Align-gram: Rethinking the Skip-gram Model for Protein Sequence Analysis

Nabil Ibtehaz 1 · S. M. Shakhawat Hossain Sourav 2 · Md. Shamsuzzoha Bayzid 1 · M. Sohel Rahman 1

Accepted: 13 February 2023 / Published online: 28 March 2023
© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2023

Abstract
The inception of next generations sequencing technologies have exponentially increased the volume of biological sequence data. Protein sequences, being quoted as the 'language of life', has been analyzed for a multitude of applications and inferences. Owing to the rapid development of deep learning, in recent years there have been a number of breakthroughs in the domain of Natural Language Processing. Since these methods are capable of performing different tasks when trained with a sufficient amount of data, off-the-shelf models are used to perform various biological applications. In this study, we investigated the applicability of the popular Skip-gram model for protein sequence analysis and made an attempt to incorporate some biological insights into it. We propose a novel $k$-mer embedding scheme, Align-gram, which is capable of mapping the similar $k$-mers close to each other in a vector space. Furthermore, we experiment with other sequence-based protein representations and observe that the embeddings derived from Align-gram aids modeling and training deep learning models better. Our experiments with a simple baseline LSTM model and a much complex CNN model of DeepGoPlus shows the potential of Align-gram in performing different types of deep learning applications for protein sequence analysis.

Keywords Deep learning · $k$-mer · Protein sequence analysis · Skip-gram model · Word embedding

1 Introduction
With the rapid development of sequencing technology, there has been an exponential increase in the amount of biological sequence data [1]. This abundant amount of sequence data enables us to apply big data and deep learning-based technologies in biological sequence analysis, which is revolutionizing almost all the different aspects of life [2]. Most notably, the application of deep learning in protein sequence analysis is showing great promises [1, 3, 4]. Previously, machine learning-based methods trained with biological features have been used in tasks like secondary structure prediction [5], surface residues [6], protein subcellular location [7] etc. Deep learning-based approaches are, arguably, much superior and convenient over the traditional machine learning based bioinformatics analysis pipeline as they make the feature extraction step unnecessary and are capable of working with the raw data instead. Since the identification and extraction of such biological features are time-consuming, expensive, and require a lot of effort, deep learning-based methods are potentially more effective to keep up with the ever-increasing number of protein sequences. As a result, more and more deep learning models are being developed to perform tasks like contact prediction [8], protein secondary structure prediction [9], protein function prediction [10], protein sub-cellular localization [11], peptide-MHC binding prediction [12] etc.

The selection of the input representation is the first step in leveraging deep learning approaches in protein sequence analysis. The idea is to break the protein sequences into tokens (amino acids or $k$-mers) and assign a numeric value or vector to each of the tokens. This assignment can be done using several approaches. The simplest and most popular method is to apply one-hot encoding [4], that assumes that different amino acids or $k$-mers are completely uncorrelated with each other and accordingly, treats them independently. The second-most widely used approach is to use the BLOcks Substitution Matrix (BLOSUM) [13] that encodes the amino acids using their corresponding row in the BLOSUM
matrix [14, 15]. This incorporates some evolutionary information of the amino acids as BLOSUM values signify which pairs of amino acids are more likely to transpose into one another during the course of evolution. This sense of relation between the amino acids may prove beneficial in certain analysis tasks. However, sometimes the simpler one-hot encoding performs better [16], and sometimes the combination of both yields superior results [12]. In addition to sequence-based approaches, evolutionary information like Position-Specific Scoring Matrices (PSSM) [17, 18], physicochemical properties [19, 20], and handcrafted features [21, 22] have also shown great promises. Although these latter representations may often yield surpassing results, this usually comes at the cost of a significant amount of analysis, studies, and experiments which are essential to compute them. When we consider the level of effort and contrast them with the almost on par results obtained from using simpler sequence-based representations in novel network architectures, their utility and the excitement rather fade away.

Distributed representation of words has been one of the most influential and ground-breaking works in Natural Language Processing (NLP) [23, 24]. Earlier works, using one-hot encoding, treated words as independent tokens. But actually, the various words are interrelated with each other, some are similar like ‘Perfect’ and ‘Excellent’, some are opposite like ‘Swift’ and ‘Sluggish’, whereas some are completely unrelated like ‘Computer’ and ‘Penguin’. Thus, it is understandable that leveraging these relations between and among the words instead of treating them as completely separate entities will greatly benefit in modeling. Although the idea of generating representations for words has been investigated quite early [25], the simplistic nature of one-hot encoding retained its acceptance, mostly because, such simple approaches proved to be robust when trained on a large amount of data [23]. Nevertheless, with their revolutionary works, Mikolov et al. popularized the concept of word embedding [23, 24]. Now, the word2vec [24] and doc2vec [26] models are being applied in multitude of applications. Like all other domains working with text data, bioinformatics has also embraced the potentials and possibilities of such word embedding models [27, 28]. In the context of bioinformatics, the concept of words is translated to k-mers, and using a large database of protein sequences instead of text corpus, embeddings are learned in an unsupervised manner. This type of approach has also obtained success in certain applications [29, 30].

Despite the success of such approaches, it seems worthwhile investigating whether off-the-shelf NLP concepts (like the popular and widely adopted skip-gram model) can be improved by infusing some biological insight for bioinformatics tasks like protein sequence analysis. Deep neural networks being the universal function approximator, it is always possible to model any problem with them, provided that we have a sufficient amount of data. Nevertheless, we seek out means to coalesce some biological insights with the concepts of NLP and while doing so, we identify some potential pitfalls of applying off-the-shelf Skip-gram models for protein sequence analysis. We attend to those issues and propose a novel model, Align-gram, which is likely to be more suitable for bioinformatics applications.

