Similarity measure for word embeddings applied to protein-protein interaction prediction

Claudio Jardim, Alta de Waal, Inger Fabris-Rotelli, Najmeh Nakhaeirad, and Jocelyn Mazarura

Department of Statistics, University of Pretoria

Abstract. Proteins often function by interacting with other proteins and forming complexes. Understanding these protein-protein interactions is critical in understanding biological processes and disease mechanisms. The traditional identification of protein-protein interactions is highly challenging, expensive and laborious. Machine learning techniques, such as deep learning, have shown promise in protein-protein interaction identification by combining protein sequences, protein structures and other sources of data as features. Although computational approaches ease the laborious process by making large-scale predictions in a short amount of time, their utilization often requires substantial computational power and extensive training time. Therefore traditional approaches are still sometimes preferred over the increase in speed, complexity and the inherent uncertainty of machine learning methods. In this study, we propose an alternative approach to protein-protein interaction prediction that overcomes the limitations of existing machine learning techniques. Our approach draws inspiration from natural language processing methods, such as Latent Dirichlet Allocation, to project protein sequence data into lower dimensional embeddings which then allows us to compare similarities from a geometric perspective. The obtained results confirm that our embeddings are able to capture the interactions between proteins using the similarity of the protein sequences alone. We have also shown that a simple heuristic can be used on the distances between the embeddings to make protein-protein interaction predictions. This simple technique was able to achieve better metrics than a competitive deep learning approach.

Keywords: Latent Dirichlet Allocation · Protein-protein interaction prediction · Sampling · Natural language processing

1 Introduction

Proteins are large molecules that perform various functions in a biological system. These large biomolecules consist of chains that are composed of amino acid residues. The sequence and structural composition of these residues uniquely determine the properties and functions of the protein [5]. Whilst some proteins function individually, the majority of proteins interact with other proteins in order to perform their biological functions. These complexes perform vital functions for many biological processes. Many diseases and disorders are also caused

by protein-protein interactions (PPIs). It is vital to predict the PPIs that are involved in a disease to understand the cause of the disease. These predicted PPIs are blocked or modulated for a therapeutic effect [26,33,18,31]. Identifying these interactions greatly assists in developing novel drug therapeutics that restore normal cellular function.

Experimental determination of PPIs is extremely difficult, expensive and time-consuming [23,31,12]. Due to the amount of known PPI data available, it is possible to screen known drug candidates for a given target using machine learning (ML) techniques. ML and other computational approaches can analyse large datasets with high-dimensional features for PPI prediction [19,35,18,33,16,24]. ML approaches are also cheaper than experimental determination of PPIs, they are highly scalable and can produce results in a shorter amount of time [19,18,33]. Although ML currently has not replaced all of the laboratory methods for PPI prediction (such as fluorescence spectroscopy) [14], it improves the process and assists in its completion. ML is used to refine a set of PPIs so that they can be validated and evaluated through experimental techniques. Some of the most commonly used techniques in this space are natural language processing (NLP) and deep learning (DL) [15,19,11,25,35,18,33,16,24]. DL and, in particular, geometric DL models have shown promising or state-of-the-art results for PPI prediction [35,34]. These approaches are either applied to generated protein surface point clouds or directly to the atomic coordinate data obtained from the Protein Data Bank (PDB)[1][1] files and other structural file formats. If the right data is unavailable, these techniques can overfit and not generalise well to unseen data. The structural data from the PDB file itself consumes more memory than the sequence data from the PDB file. There is also much more sequence data (248272897 sequences contained in the UniProt database [8] available in other file formats) than structural data [20] (208702 PDB files contained in the PDB [2]). Protein sequence data contains an arrangement of single-letter codes representing the amino acids that make up a protein. Structural data contains atomic coordinate information as well as information on the atom type, amino acids and chains.

We provide a practical and computationally efficient alternative to DL methods that is easily reproducible[2]. . We believe it is of practical benefit to implement a technique that can search a database of proteins to find the interacting partner of a given query input protein. This technique provides a molecule-level prediction for each protein pair. The atom-level interaction site of the predicted interacting partner protein or drug candidate can then be predicted through current DL techniques - if it is required. This proposed method is practically useful on its own but can also be enhanced by using atom-level or point-level techniques [35,15]. We only make use of the available text data with no structural information or added features. Any reference to text data will be the protein sequence obtained

[1] PDB files are a text-based file format containing both the structural information and sequence information of a biological molecule.

[2] The code is available at https://github.com/Claudmj/similarity_measure_for_word_embeddings_applied_to_ppi_prediction.

from the PDB files. This sequence data simply represents specific chemical and biological properties of a protein molecule, such as the amino acid residues that make up a protein. This data can be seen as an analogy to natural language data: the protein molecule is analogous to a document, and its properties to words. This allows us to apply popular NLP techniques to the data such as vector space modelling which is popular in information retrieval.

