A Summary On

dna2vec: Consistent vector representations of variable-length k-mers

Authored By: Patrick NG

Prepared By

Annajiat Yasmin Bipasha
Id : 011 143 029

Kh. Mashruba Rahman
Id : 011 153 032

Faizah Farzana
Id : 011 153 028

Aditi Debnath
Id : 011 153 073

Submitted For The Course

CS1 416: Pattern Recognition Lab [C]

To

Sajid Ahmed

Lecturer

United International University

1 Introduction

K-mer representation is widely used for analyzing long DNA sequences. But straight forward one-hot encoding of k-mer is exponential to the length of k. For example, a 10-mer needs a bit vector of dimension 4ˆ10 = 1048576. Moreover, the distance between any arbitrary pair of one-hot vectors is equidistant, even though ATGGC should be closer to ATGGG than CACGA.

In this paper, the author presents a novel method based on the popular word embedding model word2vec (which is trained on a shallow two-layer neural network) to train distributed representations of variable-length k-mers. The author provides evidence that the summing of dna2vec vectors is akin to nucleotides concatenation and also demonstrate that correlation exists between Needleman-Wunsch similarity score and cosine similarity of dna2vec vectors.

The main contribution of this work includes:

- variable-length k-mer embedding model
- experimental evidence that shows arithmetic of dna2vec vectors is akin to nucleotides concatenation
- relationship between Needleman-Wunsch alignment and cosine similarity of dna2vec vectors
- nucleotide concatenation analogy can be constructed with dna2vec arithmetic

2 Training dna2vec Model

2.1 Stage 1: Long non-overlapping DNA fragments

In this paper, the genome sequence is fragmented by gap characters (e.g. X, -, etc). To introduce more entropy in the dataset, reverse-complement the fragments (typically a couple of thousand nucleotides) was randomly chosen.

2.2 Stage 2: Overlapping variable-length k-mers

A DNA sequence sequence S is converted into overlapping fixed length k-mer by sliding a window of length k across S. For example, TAGACTGTC can be converted into five 5-mers: {TAGAC, AGACT, GACTG, ACTGT, CTGTC}. In the variable-length case, k is sampled from the discrete uniform distribution *Uniform(k_{low}, k_{high})* to determine the size of each window. For example, a sample of k-mers of k in {3, 4, 5} could be {TAGA, AGA, GACT, ACT}.

Formally, given a sequence of length n, S = (S_1, S_2, ..., S_n) where S_i in {A,C, G, T}, S is converted into $\tilde{n} = n - k_{high} + 1$ number of k-mers:

$$f(S) = (S_{1:k_1}, S_{2:2+k_2}, \ldots S_{\tilde{n}:\tilde{n}+k_{\tilde{n}}})$$
$$k_i \sim Uniform(k_{low}, k_{high})$$

2.3 Stage 3: Two-layer neural network

An aggregate DNA k-mer embedding trained by predicting the "context" surrounding a given targeted k-mer (word2vec skip-gram), where the "context" is the set of adjacent k-mers surrounding the targeted k-mer. For example, the context of k-mer GACT would be {TAGA, AGA, ACT, CTGTC}. The context size is 10, which predicts a total of 20 k-mers, and negative sampling is used to optimize the update procedure over all words.

2.4 Stage 4: Decompose aggregated model by k-mer lengths

The aggregate model is decomposed by k-mer length to form $k_{high} - k_{low} + 1$ models for searching nearest neighbors.

3 Experiments

3.1 Similarity and nearest neighbors

The nNearestNeighborsk(v) are the n-nearest neighboring k-mers to vector v. The nearest-neighbor of dna2vec vector v in R^d is a k-mer computed with:

$$NearestNeighbor_k(v) = \underset{s \in \{A,C,G,T\}^k}{\arg\max}\ sim(v, vec(s))$$

whereas, similarity between two dna2vec vectors v,w $\mathbb{R}_d$ as the cosine similarity:

$$sim(v, w) = \frac{v \cdot w}{||v||\,||w||}$$

3.2 dna2vec arithmetic and nucleotide concatenation

In this experiment, the author demonstrates that summing dna2vec embeddings is related to concatenating k-mers. The author investigated this hypothesis by adding dna2vec embeddings of two arbitrary k-mers and examining whether their vector sum's neighbors overlap with their string concatenation.