2 Materials and Methods

2.1 The Skip-gram model

We first briefly review the popular Skip-gram model. The skip-gram model [23] was developed to generate embeddings for words such that the syntactic and semantic relations between the words are preserved. The model constructs a vector space and maps the different words with the objective of mapping similar words close to each other. To achieve this, a shallow neural network with a single hidden layer is used, which tries to predict the nearby words of a given input word. To train the model, a large text corpus is used, the sentences are tokenized into words and represented by unique ids. For each word w in the sentences, we consider a window, i.e., we take a set W containing some words before and after w, and train the model to output those words (W) when w is given as input. Since the words are organized in natural human sentences obeying specific grammatical and semantic rules, this procedure, in essence, allows us to approximate the meaning of the word w. After training the model, the weights from the hidden layer are used to generate embeddings for the words.

2.2 Motivations and Higher Level Considerations

When off-the-shelf Skip-gram models are applied in protein sequence analysis (or other Bioinformatics tasks), instead of words, k-mers are naturally considered as tokens. However, unlike natural languages, where words as tokens are well defined, for k-mers in protein sequence it is not. Often when working with 3-mers, the common practice is to shift the sequence 2 times and break them into k-mers. For example, for a sequence ADTIVAVET, we have the 3 shifted sequences ADTIVAVET, DTAVET and TIVAVET. We can break the 3 sequences into sets of k-mers as \{ADT, IVA, VET\}, \{DTI, VAV\}, \{TIV, AVE\}. This approach has been followed by [27, 28], but we can observe that the same sequence is resulting in different contextual representations. This kind of situation is rare for natural texts as the sentences follow specific sets of rules. Furthermore, even if we consider contexts this way, we need to verify that the model manages to map the similar k-mers in the vector space properly, based on this contextual information about the k-mers is given. Although
a study [31] investigated the semantic meaning latent in the position or context for protein domains, to the best of our knowledge there has been no study that experiments with the semantic representation obtained from \( k \)-mer contexts in protein sequences. We inspect the vector space generated for \( k \)-mers using the Skip-gram model and discover that there is hardly a correlation between \( k \)-mer similarity scores and the vector similarity scores of the embeddings (please refer to Sect. 3.1). Moreover, homo-repeats [32] like AAAA exists in protein sequences, but the Skip-gram model does not consider repetitions as it is seldomly found in natural sentences.

### 2.3 Align-gram

The question that has driven us is whether the contextual information obtained from the location of \( k \)-mers in a protein sequence is sufficient to infer the relations between the \( k \)-mers. Traditionally the similarity between protein sequences or \( k \)-mers is determined using alignment scores [33]. During the course of evolution for point mutation, one amino acid can be transformed into another, the likelihood of which can be deduced from a substitution matrix, like BLOSUM [13]. Furthermore, gaps in the alignment correspond to indels, i.e., insertion or deletion mutations, which is regulated with suitable gap penalties. As a result, for decades, alignment scores are being used by biologists to measure the degree of similarity between protein sequences.

Therefore, to map the \( k \)-mers based on their similarity information we incorporate this biological intuition into the Skip-gram model. Instead of working with the contextual information of the surrounding \( k \)-mers, we compute the alignment scores of all the \( k \)-mer pairs. We then modify the Skip-gram model to become a regression model, by using linear activation for the output layer. Since cascaded linear layers collapse into a single linear layer, we use sigmoid activation function for the hidden layers. We represent the input \( k \)-mers to this network using one-hot encoding similar to Skip-gram, but unlike Skip-gram, we try to predict the alignment scores of all the \( k \)-mers with the input \( k \)-mer. We obtain a weight matrix from training the model and derive the embeddings for the \( k \)-mers from it, similar to Skip-gram model.

In this way, modifying the Skip-gram model, we develop Align-gram to map the \( k \)-mers based on their alignment scores instead of the contextual information (Fig. 1).

### 2.4 Implementation Details

We computed the alignment scores between the 3-mers using the BioPython [34] library. The values for gap open penalty, gap extend penalty were taken 11 and 1 respectively and we used the BLOSUM62 substitution matrix, which are the default options for BLASTP [35].

The shallow neural network with a single hidden layer has been implemented in Keras [36] with Tensorflow [37] backend. The hidden layer consists of 100 neurons which corresponds to the length of the embeddings. We used sigmoid and linear as the activation functions for the hidden layer and output layer respectively. The input one-hot vector was mapped to its corresponding alignment score vector as a regression problem. We minimized the mean squared error using the Adam [38] optimizer. The model was trained for 5000 epochs, which took about 30 min in a Tesla P-100 GPU.

### 3 Results

#### 3.1 Align-gram Establishes Equivalence Between Vector Similarity and Alignment Score

The primary objective of word embeddings is to generate such a vector representation for a word so that in the
vector space it is close to the vectors representing words with similar meaning. For example, we want a word like ‘Pizza’ to be close to words like ‘Burger’ or ‘Sandwich’, but far from words like ‘Car’ or ‘Computer’ in the vector space. Cosine similarity is a widely used measure of vector proximity or similarity. The value of cosine similarity ranges from 1 (identical) to \(-1\) (opposite), and for independent or orthogonal vectors it is around 0. Cosine similarity between two vector \(A\) and \(B\) is defined as:

\[
\text{Cosine Similarity}(A, B) = \frac{A \cdot B}{||A|| \cdot ||B||}
\]  

Therefore, for \(k\)-mer embeddings we expect the \(k\)-mers to be placed near similar \(k\)-mers. The degree of similarity between vectors and \(k\)-mers is computed using cosine similarity and alignment score respectively. Thus for both Skip-gram and Align-gram embeddings we compute the alignment score between every pair of \(k\)-mers (3-mers) and measure the cosine similarity scores of the corresponding vectors. We collected the 100-dimensional Skip-gram based embeddings for 3-mers from [27]. We study the correlation between cosine similarity and alignment score for the two embeddings and present them using box-plots in Fig. 2.