In this study, we infer lower dimensional embeddings of protein sequence data by applying Latent Dirichlet Allocation (LDA) [4] and word2vec [21]. These embeddings are simple vector space representations of text data. In vector space, one can do calculations such as vector addition or scalar-vector multiplications and the Bhattacharyya distance (BD) [3] between vectors. Our PDB* dataset is a subset of PDB files from the PDB. The PDB* dataset is labelled, meaning, we do have ground truth information on known PPIs. We make use of the nonparametric bootstrap resampling method to create a histogram of the average BD between known protein-protein interactions. This histogram represents the null hypothesis that states that the average BD between the embeddings of proteins involved in a known PPI is equal to those that are not. This test is performed to see if we can differentiate known PPIs by the BD between the embeddings of the proteins. We reject the null hypothesis that the average difference between the embeddings of proteins involved in a known PPI is equal to those that are not. After establishing that the LDA embeddings are able to capture the interactions between proteins as semantic similarity using the protein sequences alone we evaluate a heuristic approach to PPI prediction. We demonstrate the performance of our method by finding the best interacting partner from a group of proteins given a query protein using extracted data from PDB files [1]. We compare multiple embeddings to the embeddings method obtained from a state-of-the-art approach named differentiable molecular surface interaction fingerprinting (dMaSIF) [35]. The multiple embeddings we compare are word2vec [21] using cosine similarity (CS) [32], word2vec using soft cosine similarity (SCS) [32] and LDA using BD. LDA embeddings using BD performed best overall. Our simple heuristic was able to achieve a higher AUC value (0.767) than the competitive DL approach dMaSIF max (0.491) and dMaSIF mean (0.485) [35] on the test set. Further, our method performs better than word2vec with CS and word2vec with SCS on the 10-fold and 30-fold cross-validation.

The rest of this paper is structured as follows: an overview of the data as well as the core models and methods applied throughout the paper are provided in Section 2. We present the results of the proposed hypothesis test in Section 3 and PPI prediction experiments in Section 4. Lastly, the study is concluded in Section 5.

2 Materials & methods

An overview of the data as well as the core models and methods applied throughout the paper are provided in this section.

2.1 Data

We evaluate our model on a subset of 5481 ligand[3] and receptor[4] pairs that were used to train the dMaSIF model [35,15]. This results in 10962 PDB files available *here*[5] that we refer to as the PDB* dataset. We attempted to recreate the test set in the dMaSIF approach [35] however, there was missing data in the repository. Instead, we use 4569 pairs as a training set and the remaining 912 pairs are used as a test set. We also run 10-fold and 30-fold cross-validation on the full set of 5481 pairs. We extract and only make use of the protein sequence data from the PDB files. A protein can be expressed as an arrangement of amino acid residues represented by single-letter codes (see Figure 1). This arrangement of single-letter codes is text and is known as a protein sequence. Since a protein can be represented as text it makes sense to apply embedding techniques to the protein data. This allows us to reduce the dimensionality of the protein sequence data and to compare proteins of all shapes and sizes by using an embedding vector of equal dimension. These embedding techniques capture the components of the protein such that similar proteins will have vector representations that are closer to one another in the vector space. In order to embed the sequence data we make use of LDA and word2vec. In the following sections, we provide the reader with a brief summary of LDA, word2vec, BD, CS and SCS.

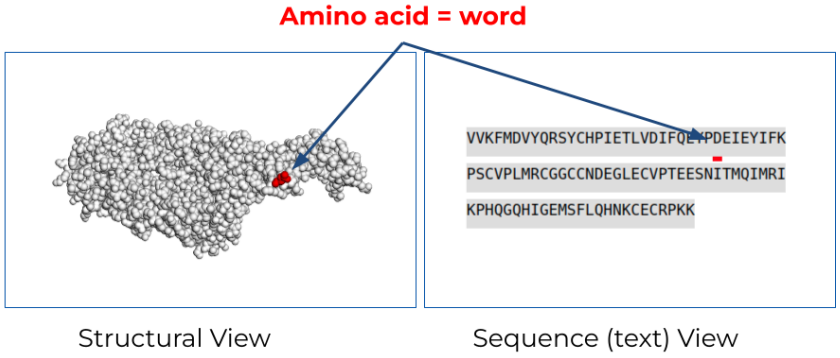

Fig. 1. An illustration of the structural view of a protein and the sequence view of the same protein. The atoms of the amino acid aspartic acid are highlighted in red in the structural view and the corresponding single letter code D is underlined in the sequence view.

2.2 Models

We infer lower dimensional embeddings of protein sequence data by making use of vector space models described in this subsection.

[3] A molecule that forms a complex with a receiving molecule or receptor.
[4] A special type of protein that functions by forming a complex with a ligand
[5] https://zenodo.org/record/2625420

Latent Dirichlet Allocation LDA is usually explained with NLP terminology, for example, a dataset is usually called a corpus [4]. Throughout this section, a protein is equivalent to a document and an amino acid residue (single-letter code) is equivalent to a word. LDA for a corpus of proteins C can be described as a generative statistical topic model where; each protein is a mixture of corpus-wide topics, each topic is a distribution over amino acid residues and each amino acid residue is drawn from one of the topics.
To discover the topics in a corpus it is easiest to reverse engineer the problem. First, we need to construct the proteins in the corpus. To do this, we can use a generative process for each protein in the corpus C with P the number of proteins in the corpus.

