# Distributed Representations for Biological Sequence Analysis

Dhananjay Kimothi[*], Akshay Soni[!], Pravesh Biyani[*], and James M. Hogan[1]

[*]IIIT Delhi, India

[!]Yahoo! Research, Sunnyvale CA, USA

[1]Queensland University of Technology (QUT), Australia

## ABSTRACT

Biological sequence comparison is a key step in inferring the relatedness of various organisms and the functional similarity of their components. Thanks to the Next Generation Sequencing efforts, an abundance of sequence data is now available to be processed for a range of bioinformatics applications. Embedding a biological sequence – over a nucleotide or amino acid alphabet – in a lower dimensional vector space makes the data more amenable for use by current machine learning tools, provided the quality of embedding is high and it captures the most meaningful information of the original sequences.

Motivated by recent advances in the text document embedding literature, we present a new method, called `seq2vec`, to represent a complete biological sequence in an Euclidean space. The new representation has the potential to capture the contextual information of the original sequence necessary for sequence comparison tasks. We test our embeddings with protein sequence classification and retrieval tasks and demonstrate encouraging outcomes.

## 1. INTRODUCTION

Nucleic acids and proteins are frequently modeled through their primary structure or *sequence*, a character string representation based on their constituent sub-units, with each nucleotide or amino acid base drawn from an appropriate alphabet. While biological sequences are necessarily an abstraction from the complex three-dimensional structure of the original molecule, the representation retains information sufficient to infer a good deal about the nature of the organism and the function of its structural components. By analogy, tailored comparison of different biological sequences or sub-sequences may allow the computational biologist to infer the functional similarity or relatedness of the organisms or their components.

General purpose sequence comparison has thus received considerable attention in the bioinformatics literature, with many approaches based on alignment of a pair of sequences, whether in their entirety (*Global Alignment*, introduced by Needleman and Wunsch [22]) or by considering local regions of high similarity (*Local Alignment*, due to Smith and Waterman [25]), with distances weighted according to their biological effect via substitution matrices such as BLOSUM [12] for protein sequences. These approaches define a formal metric between sequences and may be regarded in principle as a canonical distance measure, but their direct application is limited by efficiency considerations. Each algorithm is based on dynamic programming and has computational complexity quadratic in the length of the sequences. For large sequences such an approach can be very slow, specially for applications which require many such comparisons. These limitations led naturally to the development of heuristic methods for searching large sequence databases – the most prominent of which is BLAST, The Basic Local Alignment Search Tool [1] – and more efficient algorithms for alignment of large numbers of short sequences, such as CLUSTALW [27]. Probabilistic alternatives, such as the profile Hidden Markov Model (HMMER) [9], have enjoyed considerable success for protein database search and sequence alignment, but are less widely used than the pairwise methods described above.

The explosion in the availability of biological sequence data arising from Next Generation Sequencing (see Metzker [18] for a review) has driven work toward more scalable approaches to sequence comparison, notably through so-called *alignment-free* methods (see Song et.al. [26]). Such approaches avoid the costly reliance on dynamic programming in favor of faster alternatives, usually based on tokenization of the sequence into a set of substrings or **kmers**, words of length $k$ extracted from the sequence [15, 17, 21, 28, 29].

These methods are based on two key steps. First, a lexical representation for the sequence is obtained by extracting the underlying kmers, usually by traversing a window of size $k$ across the original sequence and recording those which appear. Note that this is normally an overlapping set, so that `ACGTTA` will yield `ACG, CGT, GTT` and `TTA` for $k = 3$, though some authors work with a reduced lexicon. Given this kmer set, the second step is to find an alternative representation in a new domain (like $\mathbb{R}^n$) where a formal distance metric is defined. The main goal is that the representation should be obtained in a manner such that the pairwise similarity of any two biological sequences is approximately preserved.

ACM ISBN 978-1-4503-2138-9.
DOI: 10.1145/1235

In other words, biological sequences which are functionally or evolutionarily similar should exhibit a relatively small separation in the new space. The main advantage of these transformations is the flexibility to apply various machine learning algorithms within the new feature space. Thus the efficacy of this new representation depends on the appropriate application of ML based tools, which in turn depends on the specific bioinformatics application at hand.

