Interpretable Deep Text Classifiers in Protein Subcellular Localization

Hao Wang
School of Electrical and Computer Engineering
Georgia Institute of Technology
hwang794@gatech.edu

Sahil Arora
College of Computing
Georgia Institute of Technology
sarora@gatech.edu

Haard Shah
College of Computing
Georgia Institute of Technology
haard@gatech.edu

1 Introduction and Problem Overview

Protein Subcellular Localization is the task of predicting where a protein is in the cell (mitochondria, nucleus, etc) given its composition. Machine learning algorithms have been used in PSL since the 2000s with the advent of cheaper high throughput sequencing and the rise of proteomics and big data. Older models have used two tiered support vector machines on crafted embeddings for meaningful sequences, which while effective were much harder to interpret and required prior knowledge [1].

In the past 5 years deep learning methods have found a niche in subcellular localization leaving older algorithms in the dust. Without prior biological knowledge and just sequence data, DeepLoc, a convolutional long short term memory network, has accurately predicted subcellular compartments better or at par with the state of the art at the time [2]. This paper focuses solely on this approach, where the model is only given the sequence of amino acids that make up the protein. PSL is a text classification problem as a protein is represented as a variable length sequence of 25 amino acids. We use padding and a one hot encoding to make our input a 1000x25 matrix to be fed to our neural network, which will output a single label from one of the 10 locations, which are our classes.

Multiple naive deep learning algorithms are succeeding in this task but there is little reliable biological interpretability [3]. With interpretability we can understand why an algorithm classified the way it did which allows for novel insights and more trustworthy results. So our goal is to make DeepLoc more interpretable so that its predictions are more reliable, and we will do this by examining the effect of attention on model performance and relevant subsequences.

Research using attention weights as attribution scores to each input feature for interpretability is performed in [4][5][6]. In our research, we reproduce their experiments in the new domain where inputs are amino acid sequence as opposed to natural language. Once we use these methods to validate whether attention is useful for the PSL task, we extend the research by introducing a new metric that can further provide insights by assessing whether the attribution scores extracted from the model are biologically consistent.

This work is not trivial as PSL is an important task in bioinformatics. Understanding where a protein is located in a cell can reveal its structure, function, and how it interacts with other proteins and organelles in the cell. This information can be applied to drug design and therapeutic target discovery. For example, subcellular localization models have been been used to improve understanding of degeneration mechanisms in Alzheimer's [7]. Improving interpretability for models in this niche has broader applications. Providing an interpretable framework for deep learning models that can glean domain insight can be used in any domain specific classification task for scientific research.

1.1 Contributions

The outline of our report is as follows: In section 2 we summarize the related literature in the biology and NLP. The aim of our research is to make PSL more interpretable in order to make the systems more trustworthy and gain more insights about the data by visualizing what the models are learning. Section 3 describes that dataset, models, and two methods we apply to analyze interpretability of attention in those PSL models.

First, we describe various approaches to investigate the interpretability of the PSL task. We run various tests motivated from prior literature to diagnose the predictive power that attention adds to our models. We follow the setup in [6] and [4] to assess whether attention can provide any additional predictive power. This allows us to assess whether attention models for PSL task behave similarly to the classification task where input is natural language (NL). Prior interpretability research mainly works with NL data, therefore, extending the attention interpretability tasks in the section *Model Driven Approach* will be our first contribution (3.1).

Second, we extract interpretable information from the model in the form of importance scores, also known to as attribution, for individual input features in contribution to model's output. To quantitatively assess the validity of attribution values, we introduce a new method to further assess the interpretability of attention in PSL models. In section 3.2, we describe the *hit rate* metric, which is used to quantify the biological consistency of the interpretation derived by inspecting attention weights. All our code can be found here: https://github.com/arolihas/interpretable_deepPSL.git.