- Let $\{1, \ldots, A\}$ be the vocabulary of amino acids. An amino acid $x_1, \ldots, x_A$ is the basic unit of a protein.
- A protein $\mathbf{x}_p$ is a combination of amino acids denoted by $\mathbf{x_p} = [x_1, \ldots, x_{N_p}]$, where x_n is the nth amino acid in the protein and N_p is the number of amino acids in protein p for $p \in \{1, \ldots, P\}$.
- A corpus C is a collection of P proteins denoted as $C = [\mathbf{x_1}, \ldots, \mathbf{x_P}]$.

For each protein $\mathbf{x_p}$, where $p \in \{1, \ldots, P\}$ in a corpus C, a generative process is assumed [4,22]:

1. Draw a topic-protein distribution $\boldsymbol{\theta_p}$ from a Dirichlet distribution. $\boldsymbol{\theta_p} \backsim Dir(\boldsymbol{\alpha})$, where $p \in \{1, \ldots, P\}$ with $\boldsymbol{\alpha}$ a vector of dimension equal to the number of topics K, where, $\sum_{k=1}^{K} \theta_{p,k} = 1$ and $\theta_{p,k} \in [0, 1]$ for all $k \in [1, \ldots, K]$.
2. For each of the amino acids $x_{p,n}$, where $p \in \{1, \ldots, P\}$, and $n \in \{1, \ldots, N_p\}$:
   (a) Generate a topic $z_{p,n} \backsim$ multinomial $(\boldsymbol{\theta_p})$.
   (b) Generate an amino acid $x_{p,n} \backsim p(x_{p,n}|z_{p,n}, \boldsymbol{\beta})$ a multinomial probability conditioned on the topic $z_{p,n}$, where $\boldsymbol{\beta}$ is the parameter of the Dirichlet prior on the per-topic amino acid distribution.

This equates to solving the following equation:

$$p(\boldsymbol{\theta}, \mathbf{z}|\mathbf{x}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \frac{p(\boldsymbol{\theta}, \mathbf{z}, \mathbf{x}|\boldsymbol{\alpha}, \boldsymbol{\beta})}{p(\mathbf{x}|\boldsymbol{\alpha}, \boldsymbol{\beta})}.$$

The normalisation factor $p(\mathbf{x}|\boldsymbol{\alpha}, \boldsymbol{\beta})$, cannot be exactly computed and so the distribution is intractable. However, there are approximate inference techniques that can be applied to the problem such as variational inference [4].
The molecular text-topic distributions are a matrix with topics as rows and columns defined by amino acids. Thus, each row of $\boldsymbol{\theta}$ is a distribution over topics and each row of the amino acid-topic distributions is a distribution over protein topics [4,10]. The amino acid-topic distribution matrix and topic-protein distribution matrix can be viewed as the decomposition of the original protein-amino acid matrix that represents the corpus of proteins being modelled. LDA can also be thought of as a dimensionality reduction technique [4,9] where the corpus is represented as an embedding in a lower dimensional form using the

topic-protein distribution matrix. The advantage of this approach is that the interpretable topics should create a semantic embedding of the proteins. Using the manifold hypothesis which states that "high dimensional spaces tend to lie in the vicinity of underlying lower dimensional spaces (manifolds)" [13] it is clear that LDA can provide us with a manifold representation of the corpus.
Thus, one can perform a similarity measure on the semantic space (amino acid-topic distribution matrix or manifolds) and query a protein using semantic indexing on the topic-protein distribution matrix. Under the LDA model a protein can be associated with more than one topic [4]. Thus, the topic probability vector for a protein will be of dimension K where K is the number of topics. For a protein $\mathbf{x}_p$ the topic probability vector can be represented as:

$$\boldsymbol{\theta_p} = [p(\alpha_1), \ldots, p(\alpha_K)],$$

where $p(\alpha_k)$ is the probability of the protein being associated with topic k. The LDA embedding will reduce the dimension of the amino acids in a protein to a probability vector with dimension equal to the number of topics chosen. This protein-topic matrix will be a semantic embedding for the corpus of proteins. In order to query a protein we make use of a similarity measure that can handle probabilities.

Word2vec Word2vec uses a group of related neural network (NN) models to compute a continuous vector representation of words (amino acids) [21]. The continuous vector of each word is chosen such that the cosine similarity between vectors is an indicator of the true semantic similarity between the words. By using word2vec a corpus of documents (proteins) can be converted into a vector space with each word in the corpus having a unique embedding vector in this feature space. The NN models are shallow networks with only two layers. Word2vec can either make use of continuous bag-of-words (CBOW) or continuous skip-gram model architectures [21]. The CBOW model takes into account the local window of words for the current word to create its embedding. Word2vec takes local features into account to create a local feature embedding vector for a word instead of a global one. The dimension of the embedding vector is directly related to the quality of the embedding. Thus it is beneficial to increase the dimension of the embedding vector until the performance metrics stop improving or the cost of increasing the dimension is too high [21]. Lastly, the size of the context window can also be adjusted to improve the quality of the embeddings [21]. The context window is the number of words around the target word that are used by the model to make predictions for the target word representation [21].