From the figure, it is apparent that the alignment scores and cosine similarity values for Align-gram embeddings are strongly correlated. On the other hand, there hardly exists a correlation for Skip-gram based embeddings, which is one particular concern we raised in Sec. 2.2. The Pearson correlation coefficient for the two cases are 90.2% and 22.6% respectively, which quantitatively expresses the compelling correlation between vector similarity and alignment score for Align-gram embeddings. This signifies that our proposed Align-gram embeddings have managed to map similar \(k\)-mers close to each other.

### 3.2 Align-gram Apparently Relates the Properties of Amino Acids in the Vector Space

In addition to maintaining the similarity between \(k\)-mers, we are interested to examine the capability of Align-gram to map the amino acids, preserving the physical-chemical properties. In order to do so, we train an Align-gram model with the 20 amino acids, instead of the \(k\)-mers. Therefore, this task reduces to predicting the pairwise scores between amino acids in the BLOSUM62 matrix in lieu of predicting the alignment scores. We follow the same protocol and just replace the inputs and outputs to the model. For the purpose of visualization, we set the number of embedding dimensions to 2, so that the amino acids can be represented in a 2-dimensional vector space (Fig. 3). We name the dimensions of the vector space as Align-gram Dim1 and Align-gram Dim2 respectively. Following this approach, we obtain a dense vector representation for the amino acids that is consistent with the BLOSUM62 score between them. For example, in Fig. 3, the amino acid Glycine (G) is nearby to amino acids Alanine (A) and Serine (S) (BLOSUM62 score = 0), whereas it is further apart from amino acids Leucine (L) and Isoleucine (I) (BLOSUM62 score = -4). Furthermore, it is observed that the values representing the different amino acids in this 2-dimensional vector space falls in the range \([-4.5, 4.5]\), which corresponds with the range of values in BLOSUM62 matrix(\([-4, 11]\)) as these two values are linearly combined with an additional bias term to predict the scores of BLOSUM62 matrix.

Venkatarajan [39] proposed a set of quantitative descriptors for the amino acids by summarizing various physical-chemical properties into a 5-dimensional property space. A total of 237 physical, structural, chemical properties, e.g., amino acid composition, hydrophobicity, net charge, molecular weight, absolute entropy etc.
were collected and they were properly normalized. Each of the amino acids were thus represented as 237-dimensional vectors, and the associated Eigenvectors ($E$) and Eigenvalues($\lambda$) were computed. The authors selected the top 5 significant Eigenvalues as they were sufficient to contrast the amino acids. Thus, the 237-dimensional property vectors were reduced to a 5-dimensional vector space. This provides a convenient way to compare the properties of amino acids, instead of inspecting all the 237 diverse physico-chemical properties.

We collect the Eigenvectors for the amino acids from [39] and compare it with our produced vector space. Figure 3 presents the relation between our vector space and the summarized 5-dimensional property space. It can be observed that for most cases for nearby points the values of the eigenvectors are quite similar. For a few exceptions like Glycine(G) and Serine(S) we can observe that in terms of E2 and E5 values they are a bit different, but for E1, E3 and E4 they are almost identical. These minor mismatches are due to the fact that we are trying to map a 2-dimensional space into a 5-dimensional one, which itself is an approximate mapping of a 237-dimensional space. Thus, overall it is apparent that amino acid properties are conserved in our vector space.

### 3.3 Align-gram Produces Better Embeddings for Protein Sequence Analysis

In this section, we demonstrate that Align-gram based embeddings are suitable to represent protein sequences in deep learning based applications. We compare with the usual baseline sequence-based representations and perform experiments with four different datasets each considering a different type of problem.

#### 3.3.1 Baseline Representations

We compare our proposed Align-gram based embeddings with the other widely used sequence based representations, namely, one-hot encoding, BLOSUM, and Skip-gram based embeddings. We represent the individual amino acids with sparse one-hot encoded vectors or corresponding rows from BLOSUM62. On the other hand, in order to use Skip-gram and Align-gram based embeddings, we tokenize the protein sequence into 3-mers and use their corresponding embeddings. We have collected the 100-dimensional Skip-gram based embeddings for 3-mers from [27]. Hence, to keep our embeddings comparable to that of Skip-gram, we have computed 100-dimensional embeddings. Protein sequences are...
seldomly of uniform length, therefore to make the sequences evenly long, we pre-pad them with zeros.

### 3.3.2 Tasks and Datasets

In order to investigate the versatility of Align-gram based embeddings, we consider four different types of machine learning problems, namely, binary classification, multiclass classification, regression, and sequence-to-sequence prediction.

For the purpose of evaluating the capability of Align-gram embeddings in performing such tasks, we use four different datasets. For binary classification, we predict DNA-Binding proteins using the PDB1075 dataset [40]. For regression, multiclass classification and sequence-to-sequence prediction we use the Protein Stability Prediction [41], Remote Homology Detection [42] and Secondary Structure Prediction [9, 43, 44] datasets from the TAPE benchmark [45]. We take the training split of these datasets to perform 5-fold cross-validation, and consider the PDB186 [46] and validation split of TAPE for testing the model. All these datasets are publicly available1,2.

The PDB1075 dataset consists of 1075 proteins sequences with a maximum length of 1323. The Protein Stability Prediction dataset consists of 53614 proteins whose lengths are confined to 49-50. The Remote Homology Detection dataset consists of a total of 12312 proteins with 1195 different fold labels. However, their lengths are distributed quite sporadically, the lengths cover a range of 17-1419 with a median value of 134 only. Therefore, both to mitigate the computational expense and avoid zero-padding over several magnitudes of length, we limit our analysis to sequences belonging to 75th percentile length, which equals 217. Furthermore, the dataset is highly imbalanced, hence, we only consider fold classes with at least 50 samples. This brings down the training set to 4814 sequences and similar selections were performed for the test data. The Secondary Structure Prediction dataset also suffers from a similar scattered distribution of sequence length, ranging from 20-1632 with a mean of 256, thus we similarly limit our experiments to 75th percentile length, i.e. 333.

The tasks and datasets are summarized in Table 1.