2 Related Work

Work on interpretability in deep PSL has been stagnant. There have been some attempts to create more interpretable models before the deep learning era but they take required heavy manual annotation and a reliance on gene ontology with carefully motivated feature selection [8]. For deeper architectures, visualizing convolutional filters has been popular as they can be analyzed as higher order protein structures. One precursor to DeepLoc has used amino acid weighting in these convolutional filters as certain pairings can reveal biochemical information [3]. However, this does not explain how a classification is actually made.

The work on visualizing the attention mechanism in this task is scant. In the previously mentioned papers, weights in the attention encoder for each instance were dimensionally reduced with t-SNE and plotted with the class labels. Similarities in the clusters were used to justify that the model was interpretable. However, interpreting t-SNE visualizations is a finicky and problematic task, where cluster sizes, shapes, and distances do not necessarily convey useful information [9]. We can improve on this with more effective visualization methods that do not affect the integrity of the data.

Attention mechanisms play an important role within recurrent neural network (RNN) models. Many researchers have found that attention may not be as interpretable as we expected. In [5], they observed in many ways that higher attention weights don't correlate with greater impact on model predictions. And while attention noisily predicts input components' overall importance to a model, it is by no means a fail-safe indicator. In [10], they found that the learned attention weights are frequently uncorrelated with gradient-based measures of feature importance, and one can identify very different attention distributions that nonetheless yield equivalent predictions. However, in [6] and [4], they designed alternative tests to find out whether attention can be used as explanation. They used these diagnostics to show that adversarial distributions are not as good the original attention distribution, indicating that prior work does not disprove the usefulness of attention mechanisms for explainability. Furthermore, [4] investigates the usefulness of attention for more diverse models.

