1-1-2024

Modeling the Relationship Between Protein Sequence and Structure Using Triangular Spatial Relationship-Based Keys

Arianna Nicole Swensen

MODELING THE RELATIONSHIP BETWEEN PROTEIN SEQUENCE AND STRUCTURE USING TRIANGULAR SPATIAL RELATIONSHIP-BASED KEYS

A Thesis
presented in partial fulfillment of requirements
for the degree of Master of Science
in the Department of Computer and Information Science
The University of Mississippi

by

ARIANNA SWENSEN

May 2024

ABSTRACT

Protein structure comparison and our understanding of protein sequence and structure relationships are intertwined. In this work, I use the TSR-based key generation algorithm to develop a model for relating sequence and structure. I develop embeddings for the sequence-based keys and the structure-based keys as the basis for a translation model between the two sets of keys, evaluating the efficacy of local and global embedding generation techniques in the context of the TSR-based comparison method.

DEDICATION

In loving memory of my grandmothers, Beverly Ann Shelton and Debra Lee Swensen.

ACKNOWLEDGEMENTS

I would like to express my deepest appreciation to my advisor, Dr. Yixin Chen, and my committee members, Drs. Charles W. Walter and Hong Xiao. In addition, I want to thank Dr. Byunghyun Jang, along with the rest of the Department of Computer and Information Science, for their invaluable support throughout my journey as a student at the University of Mississippi.

Additionally, I would like to thank Dr. Wu Xu of the University of Louisiana at Lafayette for providing important data used in this work. My calculations were executed at LONI and MCSR, and I appreciate the LONI and MCSR Support Teams, with special thanks to Siva Kasetti.

Lastly, I would like to thank my fiancé and parents, who have acted as my sounding board and biggest cheerleaders in this process. I couldn't have done this without your endless support.

TABLE OF CONTENTS

1 INTRODUCTION

Pairwise protein structure comparison is a problem that has attracted significant attention in computational biology in the past two decades; in particular because of the many ways protein structure analysis can be used. Some problems where protein structure comparison is useful include protein folding problems, protein structure prediction, and understanding molecular evolution, to name a few.

Most existing protein structure comparison methods can be broken down into four components: smallest structural unit for comparison, degree of comparison, methodology of structural comparison, and the measure of statistical significance of the similarity measure. The degree of comparison is by far one of the most varied, with many classifications existing under its umbrella including rigid vs. flexible, sequential vs. nonsequential, and local vs. global vs. local-global comparison.

The focus of this work is the triangular spatial relationship-based (TSR-based) comparison method. The TSR-based comparison method is a local-global descriptor-based pairwise protein structure comparison. The heart of this comparison method is vectorization of each protein using keys to represent amino acid triplets. Both protein sequences and protein structures can be vectorized using this method, with each one generating 1D and 3D keys, respectively.

While this method is useful for protein structure comparison, the vectorization process also allows us to examine the relationship between sequence and structure more thoroughly. In this work, I model the relationship between 1D and 3D keys; this could be extended in

the future to examine the relationship between 1D vectors and 3D vectors, allowing us to expand our understanding of the relationship between sequence and structure.

To model the relationship between 1D and 3D keys, I treat each key as a "word", with the set of 1D keys being a "language" and the set of 3D keys being its own "language". By breaking the problem down to these components, I can take an approach similar to machine translation (MT) models, where I first generate embeddings for each language and then align them to determine the closest 3D embedding for a chosen 1D embedding, or vice versa.

Several popular frameworks exist to generate embeddings for word related problems; two of the most popular methods include word2vec and GloVe. Word2vec uses a local context window method to generate embeddings; in contrast, GloVe uses a global log-bilinear regression method. Since the TSR-method is a local-global structural comparison method, it is unclear whether a local or global approach would be the better choice for embedding generation.

In this work, I generate translation models using the word2vec and GloVe embedding generation techniques to compare their effectiveness for TSR-based keys. The remainder of this paper includes a more comprehensive review of existing protein structure comparison methods followed by an in-depth description of the methodology by which I generated the translation models. Several variations of these models and their effectiveness is discussed in the evaluation section, followed by my conclusions and thoughts on future research directions.

2 LITERATURE REVIEW

Proteins play a significant role in biochemical reactions. Analysis of proteins allows us to create hypotheses on how to affect, control, or modify proteins, key components of biochemistry, microbiology, drug development, and evolutionary biology. Protein structures are more conserved than protein sequences, that is, protein structures with similar structural features often have similar functions despite differences in their sequences. Thus, quantifying these structural differences is critical to a deeper understanding of the structural, functional, and evolutionary relationships among these proteins.

Pairwise protein structure comparison has attracted a significant level of research attention from both biologists and computer scientists due to its significance. Protein comparison is applicable to a wide range of both physical and biological research questions, including various facets of the protein folding problem, protein structure prediction, homology detection, functional annotation, mechanisms of evolutionary change, and molecular evolution, just to name a few.

Over the last two decades, following the creation of the first automated structural method, an abundance of potential solutions have been proposed for aligning protein structures. However, due to the considerable variety of results from these methods, no singular method has been widely accepted for comparison or search among protein structures.

While examining all existing protein comparison methods would be impossible, most methods can be broken down into four major components. The first of these is the smallest structural unit by which to compare. This can be amino acids, secondary structures, or torsion angles. The second, degree of comparison, can include or be a combination of rigid

or flexible comparison, sequential or non-sequential comparison, and global, local, or local-global comparison. Some methods use a rigid protein state to compare structures, however, a growing variety of protein structure comparison methods include the concept of flexibility [1, 2, 3, 4].