$$NearestNeighbor_6(vec(\texttt{AAC}) + vec(\texttt{TCT})) \in \{\texttt{AACTCT}, \texttt{TCTAAC}\}$$

$$nNearestNeighbors_6(vec(\texttt{AAC}) + vec(\texttt{TCT})) \cap \{\texttt{AACTCT}, \texttt{TCTAAC}\} \neq \emptyset$$

Operands	**Concatenated**	**1-NN**	**5-NN**	**10-NN**
3-mer + 3-mer	6-mer	28.7%	80.3%	94.6%
3-mer + 4-mer	7-mer	49.9%	90.4%	97.4%
3-mer + 5-mer	8-mer	53.9%	94.0%	98.4%
4-mer + 4-mer	8-mer	73.5%	96.8%	99.2%

Table: kmers concatenation and dna2vec addition.

3.3 Relationship to global alignment similarity

In the paper, the author provided evidence that edit distance between two arbitrary k-mers is correlated with the cosine distance of their corresponding dna2vec vector.

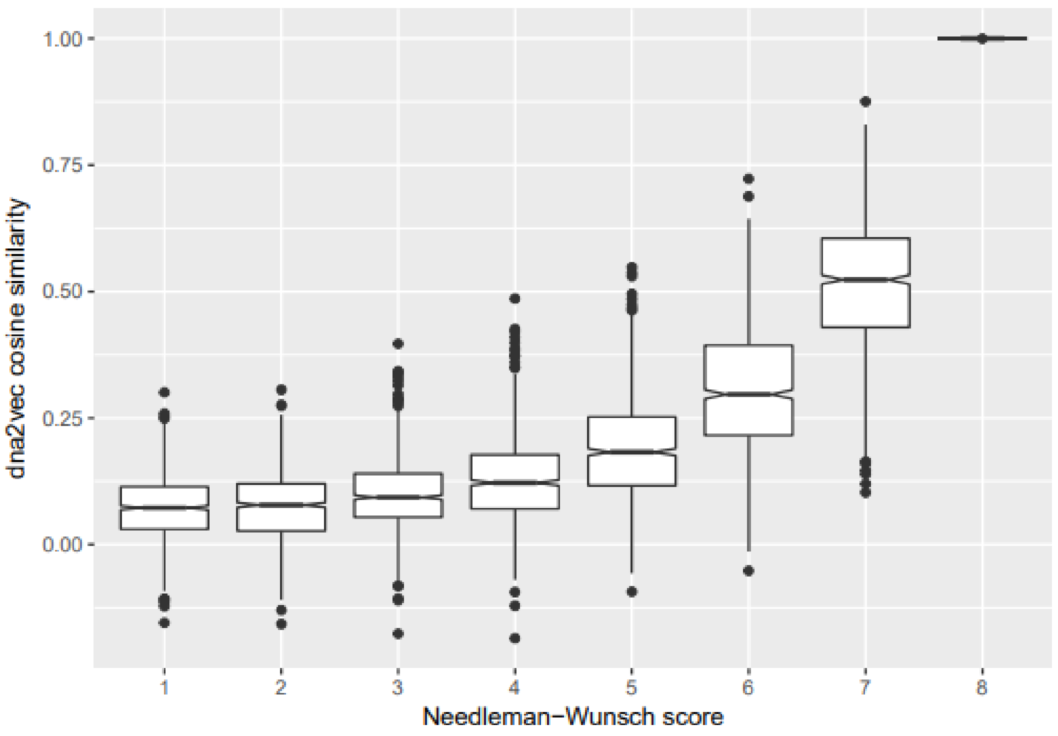

Fig: Needleman-Wunsch score and dna2vec cosine similarity.