This paper introduces a novel representation methodology for biological sequence comparison and classification. Although the proposed technique may be applied to sequences over an arbitrary alphabet, in this work we explore the effectiveness of this embedding in the context of protein sequence classification. Before presenting a more detailed view of our contribution, we consider the protein classification problem in more depth.

## 1.1 Protein Families

A protein family is a group of related proteins exhibiting significant structural similarity at both the sequence and molecular level. Large scale assignment of proteins to families grew out of the work of Dayhoff [7]. Sequence homology is regarded as a direct reflection of evolutionary relatedness, and members of the same protein family may also exhibit similar secondary structure through common functional units called *domains*. Domains present an additional organizing principle, allowing protein families to be grouped into super families [8] on the basis of their domains, even if sequence homology is limited for some pairs of protein entries. These families and super-families are documented in detail in databases such as SCOP [6] and their successors.

Direct determination of 3D protein structure is difficult and time consuming, requiring complex experimental techniques such as X-ray crystallography or NMR spectroscopy. For this reason experimental confirmation is available for only a small fraction of the proteins available, leading researchers to develop methods [3, 10, 24] which use only primary structure for family and thus functional identification.

## 1.2 Related Work

The problem of protein sequence labeling has a long history, and a mix of generative, discriminative, probabilistic and similarity-based clustering approaches has been employed in past. Early approaches are well summarized in Leslie et al.[16]; many of these works, including [4, 13] and that of the Leslie group adopt a discriminative approach via the support vector machine. While some of these studies include feature sets based on physical properties such as hydrophobicity, van der Waal volume, polarity and surface charge, Leslie et.al.[16] work directly with the properties of the sequence. This approach underpins the present work.

### *1.2.1 Distributed Representation of Sequences*

Recent work in the Natural Language Processing community (Mikolov et.al. [19] and Le and Mikolov [14]) has highlighted the value of *word embeddings* in the identification of terms with a similar linguistic context. In this approach, known as `word2vec`, words or phrases from the lexicon are mapped to vectors of real numbers in a low-dimensional space. By training a (shallow) Neural Network over a large text corpus, words with similar linguistic context may be mapped close by in the Euclidean space.

Asgari and Mofrad [2] recently applied the `word2vec` framework to the representation of, and feature extraction from, biological sequences. The embeddings generated from their algorithm is named `BioVec` for general biological sequences and `ProtVec` for the specific case of proteins. The representations `BioVec` and `ProtVec` may be utilized for several common bioinformatic tasks. Here, a biological sequence is treated like a sentence in a text corpus while the kmers derived from the sequence are treated like words, and are given as input to the embedding algorithm. The authors in [2] used the representations thus obtained for the protein classification task, reporting an accuracy of $93\% \pm 0.06$ for 7027 protein families.

## 1.3 Motivation and Contribution

Apart from this encouraging performance on the protein classification task, a word embeddings based approach offers other advantages. First, the size of the data vector obtained by using embeddings is much smaller in comparison to the original sequence, thus making the vectors obtained amenable for various machine learning algorithms. Indeed, one needs to compute the representation only once, while the machine learning tasks on the representations obtained are relatively inexpensive computationally, making this technique suitable for large scale bioinformatics applications. The efficacy of this approach is heavily dependent on the quality of the embeddings obtained, which in turn crucially depend on various factors like context size, vector size and finally the training algorithm. While Asgari and Mofrad gave a promising initiation towards the embeddings approach, the interplay of these factors is not well understood. It should be noted that Asgari and Mofrad used the representation obtained for each of the biological words or kmers in order to derive the representation for the whole biological sequence. One potential weakness of such an approach is that it does not *fully* capture the order in which the words occur in the original sequence.

