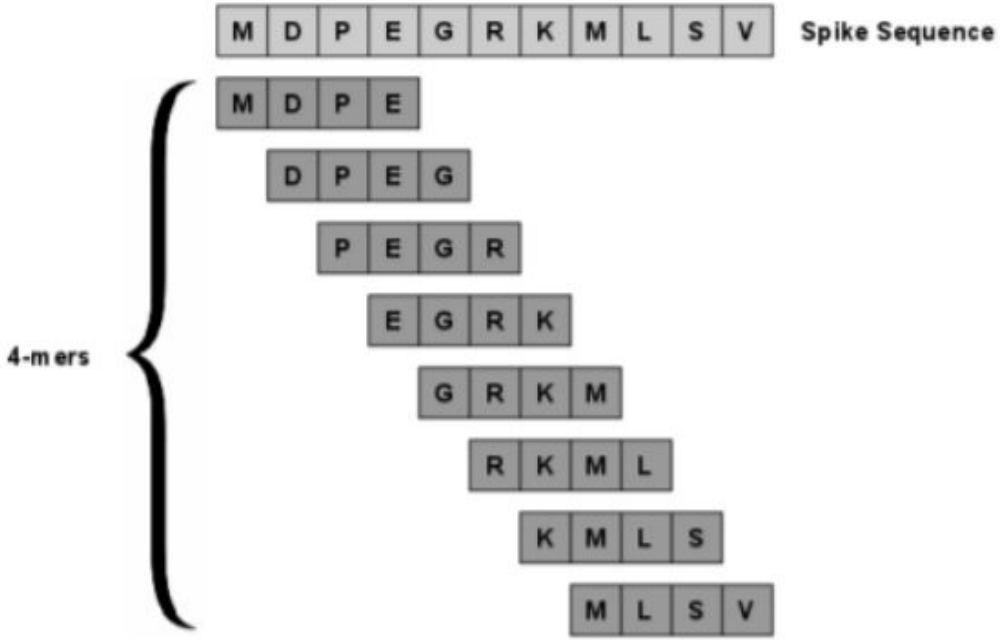
- The BioSequence2Vec representation, x for a sequence X, represents X by the random projections of $\Phi_k(X)$ on the “discrete approximation” of random directions
- It allows the application of vector space-based machine learning methods (supervised, unsupervised, visualization, etc.)
- We show that the Euclidean distance between a pair of vectors in BioSequence2Vec representation is closely related to the kernel-based proximity measure between the corresponding sequences
- We use 4-wise independent hash functions to compute $\Phi'(.)$

Proposed Approach

- Use of k-mers
- Using Hashing To Generate embedding
- Applying Classification Models

Using Hashing To Generate Sketches

Build k-mers



Using Hashing To Generate Embedding

Numerical Representation Of k-mers

- Given a kmer and Alphabet Σ => ACDEFGHIKLMNPQRSTVWXY
- For each character in k-mer
  - Find index ***i*** of the character in alphabet
  - Sort the k-mer
  - Find position ***n*** of character in sorted k-mer
  - The final numerical value **v** of the character is $\mathbf{i \times |\Sigma|^n}$
  - Repeat the above process for all characters in k-mers and concat **v** to get **nk-mer**
- Repeat the process for all k-mers

Using Hashing To Generate Embedding

Definition 1 (**4-wise Independent hash function**). *A family $\mathcal{H}$ of functions of the form $h : \Sigma^k \mapsto \{-1, 1\}$ is called 4-wise independent, or 4-universal, if a randomly chosen $h \in \mathcal{H}$ has the following properties:*