2.3 Vector space distance metrics

Once we have our simple vector space representations of the text data we can apply vector space distance metrics which are described in this subsection.

Bhattacharyya distance The BD measures the similarity between pairs of features in the vector space model [3]. It makes use of the Bhattacharyya coefficient which quantifies the amount of overlap between two probability distributions or two normalised vectors. The BD between two embedding vectors $\mathbf{u}$ and $\mathbf{v}$ of dimension N can be defined as:

$$bd\left(\mathbf{u},\mathbf{v}\right) = -\ln\left(bc\left(\mathbf{u},\mathbf{v}\right)\right)$$

where, $bc\left(\mathbf{u},\mathbf{v}\right)$ is the Bhattacharyya coefficient which is defined as:

$$bc\left(\mathbf{u},\mathbf{v}\right) = \sum_{i=1}^{N} \sqrt{u_i v_i}.$$

We have that, $0 \leq bc \leq 1$, since $u_i \leq 1$ and $v_i \leq 1\ \forall i \in 1,\ldots,N$, $\therefore 0 \leq bd \leq \infty$.

The variables u_i and v_i are less than or equal to 1 since they are components of two probability distributions or normalised vectors $\mathbf{u}$ and $\mathbf{v}$. The input vectors for the Bhattacharyya coefficient need to represent probabilities and thus it is a suitable measure to be applied to our LDA embeddings that are probability distributions. It should be noted that the Bhattacharyya coefficient is not a metric since it does not satisfy the triangle inequality [3].

Cosine similarity The CS between two vectors measures the similarity between two vectors by calculating the cosine of the angle between the vectors [32]. The CS only depends on the angle between the two vectors and not on their magnitudes. Vectors that are opposite have a CS of -1, orthogonal vectors have a CS of 0 and proportional vectors have a CS of 1. The CS for two embedding vectors $\mathbf{u}$ and $\mathbf{v}$ of dimension N can be defined as the dot product of $\mathbf{u}$ and $\mathbf{v}$ divided by the product of their lengths:

$$cs(u,v) = \cos\theta = \frac{\mathbf{u}\cdot\mathbf{v}}{||\mathbf{u}||||\mathbf{v}||} = \frac{\sum_{i=1}^{N} u_i v_i}{\sqrt{\sum_{i=1}^{N} u_i^2 \cdot v_i^2}}$$

where, u_i and v_i are the i-th components of $\mathbf{u}$ and $\mathbf{v}$.

Soft cosine similarity The SCS between two vectors considers the similarity between pairs of features in the vector space model [32]. The semantic similarity between vector embeddings that have no overlap in amino acid residues can be measured using the SCS. The SCS for two embedding vectors $\mathbf{u}$ and $\mathbf{v}$ of

dimension N can be defined as:

$$sc(\mathbf{u}, \mathbf{v}) = \frac{\sum_{i=1,j=1}^{N} s_{ij} u_i v_j}{\sqrt{\sum_{i=1,j=1}^{N} s_{ij} u_i u_j} \sqrt{\sum_{i=1,j=1}^{N} s_{ij} v_i v_j}},$$

where, s_{ij} measures the similarity between feature$_i$ and feature$_j$. It can be seen that when $s_{ii} = 1$ and $s_{ij} = 0$ for $i \neq j$ the SCS is equal to the cosine similarity.

3 Hypothesis test

In this section, we investigate the use of word embeddings for PPI prediction. We make use of the LDA embeddings on the extracted protein sequences from PDB files. We run a hypothesis test since we have the ground truth for known PPIs. Before we predict PPIs we test if the average BD between the embeddings of proteins involved in a known PPI are less than the average distances between those that are not. If this is the case then we can identify actual PPIs using these embeddings and a similarity measure such as distance. This would mean that our embeddings are able to differentiate between PPIs through embedding similarity where the magnitude of the distance between protein embeddings correlates with how likely two proteins are to interact. We describe our research approach and then summarise our results.
Let μ_{nppi} denote the average BD between the embeddings of proteins that are not in a known PPI and μ_{ppi} denote the average BD between the embeddings of proteins involved in a known PPI. Using the ground truth labels from the dataset μ_{ppi} is found to equal 0.8348. To run the hypothesis test we make use of a bootstrap algorithm [17] following the null and alternative hypotheses:

H_0 **Null Hypothesis**: the average BD between the embeddings of proteins involved in a known PPI is equal to those that are not.
Namely, $\mu_{nppi} = \mu_{ppi} = 0.8348$.

H_a **Alternative hypothesis**: the average BD between the embeddings of proteins involved in a known PPI is less than those that are not.
Namely, $\mu_{nppi} > \mu_{ppi} = 0.8348$.

In order to test the null hypothesis, we followed a bootstrap algorithm:

1. Extract the protein sequences from the 10962 PDB files in the PDB* dataset.
2. Create LDA embeddings for each protein sequence.
3. Group the sequences into ligands and receptors.
4. Calculate the BD between every receptor protein sequence embedding and every ligand protein sequence embedding (5481×5481 distances).
5. Calculate the mean distance between all of the embeddings that are part of a known PPI (5481 distances). This is $\mu_{ppi} = 0.8348$.