These methodologies attempt to capture structural similarity between proteins that have undergone conformational changes. Previous estimates suggest that between 17.4% and 35.2% of all alignments are non-sequential, depending on parameter variations [5], leading to the development of non-sequential [6], hybrid [3], and unified [7] comparison methods. Various methods have been created using local [8], global [9], or local-global [10] comparisons depending on the purpose of the comparison.

The third main component is the methodology of structural comparison, which can be broadly classified into alignment or descriptor based comparison. Alignment based comparisons require translations and rotations before performing a comparison of two structures, and can be either intramolecular [8] or intermolecular [11, 12] distance based alignments. In contrast, descriptor based comparison usually uses histograms or vectors to describe molecular shape, and only require reference transformation to perform comparison. Generally, alignment based comparison methods are not considered to be efficient for searching for similar structures from a database in real time [13]. Description based comparisons, on the other hand, have to balance information loss against redundancy and high dimensionality to be effective [14], but are considered efficient for searching large databases in a time effective manner.

The final component of a protein structure comparison method is the measurement of statistical significance of the similarity measure. Common measures of similarity include root mean squared distance (RMSD) and its variations [15, 16], Template modeling-score (TM-Score) [17, 18], and the Jaccard index [19, 20].

Many methods for protein structure comparison rely on basic computer science techniques. Some geometry-based algorithms include maximum common subgraph detection

[21], Ullman's subgraph isomorphism algorithm [22], and geometric hashing [23]. Genetic [24] and Monte-Carlo [11] algorithms have been used for secondary structure-based and distance-based comparisons, respectively, while dynamic programming algorithms have been implemented for both comparison methods [25, 26, 27, 28].

Newer approaches to finding homologous proteins focus primarily on structure; early methods used sequence string matching to derive initial structural equivalence but their performance struggled when sequence similarity was low [21]. These methods use similarities in local structural regions [29], secondary structures [30, 18], or both [31] to determine initial structural equivalence.

More recently, an innovative descriptor-based approach has been developed for examining pairwise protein structure similarity. This approach, called the triangular spatial relationship-based (TSR-based) method [32] is a non-sequential, local and global descriptor-based pairwise protein structure comparison.

The most critical component of TSR-based vectorization is the key generation. To describe the key generation process succinctly, C_α atoms from each protein's PDB file are selected, and then all three lengths and angles of each possible triangle formed by C_α are calculated. Each C_α of the 20 amino acids were assigned a unique integer identifier, which was then transformed to l_1, l_2, l_3 for the vertices of triangle i based on rule-based label-determination. θ_1 is defined using the Equation 2.0.1 and θ_Δ is a function on θ_1 values as defined in Equation 2.0.2,

$$\theta_1 = cos^{-1}\left(\frac{(d_{13}^2 - (\frac{d_{12}}{2})^2 - d_3^2}{2 * \frac{d_{12}}{2} * d_3}\right) \tag{2.0.1}$$

$$\theta_\Delta = \begin{cases} \theta_1 & \text{if } \theta \leq 90^\circ \\ 180^\circ - \theta_1 & \text{otherwise} \end{cases} \tag{2.0.2}$$

where d_{13}, d_{12}, d_3 are the distance between l_{i1} and l_{i3}, the distance between l_{i1} and l_{i2}, and the distance between l_{i3} and the midpoint of l_{i1} and l_{i2}, for triangle i respectively.

Once l_{i1}, l_{i2}, l_{i3}, D (where $D = d_{12}$) and θ_Δ have been determined, each key can be calculated using equation 2.0.3,

$$k = \theta_\tau d_\tau (l_{i1} - 1)m^2 + \theta_\tau d_\tau (l_{i2} - 1)m + \theta_\tau d_\tau (l_{i3} - 1) + \theta_\tau (d - 1) + (\theta - 1) \tag{2.0.3}$$

where m is the total number of distinct labels, θ is the bin value for the class that θ_Δ, the angle representative, falls in to achieve discretization, θ_τ is the total number of bins for the angle representative, d is the bin value for the class that D, the length representative, falls in to achieve discretization, and d_τ is the total number of bins for the length representative.

If there are q amino acids in the protein dataset, the time complexity of this method in the worst case is $O(nq^3)$ where n is the number of proteins in a given data set. Further discussion of this methodology of key generation, including the determination of bin boundary values and the number of bins, was reported [32].

The key calculation, along with the comparison and search features of the TSR-based method allow for crucial insight into the nature of protein structure relationships on both a local and a global scale. Recent work has used the TSR-based method to study the structure relations of proteins with a focus on the receptors organized in hierarchical levels. This work has found that some protein pairs have high sequence similarity but low structure similarity, or vice versa; studying these protein pairs in particular will help us understand more about sequence and structure relationships [33].

In this work, I will examine this issue as a machine translation (MT) problem and propose a novel method by which to understand the relationship between 1D and 3D keys. Developing this model requires that we first generate embeddings to represent these keys. Two popular approaches for word embedding include word2vec [34] and GloVe [35]. Word2vec has emerged as one of the most commonly used natural language processing (NLP) techniques, and uses a shallow neural network. Two models are available for training word2vec, including the continuous bag of words model (CBOW) and the continuous Skip-Gram model.

CBOW uses surrounding words to predict the target word, while Skip-Gram takes the target word and attempts to predict the surrounding words. Additional optimization techniques, such as hierarchical softmax and negative sampling, help to make this process more efficient. In contrast to word2vec, which is considered to be a local context window method, GloVe is a global log-bilinear regression model. GloVe is trained on non-zero entries of a global word-word co-occurrence matrix, which calculates how frequently words co-occur in a given corpus. Then, the ratio of probabilities between two pairs of words are used to generate the embeddings.