In this paper, we estimate the representation of a complete protein sequence as opposed to obtaining those for the individual kmers. Our algorithm is based on the `doc2vec` approach, which is an extension of the original `word2vec` algorithm. We call our proposed framework `seq2vec` and use it to obtain protein sequence embeddings. Sequences obtained by `seq2vec` compares favorably with `ProtVecs` on different classification tasks and is discussed in section 3. For evaluation purposes we apply `seq2vec` to the problem of learning vectors for the set of protein sequences provided by [2]. We then use these vectors as the basis for the protein classification task, following the approach of [2] for performance evaluation of `ProtVecs`. We also use these vectors along with `ProtVecs` for classification of sequences into their respective families based on the Euclidean neighborhood of the sequence. The classification accuracy thus obtained is used to infer the quality of the embedding, with higher accuracy suggesting a better quality embedding. Further, to examine performance relative to alignment based methods, we compare these results with sequence classification based on BLAST results. Since BLAST is a retrieval tool, we use retrieval results obtained for a query sequence to predict its class. A class is assigned to the query based on the class of each of the top-N retrieved sequences according to a majority voting scheme.

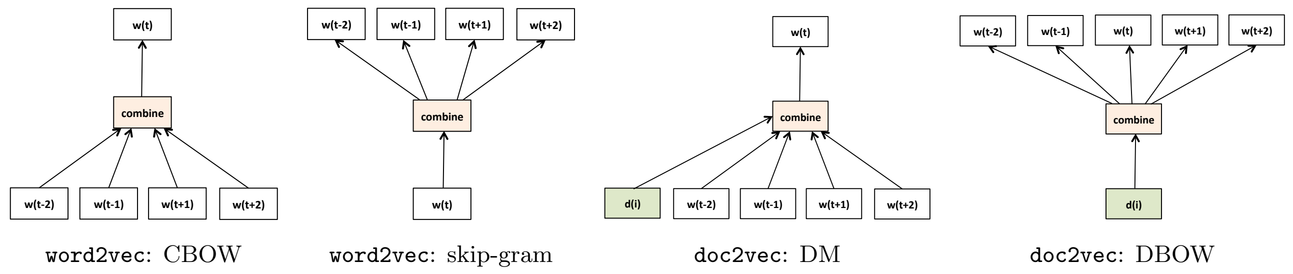

**Figure 1: Architecture for CBOW and skip-gram for `word2vec`, DM and DBOW for `doc2vec`. Notice the similarity between CBOW and DM, skip-gram and DBOW.**

## 2. OUR APPROACH

Since our approach is based on text embedding ideas from the NLP literature, we start with a brief review of `word2vec` and `doc2vec`, followed by details of `seq2vec`, which uses `doc2vec` at its core to learn embeddings for biological sequences.

### 2.1 Distributed Representations for Text

Learning representative embeddings of words and documents is a fundamental task in many NLP and machine learning applications. While simple bag-of-words based vector representation of documents have been used successfully in tasks like spam-email detection, they suffer from high dimensionality and an inability to capture complex information such as semantics and context of the underlying text. This makes their use limited for advanced tasks like multi-label learning, humor detection, hate detection, etc.

To alleviate the above mentioned shortcomings, `word2vec` [19] models – continuous bag-of-words (CBOW) and skip-gram – were recently proposed. These models are shallow, two-layer neural networks that are trained to reconstruct linguistic contexts of words. Given a corpus of text, these models assign a vector of specified dimension to each word such that words that share common contexts in the training corpus are located close to each other in the shared embedded space. `word2vec` provides two architecture choices – CBOW and skip-gram. With CBOW, the model predicts the current word by using a few surrounding context words. On the other hand, skip-gram uses the current word to predict the surrounding context words. CBOW is generally faster and is the preferred choice when a large corpus is available for training. Skip-gram is used when training data is small and it also provides better representations for infrequent words. The vectors learned from these models are powerful in terms of their ability to capture semantics i.e. words with similar meanings are embedded close to each other – for example, the word "strong" and "powerful" would be closer to each other than "strong" and "London" in the Euclidean space.