6. Calculate the test statistic which is the mean distance between all the embeddings that are not part of a known PPI. This results in a sample of size $5481 \times 5480 = 30035880$.
7. We set a significance level of 5%.
8. We bootstrap a sample of 30035880 distances from the distances between all the embeddings that are not part of a known PPI 10000 times and calculate the mean distance for each sample.
9. Finally, we calculate the bootstrap p-value by calculating the average number of sample means that are greater than or equal to the mean distance between all the embeddings that are not part of a known PPI.

The p-value obtained is 0 and since this is less than our significance level of 5% we can reject the null hypothesis that the average difference between the embeddings of proteins involved in a known PPI is equal to those that are not. The mean distance between known interactions and all interactions is visualised in the top plot shown in Figure 2 and a second view of the distribution of the sample means is provided in the bottom plot shown in Figure 2. It is clear that the distance between embeddings of proteins that are in a known PPI is less than those that are not. This means that the LDA embeddings are able to capture the interactions between proteins as semantic similarity using the protein sequences alone. In addition, Cohen's d [7] is 0.57 for this test indicating that the distances between the known interactions differ by 0.57 standard deviations to the distance between the unknown interactions. Cohen's d measures the standardised difference between the two sample means which is 0.57 standard deviations in Figure 2. Thus, we can differentiate between the two groups and make predictions on PPI using the LDA embeddings.

4 Protein-protein interaction prediction

Since we have shown that it is possible to differentiate between the known PPIs and unknown PPIs we can make predictions on PPI using the LDA embeddings. In the following sections, we define the method we use for PPI prediction. We then compare the LDA embeddings using the BD, word2vec embeddings using the CS, word2vec embeddings using the SCS and the dMaSIF embeddings [35]. After this comparison, we run 10-fold and 30-fold cross-validation in order to provide a more robust and unbiased comparison for performance evaluation [6]. We note that we do not compare our method to the dMaSIF PPI classification method- instead, we compare the LDA embeddings to the dMaSIF embeddings using our heuristic. Unfortunately, we are unable to retrain the dMaSIF model [35] as the training code is not provided and therefore it was not possible to run k-fold cross-validation experiments for these embeddings.
A simple heuristic can also be used to predict the interactions in an inventive way using the embeddings and a distance or similarity measure. We use the BD for the LDA embeddings since this measure may be used for probability distributions. For the word2vec embeddings, we use both CS and SCS. To compare with the dMaSIF embeddings we calculate the dot product between each point prediction,

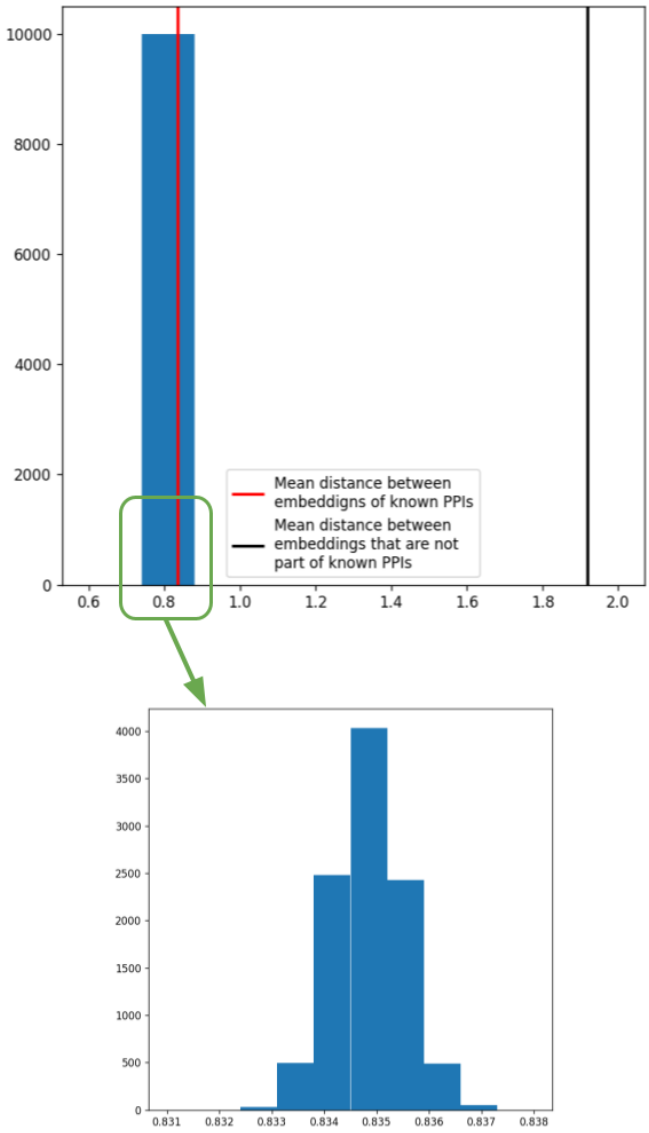

Fig. 2. Illustration of the bootstrapped sample means. The blue bar in the plot shows the sample means, the red line indicates the mean of the distances between embeddings of proteins involved in a known PPI and the black line indicates the mean of the PPI distances that are not in a known PPI. The plot below shows a zoomed-in view of the 10000 sample mean distances.

as described by the authors [35], to obtain a point-level interaction score. We calculated the minimum, maximum, median and mean point-level interaction scores. We found that the maximum and mean scores performed best so we used this as the molecule-level interaction measure. To predict whether or not two proteins interact we make use of the calculated score or measure as follows:

1. Calculate the measure between every receptor protein sequence embedding and every ligand protein sequence embedding.
2. For a given receptor:
   (a) Count how many ligands have a larger distance than the query ligand when using the BD or dMaSIF embeddings. Count how many ligands have a smaller distance than the query ligand when using the CS or SCS measure.
   (b) Take this count and divide it by the total number of ligands to get an interaction probability $P(\textit{given receptor interacting with query ligand})$.