1. *for any $\alpha \in \Sigma^k$, $h(\alpha)$ is equally likely to be -1 or 1.*
2. *for any distinct $\alpha_i \in \Sigma^k$, and $m_i \in \{-1, 1\}$ $(1 \leq i \leq 4)$,*

$$Pr[h(\alpha_1) = m_1 \wedge \ldots \wedge h(\alpha_4) = m_4] = 1/2^4$$

Definition 2. *Let p be a large prime number. For integers a_0, a_1, a_2, a_3, such that $0 \leq a_i \leq p - 1$, and $\alpha \in \Sigma^k$ (represented as integer base $|\Sigma|$), the hash function $h_{a_0,a_1,a_2,a_3} : \Sigma^k \mapsto \{-1, 1\}$ is defined as*

$$h_{a_0,a_1,a_2,a_3}(\alpha) = \begin{cases} -1 & \text{if } g(\alpha) = 0 \\ 1 & \text{if } g(\alpha) = 1 \end{cases} \tag{4}$$

where

$$g(\alpha) = \left(a_0 + a_1\alpha + a_2\alpha^2 + a_3\alpha^3 \mod p\right) \mod 2 \tag{5}$$

It is well-known that the family $\mathcal{H} = \{h_{a_0,a_1,a_2,a_3} : 0 \leq a_i < p\}$ is 4-universal. Choosing a random function from this family amounts to choosing four random coefficients of polynomial, and the hash value for a k-mer α is the value of the polynomial (with random coefficients) at α modulo the prime p and modulo 2.

- The runtime of computing the embedding is tn_x, where n_x is the number of characters in X

Algorithm 1 BioSequence2Vec Computation

Input: Set $\mathcal{S}$ of sequences, integers k, p, Σ,t
Output: Embedding R
function ComputeEmbedding(S, k, p, Σ,t)
    $R = []$
    for $X \in S$ **do** ▷ for each sequence
        $\hat{\mathbf{x}} = []$
        for $i = 1$ to t **do** ▷ # of hash functions
            $a_0, a_1, a_2, a_3 \leftarrow$ RANDOM(0, p-1)
                ▷ Four random integers for coefficients of polynomial
            for $j \in |X| - k + 1$ **do** ▷ scan sequence
                $\alpha \leftarrow X[j : j + k]$ ▷ k-mer
                $\mathrm{h} \leftarrow a_0 + a_1\alpha_\Sigma + a_2\alpha_\Sigma^2 + a_3\alpha_\Sigma^3$
                ▷ α_Σ is numerical version of α base $|\Sigma|$
                $h \leftarrow (h \mod p) \mod 2$
                if $h = 0$ **then**
                    $\hat{\mathbf{x}}[i] \leftarrow \hat{\mathbf{x}}[i]$ - 1 ▷ Eq. (4)
                else
                    $\hat{\mathbf{x}}[i] \leftarrow \hat{\mathbf{x}}[i] + 1$ ▷ Eq. (4)
            $\hat{\mathbf{x}}[i] = \frac{1}{\sqrt{t}} \times \hat{\mathbf{x}}[i]$ ▷ Eq. (6)
        R.append($\hat{\mathbf{x}}$)
    return R

Hyperparameter Values

Embedding	Hyperparameter	Possible Values	Selected Value	Description
Spike2Vec	k	1 to $\|X\|$	3	k-mers value
PWM2Vec	k	1 to $\|X\|$	9	k-mers value
String Kernel	k	1 to $\|X\|$	3	k-mers value
	m	0 to $\|X\|$	0	Mismatch value
WDGRL	epc	1 to 100	100	Epoch values
Spaced k-mer	k	1 to $\|X\|$	4	k-mers value
	g	1 to $\|X\|$	9	Gapped k-mers value
Auto Encoder	epc	1 to 100	100	Epoch values
	btc	1 to 100	16	Batch Size
BioSequence2Vec	k	1 to $\|X\|$	3	k-mers value
	t	1 to 10,000	1000	No. of Hash Functions
	p	1 to 9973	4999	Big Prime Number

Table 7: Hyperparameter values for different embedding methods.

Dataset

Dataset	Detail	Source	Total Sequences	Total classes	Sequence Length		
					Min	Max	Average
Spike7k	Aligned spike protein sequences to classify lineages of coronavirus in humans	[24]	7000	22	1274	1274	1274.00
Human DNA	Unaligned nucleotide sequences to classify gene family to which humans belong	[25]	4380	7	5	18921	1263.59

Table 1: Dataset Statistics.

