Interpretable Structured Learning with Sparse Gated Sequence Encoder for Protein-Protein Interaction Prediction

Kishan K C*, Feng Cui†, Anne R. Haake*, Rui Li*
*Golisano College of Computing and Information Sciences, Rochester Institute of Technology, NY, USA
Email: {kk3671, arhics, rxlics}@rit.edu
†Thomas H. Gosnell School of Life Sciences, Rochester Institute of Technology, NY, USA
Email: fxcsbi@rit.edu

***Abstract*—Predicting protein-protein interactions (PPIs) by learning informative representations from amino acid sequences is a challenging yet important problem in biology. Although various deep learning models in Siamese architecture have been proposed to model PPIs from sequences, these methods are computationally expensive for a large number of PPIs due to the pairwise encoding process. Furthermore, these methods are difficult to interpret because of non-intuitive mappings from protein sequences to their sequence representation. To address these challenges, we present a novel deep framework to model and predict PPIs from sequence alone. Our model incorporates a bidirectional gated recurrent unit to learn sequence representations by leveraging contextualized and sequential information from sequences. We further employ a sparse regularization to model long-range dependencies between amino acids and to select important amino acids (protein motifs), thus enhancing interpretability. Besides, the novel design of the encoding process makes our model computationally efficient and scalable to an increasing number of interactions. Experimental results on up-to-date interaction datasets demonstrate that our model achieves superior performance compared to other state-of-the-art methods. Literature-based case studies illustrate the ability of our model to provide biological insights to interpret the predictions.**

I. Introduction

Proteins are the functional units within an organism that form molecular machines organized by their protein-protein interactions (PPIs) to carry out many biological and molecular processes. The study of PPIs not only plays a crucial role in understanding biological phenomena but also provides insights about the molecular etiology of diseases as well as the discovery of putative drug targets [1]. Although a large amount of reliable PPIs has been experimentally identified, the discovery of all possible PPIs through these experiments is intractable. Thus, efficient computational methods capable of accurately predicting PPIs and speeding up new PPI discovery are needed.

The primary structure of a protein, the protein sequence, determines the protein's unique three-dimensional shape, giving rise to an assumption that knowledge of the protein sequence alone might be sufficient to model the interaction between two proteins [2], [3]. There is a longstanding interest in predicting PPIs from protein sequences which are by far the most abundant data available for proteins [4]. Traditional methods proposed to model PPIs involve extracting features based on domain expertise and training machine learning model on these features [4]–[6]. The performance of these methods relies heavily on the capability to select appropriate features, while the extracted features lack enough information about the interactions.

Recently, the state-of-the-art deep learning methods in the Siamese architecture have been proposed to automatically extract features from sequences to model interactions in an end-to-end framework [7], [8]. Siamese architecture for PPI prediction consists of two identical neural networks each taking one of the protein sequences and modeling their mutual influence. Although these methods perform quite well on small datasets, these methods involve the pairwise encoding process, making it computationally inefficient for a large number of interactions [9]. This has significant memory and runtime limitations when training with large datasets.

In this work, we propose a novel method to address these challenges. Specifically, we aim to learn a bidirectional GRU (BiGRU) model that maps variable-length sequences to a sequence of vector representations - one per amino acid position that encodes the sequential and contextual properties of amino acids. Since proteins interact with other proteins that perform different functions within a cell and even have different sequence patterns, we further encode the representation to the Gaussian distribution instead of a single point to capture uncertainty about its representation. We then define the cost function that incorporates the contrastive criteria between the latent Gaussian distributions such that the similarity between these distributions effectively captures complex interactions between proteins. It allows our model to minimize the statistical distance between interacting proteins while maximizing the distance for non-interacting proteins.

Alongside making accurate PPI predictions, it is crucial to have interpretable models that help domain experts understand how individual amino acids in the sequence contribute to the model's decisions. However, none of the state-of-the-art methods can provide such interpretability, limiting their practicality from biological perspectives. Since only a few amino acids in the interface region of the sequences are

involved in interactions with other proteins [10], we, therefore, design a sparse and structured gate mechanism to guide the model to selectively focus on few amino acids in the sequence. The sparse gating mechanism outputs sparse weights - one per amino acid position - that explains how much contribution each amino acid makes and thus enhances interpretability.

Experimental results show that our method outperforms state-of-the-art methods on two challenging datasets: yeast and human PPIs from the BioGRID interaction database. Finally, we demonstrate that the learned sparse gate values correspond to the biologically interpretable protein motifs. A literature-based case study illustrates that our model effectively learns to identify the important residues from the sequence.

II. Related work

Traditional methods focus on extracting features from protein sequences such as autocovariance (AC) [5], conjoint triads (CT) [3] and composition-transition-distribution (CTD) [4] descriptors and training a binary classifier on these features to predict PPIs [5], [6]. Since these extracted features only summarize the specific aspects of protein sequences such as physicochemical properties, frequencies of local patterns, and the positional distribution of amino acids, they lack enough information about the interactions.

Recently, deep learning architectures have been developed to address PPI prediction by automatically extracting useful features from protein sequences [7], [8]. These methods adopt deep-Siamese like neural networks to model the mutual influence between protein sequences. They use encoder based on a convolutional neural network (CNN) to capture local features and recurrent neural network (RNN) to capture sequential and contextualized features from protein sequences. The encoder encodes a pair of sequences to lower-dimensional sequence vectors and a binary classifier predicts the probability of interaction based on these sequence vectors.