We only label known PPIs as interacting and label everything else as non-interacting. We must state that this labelling may not be strictly correct. In reality, some receptors may interact with a ligand other than the known interacting ligand, but we use this labelling strategy to compare our results to other methods that use the same dataset [35]. Our method gives a probability of interaction for which a threshold can be chosen for binary prediction. The method is a simple heuristic and, as such, will not be influenced by incorrect labels as a trained model would. We evaluate the method by making a prediction on the known interacting ligand for each receptor as well as an unknown (dummy) ligand. We use the area under the receiver operating characteristic curve (AUC) [28], accuracy[6], precision[7] and recall[8] to evaluate all methods.

4.1 Evaluation: training test split

First, we compared our approach with a competitive DL method dMaSIF [35], in a simple training and test split described in Section 2.1. The training set consists of 4569 PPIs and the test set consists of 912 PPIs. The training set is the same as that used by the dMaSIF approach [35] and was used in order to compare our method to dMaSIF. We used the published dMaSIF model that is available *here*[9]. The test set is a subset of that used in the dMaSIF approach [35] since some PDB files were missing in the provided data repository.
Our simple PPI prediction heuristic is able to differentiate between known PPIs and unknown PPIs using the protein sequences only and no structural data (see

[6] Accuracy measures how many predictions are correct [27]. It can be calculated as the number of correct predictions divided by the total number of predictions.
[7] Precision measures the fraction of predicted positive cases that are true positive cases [29].
[8] Recall measures the fraction of true positive cases that are correctly predicted positive [30].
[9] https://github.com/FreyrS/dMaSIF

Table 1. PPI prediction results.

Method	AUC	Accuracy	Precision	Recall
LDA and BD	**0.767**	**0.656**	**0.623**	**0.790**
word2vec and SCS	0.746	0.645	0.616	0.772
word2vec and CS	0.717	0.611	0.593	0.706
dMaSIF max	0.491	0.489	0.489	0.505
dMaSIF mean	0.485	0.492	0.492	0.458

Table 1). We achieve a high probability of interaction between certain receptors and several ligands. We note that this is more realistic than a simple binary classification which may not be completely correct. This means that this probabilistic prediction can be used to discover potential interactions that might not be known. We leave this for future work. Our method achieves an AUC value of 0.767, which is better than the dMaSIF approach with an AUC of 0.491 for the dMaSIF max method and an AUC of 0.485 for the dMaSIF mean method. We observed that the dMaSIF embeddings basically perform the same as a random classifier as can be seen in Figure 3.

4.2 Evaluation: k-fold cross-validation

Lastly, we compared our approach to other word embedding methods with different similarity measures, using 10-fold and 30-fold cross-validation. We combined the training and test data used in Section 2.1 to create both the 10-fold and the 30-fold cross-validation split (see Table 2). We run 30-fold cross-validation in order to provide a more robust and unbiased comparison for performance evaluation as the model is trained and evaluated on various combinations of data [6]. We run the 10-fold cross-validation to check if the models' results are consistent or if they vary based on the data split. Other choices of k could be used but our choices capture the essence of the results well.

Table 2. 30-fold and a 10-fold cross-validation results

Method	30-fold cross-validation				10-fold cross-validation			
	AUC	Accuracy	Precision	Recall	AUC	Accuracy	Precision	Recall
LDA and BD	**0.806**	**0.660**	**0.623**	0.814	**0.807**	**0.665**	**0.626**	**0.822**
word2vec and SCS	0.793	0.659	0.621	**0.814**	0.795	0.663	0.625	0.814
word2vec and CS	0.796	0.656	0.619	0.813	0.796	0.652	0.615	0.810

LDA embeddings and BD perform best overall across the 10-fold and 30-fold cross-validation runs with 0.807 AUC and 0.806 AUC. It is clear that word2vec with SCS and word2vec with CS achieved similar performances.