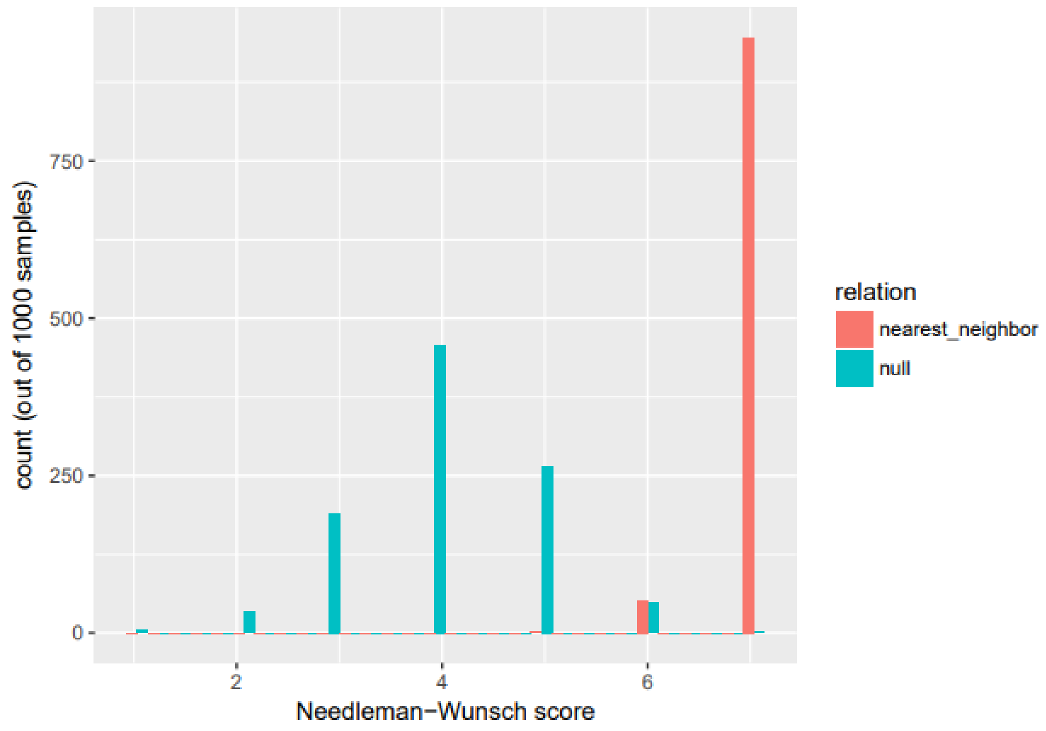

Fig: Global alignment score distribution of nearest-neighbor.

3.4 Analogy of nucleotide concatenation

$$vec(\mathbf{AC}GAT) - vec(GAT) + vec(ATC) \approx vec(\mathbf{AC}ATC)$$

$$vec(\mathbf{AC}GAT) - vec(GAT) + vec(ATC) \in_{approx} \{vec(\mathbf{AC}ATC), vec(ATC\mathbf{AC})\}$$

dimension	weak-concat scrambled-snippet 5 / 10 / 30-NN	weak-concat analogy 5 / 10 / 30-NN	strong-concat scrambled-snippet 5 / 10 / 30-NN	strong-concat analogy 5 / 10 / 30-NN
6-mer with 3-nt snippet	*1.4 / 4 / 16%*	47 / 69 / 95%	*0.6 / 1.8 / 9%*	43 / 62 / 88%
7-mer with 3-nt snippet	*2.4 / 6 / 16%*	66 / 82 / 96%	*1.5 / 3.8 / 10%*	61 / 76 / 92%
8-mer with 3-nt snippet	*3 / 6 / 19%*	67 / 82 / 95%	*2.3 / 3.8 / 11%*	62 / 77 / 91%
8-mer with 4-nt snippet	*0.7 / 1.4 / 3%*	75 / 88 / 98%	*0.3 / 1.0 / 2.4%*	69 / 83 / 95%

Fig: Global alignment score distribution of nearest-neighbor.

4 Implementation Details

- The author used gensim's Word2vec class with parameters sg=1 and window=10, which specified the usage of skipgram model and the half-size of the context window as 10, respectively.
- All of trained dna2vec vectors used in this paper has dimension size of 100 and the model used in this paper was trained with 10 epochs..
- The training step took over 3 days using gensim parameter workers=4 on a 2.66 GHz Quad-Core Intel Xeon with 8GB memory.