Specifically, DPPI [7] uses a deep-CNN based Siamese architecture that focuses on capturing local patterns from protein evolutionary profiles [11]. However, it requires extensive effort in data-preprocessing, specifically in constructing evolutionary profiles from protein sequences using Position-Specific Iterative BLAST (PSI-BLAST) [12]. To construct an evolutionary profile for a protein sequence, PSI-BLAST searches against NCBI non-redundant protein database with nearly 184 million sequences, which is time-consuming and makes it unscalable to a large number of protein sequences. However, PIPR [8] incorporates a deep residual recurrent convolutional neural network (RCNN) for PPI prediction using only the sequences of the protein pair. PIPR uses residual RCNN encoder that combines CNN to capture local features and RNN to capture sequential and contextualized features from protein sequences. The sequence representation for each protein is obtained by feeding its sequence through the deep sequence encoder with multiple layers of RCNN units. The non-intuitive mapping from protein sequences to their sequence representation makes the representation difficult to interpret.

III. Method

A protein sequence $\mathbf{s}$ is a list of amino acids $[a_1, a_2, \ldots, a_L]$ where a_l is the amino acid at position l and L is the length of the sequence. We aim to map a protein sequence $\mathbf{s}$ to a lower-dimensional Gaussian distribution with formal format as follows: $f : \mathbf{s} \rightarrow (\mu, \Sigma)$, where $\mu \in \mathbb{R}^d$, and $\Sigma \in \mathbb{R}^{d \times d}$ with $d \ll L$ denoting the dimension of the Gaussian distribution. We employ the Wasserstein distance as a measure of how different are the encoded Gaussian distributions of protein sequences. In this work, the goal of PPI prediction is to learn a model that predicts a smaller Wasserstein distance between the Gaussian distributions of interacting proteins and a larger difference for non-interacting proteins.

We introduce our deep learning framework for PPI prediction from sequences. The overall architecture of the proposed framework is illustrated in Figure 1. In step 1, the sequence encoder incorporates a bidirectional gated recurrent unit (Bi-GRU) to encode the amino acid sequence to sequence of vector representations. In step 2, the importance gate models dependencies between all the positions in the sequence, which could allow it to directly model residue-residue dependencies. This enables the model to compute the importance of amino acid in each position based on the sequence of vector representations. In step 3, the representation of sequence is encoded into Gaussian representation with mean μ and Σ. In step 4, a pairwise ranking loss is employed to minimize the statistical distance between interacting proteins while maximizing the distance for non-interacting proteins.

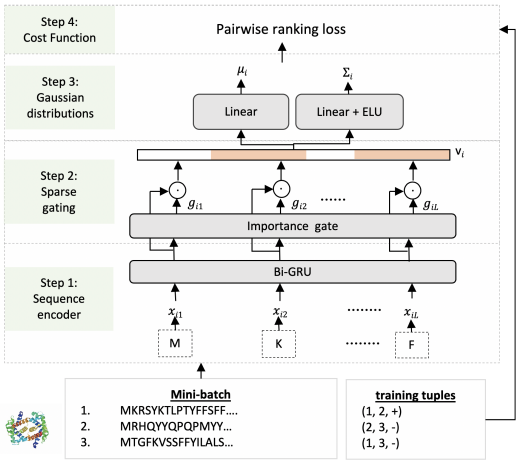


Fig. 1. Diagram of the model. A mini-batch of protein sequence is encoded to Gaussian distribution μ and Σ using BiGRU encoder with sparse regularization. Model is trained by optimizing the pairwise Wasserstein distance between the training tuples (i.e. positive and negative interactions).

A. Sparse gating sequence encoder

*1) **Protein sequence encoder**:* In step 1 as shown in Figure 1, the encoder takes a sequence of amino acids and encodes it to a sequence of vector representations - one per amino acid position. In particular, a one-hot encoded representation

of amino acid a_l is embedded to a vector representation x_l through an embedding matrix:

$$x_l = \mathbf{W}_e a_l \tag{1}$$

where $\mathbf{W}_e \in \mathbb{R}^{N \times d}$ is the weight of the embedding layer.

To learn the sequential and contextualized representation of the amino acids in the sequence $\mathbf{s}$, we adopt the bidirectional Gated Recurrent Unit (BiGRU) that summarizes the sequence information from both directions. The sequence of vector representation x_l is then fed into the bidirectional gated recurrent unit. It contains two encoding processes: a forward encoding $\overrightarrow{\text{GRU}}$ which processes the sequence $\mathbf{s}$ from position 1 to L , and a backward encoding $\overleftarrow{\text{GRU}}$ which processes the sequence $\mathbf{s}$ from position L to 1.

$$h_l = \text{BiGRU}(x_l) = [\overrightarrow{\text{GRU}}(x_l), \overleftarrow{\text{GRU}}(x_l)] \tag{2}$$

where h_l represents the GRU hidden state for amino acid a_l. The representation of amino acid h_l is the concatenation of forward hidden state $\overrightarrow{h}_l$ and backward hidden state $\overleftarrow{h}_l$ which summarizes the information of whole sequence centered around a_l.

*2) **Sparse gating mechanism**:* Proteins bind to each other at specific binding domains on each protein. These domains can be just a few peptides long or span hundreds of amino acids. For this purpose, we introduce additional gates $\mathbf{g} = \{g_1, g_2, \dots g_L\}$ indicating the activation of each amino acid. The amino acid a_l is active if $g_l > 0$ and is inactive when g_l is 0. The gate values g_l for the amino acid a_l, i.e., $g_l \in [0, 1]$ with 1 representing high importance. Let $S = \{l | g_l > 0\}$ be the set of amino acid that are active indicated by their respective gate values. We obtain the representation for protein sequences by scaling the hidden state h_l with their respective gates:

$$v_l = g_l \odot h_l \tag{3}$$

$$\mathbf{v} = \text{Concat}_{(l \in S)}(v_l) \tag{4}$$

where $\text{Concat}_{(l \in S)}$ represents the concatenation of sequence vectors for positions with $g_l > 0$. $\odot$ denotes the elementwise product between gate values g_l and GRU hidden state h_l. The sparse gate values g_l leads to the sparse representation $\mathbf{v}$ of the sequence. In contrast, for the positions with $g_l = 0$, the hidden states h_l of these positions are not included in the representation $\mathbf{v}$. The sparse gates act as the controllers to selectively activate the part of the network to account only for the subset of amino acids of the sequence.