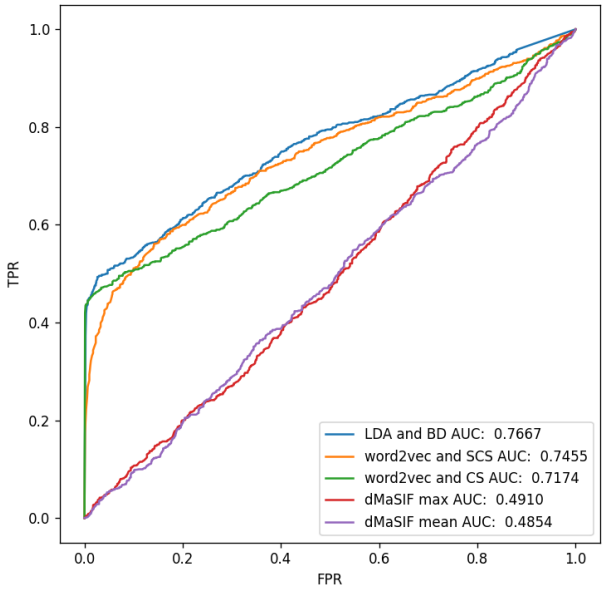

Fig. 3. ROC curves illustrating the results of our method compared to the dMaSIF method

5 Conclusion

In this research, we derived embeddings for protein sequence data and evaluated the use of these embeddings for representing a corpus of data in vector space. LDA and word2vec embeddings provide a useful transformation of protein sequence data to a vector space. These embeddings reduce the dimensionality of the data whilst keeping relative similarity between observations in their transformed state. In this way, a small amount of data can be used to understand a dataset. We evaluated the use of these embeddings for representing a corpus of data in vector space (embedding similarity) and lastly, we used semantic calculations on this representation to predict PPI.

LDA embeddings proved useful as a representation of the corpus in vector space. We rejected the null hypothesis that the average distance between the embeddings of proteins involved in a known PPI is equal to those that are not in a known PPI. This means that LDA embeddings between known interacting proteins were more similar, in vector, space than those that were not in a known interaction. The relationship between the magnitude of the distance between protein embeddings and protein-protein interaction should be investigated for future work. Further, the distances between the known interactions differ by 0.57 standard deviations from the distance between the unknown interactions.

We also showed that our PPI prediction heuristic can be used on the distances between the embeddings to make predictions on PPI. This simple technique was able to achieve a higher AUC value (0.767) than the competitive DL approach dMaSIF max (0.491) and dMaSIF mean (0.485) [35] on the test set. Further, we used 10-fold and 30-fold cross-validation to show that our method performs better than word2vec with CS and word2vec with SCS. Our heuristic is computationally efficient and requires much less computation than other approaches. The method does not require any training data and, as such, will not be influenced by incorrect labels or a class imbalance as a trained model would. This is particularly important when in reality, some receptors may interact with a ligand other than the known interacting ligand. Thus it is better to provide a probabilistic prediction instead of a binary one. Further, a threshold can be chosen to obtain binary predictions for a specific task.
We note that we do not compare our method to the dMaSIF PPI classification method- instead, we provide an alternative approach that requires a fraction of the computation required for DL. Furthermore, future work could include a comparison of their classification method, should reproducible code exist, which, to the best of our knowledge doesn't exist. This research underscores the critical need for investigation into the optimal window size for word2vec embeddings in the context of amino acid sequences. By systematically exploring various window sizes, we aim to discern whether the nearest neighbours of an amino acid alone encapsulate the most informative semantic features, or if a broader contextual window contributes significantly to approximating the intricate nature of these biomolecules. Simultaneously- in the context of LDA applied to amino acid sequences- a sentence can be extended to represent specific domains with common sequences. In the NLP paradigm, where sentence boundaries are crucial for contextual analysis, the delineation of specific domains within a protein similarly aids the embedding process by focusing on the localized context of specific functional or structural elements within the protein, thereby providing a more detailed representation of its biological significance. Lastly, it is imperative to extend this inquiry by exploring various dimensions of the word2vec model and investigating different numbers of topics for the LDA model.

References

1. Bank, R.P.D.: Homepage, https://www.rcsb.org/
2. Bank, R.P.D.: PDB statistics: Overall growth of released structures per year. https://www.rcsb.org/stats/growth/growth-released-structures, accessed: 2023-08-19
3. Bhattacharyya, A.: On a measure of divergence between two multinomial populations. In: Sankhyā: the Indian Journal of Statistics, pp. 401–406. JSTOR (1946)
4. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent Dirichlet Allocation. Journal of Machine Learning Research **3**(Jan), 993–1022 (2003)
5. Britannica, E.: Protein. https://www.britannica.com/science/protein, accessed: 2023-08-19
6. Cawley, G.C., Talbot, N.L.: On over-fitting in model selection and subsequent selection bias in performance evaluation. The Journal of Machine Learning Research **11**, 2079–2107 (2010)