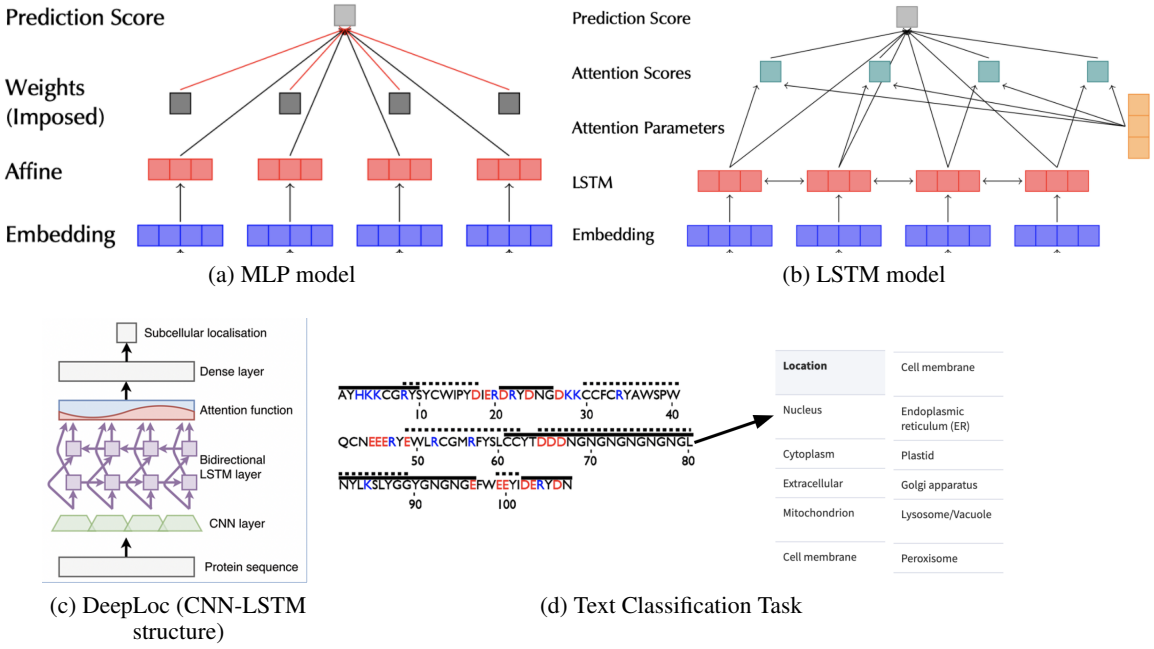
3 Methods

3.1 Dataset

The DeepLoc dataset, which was curated from the proteomic database UniProt [11] is used for training and testing. Designed for training DeepLoc, this dataset has over 13000 eukaryotic protein sequences belonging to 10 different possible subcellular components [3]. Since all of the architectures are similar to DeepLoc, we are not concerned about any possible bias towards DeepLoc in the design

of the dataset. The Swiss-Prot knowledge base is a manually annotated resource from UniProt that contains the subcellular localizations of 195 million referenced proteins in biological research [12]. This knowledge base was used with a fast API [13] to search for subsequence matches when calculating the *hit rate*.

3.2 Models



(a) MLP model

(b) LSTM model

(c) DeepLoc (CNN-LSTM structure)

(d) Text Classification Task

Figure 1: Model Architectures and Task

In this project, we implement various neural network models inspired by the DeepLoc architecture. All the models can be seen in 1. The BiLSTM and CNN-BiLSTM models are similar to the DeepLoc architecture. BiLSTM without CNN is included to simplify interpretability. The only difference between the two is that CNN-BiLSTM has convolutional layers between embedding and BiLSTM layers. In our experiments, we introduce two sizes of convolutional layers. The smaller model contains two convolutional kernels with sizes 3 and 5, whereas the larger model contains four kernels with sizes 3, 5, 7, and 11. The outputs of the different kernels are concatenated before feeding them into the BiLSTM layers. Both these models are tested with and without attention. Finally, the outputs of attention or BiLSTM layer are fed into a dense layer to output raw scores, which are then converted into probability distribution using a softmax function. The final probabilities describe the model's prediction for each of the 10 cell locations. Furthermore, the MLP model is used for tests in 3.3. Here we simply replace the BiLSTM layer with simple feedforward layers to remove the contextual information captured with BiLSTM. The reasons for this are motivated further in 3.3.

We use F1 macro and F1 micro to evaluate models' performance on test data. The F1 score of single class is defined as 1. F1 macro is computed by averaging all F1 score of single class while F1 micro is generated by calculating Precision and Recall by summing all the TPs and Type Errors instead of calculating for each label. If our model have a good performance on large class, then the F1 macro score should be higher than the F1 micro score.

$$F_1 = \frac{2}{\text{recall}^{-1} + \text{precision}^{-1}} = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} = \frac{\text{tp}}{\text{tp} + \frac{1}{2}(\text{fp} + \text{fn})} \tag{1}$$

3.3 Model Driven Approach

Our experiments in the first part are based on [6]. In the original paper, they propose four tests and we implement two of them. We introduce a **uniform** model variant, identical to the BiLSTM model setup except that the attention distribution is frozen to uniform weights over the hidden states. Similarly, as suggested by [4], we obtain other variants of the models where attention weights are **permuted** and set to **random** values. These experiments are run on the BiLSTM and CNN-BiLSTM models. Once the attention weights are modified according to the given test, we perform inference and compare its performance with the original BiLSTM or CNN-BiLSTM model with unaltered attention weights. If altering the attention weights significantly impairs model's performance, then attention provides meaningful information to assist with classification and can be used for interpretation.

Next, we replicate another test from [6] where we create a diagnostic model that we call the MLP model 1a. Token-level affine hidden layer with tanh activation is used here to encode the inputs. Then we use different attention distributions to guide the simple model during the training and testing. The guiding weights we impose are as follows: **Uniform** (where all MLP outputs are weighted equally), **Trained-MLP** (where we the model learns its own parameters, and **Pre-set** (where we directly take learned attention weights from BiLSTM model and freeze them during training). Here, we introduced a post-hoc training protocol of a non-contextual model guided by pre-set weight distributions. The idea is to examine the prediction power of attention distributions in a 'clean' setting, where the trained parts of the model have no access to neighboring tokens of instance. If the pre-trained scores from the BiLSTM model can have a good performance, we take this to mean that they are helpful and attention weights can provide explanation and contextual information.