We introduce an auxillary network that takes the GRU hidden states $h_{(.)}$ and generates the gate values for each position to determine whether the amino acid at that position is important for PPI prediction. The auxillary network models the long-range pairwise dependencies between amino acids in the sequence. With the auxillary network, our model explicitly considers dependencies between all position in the sequence, which could allow it to directly model residue-residue dependencies. In step 2 of Figure 1, GRU hidden states h_l is transformed to score p_l as:

$$p_l = \mathbf{W}_2(\tanh(\mathbf{W}_1 h_l + \mathbf{b}_1)) + \mathbf{b}_2 \tag{5}$$

where $\mathbf{W}_1 \in \mathbb{R}^{d \times d}, \mathbf{W}_2 \in \mathbb{R}^{d \times 1}$, $\mathbf{b}_1 \in \mathbb{R}^d$ and $\mathbf{b}_2 \in \mathbb{R}^1$ are the weight matrices and biases for the linear layers. Let $\mathbf{p} = [p_1, p_2, \dots, p_L]$ is a vector of scores for amino acids in the sequence $\mathbf{s}$. Next, we convert the vector of scores $\mathbf{p}$ to a probability distribution $\mathbf{g}$ so that $\sum_{l=1}^{L} g_l = 1$ and $0 \leq g_l \leq 1$. This allows us to quantify the relative contribution of each amino acid for the sequence representation. The softmax function is a simple choice to map the vector $\mathbf{p}$ to a probability distribution defined as:

$$\text{softmax}_i(\mathbf{p}) = \frac{\exp(p_i)}{\sum_j \exp(p_j)} \tag{6}$$

Since the resulting softmax distribution has full support, i.e., $\text{softmax}(p_l) > 0$ for every a_l, it forces all the amino acids in the sequence to receive some probability mass. Sotmax distribution assigns weights to each amino acid and even an unimportant amino acids have small weights, the weights on important amino acids become much smaller for long sequences, leading to degraded performance. However, not all of the amino acids in the sequence contribute towards the certain functions or interactions.

We introduce a sparsity regularization on $\mathbf{p}$ to only select the important set of amino acids, which may corresponds to the interface and may be important for interaction prediction. We employ sparsemax [13] as gate mechanism:

$$\text{sparsemax}(\mathbf{p}) := \underset{\mathbf{g} \in \Delta^K}{\text{argmin}} \|\mathbf{g} - \mathbf{p}\|^2, \tag{7}$$

where $\Delta^K := \{\in \mathbb{R}^K | \mathbf{g} \geq 0, \mathbf{1}^T \mathbf{g} = 1\}$ and $\mathbf{g}$ represents the Euclidean projection of $\mathbf{p}$ onto the $(K-1)$-dimensional probability simplex. The projection is likely to hit the boundary of the simplex, leading to sparse outputs, which allows the encoder to select only an informative subset of amino acids in the sequence. Although sparsemax leads to sparse representation, the sparse weights may only capture few important amino acids but may not identify a relatively long stretches of amino acids.

Furthermore, to enable our model to selectively focus on relatively longer stretches of amino acids, we present fused-max regularization [14] that not only results in the sparse representations but also encourages the encoder to assign equal weights for contiguous sets of amino acids of the sequence while predicting interactions:

$$\text{fusedmax}(\mathbf{p}) := \underset{\mathbf{g} \in \Delta^K}{\text{argmin}} \frac{1}{2}\|\mathbf{g} - \frac{\mathbf{p}}{\gamma}\|^2 + \lambda \sum_{j=1}^{L-1} |\mathbf{g}_{j+1} - \mathbf{g}_j| \tag{8}$$

where γ controls the regularization strength and λ is the tuning parameter that balances the attention to produce sparse outputs and to assign equal weights to the amino acids within each segment. In Eq. 8, the first part projects the scores $\mathbf{p}$ to the

probability simplex and the second part encourages paying equal attention to adjacent amino acids in the sequence. This allows the model to identify long stretches of amino acids that may bind with the residues from the interacting proteins.

*3) **Encoding sequences as Gaussian distributions**:* Within an organism, a given protein may be involved in a complex interplay with various proteins that perform different functions within a cell and even have different sequence patterns. Such differences should be reflected in the uncertainty of its representation $\mathbf{v}$. To model the uncertainty about the representation [15], [16], sequence representation $\mathbf{v}$ is then encoded to μ and Σ in the final layer of the architecture as shown in step 3 in Figure 1. To ensure that covariance matrices Σ is positive definite, we use Exponential Linear Unit [17] in the final layer.

$$\mu = \mathbf{W}_\mu \mathbf{v} + \mathbf{b}_\mu \tag{9}$$

$$\Sigma = \text{ELU}(\mathbf{W}_\Sigma \mathbf{v} + \mathbf{b}_\Sigma) + 1 \tag{10}$$

where $\mathbf{W}_\mu, \mathbf{W}_\Sigma, \mathbf{b}_\mu$ and $\mathbf{b}_\Sigma$ denote the weight matrices and biases of linear layers that project intermediate representation $\mathbf{v}$ to mean μ and variance Σ of Gaussian representation of the sequence $\mathbf{s}$. We denote a protein sequence $\mathbf{s}$ with a d-dimensional Gaussian distribution (μ, Σ), where μ represents the center point of the sequence representation in the latent space and Σ represents the uncertainty associated with the representation. With an assumption that the different dimensions of the latent Gaussian distribution are independent of each other, the covariance matrix Σ is a diagonal matrix.