7. Cohen, J.: Statistical Power Analysis for the Behavioral Sciences. Academic Press (2013)
8. Consortium, T.U.: Current release statistics. https://www.ebi.ac.uk/uniprot/TrEMBLstats, accessed: 2023-08-19
9. Crain, S.P., Zhou, K., Yang, S.H., Zha, H.: Dimensionality reduction and topic modeling: From Latent Semantic Indexing to Latent Dirichlet Allocation and beyond. In: Mining Text Data, pp. 129–161. Springer (2012)
10. De Waal, A., Barnard, E.: Evaluating topic models with stability. In: Nineteenth Annual Symposium of the Pattern Recognition Association of South Africa (PRASA 2008). vol. 5221, pp. 79–84 (2008)
11. Deng, J., Yang, Z., Ojima, I., Samaras, D., Wang, F.: Artificial intelligence in drug discovery: applications and techniques. Briefings in Bioinformatics **23**(1), bbab430 (2022)
12. Durham, J., Zhang, J., Humphreys, I.R., Pei, J., Cong, Q.: Recent advances in predicting and modeling protein–protein interactions. Trends in Biochemical Sciences (2023)
13. Fefferman, C., Mitter, S., Narayanan, H.: Testing the manifold hypothesis. Journal of the American Mathematical Society **29**(4), 983–1049 (2016)
14. Free, R.B., Hazelwood, L.A., Sibley, D.R.: Identifying novel protein-protein interactions using co-immunoprecipitation and mass spectroscopy. Current Protocols in Neuroscience **46**(1), 5–28 (2009)
15. Gainza, P., Sverrisson, F., Monti, F., Rodola, E., Boscaini, D., Bronstein, M., Correia, B.: Deciphering interaction fingerprints from protein molecular surfaces using geometric deep learning. Nature Methods **17**(2), 184–192 (2020)
16. Gainza, P., Wehrle, S., Van Hall-Beauvais, A., Marchand, A., Scheck, A., Harteveld, Z., Buckley, S., Ni, D., Tan, S., Sverrisson, F., et al.: De novo design of protein interactions with learned surface fingerprints. Nature pp. 1–9 (2023)
17. Hall, P., Wilson, S.R.: Two guidelines for bootstrap hypothesis testing. Biometrics pp. 757–762 (1991)
18. Hu, L., Wang, X., Huang, Y.A., Hu, P., You, Z.H.: A survey on computational models for predicting protein–protein interactions. Briefings in Bioinformatics **22**(5), bbab036 (2021)
19. Jarada, T.N., Rokne, J.G., Alhajj, R.: A review of computational drug repositioning: strategies, approaches, opportunities, challenges, and directions. Journal of Cheminformatics **12**(1), 1–23 (2020)
20. Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., Tunyasuvunakool, K., Bates, R., Žídek, A., Potapenko, A., et al.: Highly accurate protein structure prediction with alphafold. Nature **596**(7873), 583–589 (2021)
21. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013)
22. Pritchard, J.K., Stephens, M., Donnelly, P.: Inference of population structure using multilocus genotype data. Genetics **155**(2), 945–959 (2000)
23. Rao, V.S., Srinivas, K., Sujini, G., Kumar, G.: Protein-protein interaction detection: methods and analysis. International Journal of Proteomics **2014** (2014)
24. Riahi, S., Lee, J.H., Sorenson, T., Wei, S., Jager, S., Olfati-Saber, R., Zhou, Y., Park, A., Wendt, M., Minoux, H., et al.: Surface ID: a geometry-aware system for protein molecular surface comparison. Bioinformatics **39**(4), btad196 (2023)
25. Rifaioglu, A.S., Atas, H., Martin, M.J., Cetin-Atalay, R., Atalay, V., Doğan, T.: Recent applications of deep learning and machine intelligence on in silico drug discovery: methods, tools and databases. Briefings in Bioinformatics **20**(5), 1878–1912 (2019)

26. Ryan, D.P., Matthews, J.M.: Protein–protein interactions in human disease. Current Opinion in Structural Biology **15**(4), 441–446 (2005)
27. Sammut, C., Webb, G.I. (eds.): Accuracy, pp. 8–8. Springer US, Boston, MA (2017), https://doi.org/10.1007/978-1-4899-7687-1_3
28. Sammut, C., Webb, G.I. (eds.): AUC, pp. 75–75. Springer US, Boston, MA (2017), https://doi.org/10.1007/978-1-4899-7687-1_10025
29. Sammut, C., Webb, G.I. (eds.): Precision, pp. 990–990. Springer US, Boston, MA (2017), https://doi.org/10.1007/978-1-4899-7687-1_658
30. Sammut, C., Webb, G.I. (eds.): Recall, pp. 1056–1056. Springer US, Boston, MA (2017), https://doi.org/10.1007/978-1-4899-7687-1_702
31. Scott, D.E., Bayly, A.R., Abell, C., Skidmore, J.: Small molecules, big targets: drug discovery faces the protein–protein interaction challenge. Nature Reviews Drug Discovery **15**(8), 533–550 (2016)
32. Sidorov, G., Gelbukh, A., Gómez-Adorno, H., Pinto, D.: Soft similarity and soft cosine measure: Similarity of features in vector space model. Computación y Sistemas **18**(3), 491–504 (2014)
33. Soleymani, F., Paquet, E., Viktor, H., Michalowski, W., Spinello, D.: Protein–protein interaction prediction with deep learning: A comprehensive review. Computational and Structural Biotechnology Journal (2022)
34. Stärk, H., Ganea, O.E., Pattanaik, L., Barzilay, R., Jaakkola, T.: Equibind: Geometric deep learning for drug binding structure prediction. arXiv preprint arXiv:2202.05146 (2022)
35. Sverrisson, F., Feydy, J., Correia, B.E., Bronstein, M.M.: Fast end-to-end learning on protein surfaces. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 15272–15281 (2021)