Follow-up research extended `word2vec` models to embed sentences and paragraphs – more generally, documents – in a vector space [11, 20]. The simplest approach is to use the weighted sum of word-vectors learned using `word2vec`. This is conceptually simple but does not take the order of words into consideration, which implies that sentences composed of the same words, but in a different order, will be represented by same vectors. To address these issues, `doc2vec` [14] models – distributed-memory (DM) and distributed bag-of-words (DBOW) – have been proposed. DM is akin to the CBOW architecture in `word2vec` and learns a vector representation of each word in the corpus of documents, along with the vectors for documents as well. This is done by predicting the current word using the context words and the document vector. DBOW is similar to the skip-gram architecture in `word2vec` and uses the document vector to predict the context words. DBOW is conceptually simple and does not require storage of the word vectors if the eventual task is to learn the document vectors. Note that in `doc2vec`, the word vectors are global and are updated for each context, while document vectors are local and are only updated for contexts from this document.

An overall architecture diagram for CBOW, skip-gram, DM, and DBOW is shown in Figure 1.

### 2.2 seq2vec

In order to port the advantages of using `doc2vec` in place of combining word vectors learned using `word2vec`, we improve upon `BioVec` [2], which is a `word2vec` based approach, by using the `doc2vec` framework instead. We call our approach **`seq2vec`**, a method to embed an arbitrary biological sequence into a vector space.

Unlike a document, which may be seen as an array of words with a certain linguistic structure, biological sequences are strings of letters selected from an alphabet – for DNA sequences, {A, C, G, T}, the set of nucleotides Adenine, Cytosine, Guanine and Thymine, while for protein sequences the alphabet consists of 20 letters, each representing an amino acid. Since there is no clear notion of words in biological sequences, we need to break these sequences during the preprocessing stage to represent them as an array of kmers – a unit made of $k$ consecutive letters.

We employ two ways of processing the sequences – non-overlapping and overlapping processing. In non-overlapping processing, we generate $k$ new sequences from a given sequence by shifting the starting point from where we start constructing the kmers. For example, given a hypothetical sequence

QWERTYQWERTY

and $k = 3$, we generate three new sequences as follows:

Seq 1: QWE RTY QWE RTY
Seq 2: WER TYQ WER
Seq 3: ERT YQW ERT.

In overlapping processing, we generate one sequence by breaking the original sequence into overlapping kmers. While the overlap itself is a parameter, in this paper we focus on kmers shifted by one letter at a time. For instance, overlapping

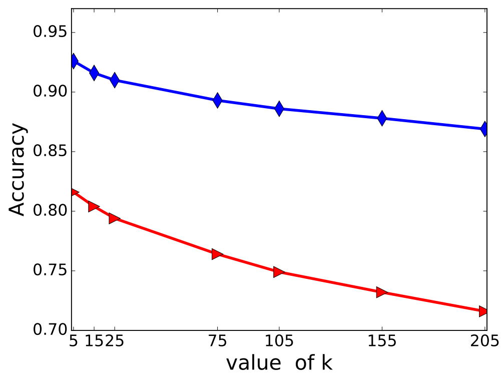

**Figure 2: Performance of non-overlapping (–◆–◆–) vs overlapping (–▶–▶–) pre-processing for the task of kNN based classification for different number of nearest neighbors. This is with best choice of other parameters – dimension: 250, `kmer` size: 3, context size: 5 (25 for overlapping). It is clear that non-overlapping processing performs significantly better than the overlapping processing.**

processing of the hypothetical sequence above yields:

QWE WER ERT RTY TYQ YQW QWE WER ERT RTY.

After this preprocessing, each sequence is treated as a document with kmers as corresponding words. We then apply the standard `doc2vec` model to learn the vector representation of each sequence in the corpus. The choice of using either DM, or DBOW, or both (as in combining representations learned from DM and DBOW) depends on the underlying task, and needs to be made using cross-validation. Similarly, other learning hyper-parameters like context window size, vector space dimension, hierarchical softmax and/or negative sampling for learning procedure, and the threshold for sub-sampling high-frequency words are all selected using cross-validation.