B. Loss function definition

We employ the Wasserstein distance to measure the similarity between the Gaussian distributions of the proteins to make PPI prediction. Wasserstein distance allows the model to capture the transitivity property of PPIs that measures the tendency of proteins to cluster together into functional modules and protein complexes [18].

The p^{th} Wasserstein distance between two probability measures μ and ν is defined as:

$$W_p(\mu, \nu)^p = \inf \mathbb{E}[d(X, Y)^p] \tag{11}$$

where $\mathbb{E}[Z]$ denotes the expected value of a random variable Z and the infimum is taken over all joint distributions of random variables X and Y with marginals μ and ν respectively. Wasserstein distance is a well-defined measure that preserves both the symmetry and triangular inequality [19].

Wasserstein distance has a closed-form solution for two multivariate Gaussian distributions. This allows us to employ 2-Wasserstein distance (abbreviated as W_2) as similarity measure between the latent Gaussian distributions $\mathcal{N}(\mu_i, \Sigma_i)$ and $\mathcal{N}(\mu_j, \Sigma_j)$:

$$dist = W_2(\mathcal{N}(\mu_i, \Sigma_i), \mathcal{N}(\mu_j, \Sigma_j)) \tag{12}$$

$$dist^2 = ||\mu_i - \mu_j||_2^2 + \text{Tr}(\Sigma_i + \Sigma_j - 2(\Sigma_i^{\frac{1}{2}} \Sigma_j \Sigma_i^{\frac{1}{2}})^{\frac{1}{2}}) \tag{13}$$

Since we focus on diagonal covariance matrices, thus $\Sigma_i \Sigma_j = \Sigma_j \Sigma_i$:

$$dist^2 = ||\mu_i - \mu_j||_2^2 + ||\Sigma_i^{\frac{1}{2}} - \Sigma_j^{\frac{1}{2}}||_F^2 \tag{14}$$

According to the above equation, the time complexity to compute the W_2 between two multivariate Gaussian distributions is linear with the embedding dimension d. Since the computation of W_2 no longer constitutes a computational challenge, we choose W_2 as a measure of distance.

We use a pairwise ranking formulation with respect to the Wasserstein distance W_2 to model PPIs:

$$W_2(\mathcal{N}(\mu_i, \Sigma_i), \mathcal{N}(\mu_j, \Sigma_j)) < W_2(\mathcal{N}(\mu_i, \Sigma_i), \mathcal{N}(\mu_k, \Sigma_k)) \tag{15}$$

where $\mathcal{N}(\mu_i, \Sigma_i)$ is the latent Gaussian distribution of sequence $\mathbf{s}_i$, and $(\mathbf{s}_i, \mathbf{s}_j)$ and $(\mathbf{s}_i, \mathbf{s}_k)$ represents positive and negative interaction respectively. Specifically, the idea of ranking formulation is to penalize ranking errors based on the the Wasserstein distances between the pairs. The smaller the Wasserstein distance, the larger the possibility of interactions.

Finally, we employ square-exponential loss [20] to enable learning from the known pairwise interactions. Mathematically, the energy between the protein pairs can be defined as $E_{ij} = W_2(\mathcal{N}(\mu_i, \Sigma_i), \mathcal{N}(\mu_j, \Sigma_j))$. Then, the loss function to be optimized is:

$$\mathcal{L} = \sum_i \sum_{(i,j) \in \mathbf{Y}^+} \sum_{(i,k) \in \mathbf{Y}^-} ({E_{ij}}^2 + \exp(-E_{ik})) \tag{16}$$

where $\mathbf{Y}^+$ represents set of positive interactions and $\mathbf{Y}^-$ represents set of negative interactions. In our setting, the objective function penalizes the pairwise errors by the energy of the pairs, such that the energy of positive interactions is lower than the energy of negative interactions. Furthermore, such ranking formulation also maximizes the difference in energy between positive pair (i, j) and negative pair (i, k). Equivalently, this will make the possibility of interactions between the interacting proteins larger than that of non-interacting proteins.

Finally, we can optimize the parameters Θ (i.e. weights and biases) of the model such that the loss $\mathcal{L}$ is minimized and the pairwise rankings are satisfied. Specifically, for each protein, the distance with interacting proteins should be smaller than with non-interacting proteins. We term this as the ranking approach since interacting proteins have smaller W_2 distance and are ranked higher than non-interacting proteins.

C. PPI prediction

The Wasserstein distances between the latent Gaussian distributions of protein sequences corresponds to the possibility of their interaction. However, predicting PPIs by only computing the Wasserstein distance fails to take into account the homodimers, the proteins with identical sequences [7]. The encoded Gaussian representations of these protein sequences will be the same and their Wasserstein distance will be 0 indicating they must interact.

To overcome this limitation, we define pairwise features for all protein pairs by the concatenation of the absolute element-wise differences of means and variances and the element-wise multiplications of the means of their Gaussian representations, $[|\mu_i - \mu_j|; |\Sigma_i - \Sigma_j|; \mu_i \odot \mu_j]$. This featurization is effective in modeling the symmetric relationship between proteins. To predict binary interactions, we train a binary classifier on these pairwise features to learn a decision boundary δ that separates interacting proteins from non-interacting pairs.

D. Efficient training

Siamese networks are suitable to train with contrastive loss mentioned in Eq. 16. However, Siamese training is inefficient when the amount of PPIs increases. In particular, the possible number of interactions for N proteins is $(N^2+N)/2$ (including self-interactions), which is computationally intensive for Siamese training. A mini-batch of m interactions in Siamese training may have multiple occurrences of the same proteins, leading the sequence to be feed-forwarded for each interaction. It is sufficient to feed-forward each sequence once to compute the loss. To address this problem, we encode the minibatch of n protein sequences and retrieve positive and negative interactions that involve them to compute the loss in the minibatch. With this setting, we are only required to feed-forward N protein sequences compared to $(N^2+N)/2$ pairs in the Siamese setting, which makes our method computationally efficient and scalable to a large number of interactions.