GloVe is mostly used in natural language processing (NLP) contexts. In contrast, word2vec - along with its sister tool, doc2vec [36], which vectorizes documents rather than words - have been used effectively in non-NLP contexts in a variety of fields, including product recommender tools [37], generating embeddings for categorical variables [38], and malware detection [39], to name a few.

In brief, despite the significant research effort into protein structure comparison, there are still a lot of gaps in our understanding of the relationship between protein structures, and the relationship between protein sequences and protein structures. A promising approach to structure comparison is the TSR-based model, however, the relationship between 1D and 3D keys is not well established. In this work, we use an approach akin to machine translation to create models that can "translate" a 1D key to a 3D key and vice versa, based on context. We utilize word2vec and GloVe, local and global embedding methods of vectorization, to build these models. The resulting models will provide important insights into the relationship between 1D and 3D keys, and allow future researchers to develop new algorithms examining the relationships between protein sequences and structures.

3 METHODOLOGY

I selected a small sample grouping of 5 protein structures to test the modeling system on a small scale. Additional information about each protein can be found in Appendix A. The key generation method described in Section 2.2 was applied to each protein, generating the 1D and 3D keys for each amino acid triplet within a protein for each amino acid triplet, including the representative coordinates each amino acid within the triplet. These coordinates can be denoted as p_1, p_2, p_3 for the location of each amino acid in sequence and (x_1, y_1, z_1), (x_2, y_2, z_2), and (x_3, y_3, z_3) for the locations of each amino acid in the structure.

$$c(p_1, p_2, p_3) = \frac{p_1 + p_2 + p_3}{3} \tag{3.0.1}$$

$$C((x_1, y_1, z_1), (x_2, y_2, z_2), (x_3, y_3, z_3)) = (\frac{x_1 + x_2 + x_3}{3}, \frac{y_1 + y_2 + y_3}{3}, \frac{z_1, z_2, z_3}{3}) \tag{3.0.2}$$

For each amino acid triplet, centroids c and C are calculated using Equation 3.0.1 and 3.0.2, respectively, with c denoting the 1D centroid and C denoting the 3D centroid.

The 1D and 3D keys for two amino acids are denoted as k_1, k_2, and K_1, K_2, respectively. The similarity function for two one dimensional keys $sim_1(k_1, k_2)$ is defined as a function of the distance between the two keys in sequence:

$$f(k_1, k_2) = \begin{cases} 1 & \text{if } dist(c_1, c_2) \leq d \\ 0 & \text{otherwise} \end{cases} \tag{3.0.3}$$

Likewise, the similarity function $sim_3(K_1, K_2)$ is defined as a function of the distance between the two keys in the protein structure:

$$F(K_1, K_2) = \begin{cases} 1 & \text{if } dist(C_1, C_2) \leq d \\ 0 & \text{otherwise} \end{cases} \tag{3.0.4}$$

where k_1 and k_2 are the 1D keys representing two amino acids, K_1 and K_2 are the equivalent 3D keys, c_1 and c_2 are the 1D centroids, C_1 and C_2 are the 3D centroids, $dist(x, y)$ is the distance function between two points, and d is a user input.

For the purposes of this work, I chose d by selecting the distances between approximately 150,000 pairs of amino acid triplets. This sampling approximates the normal distribution, with the sample distances in 1D having a mean of 83.521 and a standard deviation of 29.875 and the sample distances in 3D having a mean of 19.966 and a standard deviation of 4.947. The goal of setting a value d is to find triplet pairs that are "close" to each other. Thus, I select d = 8.250 to approximate the smallest 2.5% of distances from both sets.

Each protein has an average of 406,663 total keys, which can be combined in pairs of 2 in approximately 80 billion ways. Even with advanced techniques, doing this across even a small dataset is costly and time consuming. Initial observations suggest that triplets that are close to each other in either dimension are generally near each other in the files. Thus, I use a batching technique, calculating all the potential combinations in batches of 1,000 triplets and cross batches containing the last 500 triplets from the first batch and first 500 triplets from the second batch. Then, I apply the similarity function to each pair of triplets, counting how many times a pair $sim(k_1, k_2) = 1$, or the co-occurrence frequency.

These values are stored as an adjacency list of the undirected weighted graph $G = (V, E)$, where V is the list of unique keys in a protein p and E contains the weighted edges representing the co-occurrence frequencies. Two graphs are computed: one for the 1D keys and one for the 3D keys. A Python implementation of B^+ [40] trees is used to store these adjacency lists; further reasoning behind this storage mechanism has been reported [41]. After computing the adjacency lists for each file, these are aggregated to one global adjacency list for each dimension.

Each one-dimensional model uses vectors of size $n_1 * 150$, where n_1 is the number of unique 1D keys and 150 is the embedding dimension. The same embedding dimension is used for the three-dimensional models, which use vectors of size $n_3 * 150$ where n_3 is the number of unique 3D keys.The list of protein files was split into training and test sets where 80% of the proteins were allocated to training and the remaining 20% for testing. Each training process was performed on a NVIDIA Tesla V100 GPU node on Maple cluster using pytorch [42].