Lineage	Region	Labels	No. Mutation S/Gen.	No. of sequences
B.1.1.7	UK [21]	Alpha	8/17	3369
B.1.617.2	India	Delta	8/17	875
AY.4	India	Delta	-	593
B.1.2	USA	-	-	333
B.1	USA	-	-	292
B.1.177	Spain [25]	-	-	243
P.1	Brazil [31]	Gamma	10/21	194
B.1.1	UK		-	163
B.1.429	California	Epsilon	3/5	107
B.1.526	New York [41]	Iota	6/16	104
AY.12	India	Delta	-	101
B.1.160	France	-	-	92
B.1.351	South Africa [21]	Beta	9/21	81
B.1.427	California [43]	Epsilon	3/5	65
B.1.1.214	Japan	-	-	64
B.1.1.519	USA	-	-	56
D.2	Australia	-	-	55
B.1.221	Netherlands	-	-	52
B.1.177.21	Denmark	-	-	47
B.1.258	Germany	-	-	46
B.1.243	USA	-	-	36
R.1	Japan	-	-	32
Total	-	-	-	7000

Table 4: Spike7k (SARS-CoV-2) dataset statistics for 22 variants. The character '-' means that information is not available. The fourth column shows the total number of mutations in S (spike region) and full-length genome (Gen.).

Gene Family	Num. of Sequences
G Protein Coupled	531
Tyrosine Kinase	534
Tyrosine Phosphatase	349
Synthetase	672
Synthase	711
Ion Channel	240
Transcription Factor	1343
Total	4380

Table 6: Dataset Statistics for Human DNA data.

Baseline Models

Method	Category	Detail	Source	Alignment Free	Computationally Efficient	Space Efficient	Low Dim. Embedding
Spike2Vec	Feature Engineering	Take biological sequence as input and design fixed-length numerical embeddings	[8]	✓	✓	✓	✗
Spaced k-mers			[32]	✓	✓	✓	✗
PWM2Vec			[2]	✗	✓	✓	✓
WDGRL	Neural Network (NN)	Take one-hot representation of biological sequence as input and design NN-based embedding method by minimizing loss	[30]	✗	✗	✓	✓
AutoEncoder			[37]	✗	✗	✓	✓
String Kernel	Kernel Matrix	Designs $n \times n$ kernel matrix that can be used with kernel classifiers or with kernel PCA to get feature vector based on principal components	[19]	✓	✗	✗	✓
SeqVec	Pretrained Language Model	Takes biological sequences as input and fine-tunes the weights based on a pre-trained model to get final embedding	[22]	✓	✗	✓	✓
ProteinBERT	Pretrained Transformer	A pretrained protein sequence model to classify the given biological sequence using Transformer/Bert	[13]	✓	✗	✓	✓
BioSequence2Vec (ours)	Hashing	Takes biological sequence as input and design embeddings based on the kernel property of preserving pairwise distance	-	✓	✓	✓	✓

Table 2: Different methods (ours and SOTA) description.

Evaluation

Machine Learning Classifiers

- Support Vector Machine (SVM)
- Naive Bayes (NB)
- Multi-Layer Perceptron (MLP)
- K-Nearest Neighbour (KNN) (with K = 5)
- Random Forest (RF)
- Logistic Regression (LR)
- Decision Tree (DT)

Evaluation

Metrics

- Accuracy
- Precision
- Recall
- F1 (Macro)
- F1 (Weighted)
- ROC-AUC
- Training Runtime (sec.)

- We repeat experiments 5 times and report average and std. results

Results

Results

Embeddings	Algo.	Spike7k							Human DNA						
		Acc. ↑	Prec. ↑	Recall ↑	F1 (Weig.) ↑	F1 (Macro) ↑	ROC AUC ↑	Train Time (sec.) ↓	Acc. ↑	Prec. ↑	Recall ↑	F1 (Weig.) ↑	F1 (Macro) ↑	ROC AUC ↑	Train Time (sec.) ↓
BioSequence2Vec (ours)	SVM	0.848	0.858	0.848	0.841	0.681	0.848	9.801	0.555	0.554	0.555	0.543	0.497	0.700	13.251
	NB	0.732	0.776	0.732	0.741	0.555	0.771	1.440	0.263	0.518	0.263	0.244	0.239	0.572	0.095
	MLP	0.835	0.825	0.835	0.825	0.622	0.819	13.893	0.583	0.598	0.583	0.571	0.541	0.717	70.463
	KNN	0.821	0.818	0.821	0.811	0.616	0.803	1.472	0.613	0.625	0.613	0.615	0.565	0.748	0.313
	RF	**0.863**	**0.867**	**0.863**	**0.854**	**0.703**	**0.851**	2.627	**0.786**	**0.816**	**0.786**	**0.787**	**0.779**	**0.846**	1.544
	LR	0.500	0.264	0.500	0.333	0.031	0.500	11.907	0.527	0.522	0.527	0.501	0.457	0.674	29.029
	DT	0.845	0.856	0.845	0.841	0.683	0.839	0.956	0.663	0.666	0.663	0.664	0.639	0.795	4.064