## 3. EXPERIMENTAL RESULTS

In order to demonstrate the advantages of `seq2vec` over `BioVec`, we carefully designed our experiments to compare the two sets of embeddings on multiple fronts. First, we use the protein vectors learned using `seq2vec` and `ProtVecs` as features and compare them for the task of protein classification using SVMs. This involves experiments with both binary and multiclass classification problems. Next, since distributed representations embed similar sequences in proximity to each other, we use $k$-nearest neighbors (kNN) to retrieve the $k$ nearest sequences in the vector space and see how successful are we in predicting the family of a test sequence based on a majority vote. Finally, we compare this retrieval task with BLAST, the standard approach for rapid determination of sequence similarity, and notice that at present BLAST performs significantly better than the embedding based approaches. This motivates our discussion on future work in the next section.

We employ `seq2vec` for learning distributed vector representations of 324018 protein sequences from [2] which are originally taken from the Swiss-Prot database [5], they also provided a meta file which contains the family labels along with other description of protein sequences. We extracted the family label information from this meta file. To our surprise we found that the unique family labels extracted from the meta file are 6097, which is different than the total number of families(7027) reported by [2]. On further probing we found that in classification result file provided by [2] there are multiple entries for some families but the number of samples corresponding to such entries are different. Adding up the number of sequences for such entries for a given family matches to the actual number of sequences as per the meta file, which suggests that the samples of few families are splitted into sub groups by [2] in their experimental setup, which is the reason for difference in number of families reported in this work and [2]. In our experiments we utilized the labels from meta file and based on it categorized sequences into 6097 families, out of these, 3861 families have fewer than 10 sequences, 2472 families have number of sequences in the range 11 - 100, 609 families have number of sequences in the range 101 - 1000, and remaining 25 families contain more than 1000 sequences.

In order to choose a good set of hyper-parameters – including context size, vector space dimension, non-overlapping vs. overlapping pre-processing, model architecture – we did a search over a range of these parameters for the task of kNN based classification and chose parameters which gave the best accuracy for sequence retrieval from the same family as the query sequence on a separate validation set. We did this experiment using sequences from the 25 biggest families having more than 1000 sequences. Overall, from this experiment we chose dimension: 250, kmer size: 3, context size: 5, pre-processing: non-overlapping, and architecture: DM (see Figure.2).All the experiments were performed using Gensim [23]

### 3.1 Protein Classification

As discussed above we first compare representations learned from `seq2vec` and `BioVec` for the task of binary and multi-class classification on the following three metrics:

$$\begin{aligned}\text{Specificity} &= \frac{\text{TN}}{\text{TN}+\text{FP}}\\ \text{Sensitivity} &= \frac{\text{TP}}{\text{TP}+\text{FN}}\\ \text{Accuracy} &= \frac{\text{TN}+\text{TP}}{\text{TN}+\text{TP}+\text{FN}+\text{FP}}\end{aligned}$$

where TN is true-negatives, TP is true-positives, FP is false-positives, and FN is false-negatives. Specificity (true negative rate) measures the proportion of negatives that are correctly classified. Sensitivity (true positive rate or recall) measures the proportion of positives that are correctly classified. Accuracy measures the proportion of examples that are correctly classified. Depending on the application, the relative importance of these metrics may change.

Note that we use the vectors provided by authors of [2] for doing experiments involving `ProtVec`[1].

#### 3.1.1 *Binary Classification*

This experiment exactly follows the setting used by [2]. The task is to distinguish sequences of a given family from

[1]http://llp.berkeley.edu/

| | # of classes | Specificity (%) | Sensitivity (%) | Accuracy (%) |
|---|---|---|---|---|
| `seq2vec` | 1000 | **97.49 ± 0.05** | 92.72 ± 0.06 | **95.10 ± 0.05** |
| | 2000 | **97.01 ± 0.04** | 93.07 ± 0.07 | **95.04 ± 0.05** |
| | 3000 | **96.67 ± 0.05** | 93.18 ± 0.08 | **94.92 ± 0.06** |
| `ProtVec` | 1000 | 91.61 ± 0.05 | **95.40 ± 0.04** | 93.50 ± 0.05 |
| | 2000 | 90.10 ± 0.07 | **95.75 ± 0.05** | 92.92 ± 0.05 |
| | 3000 | 88.61 ± 0.09 | **95.95 ± 0.05** | 92.27 ± 0.06 |