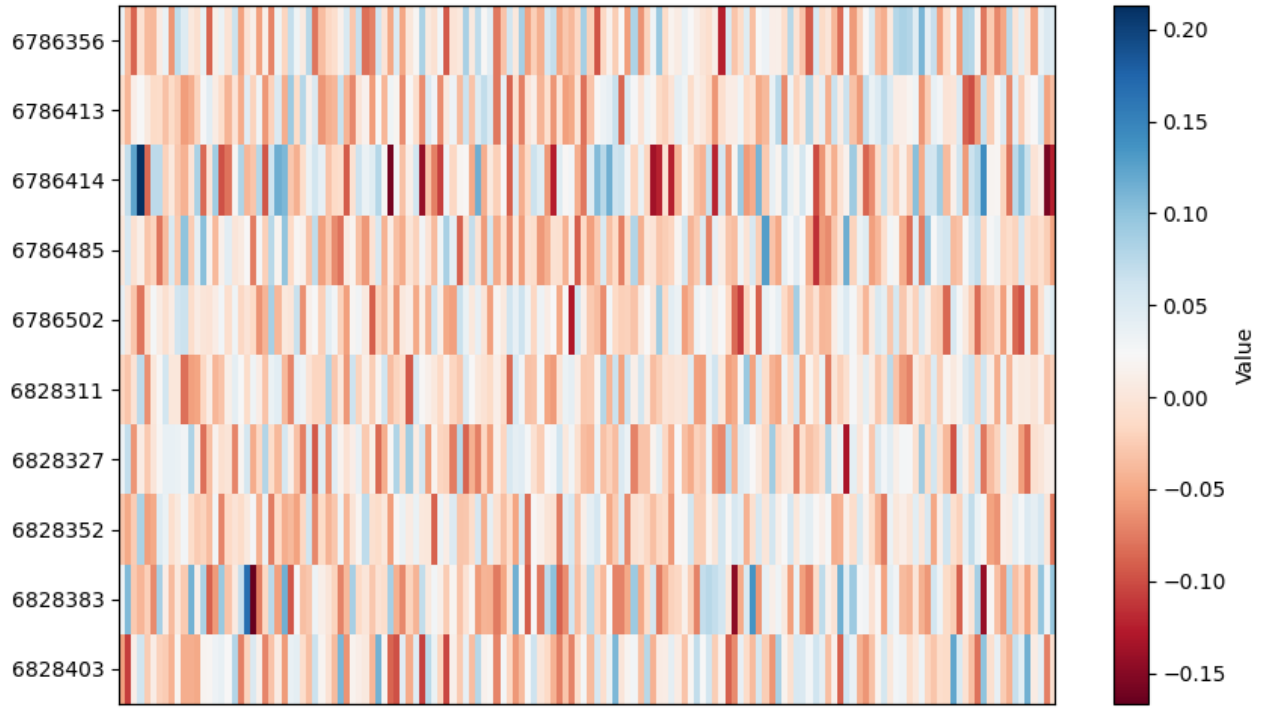

Fig. 1: Visualization of the embeddings after training.

I implemented word2vec using the SkipGram with negative sampling (SGNS) model. I chose to use 5 positive samples so the model focused on finding interchangeable keys more than contextual similarity. For a key k, the positive samples are the 5 edges containing k with the largest weights. The number of negative samples per positive sample, γ, allowed me to tweak the model training more, since increasing the number of positive samples negatively impacted the model's ability to find "synonyms", or interchangeable keys. For model

training, I used $\gamma = 3$, $\gamma = 5$, and $\gamma = 7$ to evaluate the effect the number of negative samples would have on building the word2vec models.

The GloVe model required additional hyperparameter tuning to obtain results; especially due to the amount of co-occurrences. Unlike the word2vec model which selects a subset of the data by default, the GloVe model uses the full global dataset. The loss function J is defined as the following:

$$J = \sum_{i,j} f(w_{ij})(k_i^T k_j - log(w_{ij}))^2 \tag{3.0.5}$$

where w_{ij} is the weight associated with the edge $e = \{k_i, k_j\}$, k_i^T and k_j are the embeddings for k_i and k_j, respectively, and $f(w_{ij})$ is the weighting function, defined below:

$$f(w_{ij}) = \begin{cases} (\frac{w_{ij}}{x_{max}})^\alpha & \text{if } w_{ij} \leq x_{max} \\ 1 & \text{otherwise} \end{cases} \tag{3.0.6}$$

where w_{ij} is the number of co-occurrences and x_{max} is the maximum number of co-occurrences considered significant. Further discussion of the selection of the weighting function has been reported; the original authors found that an α value of 0.75 was most effective in their work [35]. However, I was interested to see the impact of the weighting function in the model training, so I used three alpha values: $\alpha = 0.5$, $\alpha = 0.75$, and $\alpha = 1$ to investigate the effect in this particular context.

After training the models, each pair of 1D and 3D models were aligned to each other using a simple rotation technique. The rotation matrix R was computed using the scipy [43] implementation of Orthogonal Procrustes Analysis [44], which finds the matrix that minimizes the squared Frobenius norm of the difference between the two embeddings. The matrix is then applied to the 3D embeddings by multiplying the two together; this newly