Table 3: Classification results (averaged over 5 runs) on **Spike7k** and **Human**

Results

- Comparison with SOTA (average results)

Embeddings	Algo.	Spike7k Acc. ↑	Spike7k Prec. ↑	Spike7k Recall ↑	Spike7k F1 (Weig.) ↑	Spike7k F1 (Macro) ↑	Spike7k ROC AUC ↑	Spike7k Train Time (sec.) ↓	Human DNA Acc. ↑	Human DNA Prec. ↑	Human DNA Recall ↑	Human DNA F1 (Weig.) ↑	Human DNA F1 (Macro) ↑	Human DNA ROC AUC ↑	Human DNA Train Time (sec.) ↓
Spike2Vec	SVM	0.855	0.853	0.855	0.843	0.689	0.843	61.112	0.597	0.602	0.597	0.589	0.563	0.733	4.612
	NB	0.476	0.716	0.476	0.535	0.459	0.726	13.292	0.175	0.143	0.175	0.106	0.128	0.532	0.039
	MLP	0.803	0.803	0.803	0.797	0.596	0.797	127.066	0.618	0.618	0.618	0.613	0.573	0.747	22.292
	KNN	0.812	0.815	0.812	0.805	0.608	0.794	15.970	0.640	0.653	0.640	0.642	0.608	0.772	0.561
	RF	0.856	0.854	0.856	0.844	0.683	0.839	21.141	0.752	0.773	0.752	0.749	0.736	0.824	2.558
	LR	0.859	0.852	0.859	0.844	0.690	0.842	64.027	0.569	0.570	0.569	0.555	0.525	0.710	2.074
	DT	0.849	0.849	0.849	0.839	0.677	0.837	4.286	0.621	0.624	0.621	0.621	0.594	0.765	0.275
PWM2Vec	SVM	0.818	0.820	0.818	0.810	0.606	0.807	22.710	0.302	0.241	0.302	0.165	0.091	0.505	10011.3
	NB	0.610	0.667	0.610	0.607	0.218	0.631	1.456	0.084	0.442	0.084	0.063	0.066	0.511	4.565
	MLP	0.812	0.792	0.812	0.794	0.530	0.770	35.197	0.310	0.350	0.310	0.175	0.107	0.510	320.555
	KNN	0.767	0.790	0.767	0.760	0.565	0.773	1.033	0.121	0.337	0.121	0.093	0.077	0.509	2.193
	RF	0.824	0.843	0.824	0.813	0.616	0.803	8.290	0.309	0.332	0.309	0.181	0.110	0.510	65.250
	LR	0.822	0.813	0.822	0.811	0.605	0.802	471.659	0.304	0.257	0.304	0.167	0.094	0.506	23.651
	DT	0.803	0.800	0.803	0.795	0.581	0.791	4.100	0.306	0.284	0.306	0.181	0.111	0.509	1.861
String Kernel	SVM	0.845	0.833	0.846	0.821	0.631	0.812	7.350	0.618	0.617	0.618	0.613	0.588	0.753	39.791
	NB	0.753	0.821	0.755	0.774	0.602	0.825	0.178	0.338	0.452	0.338	0.347	0.333	0.617	0.276
	MLP	0.831	0.829	0.838	0.823	0.624	0.818	12.652	0.597	0.595	0.597	0.593	0.549	0.737	331.068
	KNN	0.829	0.822	0.827	0.827	0.623	0.791	0.326	0.645	0.657	0.645	0.646	0.612	0.774	1.274
	RF	0.847	0.844	0.841	0.835	0.666	0.824	1.464	0.731	0.776	0.731	0.729	0.723	0.808	12.673
	LR	0.845	0.843	0.843	0.826	0.628	0.812	1.869	0.571	0.570	0.571	0.558	0.532	0.716	2.995
	DT	0.822	0.829	0.824	0.829	0.631	0.826	0.243	0.630	0.631	0.630	0.630	0.598	0.767	2.682
WDGRL	SVM	0.792	0.769	0.792	0.772	0.455	0.736	0.335	0.318	0.101	0.318	0.154	0.069	0.500	0.751
	NB	0.724	0.755	0.724	0.726	0.434	0.727	0.018	0.232	0.214	0.232	0.196	0.138	0.517	**0.004**
	MLP	0.799	0.779	0.799	0.784	0.505	0.755	7.348	0.326	0.286	0.326	0.263	0.186	0.535	8.613
	KNN	0.800	0.799	0.800	0.792	0.546	0.766	0.094	0.317	0.317	0.317	0.315	0.266	0.574	0.092
	RF	0.796	0.793	0.796	0.789	0.560	0.776	0.393	0.453	0.501	0.453	0.430	0.389	0.625	1.124
	LR	0.752	0.693	0.752	0.716	0.262	0.648	0.091	0.323	0.279	0.323	0.177	0.095	0.507	0.041