### 3.3.3 Baseline Model

Since the objective of this work has been developing a representation for protein sequences, it is beyond our scope to design suitable deep network architectures for solving the different problems and properly tune their hyperparameters. Therefore, we develop a simple baseline model and experiment with the different baseline representations, using the same model.

The baseline model consists of 3 LSTM layers each with 256 cells. 2 fully connected layers with 128 and 64 neurons respectively follow them, each with relu activation function and 20% dropout. After these layers, we define the output layer, for regression and binary classification task the output layer consists of a single neuron with linear and sigmoid activation function respectively. For the multiclass classification problem, the output layer consists of neurons equalling the number of classes and they are activated using softmax activation function. On the other hand, for sequence-to-sequence prediction, we use time distributed fully connected layers.

### 3.3.4 Experimental Results

We train the baseline model with the different input representations for the different tasks. We evaluate the models on the test dataset, using task-specific evaluation metrics. For binary classification, we use Precision, Recall, F1, and Accuracy, for multiclass classification we use Top-1, Top-5, and Top-10 Accuracies, for regression we use Mean Squared Error (MSE), Mean Absolute Error (MAE), and Spearman’s rank correlation coefficient (ρ), for the specific sequence-to-sequence task we compute the suitable 3 Class and 8 Class Accuracies over the entire predicted sequence. The experimental results for the different tasks are presented in Table 2.

From the table, it is evident that Align-gram based embeddings have obtained superior results in all the different tasks for all the evaluation metrics. The individual performance metric scores may appear mediocre, but this is due to the simplicity of the baseline model. For these

---

1 [http://server.malab.cn/Local-DPP/Datasets.html](http://server.malab.cn/Local-DPP/Datasets.html)
2 [https://github.com/songlab-cal/tape](https://github.com/songlab-cal/tape)
Table 2 Experimental results

| Embedding   | DNA-binding-protein prediction | Protein secondary structure prediction |
|-------------|--------------------------------|----------------------------------------|
|             | Precision | Recall | F1 | Accuracy | 3 Class accuracy | 8 Class accuracy |
| One Hot     | 48 ± 24   | 60 ± 32 | 53 ± 27 | 60 ± 6 | 50.2 ± 1.7       | 26.8 ± 1.7       |
| BLOSUM62    | 56 ± 1.9  | 82.2 ± 3.9 | 66.3 ± 2.4 | 58.3 ± 2.8 | 47.4 ± 1.1       | 25.1 ± 1.1       |
| Skip-gram   | 57.8 ± 2.6 | 55.9 ± 7.8 | 56.7 ± 4.7 | 57.6 ± 2.8 | 46.9 ± 3.4       | 27.6 ± 1.6       |
| Align-gram  | 61.7 ± 4.2 | 85.6 ± 3.2 | 71.6 ± 3.4 | 65.9 ± 5.1 | 54.9 ± 0.7       | 32.7 ± 1.6       |

| Embedding   | Remote homology detection | Protein stability prediction |
|-------------|---------------------------|------------------------------|
|             | Top-1 Acc | Top-5 Acc | Top-10 Acc. | MSE | MAE | ρ |
| One Hot     | 17.57 ± 3.86 | 40.64 ± 8.55 | 53.18 ± 6.81 | 0.17 ± 0.003 | 0.31 ± 0.003 | 0.78 ± 0.005 |
| BLOSUM62    | 18.04 ± 2.62 | 41.34 ± 6.97 | 55.71 ± 4.30 | 0.20 ± 0.004 | 0.35 ± 0.005 | 0.74 ± 0.007 |
| Skip-gram   | 19.85 ± 3.03 | 52.46 ± 3.37 | 70.03 ± 2.25 | 0.20 ± 0.009 | 0.35 ± 0.007 | 0.73 ± 0.012 |
| Align-gram  | 24.31 ± 2.06 | 55.33 ± 2.06 | 71.70 ± 3.77 | 0.15 ± 0.003 | 0.29 ± 0.004 | 0.82 ± 0.005 |

Here, we present the results on the test data for the different tasks, using different representations. For different type of predictions suitable metrics have been used. Although the results are quite mediocre due to the simplicity of the baseline model, it can be observed that Align-gram based embeddings led to the best results (presented in bold).

In this way, though Align-gram could not surpass evolutionary features, still it is able to supplement them by improving the overall performance. The results are presented in Table 3.

3.4 Align-gram Embeddings Supplements Evolutionary Features

Evolutionary features, although are the most capable ones for modeling biological sequences, are not always applicable for a number of reasons. For example, PSSM [17, 18] is always changing with the discovery of new proteins [16], HMM state-transition probabilities [9] requires sequence alignment which is time consuming and computationally expensive. As a result, language model based approaches are gaining more popularity for simplicity and convenience, despite the evolutionary features being more capable of modeling state of the art methods [45].

Thus, it is expected that evolutionary features like PSSM or HMM will outperform simple sequence based representations including Align-gram embeddings. Nevertheless, we conduct experiments with the baseline model using evolutionary features. For Remote Homology Detection we use PSSM [42] and for Secondary Structure Prediction we use HMM [9]. Using them as input we observe a significant improvement over the sequential representations, which is expected. However, most often these evolutionary features are coupled with one-hot-encoding [9, 47, 48]. Therefore, we examined coupling Align-gram embeddings with these features to see if Align-gram manages to contribute to the score. From the experiments, it was observed that including Align-gram embeddings with evolutionary features improves the result over using only evolutionary features or combining them with one-hot encoding, which is the popular practice. It may be noted here that since PSSM results in a 20-dimensional vector, we found that using 20-dimensional Align-gram embeddings were better to combine with PSSM. Align-gram offers us to use embeddings of various dimensions and more investigation is necessary to determine the optimal dimension for Align-gram embeddings.

As mentioned in Sect. 3.3.3, since we primarily worked with protein sequence encoding, it was beyond our scope to design suitable deep network architectures for solving different problems. This limited us to conduct our experiments with a generic, boilerplate baseline model. Therefore, in order to demonstrate the efficacy and applicability of Align-gram in deep networks, we consider DeepGOpPlus [10], one of the state-of-the-art methods for protein function prediction.