IV. Experiments

We evaluate our method on two real-world datasets: yeast and human proteins to predict their interactions. We use the area under the ROC curve (AUROC) and the average precision (AP) scores as the evaluation metrics. With these evaluation metrics, we expect the positive protein pair to have higher interaction probability compared to negative protein pair.

A. Experimental Setup

*1) **Datasets**:* The datasets for protein sequences of yeast and human proteins are from the EMBL-EBI Reference Proteome [21]. The information about the subcellular localization of proteins is extracted from UniProt database [22]. The evolutional protein profiles for yeast and human protein sequences are collected from Rost Lab [23]. We evaluate our proposed model with two types of protein features: (a) amino acid sequences and (b) evolutionary protein profiles constructed from these sequences.

To evaluate the performance of deep learning models, the interaction datasets are downloaded from the up-to-date BioGRID interaction database (Release 3.5.169) [24]. The BioGRID database provides a large number of PPIs allowing us to evaluate the scalability of different approaches as well. Only the interactions that correspond to the physical binding between the protein pairs (say $\mathbf{Y}^+$) are considered since these interactions are supported by experimental evidence. The negative samples $\mathbf{Y}^-$ are generated by randomly sampling from all protein sequence pairs ($(\mathbf{s}_i, \mathbf{s}_j) \notin \mathbf{Y}^+$), that are not yet confirmed by experimental evidence. Furthermore, these negative interactions are filtered based on their subcellular localization, assuming that proteins in the different locations are unlikely to interact although some proteins do translocate [23].

Also, we perform the cluster analysis with the CD-HIT [25] to cluster protein sequences based on a certain similarity threshold that represents sequence identity. We remove the interactions such that no two proteins have a pairwise sequence identity greater than 10%. Table I shows the statistics of the interaction datasets.

TABLE I
Statistics of interaction datasets

Data	No. of proteins	No. of positive pairs	No. of negative pairs
Yeast	3,651	50,344	50,376
Human	7,028	73,624	73,628

*2) **Hyperparameters and training details**:* For both datasets, we train a sequence encoder with the same configuration. The best hyperparameters of our model are selected based on validation performance. The maximum length of the input sequence to the encoder is 1024 for efficient training. In the datasets, 91.2% of yeast sequences and 86% of human sequences are shorter than 1024 residues.

The encoder consists of a BiGRU layer with 16 hidden units each, to map a protein sequence to a sequence of 32-dimensional representation, one per amino acid. Then, this representation is encoded to a latent Gaussian distribution with dimension $\mathbf{d}$ = 256 ($\mathbf{d} = 2 \times d$ = 128 for the mean and 128 for the variance of the Gaussian).

All the weight matrices of the encoder layer are initialized using Xavier initialization [26]. The model is trained on a single NVIDIA GeForce RTX 2080 Ti GPU for all experiments using Adam optimizer [27] with learning rate 0.003 and other default parameters provided by PyTorch [28]. Our unique approach of encoding unique sequences allows us to train the model efficiently even with large batch sizes. We empirically find that our model converges in a small number of iterations ($\leq$ 50 for all shown experiments).

B. Results on PPI prediction

We compare our method against the state-of-the-art deep learning methods on the up-to-date BioGRID interaction datasets. We split the interactions into training, validation, and test sets (0.6:0.2:0.2). All the models are trained on the same training set and the best set of hyperparameters are selected based on their performances on the validation set. Finally, the models are evaluated on independent test sets. Table II reports the mean AUROC and AP and their standard errors on five independent runs. We perform a two-tailed Welch's t-test and Benjamini-Hochberg procedure to adjust p-value and find that the improvement over PIPR, the state-of-the-art method is statistically significant.

With the ranking approach, we expect our model to rank positive interactions higher than negative interactions, i.e. the

TABLE II

AVERAGE AUROC AND AP SCORES (WITH STANDARD DEVIATION) AVERAGED OVER FIVE INDEPENDENT RUNS FOR PPI PREDICTION. * REPRESENTS STATISTICALLY SIGNIFICANT DIFFERENCES WITH PIPR (P-VALUE < 0.005).

Method		Data	Yeast		Human	
			AUROC	AP	AUROC	AP
DPPI [7]		Profiles	0.891±0.004	0.857±0.007	0.870±0.004	0.835±0.005
PIPR [8]		sequences	0.909±0.003	0.912±0.004	0.878±0.002	0.882±0.003
Our method (sparsemax)	Ranking	Profiles	0.882±0.003	0.888±0.002	0.884±0.003	0.893±0.004
		Sequences	0.901±0.002	0.904±0.002	0.881±0.002	0.889±0.001
	Random Forest	Profiles	0.908±0.002	0.913±0.003	**0.891 ±0.005***	**0.896±0.005***
		Sequences	**0.924±0.002***	**0.925±0.001***	0.887±0.002	0.894±0.001
Our method (fusedmax)	Ranking	Profiles	0.882±0.006	0.885±0.006	0.873±0.09	0.881±0.01
		Sequences	0.898±0.001	0.900±0.002	0.874±0.002	0.883±0.001
	Random Forest	Profiles	0.906±0.004	0.912±0.005	0.872±0.015	0.877±0.015
		Sequences	0.919±0.003	0.921±0.002	0.881±0.002	0.886±0.001

probability of interactions between interacting protein pairs is greater than that of non-interacting protein pairs. Table II demonstrates that our model ranks positive interactions higher than negative interactions. The ranking based model with sparsemax regularization achieves comparable performance with PIPR. Furthermore, the random forest classifier trained to account for homodimeric interactions improve the model's performance on both datasets. The best parameters for random forest classifier are selected via grid search.