	DT	0.790	0.799	0.790	0.788	0.557	0.768	**0.009**	0.368	0.372	0.368	0.369	0.328	0.610	0.047
Spaced k-mers	SVM	0.852	0.841	0.852	0.836	0.678	0.840	2218.347	0.746	0.749	0.746	0.746	0.728	0.844	26.957
	NB	0.655	0.742	0.655	0.658	0.481	0.749	267.243	0.177	0.233	0.177	0.122	0.142	0.533	0.467
	MLP	0.809	0.810	0.809	0.802	0.608	0.812	2072.029	0.722	0.723	0.722	0.720	0.689	0.817	126.584
	KNN	0.821	0.810	0.821	0.805	0.591	0.788	55.140	0.699	0.704	0.699	0.698	0.667	0.804	1.407
	RF	0.851	0.842	0.851	0.834	0.665	0.833	646.557	0.784	0.814	0.784	0.782	0.773	0.843	13.397
	LR	0.855	0.848	0.855	0.840	0.682	0.840	200.477	0.712	0.712	0.712	0.709	0.693	0.812	37.756
	DT	0.853	0.850	0.853	0.841	0.685	0.842	98.089	0.656	0.658	0.656	0.656	0.626	0.784	2.985
Auto-Encoder	SVM	0.699	0.720	0.699	0.678	0.243	0.627	4018.028	0.621	0.638	0.621	0.624	0.593	0.769	22.230
	NB	0.490	0.533	0.490	0.481	0.123	0.620	24.6372	0.260	0.426	0.260	0.247	0.268	0.583	0.287
	MLP	0.663	0.633	0.663	0.632	0.161	0.589	87.4913	0.621	0.624	0.621	0.620	0.578	0.756	111.809
	KNN	0.782	0.791	0.782	0.776	0.535	0.761	24.5597	0.565	0.577	0.565	0.568	0.547	0.732	1.208
	RF	0.814	0.803	0.814	0.802	0.593	0.793	46.583	0.689	0.738	0.689	0.683	0.668	0.774	20.131
	LR	0.761	0.755	0.761	0.735	0.408	0.705	11769.02	0.692	0.700	0.692	0.693	0.672	0.799	58.369
	DT	0.803	0.792	0.803	0.792	0.546	0.779	102.185	0.543	0.546	0.543	0.543	0.515	0.718	10.616
SeqVec	SVM	0.796	0.768	0.796	0.770	0.479	0.747	1.0996	0.656	0.661	0.656	0.652	0.611	0.791	0.891
	NB	0.686	0.703	0.686	0.686	0.351	0.694	0.0146	0.324	0.445	0.312	0.295	0.282	0.624	0.036
	MLP	0.796	0.771	0.796	0.771	0.510	0.762	13.172	0.657	0.633	0.653	0.646	0.616	0.783	12.432
	KNN	0.790	0.787	0.790	0.786	0.561	0.768	0.6463	0.592	0.606	0.592	0.591	0.552	0.717	0.571
	RF	0.793	0.788	0.793	0.786	0.557	0.769	1.8241	0.713	0.724	0.701	0.702	0.693	0.752	2.164
	LR	0.785	0.763	0.785	0.761	0.459	0.740	1.7535	0.725	0.715	0.726	0.725	0.685	0.784	1.209
	DT	0.757	0.756	0.757	0.755	0.521	0.760	0.1308	0.586	0.553	0.585	0.577	0.557	0.736	0.24
Protein Bert	-	0.836	0.828	0.836	0.814	0.570	0.792	14163.52	0.542	0.580	0.542	0.514	0.447	0.675	58681.57
BioSequence2Vec (ours)	SVM	0.848	0.858	0.848	0.841	0.681	0.848	9.801	0.555	0.554	0.555	0.543	0.497	0.700	13.251
	NB	0.732	0.776	0.732	0.741	0.555	0.771	1.440	0.263	0.518	0.263	0.244	0.239	0.572	0.095
	MLP	0.835	0.825	0.835	0.825	0.622	0.819	13.893	0.583	0.598	0.583	0.571	0.541	0.717	70.463
	KNN	0.821	0.818	0.821	0.811	0.616	0.803	1.472	0.613	0.625	0.613	0.615	0.565	0.748	0.313
	RF	**0.863**	**0.867**	**0.863**	**0.854**	**0.703**	**0.851**	2.627	**0.786**	**0.816**	**0.786**	**0.787**	**0.779**	**0.846**	1.544
	LR	0.500	0.264	0.500	0.333	0.031	0.500	11.907	0.527	0.522	0.527	0.501	0.457	0.674	29.029
	DT	0.845	0.856	0.845	0.841	0.683	0.839	0.956	0.663	0.666	0.663	0.664	0.639	0.795	4.064