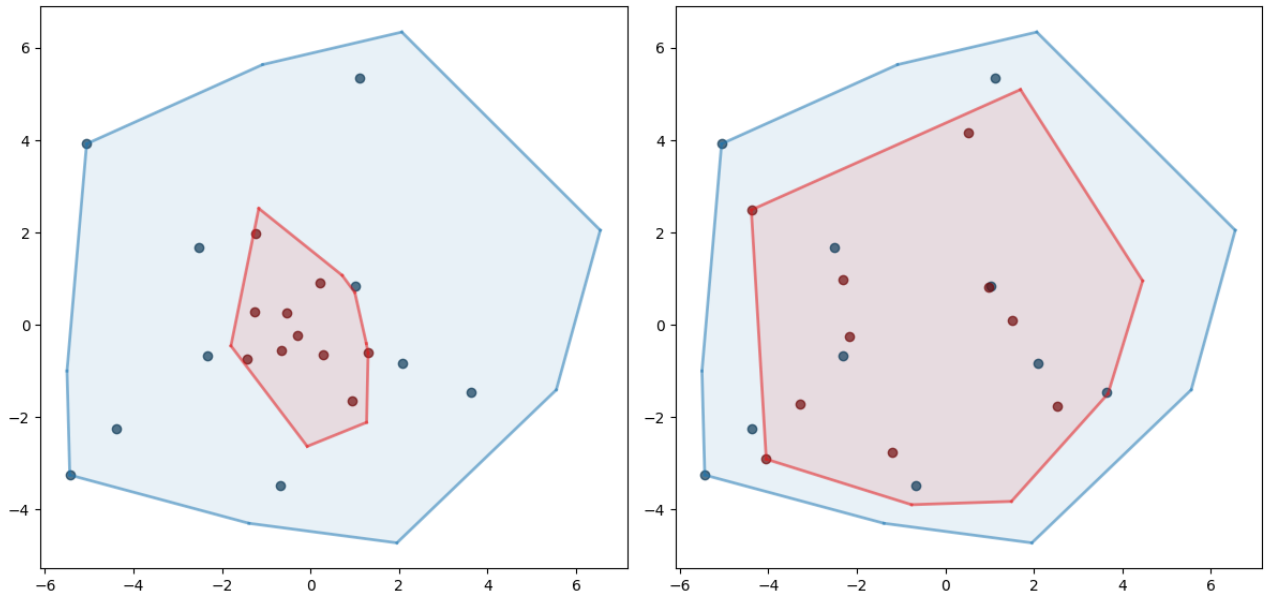

Fig. 2: Before and after rotational alignment.

aligned set of 3D embeddings is used for the translation model. Experiments using the non-aligned set of embeddings and the aligned set of embeddings show a significant increase in accuracy when using the aligned embeddings.

After alignment, the translation model requires the two sets of embeddings, a key k, and length P of similarity allowance. If k is a 1D key, then the P closest embeddings are returned from the 3D embeddings, where closeness is measured by cosine similarity; the nature of the model means it is effective for translation from 1D to 3D and from 3D to 1D.

4 EVALUATION AND RESULTS

Each model was evaluated on the same set of test data to ensure consistency in the comparison process. The test data was read in as a set of 1D keys with their actual corresponding 3D keys in the test set. For each direction - 1D to 3D and 3D to 1D - the translation model was given a key k_1 and a range P, and it generated the P closest keys for the opposite dimension. If the corresponding key k_3 was found in the set of P closest keys, it was considered a positive; otherwise, it was not. Accuracy was given as a score $\frac{numpositives}{totalkeys}$. As there are a significant number of keys and the average number of closely related keys is unknown, I chose to evaluate using $P = 20$.

The evaluation function can be written as the following:

$$accuracy_x = (\sum_{i=1}^{n_x} f(k_{xi})) \div n_x \tag{4.0.1}$$

$$f(k_x) = \begin{cases} 1 & \text{if } k_y \in translate(k_x, P) \\ 0 & \text{otherwise} \end{cases} \tag{4.0.2}$$

where the x and y are dimensions, n_x is the number of keys in that dimension, k_x and k_y are corresponding keys in different dimensions, $translate(k_x, P)$ is the set of keys $\{k_{y1}, k_{y2} \ldots k_{y(P-1)}, k_{yP}\}$ containing the keys from dimension y closest to k_x, and the $accuracy_x$ represents the accuracy of translation from x dimension to y dimension.

The findings of this work have to be seen in light of some limitations. Protein data is notoriously large and computationally expensive to work with; the amount of data I was able to process was affected by the memory and time constraints present in the supercomputing

nodes I had access to. The accuracy of the models is therefore subject to the limitations of being able to process only a small portion of the protein data available, and may not be fully realizing the potential of this methodology for translation between keys. Further work on this topic will require more extensive parallelization, specifically in the calculation of the adjacency lists, likely using CUDA or a similar framework to fully realize the potential of the GPUs available; however, implementation of CUDA-based processing was outside the scope of this work. However, even the worst performing models do still perform significantly better than random chance, indicating that this is a viable technique.

TABLE I
WORD2VEC ACCURACY WHEN P = 20

	accuracy	
γ	1D $\rightarrow 3D$	3D $\rightarrow 1D$
5	0.00005	0.000042
7	0.00016	0.000084

The word2vec model shows an small increase in performance when a greater number of negative samples per positive sample is used, particularly over several epochs; however, too small or too large of a γ value results in the model not converging. I believe that this is at least due in part to the fact that even if a key re-occurs multiple times, it is only trained on once due to the structure of the model, where in a typical word2vec model it would generate a number of positive and negative samples corresponding to the frequency of appearance in the documents. Potential future optimizations to this model may include tracking a frequency counter when generating the adjacency lists, where the number of positive samples is a function of the frequency count rather than a fixed value. Also notable is the fact that 1D to 3D translation generally outperformed 3D to 1D translation, although the cause is less clear; there are generally fewer 1D keys than 3D keys but there are otherwise no significant differences between the two.