Furthermore, we evaluate our proposed method on evolutionary protein profiles. Protein profiles constructed from protein sequences capture the correlation between different proteins as well as between different parts of the sequences [29]. Our model trained with protein profiles outperforms DPPI, the state-of-the-art deep architecture that uses profiles as the input to their deep Siamese model. Table II shows that our model with sequences achieves comparable or better performance compared to profiles across both datasets. This demonstrates that our model is capable of extracting useful information about interactions from protein sequences and alleviates the expensive process of profile construction from sequences.

C. Ablation study of framework components

We next evaluate individual model components on the PPI prediction task with yeast dataset.

1) ***Gaussian representations outperform point representations****:* We first explore whether the Gaussian representation of sequences improves the performance of the model over point representation. We encode intermediate representation $\mathbf{v}$ of sequence $\mathbf{s}$ to point representation $\mathbf{z} = \mathbf{W_z v} + \mathbf{b_z}$ instead of Gaussian representation (as in Eq. 9 and 10) and define L2 norm as the similarity between the point representations of sequences $\mathbf{s}_i$ and $\mathbf{s}_j$ instead of Wasserstein distance (Eq. 12):

$$dist^2 = ||\mathbf{z}_i - \mathbf{z}_j||_2^2 \tag{17}$$

where $\mathbf{z}$ represents point representation of sequences s. Table III shows that Gaussian representations are better predictors of PPIs compared to point representations.

2) **Sparse gating mechanism improves performance***:* We further demonstrate the importance of the proposed sparse gating mechanism by comparing the performance of the sequence encoder trained with and without various gating mechanism. We train models with different settings of the gate mechanism. Table III demonstrates that the sparse gating mechanism provides a significant improvement over no gate mechanism and softmax in PPI prediction. This indicates that sparse regularization helps the model to selectively activate the important stretches of amino acids that are important to model and predict PPIs.

TABLE III

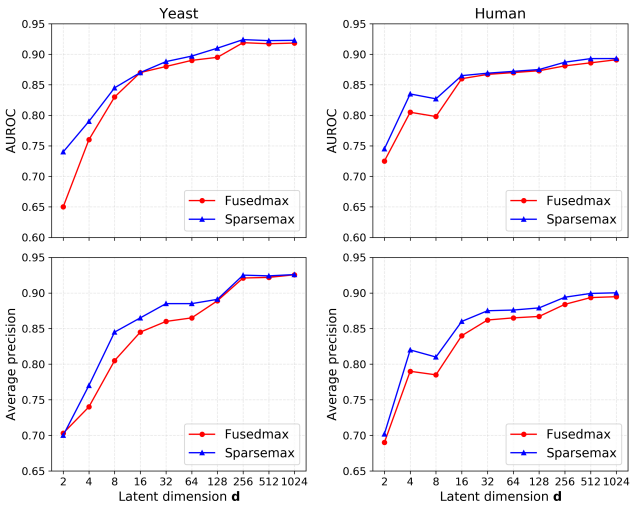
STUDY OF INDIVIDUAL MODEL COMPONENTS ON YEAST DATASET. THE MODEL TRAINED WITHOUT ANY GATE MECHANISM (NO GATING), WITH DIFFERENT REPRESENTATIONS (POINT VS GAUSSIAN) COUPLED WITH DIFFERENT REGULARIZATION AND RANDOM FOREST (RF) CLASSIFIER.

Model configuration		AUROC	AP
No gating		0.880±0.001	0.875±0.003
Point + RF	Softmax	0.881±0.001	0.877±0.001
	Fusedmax	0.909±0.001	0.912±0.002
	Sparsemax	0.913±0.001	0.916±0.002
Gaussian + RF	Softmax	0.882±0.001	0.879±0.002
	Fusedmax	0.919±0.003	0.921±0.001
	Sparsemax	**0.924±0.002**	**0.925±0.001**

3) **Dimension of Gaussian distribution is important***:* Finally, we investigate how the dimensionality of the latent Gaussian distributions can affect the model's performance. Figure 2 shows the plot of the AUROC and the AUPR scores of our method across the two organisms. When the dimension $\mathbf{d}(= 2 \times d)$ of the Gaussian distribution increases from 2 to 256, the performance also increases. When $\mathbf{d} \geq 128$, two regularization strategies result in similar performance. Moreover, the performance also remains stable when $\mathbf{d} \geq 256$.

D. Training time per epoch

Here, we compare the training time of our method and PIPR, the state-of-the-art deep learning model [8]. For fair comparison, we train both models in the same machine on the same dataset and compare only the average training time per epoch in Figure 3. For this experiment, we randomly sample 8k, 16k, 24k, 32k, 40k, 48k, 56k, 64k, 72k, 80k, and 88k training interactions.



(a) (b)

Fig. 2. Effect of dimension of latent Gaussian distributions on the model's performance with different sparse gating mechanism for (a) yeast (b) human.

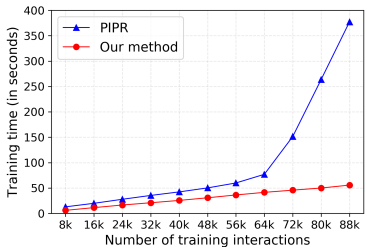


Fig. 3. Average training time per epoch. Our method is much faster than PIPR. Encoding the unique set of protein sequences to their latent Gaussian distribution and optimizing the loss based on their interactions makes our model efficient.