Results

- Comparison with SOTA (std. results)

Embed. Method	ML Algo.	Acc.	Prec.	Recall	F1 weigh.	F1 Macro	ROC-AUC	Train. runtime (sec.)
BioSequence2Vec	SVM	0.007835	0.007516	0.007835	0.007297	0.022666	0.011886	0.215113
	NB	0.008105	0.023467	0.008105	0.012723	0.024591	0.010523	0.039378
	MLP	0.004398	0.005859	0.004398	0.005486	0.016899	0.006182	1.631292
	KNN	0.013624	0.010276	0.013624	0.014326	0.01157	0.006457	0.015902
	RF	0.004179	0.003099	0.004179	0.004322	0.010787	0.007188	0.118258
	LR	0.009168	0.011003	0.009168	0.01032	0.000823	0.000216	0.160578
	DT	0.008422	0.005511	0.008422	0.008296	0.017698	0.008793	0.041413

Table 9: Standard Deviation values of 5 runs for Classification results on the proposed and SOTA methods for the **Spike7k** dataset. The average results are reported in Table 3 in the main paper.

Embed. Method	ML Algo.	Acc.	Prec.	Recall	F1 weigh.	F1 Macro	ROC-AUC	Train. runtime (sec.)
BioSequence2Vec	SVM	0.00371	0.00483	0.00371	0.00581	0.01462	0.00729	1.33486
	NB	0.01361	0.01315	0.01361	0.01408	0.01650	0.00907	0.50232
	MLP	0.00623	0.00815	0.00623	0.00781	0.01115	0.00580	5.02417
	KNN	0.01255	0.01042	0.01255	0.01283	0.02426	0.01427	1.76795
	RF	0.00256	0.00630	0.00256	0.00227	0.01401	0.00696	0.96633
	LR	0.00409	0.00444	0.00409	0.00429	0.00683	0.00546	2.44877
	DT	0.00543	0.00454	0.00543	0.00594	0.01928	0.00784	0.41238

Table 11: Standard Deviation values of 5 runs for Classification results on the proposed and SOTA methods for the **Human DNA** dataset.