While the word2vec model showed improvement with increasing numbers of negative samples, the GloVe model was not able to converge despite significant experimentation with

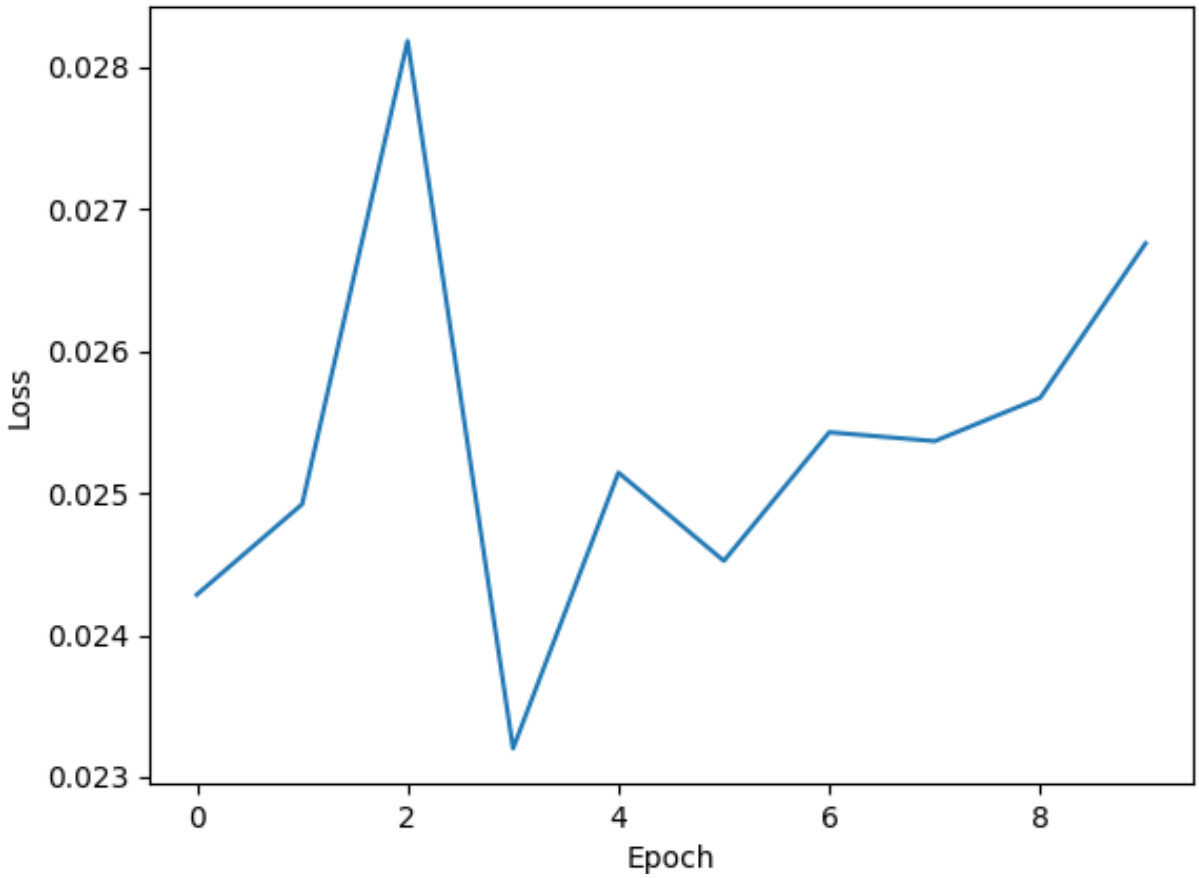

Fig. 3: GloVe training loss over multiple epochs.

hyperparameters. While local testing on a small scale with a validation set proved successful, I was unable to replicate it on the larger dataset. Hyperparameter experimental values included x_{max} values from 50-1000, alpha values of 0.5, 0.75, and 1.0, number of epochs from 3-10. Experimentation with reducing the number of values present in the co-occurrence matrix was also done in an attempt to encourage faster convergence. The primary limitation with the GloVe training was time - when using the full co-occurrence dataset, the model was not reaching convergence over days of training; however, the smaller dataset does not appear to support convergence either, as seen in Fig. 3.

Further improvements of this technique should be investigated. Similar techniques for embedding alignment generally use translation reference points; while this may require additional pre-processing and further domain knowledge that may not currently be available, it could further improve the alignment process. Potential future research directions include

utilizing back-translation to improve the translation; this has been proven to significantly improve unsupervised learning methods for language translation models [45].

Given that the word2vec method outperformed the GloVe method I conclude that local-based embedding techniques should be further explored for this specific problem. Word2vec is one of the most common techniques used, however, similar and more advanced local techniques such as FastText [46], ELMo (Embeddings from Langauge Models) [47], BERT (Bidirectional Encoder Representations for Language Understanding) [48], or ULMFiT (Universal Language Model Fine-tuning) [49]. In particular, I think FastText may provide an interesting basis for future work due the way keys are generated; it may be able to find patterns inside the keys that can be recognized across dimensions.

5 CONCLUSION

Understanding the relationship between sequence and structure is a critical problem in protein studies; protein structure comparison is at the heart of this problem. Using the TSR-based key generation method, I examined the relationship between 1D and 3D keys as a machine translation problem. As the TSR-based method is a local-global structural comparison method, I investigated both local and global embedding techniques as potential bases for the translation model.

Embeddings were generated based on co-occurrence frequencies; where co-occurrence frequency represented the number of times the distance between two keys was less than or equal to a variable d. Word2vec was used as the local embedding method, while GloVe was chosen as the global embedding method. Both methods had individual challenges and limitations on how the data could be adapted to be compatible with the models; additionally, both methods faced challenges due to the amount of data being processed. A variety of hyperparameters were tuned to improve these models, however, the results focus specifically on the effect of the number of negative samples for the word2vec model, and the effect of the weighting function parameter alpha on the GloVe model.

After training the sets of embeddings for 1D and 3D keys, they were aligned using a rotational matrix found through Orthogonal Procrustes Analysis to ensure that the embedding spaces trained separately could be overlapped. The translation model, given a key and a variable P, returns the closest P embeddings from the opposite dimension. Due to the number of keys present - approximately 340,000 unique 1D keys and 435,000 unique 3D keys - I evaluated the models using a larger P to determine an accuracy score.