**Table 1: Binary Classification: performance of `seq2vec` and `ProtVecs` for top 1000, 2000 and 3000 protein families.**

| | Precision(%) | Sensitivity (%) | Accuracy (%) |
|---|---|---|---|
| `seq2vec` | **83.37 ± 0.052** | **81.69 ± 0.067** | **81.29 ± 0.057** |
| `ProtVec` | 79.01 ± 0.071 | 76.78 ± 0.082 | 76.70 ± 0.080 |

**Table 2: Multiclass Classification: Performance of `seq2vec` and `ProtVec` for classification of sequences from top 25 families.**

all other families. For each family, the corresponding sequences constitute the positive class, and the same number of sequences, randomly chosen from rest of the families, forms the negative class. A binary classifier is trained and the metrics are computed using 10-fold cross validation.

While the results reported in [2] also trained classifiers for families with fewer than 10 sequences, we included families with at least 10 sequences in order to perform a proper 10-fold validation. Also, more data makes the learned classifiers more stable and the reported results are stable and reproducible.

The exact choice of SVM hyper-parameters, including the type of kernel and $C$, are not clearly mentioned in [2]. We chose to use an SVM classifier with linear kernel and C equal to 1.0, these choices being made based on grid search. We did three set of experiments with the 1000, 2000, and 3000 biggest families. The results reported in Table 1 show that the accuracy and specificity obtained using `seq2vec` is consistently better than `ProtVec` while `ProtVec` performs slightly better in terms of sensitivity than `seq2vec`.

### *3.1.2 Multiclass Classification*

The very high values for metrics as reported in Table 1 for the binary classification task is possibly due to the way in which the negative class is constructed – by randomly selecting sequences from all the remaining families. Since the number of possible negative sequences is very large, the randomly selected sequences would generally be far away from the positive sequences, making the positive and negative classes far apart, resulting in an easy classification problem.

To alleviate this issue with the experimental setup of [2], we adopt a different approach. In order to compare `seq2vec` with `ProtVecs`, we do a multiclass classification to classify the sequences into their corresponding families. We did this experiment using the 25 biggest families consisting of more than 1000 sequences each. Then we trained a multiclass SVM classifier with a linear kernel using the one-vs-rest strategy. We set $C = 1$ for `seq2vec` and $C = 7.5$ for `ProtVecs` based on grid search. The evaluation metrics are then computed using 10-fold cross validation and the results are reported in Table 2.

As shown in Table 2, `seq2vec` perform much better than `ProtVecs` and the improvement is 4-6% for all three metrics. Note that this experimental setting provides a much harder challenge than the earlier binary classification and the results prove that the embeddings learned using `seq2vec` are superior t learned using `ProtVecs` for this task.

## 3.2 kNN based Classification

In an ideal scenario, distributed representations should embed sequences in such a manner that different families are separated from each other and the sequences belonging to same class are clustered together. This provides us with a way to do classification using a $k$-nearest neighbor based approach. We first embed the training and test data in a vector space using either `seq2vec` or `ProtVecs`. In order to assign a family to a test sequence, we find its $k$-nearest training sequences in the vector space and predict the family using a majority vote. Here we assume that we embed both the train and test data simultaneously. If test data is not available for embedding, we can later use gradient-descent to learn the vector representation of the test sequences as explained in [14].

We performed this experiment for the 25 biggest families consisting of more than 1000 sequences each, reporting results based on 10-fold cross validation in Figure 3. It is clear that `seq2vec` based embeddings are significantly better than the ones produced by `ProtVecs`.