DeepGOpPlus employs a novel deep convolutional neural network (CNN) architecture to extract motifs, that are likely indicators of possible protein functions. Furthermore, it combines the neural network predictions with Diamond...
Therefore, using Align-gram based embeddings has the potential to improve deep, sophisticated networks as well.

### 3.6 Analysis: Hyperparameter Tuning

#### 3.6.1 Selection of $k$

We have conducted a study with different values of $k$. We tokenize the protein sequences as a list of $k$-mers. Therefore, as we increase the value of $k$ the amount of contextual information is increased, which is likely to be beneficial up to a certain degree. We experimented with values of $k = 1, 2, 3$ and generate different versions of our Align-gram embeddings. We perform the DNA-Binding Protein Prediction task and observe the performance on the validation data (Table 5).

From the table, it is evident that as we increase the value of $k$, the performance improves. This goes in line with our hypothesis that the notion of contextual information in higher values of $k$ should be beneficial. However, for much larger values of $k$, not only the distribution becomes sparse, it may also be the case that the model overfits on a set of large $k$-mers. Unfortunately, we could not work with $k > 3$ for the exponential increase of computational requirements. For instance, when we consider $k = 4$, we have a total of $20^4 = 160,000$ 4-mers and the pairwise alignment matrix becomes a $160,000 \times 160,000$ matrix. Thus, we failed to train Align-gram for 4-mers and beyond.

Based on the consideration of both performance and computational complexity, $k = 3$ is the most recommended option.

#### 3.6.2 Remarks on gaps in alignment

Our primary objective has been to consider the semantic relation between the $k$-mers. In order to do so we utilized the BLOSUM substitution matrices, as they represent the statistical likelihood of one amino acid to transpose into another amino acid, in the course of evolution. For 1-mers, i.e.,

### Table 3 Results obtained from using evolutionary features

| Embedding               | Task: remote homology detection | Task: secondary structure prediction |
|-------------------------|---------------------------------|-------------------------------------|
|                         | Top-1 Acc                       | Top-5 Acc                           | 3 Class Acc | 8 Class Acc |
| Evolutionary features   | 28.36 ± 1.64                    | 59.15 ± 2.75                        | 68.37 ± 2.03 | 44.06 ± 1.07 |
| Evo.+One Hot Enc        | 27.72 ± 1.79                    | 55.64 ± 1.64                        | 66.44 ± 1.56 | 44.01 ± 2.84 |
| Evo.+Skip-gram          | 27.12 ± 2.57                    | 61.68 ± 2.74                        | 66.82 ± 2.5  | 44.18 ± 1.86 |
| Evo.+Align-gram         | **29.8 ± 1.5**                  | **61.9 ± 2.38**                      | **70.57 ± 0.8** | **47.05 ± 1.04** |

The best results are marked in bold font.

For remote homology detection we use PSSM and for Secondary structure prediction we use HMM features. It can be observed that using evolutionary features yields much better results compared to that obtained from the sequence based representations. Although Align-gram embeddings fail to outperform evolutionary features, using them together in unison yields more superior results.

[49] based sequence similarity predictions using a weighted sum model. DeepGOPlus shows competitive performance with contemporary best-performing methods. Moreover, they demonstrate astounding results in CAFA3 evaluation [50], putting them as one of the three best scoring predictors for CCO and the second-best for BPO and MFO evaluations.

The authors of DeepGOPlus have made both their codes and data publicly available. Therefore, we have collected their codes and reproduced their results using their provided data-splits, following the CAFA3 evaluations. DeepGOPlus uses one-hot-encoding to represent the protein sequences, thus, we replace the input scheme with our proposed Align-gram based embedding and rerun the experiments following precisely the same protocol. It should be noted that DeepGOPlus, being a model with over 54 million parameters, for the massive computational requirements, we ran the experiments with 20-dimensional embeddings generated using Align-gram for 3-mers. Also, the results reproduced from the official implementation of DeepGOPlus were slightly different from the results presented in the paper. This may be due to differences in software or library versions, floating precision in hardware, random seed, etc. Therefore, to make the comparisons in an even-ground, we compare the results obtained from using Align-gram embeddings with the results we reproduced. The authors also shared their computed diamond scores and thus we perform the weighted sum ensemble as well. The results are presented in Table 4.

From the results, it can be observed that using Align-gram based embeddings instead of one-hot-encoding improved the various metrics of DeepGOPlus. All the three metrics for MFO and BPO has been improved, and a comparatively slighter dip has been observed for CCO. The gains are more striking when we consider the predictions from the CNN model only, as in the ensemble results the Diamond scores seem to dominate a bit.

---

3 https://github.com/bio-ontology-research-group/deepgoplus
4 http://deepgoplus.bio2vec.net/data/data-cafa.tar.gz
amino acids this relationship is trivial as it directly follows the BLOSUM substitution matrix. However, for \(k\)-mers with \(k > 1\), we can still obtain a logical estimation of this by computing the alignment score, which has been our motivation. Nevertheless, when we compute alignments between sequences (or \(k\)-mers in our case), the methods introduce some gaps. Although gaps are helpful to represent sequence variations like insertions and deletions, and cases not expected to occur very often [51], they are unfavorable for our algorithm. If the alignment we are working with contains gaps it will reduce incompatibility measures between \(k\)-mers to a certain degree, as large negative numbers will be replaced by a comparatively smaller gap penalty. Therefore, it is desired that the alignments used in the training are free from gaps.

The default value of gap penalty, i.e., 11 appears convenient for our purpose. Since the lowest value in BLOSUM62 matrix is \(-4\), for 3-mers the alignment algorithms will never introduce any gaps. In the extreme case, for example, for the 3-mer pair \(NNN\) and \(WWW\), the penalty from BLOSUM62 is \(-4 \times 3 = -12\), whereas opening a single gap would penalize by \(-11\). Thus, the alignment methods never introduce any gaps and the training data we used are gap free, preserving solely the similarities or dissimilarities between the 3-mers.