The global-based embedding technique (GloVe) struggled to converge despite extensive experimentation with hyperparameters. In contrast, the local-based embedding technique (word2vec) showed improvement with increasing numbers of negative samples. Future work should explore additional local-based embedding techniques like FastText or BERT to further improve the translation model outlined in this work. Additional optimizations to the word2vec model should be developed as well to improve its accuracy. Models with a variable number of positive samples, alternative values of d, and a variety of hyperparameters such as learning rate, number of epochs, should be explored, however, these specific optimizations are outside the scope of this work.

In summary, I proposed a novel methodology for examining the relationship between sequence and structure using TSR-based keys, developed a similarity function for understanding the relationship between keys in a singular dimension, and evaluated local and global-based embedding techniques. This work shows that local embedding techniques have a significantly improved performance over global-based embedding techniques, and suggests several directions forward for similar work. The novel translation method introduced in this paper provides a basis for future translation models seeking to examine the relationship between sequence and structure.

BIBLIOGRAPHY

[1] Z. Li, L. Jaroszewski, M. Iyer, M. Sedova, and A. Godzik, "Fatcat 2.0: towards a better understanding of the structural diversity of proteins," *Nucleic acids research*, vol. 48, no. W1, pp. W60–W64, 2020.

[2] J. Razmara, S. Fotoohi, and S. Parvizpour, "Flexible protein structure alignment based on topology string alignment of secondary structure," *International Journal of e-Education, e-Business, e-Management and e-Learning*, vol. 4, no. 1, p. 19, 2014.

[3] S. Salem, M. J. Zaki, and C. Bystroff, "Flexsnap: flexible non-sequential protein structure alignment," *Algorithms for Molecular Biology*, vol. 5, pp. 1–13, 2010.

[4] H.-W. Wang, C.-H. Chu, W.-C. Wang, and T.-W. Pai, "A local average distance descriptor for flexible protein structure comparison," *BMC bioinformatics*, vol. 15, pp. 1–13, 2014.

[5] A. Abyzov and V. A. Ilyin, "A comprehensive analysis of non-sequential alignments between all protein structures," *BMC structural biology*, vol. 7, pp. 1–20, 2007.

[6] P. Brown, W. Pullan, Y. Yang, and Y. Zhou, "Fast and accurate non-sequential protein structure alignment using a new asymmetric linear sum assignment heuristic," *Bioinformatics*, vol. 32, no. 3, pp. 370–377, 2016.

[7] C. Zhang and A. M. Pyle, "A unified approach to sequential and non-sequential structure alignment of proteins, rnas, and dnas," *Iscience*, vol. 25, no. 10, 2022.

[8] B. Zhu, "Protein local structure alignment under the discrete fréchet distance," *Journal of Computational Biology*, vol. 14, no. 10, pp. 1343–1351, 2007.

[9] B. Pang, N. Zhao, M. Becchi, D. Korkin, and C.-R. Shyu, "Accelerating large-scale protein structure alignments with graphics processing units," *BMC research notes*, vol. 5, pp. 1–11, 2012.

[10] A. Zemla, "Lga: a method for finding 3d similarities in protein structures," *Nucleic acids research*, vol. 31, no. 13, pp. 3370–3374, 2003.

[11] L. Holm and C. Sander, "Protein structure comparison by alignment of distance matrices," *Journal of molecular biology*, vol. 233, no. 1, pp. 123–138, 1993.

[12] L. Holm, "Using dali for protein structure comparison," *Structural Bioinformatics: Methods and Protocols*, pp. 29–42, 2020.

[13] L. Sael, B. Li, D. La, Y. Fang, K. Ramani, R. Rustamov, and D. Kihara, "Fast protein tertiary structure retrieval based on global surface shape similarity," *Proteins: Structure, Function, and Bioinformatics*, vol. 72, no. 4, pp. 1259–1273, 2008.

[14] J. Emonts and J. F. Buyel, "An overview of descriptors to capture protein properties–tools and perspectives in the context of qsar modeling," *Computational and Structural Biotechnology Journal*, 2023.

[15] M. R. Betancourt and J. Skolnick, "Universal similarity measure for comparing protein structures," *Biopolymers: Original Research on Biomolecules*, vol. 59, no. 5, pp. 305–309, 2001.

[16] I. Kufareva and R. Abagyan, "Methods of protein structure comparison," *Homology modeling: Methods and protocols*, pp. 231–257, 2012.

[17] Y. Zhang and J. Skolnick, "Scoring function for automated assessment of protein structure template quality," *Proteins: Structure, Function, and Bioinformatics*, vol. 57, no. 4, pp. 702–710, 2004.

[18] Y. Zhang and J. Skolnick, "Tm-align: a protein structure alignment algorithm based on the tm-score," *Nucleic acids research*, vol. 33, no. 7, pp. 2302–2309, 2005.

[19] T. Narayanan, M. Gersten, S. Subramaniam, and A. Grama, "Modularity detection in protein-protein interaction networks," *BMC research notes*, vol. 4, pp. 1–6, 2011.

[20] P. Mier, G. Alanis-Lobato, and M. A. Andrade-Navarro, "Protein-protein interactions can be predicted using coiled coil co-evolution patterns," *Journal of Theoretical Biology*, vol. 412, pp. 198–203, 2017.

[21] I. Koch and T. Lengauer, "Detection of distant structural similarities in a set of proteins using a fast graph-based method.," in *ISMB*, pp. 167–178, 1997.