Data Visualization

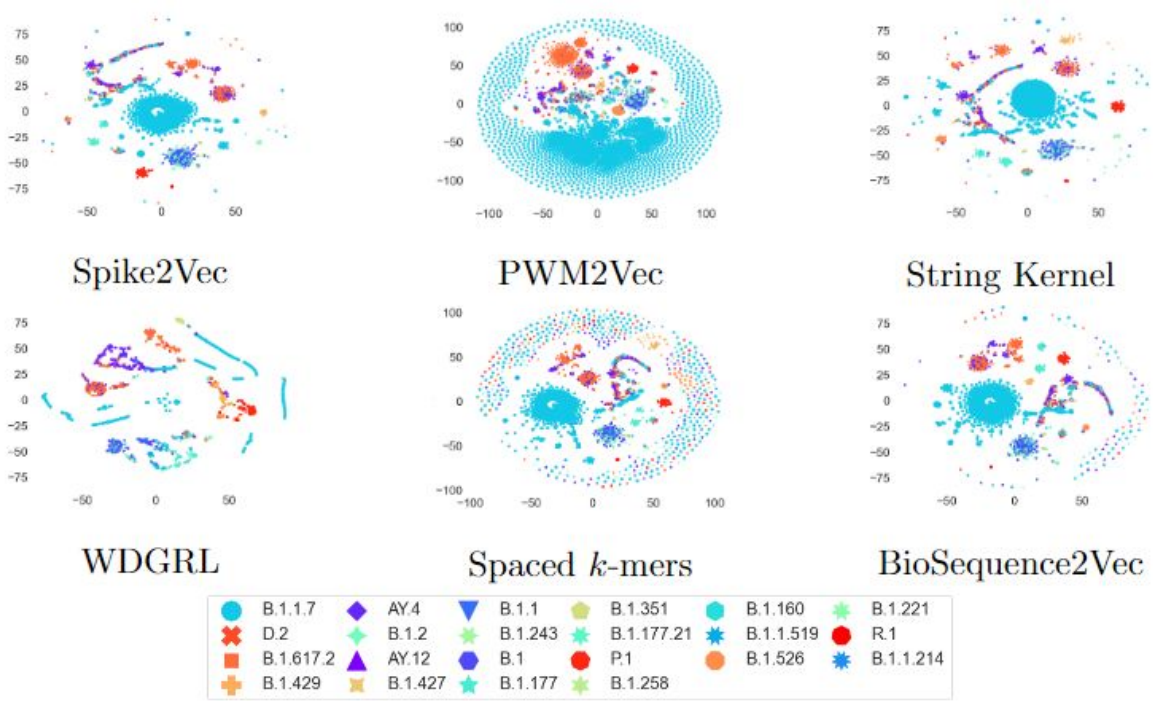
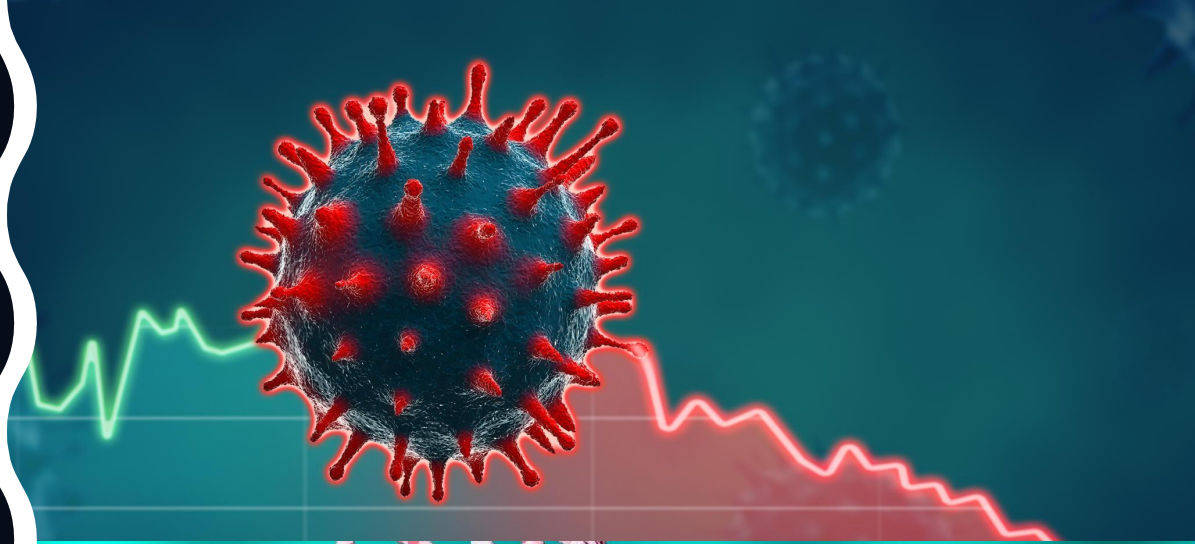
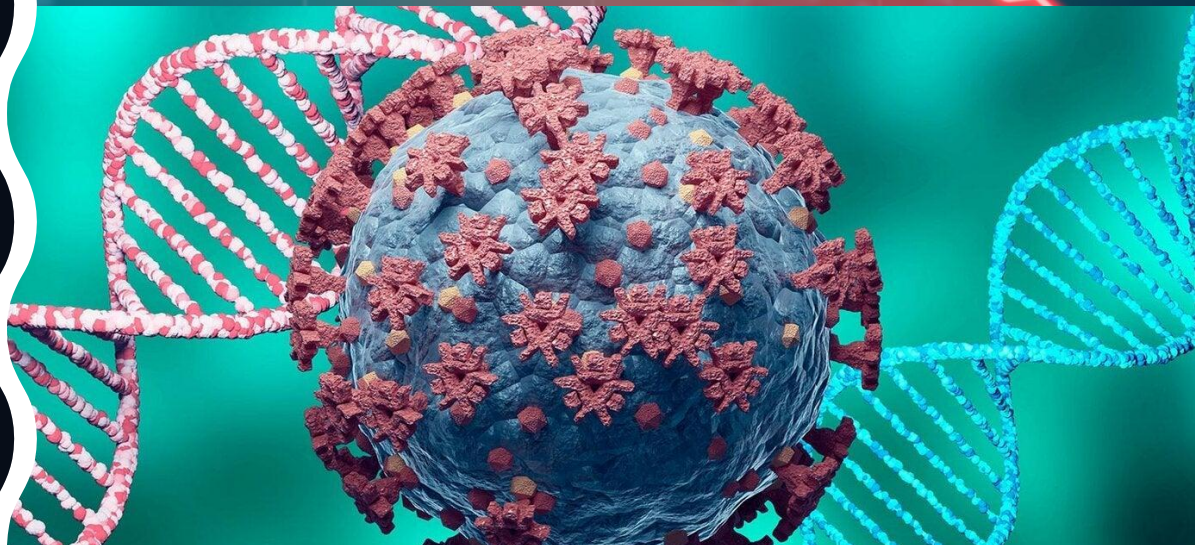


Fig. 1: t-SNE plots using different embeddings for the Spike7k dataset. This figure is best seen in color.

Conclusion

- We propose an efficient and alignment-free method to generate embeddings for biological sequences that have properties of kernel method.
- We show that our method improved the classification results compared to SOTA
- We performed extensive experiments on real-world biological sequence data to validate the proposed model using different evaluation metrics





Future Work

- Evaluating the method for larger sets of sequence data (multi-million sequences)
- Applying the proposed method to other virus data such as Zika
- Use Deep Learning models
- Evaluate the robustness

Questions!!

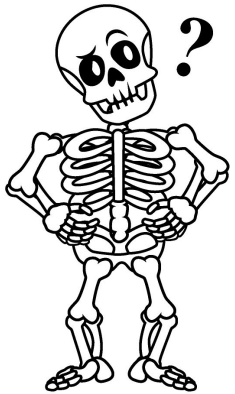


Image Source: https://charatoon.com/?id=5491

Do Reach Out For Any Questions

- **Email:** sali85@student.gsu.edu

- **Website:** https://sarwanpasha.github.io/

- **Google Scholar:** https://scholar.google.com/citations?user=9dtXSoAAAAAJ&hl=en