Figure 3 shows that our method is efficient in comparison to PIPR. PIPR uses a pairwise training process that requires a higher number of matrix multiplications for each interaction. On the other hand, given the large batch of interactions, our model finds the unique set of protein sequences involved in these interactions and encodes them, which significantly reduces the number of matrix multiplication. Once we have the embeddings for these sequences, we can compute the loss based on their respective interactions. As discussed in Section III-D, for instance, if there are 1000 interactions in a batch with 100 unique proteins, our model encodes only 100 protein sequences instead of 1000 protein pairs as in PIPR. Note that this approach allows our model to train on a large batch of interactions, and thus takes less time to train. This demonstrates that our method scales with the number of interactions and is significantly faster than PIPR.

E. Interpretability

Since our proposed model selectively activates the part of a given sequence, it is important to evaluate whether the selected parts are important. To explore this, we perform the quantitative evaluation on how the amino acids selected by the sparse gating mechanism in our proposed method align with the motifs from the Pfam motif library [30] from GenomeNet [31]. The gate vector $\mathbf{g}$ for a sequence $\mathbf{s}$ helps us interpret how much contribution an amino acid on that position signaled by GRU hidden state makes. Since our model computes gate values between 0 and 1 for each amino acid, we consider amino acids with $g_l > 0$ to be active i.e. used by the model for the representation of proteins. Table IV shows the average percentage of amino acids selected by the sparse gating mechanism and their alignment with motifs having biological significance. For instance, for yeast dataset, only 19.24% amino acids (on average) are selected with fusedmax and 59.05% of these selected amino acids aligns with the motif. This illustrates that the amino acids in the sequence selectively activated by our model to learn protein representation align with biologically interpretable motifs.

TABLE IV
COMPARISON OF SELECTED AMINO ACIDS WITH THE MOTIFS FROM PFAM MOTIF DATABASE

Dataset	Gating	Selected amino acids (%)	Alignment with motifs (%)
Yeast	Sparsemax	8.06	49.96
	Fusedmax	19.24	59.05
Human	Sparsemax	9.15	48.33
	Fusedmax	23.33	65.63

In addition, we visualize the activated amino acids and the motifs from Pfam motif library [30] from GenomeNet [31] in Figure 4. For this experiment, we select three proteins: LSM8, SMD2, and RPC11 from the yeast dataset with motifs in different parts of the sequences. Figure 2 and Table IV shows that the model trained with fusedmax achieves similar performance to sparsemax when $\mathbf{d} \geq 256$ but gains better alignment with the motifs from Pfam database. So, we train the model with fusedmax and obtain the gate values for each amino acid in these sequences. Red lines in each sub-figure of Figure 4 are the important regions identified by our model.

In particular, LSM8 with the sequence of length 109 has two motifs: PF01423, LSM domain at the position from 4 to 65, and PF14807, adaptin AP4 complex epsilon appendage platform at the position from 9 to 65 shown in Figure 4a. The learned gate value corresponds to the subset of amino acids from position 1 to 65 and aligns with motifs. Similarly, the selected parts of sequences for SMD2 and RPC11 aligns with their motifs even though the motif lies in different parts of sequences. The quantitative and qualitative evaluation of gate vectors shows that our model successfully identifies important amino acids in the sequence for PPI prediction.

V. CONCLUSION

We present a novel deep neural network to model and predict PPIs from variable-length protein sequences. Our proposed framework adopts a recurrent neural network to capture contextualized and sequential information from amino acid

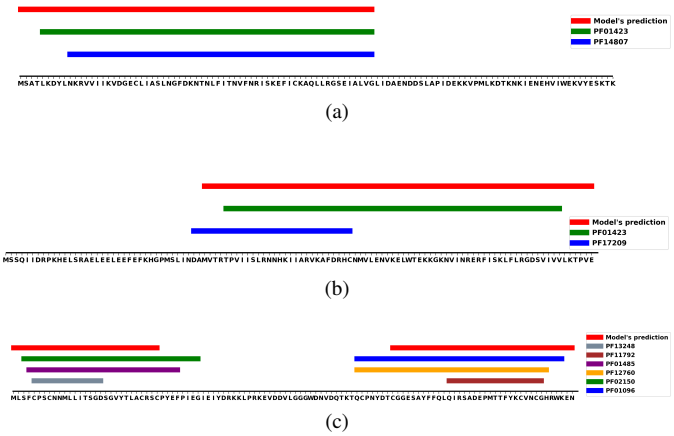


Fig. 4. Visualization of motifs and the activated amino acids for three proteins (a) LSM8 (b) SMD2 and (c) RPC11. The legend includes the motif identifier from Pfam database. The alphabet on the x-axis is the amino acid at the respective position. The line segments corresponds to the part of sequence that is model's prediction (red) or motifs (other color). The selected subset of amino acid sequences aligns with motifs from Pfam motif library.

sequences. By incorporating a structured and sparse gating mechanism into the sequence encoder, our model successfully selects the important residues on the sequence to learn the sequence representation. Furthermore, our novel approach of encoding sequences to their representation makes the model efficient and scalable to a large number of interactions. Extensive experimental evaluations on various up-to-date datasets show its promising performance on binary PPI prediction task. Various case studies demonstrate the ability of our model to provide biological insights to interpret the predictions.

Acknowledgment

This work was supported by the National Science Foundation [NSF-1062422 to A.H.], [NSF-1850492 to R.L.] and the National Institutes of Health [GM116102 to F.C.]

References

[1] S. Martin, D. Roe, and J.-L. Faulon, "Predicting protein–protein interactions using signature products," *Bioinformatics*, vol. 21, no. 2, pp. 218–226, 2004.

[2] C. B. Anfinsen, "Principles that govern the folding of protein chains," *Science*, vol. 181, no. 4096, pp. 223–230, 1973.

[3] J. Shen, J. Zhang, X. Luo, W. Zhu, K. Yu, K. Chen, Y. Li, and H. Jiang, "Predicting protein–protein interactions based only on sequences information," *Proceedings of the National Academy of Sciences*, vol. 104, no. 11, pp. 4337–4341, 2007.