However, to observe the impact of gaps, we also experimented with two relaxed variants of Align-gram, with gap penalties of 5 and 3. Under these settings, \(8.01\%\) and \(26.53\%\) of the computed alignments contain gaps respectively. We train Align-gram embeddings with these alignments and observe the performance on the validation data of the DNA-Binding Protein Prediction task (Table 6).

From the table it can be observed that the model with gap penalty of 11, i.e., without any gaps in alignments performs the best. The presence of gaps decreases the performance, as they bring ambiguities in the \(k\)-mer relationships. Therefore, sufficiently high value of gap penalty should be selected to avoid gaps in alignment.

### 4 Data and Software Availability

Align-gram is available as free, open-source software at: https://github.com/nibtehaz/align-gram

We have made our trained Align-gram embedding public in the same link. Furthermore, we have published Align-gram as a customizable interface, with sufficient documentation to aid researchers in the advancement of protein sequence analysis.

The datasets used to develop and evaluate Align-gram are publicly available and the sources have been properly mentioned throughout the paper.

---

**Table 4** Results obtained from the DeepGOPlus model

| Method               | \(F_{\text{max}}\) | \(S_{\text{min}}\) | AUPR |
|---------------------|--------------------|--------------------|------|
|                     | MFO | BPO | CCO | MFO | BPO | CCO | MFO | BPO | CCO |
| Deep CNN results    |     |     |     |     |     |     |     |     |     |
| In paper            | 0.420 | 0.378 | 0.607 | 9.711 | 24.234 | 8.153 | 0.355 | 0.323 | 0.616 |
| Reproduced          | 0.405 | 0.380 | 0.598 | 9.835 | 24.120 | 8.192 | 0.337 | 0.314 | 0.593 |
| Align-gram          | **0.420** | **0.395** | 0.589 | **9.688** | **23.986** | **8.192** | **0.353** | **0.335** | **0.593** |
| Diamond score results |     |     |     |     |     |     |     |     |     |
| In paper            | 0.509 | 0.427 | 0.557 | 9.031 | 22.860 | 8.198 | 0.340 | 0.267 | 0.335 |
| Ensemble with diamond score |     |     |     |     |     |     |     |     |     |
| In paper            | 0.544 | 0.469 | 0.623 | 8.724 | 22.573 | 7.823 | 0.487 | 0.404 | 0.627 |
| Reproduced          | 0.528 | 0.458 | **0.617** | 8.732 | **22.489** | 7.964 | 0.476 | 0.392 | **0.611** |
| Align-gram          | **0.535** | **0.462** | 0.611 | **8.67** | 22.56 | **7.88** | **0.479** | **0.4** | **0.610** |

The best results are marked in bold font.

We reproduce the original results presented in the DeepGOPlus paper. Furthermore, we rerun the experiments using Align-gram based protein sequence representation and also ensemble with Diamond scores. Using Align-gram improves the MFO and BPO metrics and for CCO a slight dip is noticed. The improvements are more noticeable for the results using the CNN model only as Diamond score seems to prevail the ensemble result a bit.

**Table 5** Selection of \(k\)

| \(k\) | Precision | Recall | F1 | Accuracy |
|-------|-----------|-------|----|----------|
| 1     | 71.57     | 69.52 | 70.53 | 71.63    |
| 2     | **73.74** | 69.52 | 71.57 | 73.02    |
| 3     | 71.43     | **76.19** | **73.73** | **73.49** |

The best results are marked in bold font.

Varients of Align-gram embeddings with different values of \(k\) are trained and the performance on the validation data of the DNA-binding protein prediction task is observed.
Table 6  Variation in gap penalty

| Gap penalty                        | Precision | Recall | F1  | Accuracy |
|-----------------------------------|-----------|--------|-----|----------|
| 3 (gaps in 26.53% alignments)    | 67.57     | 71.43  | 69.44 | 69.3     |
| 5 (gaps in 8.01% alignments)     | 66.41     | 72.86  | 69.49 | 71.16    |
| 11 (no gaps)                     | 71.43     | 76.19  | 73.73 | 73.49    |

The best results are marked in bold font

Variants of Align-gram embeddings with different values of gap Penalty are trained and the performance on the validation data of the DNA-binding protein prediction task is observed

5 Conclusion

Thus, in this work, we have developed Align-gram, a novel word or k-mer embedding scheme for protein sequence analysis, maintaining the analogies with the successful and popular Skip-gram model from NLP. Align-gram manages to correlate the embedding vector similarity of the k-mers with the alignment score, signifying that evolutionary and biologically similar k-mers are projected together. In addition to bridging the gap between NLP model and Biological insights, Align-gram provides us with a better embedding to train deep learning models. Although Align-gram based embeddings are not on-par with computationally expensive PSSM and HMM based evolutionary features, the embeddings can supplement them well, improving the overall performance further.

Very recently, some attempts of using complex NLP models (e.g., Elmo, BERT, etc.) for protein sequences have been made [52, 53], albeit off-the-shelf NLP models without much specific modifications have been employed. Comparing Skip-gram models (~ a few thousand parameters) with such large-scale models (~ hundreds of millions of parameters) is hardly an even comparison. It is expected that context-aware representations will be capable of extracting more patterns, just like PSSM or HMM profile have shown great success, but these context-aware representations miss the individual residue level information. For example, when PSSM or HMM is used, amino acid information is often included as one hot encoded vectors [9]. Similarly, methods using embeddings from context-aware, complex NLP models also consider residue level information in form of one hot encoding [54]. Therefore, our goal has been to improve the representation for such residue level information, as supplements to complex context-aware representations, not to substitute them.