[22] J. R. Ullmann, "An algorithm for subgraph isomorphism," *Journal of the ACM (JACM)*, vol. 23, no. 1, pp. 31–42, 1976.

[23] O. Bachar, D. Fischer, R. Nussinov, and H. Wolfson, "A computer vision based technique for 3-d sequence-independent structural comparison of proteins," *Protein Engineering, Design and Selection*, vol. 6, no. 3, pp. 279–287, 1993.

[24] J. D. Szustakowski and Z. Weng, "Protein structure alignment using a genetic algorithm," *Proteins: Structure, Function, and Bioinformatics*, vol. 38, no. 4, pp. 428–440, 2000.

[25] T. Blundell, D. Carney, S. Gardner, F. Hayes, B. Howlin, T. Hubbard, J. Overington, D. A. Singh, B. L. Sibanda, and M. Sutcliffe, "Knowledge-based protein modelling and design," *European Journal of Biochemistry*, vol. 172, no. 3, pp. 513–520, 1988.

[26] W. R. Taylor and C. A. Orengo, "Protein structure alignment," *Journal of molecular biology*, vol. 208, no. 1, pp. 1–22, 1989.

[27] P. Lackner, W. A. Koppensteiner, M. J. Sippl, and F. S. Domingues, "Prosup: a refined tool for protein structure alignment," *Protein Engineering*, vol. 13, no. 11, pp. 745–752, 2000.

[28] A.-S. Yang and B. Honig, "An integrated approach to the analysis and modeling of protein sequences and structures. i. protein structural alignment and a quantitative measure for protein structural distance," *Journal of molecular biology*, vol. 301, no. 3, pp. 665–678, 2000.

[29] I. N. Shindyalov and P. E. Bourne, "Protein structure alignment by incremental combinatorial extension (ce) of the optimal path.," *Protein engineering*, vol. 11, no. 9, pp. 739–747, 1998.

[30] E. Krissinel and K. Henrick, "Secondary-structure matching (ssm), a new tool for fast protein structure alignment in three dimensions," *Acta Crystallographica Section D: Biological Crystallography*, vol. 60, no. 12, pp. 2256–2268, 2004.

[31] J. Shapiro and D. Brutlag, "Foldminer: structural motif discovery using an improved superposition algorithm," *Protein Science*, vol. 13, no. 1, pp. 278–294, 2004.

[32] S. Kondra, T. Sarkar, V. Raghavan, and W. Xu, "Development of a tsr-based method for protein 3-d structural comparison with its applications to protein classification and motif discovery," *Frontiers in Chemistry*, vol. 8, p. 602291, 2021.

[33] S. Kondra, F. Chen, Y. Chen, Y. Chen, C. J. Collette, and W. Xu, "A study of a hierarchical structure of proteins and ligand binding sites of receptors using the triangular spatial relationship-based structure comparison method and development of a size-filtering feature designed for comparing different sizes of protein structures," *Proteins: Structure, Function, and Bioinformatics*, vol. 90, no. 1, pp. 239–257, 2022.

[34] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[35] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543, 2014.

[36] Q. Le and T. Mikolov, "Distributed representations of sentences and documents," in *International conference on machine learning*, pp. 1188–1196, PMLR, 2014.

[37] M. Grbovic, V. Radosavljevic, N. Djuric, N. Bhamidipati, J. Savla, V. Bhagwan, and D. Sharp, "E-commerce in your inbox: Product recommendations at scale," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '15, (New York, NY, USA), p. 1809–1818, Association for Computing Machinery, 2015.

[38] H. De Meulemeester and B. De Moor, "Unsupervised embeddings for categorical variables," in *2020 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, IEEE, 2020.

[39] M. Mimura and R. Ito, "Applying nlp techniques to malware detection in a practical environment," *International Journal of Information Security*, vol. 21, no. 2, pp. 279–291, 2022.

[40] Z. Foundation, Feb 2024.

[41] A. Swensen, "Improving adjacency list storage methods for polypeptide similarity analysis," 2022.

[42] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, vol. 32, 2019.

[43] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020.

[44] P. H. Schönemann, "A generalized solution of the orthogonal procrustes problem," *Psychometrika*, vol. 31, no. 1, pp. 1–10, 1966.

[45] M. Ott, M. Ott, M. Ranzato, and G. Lample, "Unsupervised machine translation: A novel approach to provide fast, accurate translations for more languages," Jun 2020.

[46] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *Transactions of the association for computational linguistics*, vol. 5, pp. 135–146, 2017.

[47] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations. corr abs/1802.05365 (2018)," *arXiv preprint arXiv:1802.05365*, vol. 42, 1802.

[48] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[49] J. Howard and S. Ruder, "Universal language model fine-tuning for text classification," *arXiv preprint arXiv:1801.06146*, 2018.

APPENDIX

APPENDIX A

TABLE II
PROTEIN DATA DETAILS

protein	**chain**	**group**	**group1**	**percent_threshold**	**all_keys**
1A25	A	PKC	PKABC	100	374660
1DSY	A	PKC	PKABC	100	419220
1GMI	A	PKC	PKABC	100	400995
3GPE	A	PKC	PKABC	100	419220
3RDJ	A	PKC	PKABC	100	419220

VITA

Arianna N. Swensen was born in Rogers, Arkansas on April 1, 2001. She graduated from Nixa High School with honors in 2019 and enrolled in the University of Mississippi the following August. In December 2022 she graduated with honors as a Sally McDonnell Barksdale Honors College Scholar, earning her Bachelor of Science in Computer Science with an emphasis in Data Science and a minor in Math for School of Engineering. She enrolled as a graduate student at the University of Mississippi in January of 2023.