[4] L. Yang, J.-F. Xia, and J. Gui, "Prediction of protein-protein interactions from protein sequence using local descriptors," *Protein and Peptide Letters*, vol. 17, no. 9, pp. 1085–1090, 2010.

[5] Y. Guo, L. Yu, Z. Wen, and M. Li, "Using support vector machine combined with auto covariance to predict protein–protein interactions from protein sequences," *Nucleic acids research*, vol. 36, no. 9, pp. 3025–3030, 2008.

[6] Z.-H. You, L. Zhu, C.-H. Zheng, H.-J. Yu, S.-P. Deng, and Z. Ji, "Prediction of protein-protein interactions from amino acid sequences using a novel multi-scale continuous and discontinuous feature set," in *BMC bioinformatics*, vol. 15. BioMed Central, 2014, p. S9.

[7] S. Hashemifar, B. Neyshabur, A. A. Khan, and J. Xu, "Predicting protein-protein interactions through sequence-based deep learning," *Bioinformatics*, vol. 34, no. 17, pp. i802–i810, 09 2018.

[8] M. Chen, C. Ju, G. Zhou, X. Chen, T. Zhang, K.-W. Chang, C. Zaniolo, and W. Wang, "Multifaceted protein-protein interaction prediction based on siamese residual rcnn," *Bioinformatics*, vol. 35, no. 14, pp. i305–i314, 07 2019.

[9] Y.-C. Hsu and Z. Kira, "Neural network-based clustering using pairwise constraints," *Proc. of International Conference on Learning Representations (ICLR) Workshop Track*, 2016.

[10] S. Tonddast-Navaei and J. Skolnick, "Are protein-protein interfaces special regions on a protein's surface?" *The Journal of chemical physics*, vol. 143, no. 24, p. 12B631_1, 2015.

[11] T. Hamp and B. Rost, "Evolutionary profiles improve protein–protein interaction prediction from sequence," *Bioinformatics*, vol. 31, no. 12, pp. 1945–1950, 2015.

[12] S. F. Altschul, T. L. Madden, A. A. Schäffer, J. Zhang, Z. Zhang, W. Miller, and D. J. Lipman, "Gapped blast and psi-blast: a new generation of protein database search programs," *Nucleic acids research*, vol. 25, no. 17, pp. 3389–3402, 1997.

[13] A. Martins and R. Astudillo, "From softmax to sparsemax: A sparse model of attention and multi-label classification," in *Proc. of International Conference on Machine Learning*, 2016, pp. 1614–1623.

[14] V. Niculae and M. Blondel, "A regularized framework for sparse and structured neural attention," in *Proc. of Advances in Neural Information Processing Systems*, 2017, pp. 3338–3348.

[15] A. Bojchevski and S. Günnemann, "Deep gaussian embedding of graphs: Unsupervised inductive learning via ranking," in *Proc. of International Conference on Learning Representations*, 2018.

[16] D. Zhu, P. Cui, D. Wang, and W. Zhu, "Deep variational network embedding in wasserstein space," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2018, pp. 2827–2836.

[17] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (elus)," in *Proc. of International Conference on Learning Representations*, 2015.

[18] W. Liu and J. C. Rajapakse, "Fusing gene expressions and transitive protein-protein interactions for inference of gene regulatory networks," *BMC systems biology*, vol. 13, no. 2, p. 37, 2019.

[19] C. R. Givens, R. M. Shortt *et al.*, "A class of wasserstein metrics for probability distributions." *The Michigan Mathematical Journal*, vol. 31, no. 2, pp. 231–240, 1984.

[20] Y. LeCun, S. Chopra, R. Hadsell, F. J. Huang, and et al., "A tutorial on energy-based learning," *Predicting Structured Data*, vol. 1, p. 0, 2006.

[21] C. Dessimoz, T. Gabaldón, D. S. Roos, E. L. Sonnhammer, J. Herrero, and Q. for Orthologs Consortium, "Toward community standards in the quest for orthologs," *Bioinformatics*, vol. 28, no. 6, pp. 900–904, 2012.

[22] U. Consortium, "Uniprot: a worldwide hub of protein knowledge," *Nucleic acids research*, vol. 47, no. D1, pp. D506–D515, 2018.

[23] T. Hamp and B. Rost, "More challenges for machine-learning protein interactions," *Bioinformatics*, vol. 31, no. 10, pp. 1521–1525, 2015.

[24] R. Oughtred, C. Stark, B.-J. Breitkreutz, J. Rust, L. Boucher, C. Chang, N. Kolas, L. O'Donnell, G. Leung, R. McAdam *et al.*, "The biogrid interaction database: 2019 update," *Nucleic acids research*, vol. 47, no. D1, pp. D529–D541, 2018.

[25] W. Li and A. Godzik, "Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences," *Bioinformatics*, vol. 22, no. 13, pp. 1658–1659, 2006.

[26] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. of the thirteenth international conference on artificial intelligence and statistics*, 2010, pp. 249–256.

[27] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. of International Conference on Learning Representations*, 2015.

[28] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," in *NIPS-W*, 2017.

[29] Q. Cong, I. Anishchenko, S. Ovchinnikov, and D. Baker, "Protein interaction networks revealed by proteome coevolution," *Science*, vol. 365, no. 6449, pp. 185–189, 2019.

[30] R. D. Finn, A. Bateman, J. Clements, P. Coggill, R. Y. Eberhardt, S. R. Eddy, A. Heger, K. Hetherington, L. Holm, J. Mistry *et al.*, "Pfam: the protein families database," *Nucleic acids research*, vol. 42, no. D1, pp. D222–D230, 2014.

[31] M. Kanehisa, "Linking databases and organisms: Genomenet resources in japan," *Trends in biochemical sciences*, vol. 22, no. 11, pp. 442–444, 1997.