### *3.2.1 Classification using BLAST*

The BLAST heuristic finds the most similar sequences to a given query sequence based on a hit extension strategy. BLAST ranks matches to a query sequence according to an estimate of the statistical significance of the match, and reports the base identity via a bit score. This requires processing of actual sequences to compute similarity, while we represent these sequences in a low-dimensional space us-

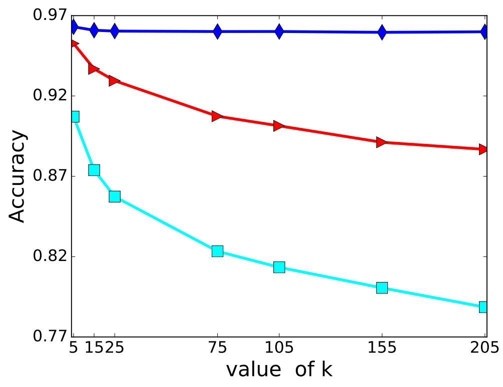

**Figure 3: kNN Classification: Performance of seq2vec (–▶–▶–) and ProtVecs (–■–■–) as compared to that of BLAST (–◆–◆–) as a function of $k$ used for kNN.**

ing `seq2vec`. Since BLAST is in some sense a retrieval tool, in order to do classification, we look at the top-$k$ results returned and use majority voting to assign the query to a protein family. This is similar to the setup of kNN classification in Section 3.2 and hence can be compared directly. The results are reported in Figure 3.

It is clear that while `seq2vec` managed to perform significantly better than `ProtVecs`, BLAST is markedly superior for this task. We note that the scoring inherent in BLAST closely follows the evolutionary relationships observed across large numbers of protein alignments, and on which protein families are defined. An important focus for future work is thus to explore the extent to which this information can be captured implicitly in the context modelling of our approach, and the performance of embedding based approaches improved to become genuinely competitive with BLAST.

## 4. CONCLUSIONS AND FUTURE WORK

In this paper, we proposed `seq2vec` – a distributed representation framework for biological sequences. We demonstrated the advantage of our approach by utilizing the learned embeddings for protein classification problems. `seq2vec` performs significantly better than the current state-of-the-art embedding based approach.

Going forward, we would like to learn embeddings for other biological sequences and use them for tasks such as homology detection in nucleotide sequences, functional annotation, and the prediction of structure and regulatory relationships. As noted in the last section, there is a significant performance gap between `seq2vec` and BLAST for protein classification, and this may perhaps be addressed by embeddings tailored to reflect the similarity weightings inherent in established substitution matrices. Moreover, the embedding approach may allow us to capture important context where substitution matrices or other alignment derived scoring methods are not available, and to improve the state of the art for a range of bioinformatics tasks which presently rely on more complex and costly feature sets.

## References

[1] Altschul, S. F., Gish, W., Miller, W., Myers, E. W., and Lipman, D. J. Basic local alignment search tool. *Journal of molecular biology 215*, 3 (oct 1990), 403–10.

[2] Asgari, E., and Mofrad, M. R. K. Continuous Distributed Representation of Biological Sequences for Deep Proteomics and Genomics. *PLOS ONE 10*, 11 (nov 2015), e0141287.

[3] Bork, P., Dandekar, T., Diaz-Lazcoz, Y., Eisenhaber, F., Huynen, M., and Yuan, Y. Predicting function: from genes to genomes and back. *Journal of molecular biology 283*, 4 (nov 1998), 707–25.

[4] Cai, C. Z., Han, L. Y., Ji, Z. L., Chen, X., and Chen, Y. Z. SVM-Prot: Web-based support vector machine software for functional classification of a protein from its primary sequence. *Nucleic acids research 31*, 13 (jul 2003), 3692–7.

[5] Consortium, U., et al. Uniprot: a hub for protein information. *Nucleic acids research* (2014), gku989.

[6] Conte, L. L., Ailey, B., Hubbard, T. J., Brenner, S. E., Murzin, A. G., and Chothia, C. Scop: a structural classification of proteins database. *Nucleic acids research 28*, 1 (2000), 257–259.

[7] Dayhoff, M. Computer analysis of protein sequences. In *Computers in Life Science Research*. Springer, 1974, pp. 9–14.