The future directions of this research can be manifold. Firstly, we are interested in applying Align-gram embedding in solving a multitude of diverse problems. It would be interesting to analyze how Align-gram based embeddings affect the performance of the existing state of the art sequence based deep learning models for different tasks in protein sequence analysis. Furthermore, we are intrigued to perform similar investigations on more sophisticated NLP models like Elmo [55] and BERT [56], as it is worth exploring the possibilities of augmenting such models with biological insights and intuitions to develop more refined models, better suited for bioinformatics applications.

References

1. Larranaga P, Calvo B, Santana R, Bielza C, Galdiano J, Inza I, Lozano JA, Armananzas R, Santafe G, Perez A et al (2006) Machine learning in bioinformatics. Brief Bioinform 7(1):86–112
2. Manyika J, Chui M, Brown B, Bughin J, Dobbs R, Roxburgh C, Byers AH et al (2011) Big data: the next frontier for innovation, competition, and productivity. McKinsey Global Institute, San Francisco
3. Min S, Lee B, Yoon S (2017) Deep learning in bioinformatics. Brief Bioinform 18(5):851–869
4. Bo W, Zeng W, Liao Y, Shi Z, Savage SR, Jiang W, Zhang B (2020) Deep learning in proteomics. Proteomics 1900335
5. Selbig J, Mevissen T, Lengauer T (1999) Decision tree-based formation of consensus protein secondary structure prediction. Bioinformatics 15(12):1039–1046
6. Yan C, Dobbs D, Honavar V (2004) A two-stage classifier for identification of protein-protein interface residues. Bioinformatics 20(suppl 1):i371–i378
7. Huang Y, Li Y (2004) Prediction of protein subcellular locations using fuzzy k-nn method. Bioinformatics 20(1):21–28
8. Di Lena P, Nagata K, Baldi P (2012) Deep architectures for protein contact map prediction. Bioinformatics 28(19):2449–2457
9. Klausen MS, Jespersen MC, Nielsen H, Jensen KK, Jurtz VL, Sønderby CK, Otto Alexander Sommer M, Winther O, Nielsen M, Petersen B, Marcatili P (2019) Netsurf-p: improved prediction of protein structural features by integrated deep learning. Proteins 87(6):520–527
10. Kulmanov M, Hoehndorf R (2020) Deepgoplus: improved protein function prediction from sequence. Bioinformatics 36(2):422–429
11. Armenteros JJA, Sønderby CK, Sønderby SK, Nielsen H, Winther O (2017) Deeploc: prediction of protein subcellular localization using deep learning. Bioinformatics 33(21):3387–3395
12. Zeng H, Gifford DK (2019) Quantification of uncertainty in peptide-mhc binding prediction improves high-affinity peptide selection for therapeutic design. Cell Syst 9(2):159–166
13. Henikoff S, Henikoff JG (1992) Amino acid substitution matrices from protein blocks. Proc Natl Acad Sci 89(22):10915–10919
14. O’Donnell TJ, Rubinsteyn A, Bonsack M, Riemer AB, Laserson U, Hammerbacher J (2018) Mhcflurry: open-source class i mhc binding affinity prediction. Cell Syst 7(1):129–132
15. Jin J, Liu Z, Nasiri A, Cui Y, Louis S-Y, Zhang A, Zhao Y, Jianjun H (2021) Deep learning pan-specific model for interpretable mhc-i peptide binding prediction with improved attention mechanism. Proteins Struct Funct Bioinf 89(7):866–883
16. Hein A, Cole C, Valafar H (2021) An investigation in optimal encoding of protein primary sequence for structure prediction by artificial neural networks. In: Advances in computer vision and computational biology. Springer International Publishing, Berlin, pp 685–699
17. Jones DT (1999) Protein secondary structure prediction based on position-specific scoring matrices. J Mol Biol 292(2):195–202
18. Hanson J, Palival K, Liñán T, Yang Y, Zhou Y (2019) Improving prediction of protein secondary structure, backbone angles, solvent accessibility and contact numbers by using predicted contact
maps and an ensemble of recurrent and residual convolutional neural networks. Bioinformatics 35(14):2403–2410

19. Wang D, Liang Y, Dong X (2019) Capsule network for protein post-translational modification site prediction. Bioinformatics 35(14):2386–2394

20. Hongli F, Yang Y, Wang X, Wang H, Yan X (2019) Deepubi: a deep learning framework for prediction of ubiquitination sites in proteins. BMC Bioinf 20(1):1–10

21. Abelin JG, Harjanto D, Malloy M, Suri P, Colson T, Goulding SP, Creech AL, Serrano LR, Nasir G, Nasrallah Y et al (2019) Defining hla-ii ligand processing and binding rules with mass spectrometry enhances cancer epitope prediction. Immunity 51(4):766–779

22. Bin Y, Zhaoxin Y, Chen C, Ma A, Liu B, Tian B, Ma Q (2020) Dnmac: Prediction of prokaryote lysine acetylation sites through deep neural networks with multi-information fusion. Chemometrics and Intelligent Laboratory Systems 1039999

23. Mikolov Tomás, Chen Kai, Corrado Greg, Dean Jeffrey (2013) Efficient estimation of word representations in vector space. In 1st international conference on learning representations, ICLR

24. Mikolov T, Sutskever I, Chen K, Corrado GS, Dean J (2013) Distributed representations of words and phrases and their compositionality. In: Advances in neural information processing systems, pp 3111–3119

25. Rumelhart DE, Hinton GE, Williams RJ (1986) Learning representations by back-propagating errors. Nature 323(6088):533–536

26. Jey HL, Timothy B (2016) An empirical evaluation of doc2vec with practical insights into document embedding generation. In: Proceedings of the 1st Workshop on Representation Learning for NLP. Association for Computational Linguistics, pp 78–86

27. Asgari E, Mofrad MRK (2015) Continuous distributed representation of biological sequences for deep proteomics and genomics. PloS ONE 10(11):e0141287

28. Yang KK, Zachary W, Bedbrook CN, Arnold FH (2018) Learned protein embeddings for machine learning. Bioinformatics 34(15):2642–2648

29. Pholyphusit P, Pornputtapong N, Srisawati S, Chuangsawanich E (2019) Mhsecquet: a deep neural network model for universal mh binding prediction. BMC Bioinf 20(1):270

30. Viehjken J, Wenzel M, Samek W, Strodthoff N (2020) Usmpet: universal sequence models for major histocompatibility complex binding affinity prediction. BMC Bioinf 21(1):1–16

31. Buchan DWA, Jones DT (2020) Learning a functional grammar of protein domains using natural language word embedding techniques. Proteins Str Func Bioinf 88(4):616–624

32. Michail YL, Petr K, Igor VS, Gian GT, Oxana VG (2016) Non-random distribution of homo-repeats: links with biological functions and human diseases. Sci Rep 6:26941

33. Swathik CP, Jaspreet KD, Vidhi M, Navaneethan R, Mannu J, Durai S, Durai S, Mannu J (2018) Encyclopaedia of bioinformatics and computational biology, Ranganathan S, Grib-skov M, Nakai K, Schönbach C (eds), pp 661–676

34. Cock PJA, Antao T, Chang JT, Chapman BA, Cox CJ, Dalke A, Friedberg I, Hamelryck T, Kauff F, Wilczynski B et al (2009) Biopython: freely available python tools for computational molecular biology and bioinformatics. Bioinformatics 25(11):1422–1423

35. Blast options and defaults (2020)

36. François Chollet et al, Keras (2015)

37. Abadi M, Barham P, Chen J, Chen Z, Davis A, Dean J, Devin M, Ghemawat S, Irving G, Isard M et al (2016) Tensorflow: a system for large-scale machine learning. In OSDI 16:265–283

38. Kingma DP, Ba J (2015) Adam: a method for stochastic optimization. In: 3rd international conference on learning representations, ICLR

39. Mathura SV, Werner B (2001) New quantitative descriptors of amino acids based on multidimensional scaling of a large number of physical-chemical properties. Mol Model Annu 7(12):445–453

40. Liu B, Jinghao X, Lan X, Rufeng X, Zhou J, Wang X, Chou K-C (2014) idna-prot dis: identifying dna-binding proteins by incorporating amino acid distance-pairs and reduced alphabet profile into the general pseudo amino acid composition. PloS ONE 9(9):e106691

41. Rocklin GJ, Chidyaukisu TM, Groesink H, Ford A, Houlston S, Lemak A, Carter L, Ravichandran R, Mulligan VK, Chevalier A et al (2017) Global analysis of protein folding using massively parallel design, synthesis, and testing. Science 357(6347):168–175

42. Fox NK, Brenner SE, Chandonia JM (2013) Scope: Structural classification of proteins-extended, integrating scop and astra data and classification of new structures. Nucleic Acids Res 42(D1):D304–D309

43. Berman HM, Westbrook J, Feng Z, Gilliland G, Bhat TN, Weissig H, Shindyalov IN, Bourne PE (2000) The protein data bank. Nucleic Acids Res 28(1):235–242

44. Moult J, Fidelis K, Kryshtaforovich A, Schwede T, Tramontano A (2018) Critical assessment of methods of protein structure prediction (CASP)-Round XII. Proteins Str Func Bioinf 86:7–15

45. Rao R, Bhattacharya N, Thomas N, Duan Y, Chen P, Canny J, Abbeel P, Song Y (2019) Evaluating protein transfer learning with tape. In: Advances in neural information processing systems, pp 9690–9701

46. Lou W, Wang X, Chen F, Chen Y, Jiang B, Zhang H (2014) Sequence based prediction of dna-binding proteins based on hybrid feature selection using random forest and gaussian naive bayes. PloS ONE 9(5):e86703

47. He F, Wang R, Li J, Bao L, Dong X, Zhao X (2018) Large-scale prediction of protein ubiquitination sites using a multimodal deep architecture. BMC Syst Biol 12(6):109

48. Huang K-Y, Hou JB-K, Lee T-Y (2019) Characterization and identification of lysine succinylation sites based on deep learning method. Sci Rep 9(1):1–15

49. Buchfink B, Xie C, Huson DH (2015) Fast and sensitive protein alignment using diamond. Nat Methods 12(1):59–60

50. Zhou N, Jiang Y, Bergquist TR, Lee AJ, Kaczob BZ, Croacker AW, Lewis KA, Georgiou G, Nguyen HN, Hamid MN et al (2019) The cafa challenge reports improved protein function prediction and new functional annotations for hundreds of genes through experimental screens. Genom Biol 20(1):1–23

51. Mount DW (2008) Using gaps and gap penalties to optimize pairwise sequence alignments. Cold Spring Harbor Protoc 2008(6):pdb–top40

52. Heinzinger M, Ahmed Elnaggar Yu, Wang CD, Nechaev D, Matthes F, Rost B (2019) Modeling aspects of the language of life through transfer-learning protein sequences. BMC Bioinf 20(1):723

53. Elnaggar A, Heinzinger M, Dallago C, Rehawi G, Yu W, Jones L, Gibbs T, Feher T, Angerer C, Steinegger M, Bhowmik D, Rost B (2021) Protrans: towards cracking the language of life codes through self-supervised deep learning and high performance computing. In: IEEE transactions on pattern analysis and machine intelligence, p 1

54. Zeng H, Gifford DK (2019) Deepeligand: accurate prediction of mhc class i ligands using peptide embedding. Bioinformatics 35(14):i278–i283

55. Peters ME, Neumann M, Iyyer M, Gardner M, Clark C, Lee K, Zettlemoyer L (2018) Deep contextualized word representations. In: Proceedings of the 2018 conference of the North American chapter of the association for computational linguistics. Association for Computational Linguistics, pp 2227–2237

56. Devlin J, Chang M-W, Lee K, Toutanova K (2019) BERT: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 conference of the North American chapter of the Association for Computational Linguistics. Association for Computational Linguistics, pp 4171–4186
**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.