[8] Dayhoff, M., McLaughlin, P., Barker, W., and Hunt, L. Evolution of sequences within protein superfamilies. *Naturwissenschaften 62*, 4 (1975), 154–161.

[9] Eddy, S., et al. Hmmer-biosequence analysis using profile hidden markov models. *URL http://hmmer. janelia. org* (2007).

[10] Enright, A. J., Van Dongen, S., and Ouzounis, C. A. An efficient algorithm for large-scale detection of protein families. *Nucleic acids research 30*, 7 (apr 2002), 1575–84.

[11] Grefenstette, E., Dinu, G., Zhang, Y.-Z., Sadrzadeh, M., and Baroni, M. Multi-step regression learning for compositional distributional semantics. *arXiv preprint arXiv:1301.6939* (2013).

[12] Henikoff, S., and Henikoff, J. G. Amino acid substitution matrices from protein blocks. *Proceedings of the National Academy of Sciences 89*, 22 (1992), 10915–10919.

[13] Huynen, M., Snel, B., Lathe, W., and Bork, P. Predicting protein function by genomic context: quantitative evaluation and qualitative inferences. *Genome research 10*, 8 (aug 2000), 1204–10.

[14] Le, Q. V., and Mikolov, T. Distributed representations of sentences and documents. In *ICML* (2014), vol. 14, pp. 1188–1196.

[15] Leimeister, C.-A., Boden, M., Horwege, S., Lindner, S., and Morgenstern, B. Fast alignment-free sequence comparison using spaced-word frequencies. *Bioinformatics* (2014), btu177.

[16] Leslie, C. S., Eskin, E., Cohen, A., Weston, J., and Noble, W. S. Mismatch string kernels for discriminative protein classification. *Bioinformatics 20*, 4 (2004), 467–476.

[17] Melko, O. M., and Mushegian, A. R. Distribution of words with a predefined range of mismatches to a dna probe in bacterial genomes. *Bioinformatics 20*, 1 (2004), 67–74.

[18] Metzker, M. L. Sequencing technologies [mdash] the next generation. *Nat Rev Genet 11*, 1 (2010), 31–46.

[19] Mikolov, T., Corrado, G., Chen, K., and Dean, J. Efficient Estimation of Word Representations in Vector Space. *Proceedings of the International Conference on Learning Representations (ICLR 2013)* (2013), 1–12.

[20] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems* (2013), pp. 3111–3119.

[21] Mizuta, S., and Yamaguchi, K. A new graphical representation of dna sequences using symmetrical vector assignment. *Review of Bioinformatics and Biometrics* (2014).

[22] Needleman, S. B., and Wunsch, C. D. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of molecular biology 48*, 3 (mar 1970), 443–53.

[23] Řehůřek, R., and Sojka, P. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks* (Valletta, Malta, May 2010), ELRA, pp. 45–50. http://is.muni.cz/publication/884893/en.

[24] Remmert, M., Biegert, A., Hauser, A., and Söding, J. HHblits: lightning-fast iterative protein sequence searching by HMM-HMM alignment. *Nature Methods 9*, 2 (dec 2011), 173–175.

[25] Smith, T. F., and Waterman, M. S. Identification of common molecular subsequences. *Journal of molecular biology 147*, 1 (mar 1981), 195–7.

[26] Song, K., Ren, J., Reinert, G., Deng, M., Waterman, M. S., and Sun, F. New developments of alignment-free sequence comparison: measures, statistics and next-generation sequencing. *Briefings in bioinformatics 15*, 3 (may 2014), 343–53.

[27] Thompson, J. D., Gibson, T., Higgins, D. G., et al. Multiple sequence alignment using clustalw and clustalx. *Current protocols in bioinformatics* (2002), 2–3.

[28] Wen, J., Chan, R. H., Yau, S.-C., He, R. L., and Yau, S. S. K-mer natural vector and its application to the phylogenetic analysis of genetic sequences. *Gene 546*, 1 (2014), 25–34.

[29] Yu, C., Liang, Q., Yin, C., He, R. L., and Yau, S. S.-T. A novel construction of genome space with biological geometry. *DNA research* (2010), dsq008.