One fundamental difference between our task and the task in [6] is that in the original paper they focus on binary classification and PSL is a multi-class classification problem. The conclusion in the original paper may not apply to our data and therefore we recreate the experiments here.

3.4 Subsequence Driven Approach

In this section, we introduce techniques to retrieve attribution scores from a trained model and propose a new metric to assess their validity.

3.4.1 Attribution

Attribution is a numerical value inferred from the model about how much an input contributes to the model's output. We obtain these values using two primary methods: attention and integrated gradients [14][15].

In the attention method, we directly take the αs computed by the model using weights that are learned with backpropagation. The additive attention used by most text classification neural networks is shown in 2.

$$\mathbf{u}_i = \tanh\left(\mathbf{W}\mathbf{h}_i + \mathbf{b}\right); \quad \alpha_i = \frac{\exp\left(\mathbf{u}_i^T \mathbf{c}\right)}{\sum_j \left(\mathbf{u}_j^T \mathbf{c}\right)} \tag{2}$$

Here $\mathbf{W}^{dxd'}, b, c \in^{d'}$ are the parameters of the model. The attention weightings, α can be multiplied to the hidden representations from previous layers and then fed to a dense layer followed by a softmax to obtain predictions. The attention weightings can be used to attribute importance scores to individual features of the input.

The attention layer can be inserted in most neural network architectures to give the model capability to select which input features are more important and which are less important. It usually follows some sort of encoding layer that transforms inputs into a compact fixed vector representation. The intuition behind attention mechanism is to replicate how humans pay attention to different regions an image or words in a sentence. Approaches like Hierarchical Attention Network, utilize multiple levels of attention [16]. They motivate their use for natural language inputs by observing that "different words and sentences in a document are differently informative". In a similar sense, different subsequences of amino acids in a protein sequence may be differently informative about its location in the cell. Since, models that incorporate attention layers learn this information automatically, we use this information to extract meaningful subsequences that are assumed to have the highest impact on the model's prediction decision.

Integrated gradients is another method that can enable us to attribute similar importance scores for individual input features of any machine learning model [15]. Integrated gradients is an axiomatic interpretability algorithm because it derives the importance scores by directly looking at the partial derivatives of a model's output with respective to the individual input feature. The authors of [15] notice that directly calculating gradients with respect to inputs results in noisy attribution scores that don't faithfully capture input feature attribution if the model function is flat in the vicinity of the input. The authors combat this by calculating the integral of gradients of the model's output with respect to the inputs along a straight line path from a given baseline to input. In practice, they obtain multiple inputs along the straight-line path from a baseline input to the original input and average the gradients. For images, the baseline input could a black image with all 0 inputs and we slowly raise brightness of the image until we get our original image. For text, the baseline could be a zero embedding vector. The attribution score of input feature i for model F can be obtained using integrated gradients as follows:

$$\text{IntegratedGrads}_i(x) ::= (x_i - d'_i) \times \int_{\alpha=0}^{1} \frac{\partial F(x' + \alpha \times (x - x'))}{\partial x_i} d\alpha \tag{3}$$

For details on the approach, we encourage the reader to refer the original paper [15]. Here integrated gradients method is treated as a baseline approach to compare the quality of attribution scores obtained using attention weightings. Next, we propose a novel metric to quantitatively assess the reliability of most meaningful subsequences as provided by attribution scores.

3.4.2 Hit Rate

Once we have attribution scores for each of the features in our input, we can use the subsequences with the highest attribution scores to better interpret the model. We first extract subsequences of length 5 that have an attribution score 2 standard deviations above the mean attribution score for a sequence. Among these subsequences we merge any overlapping ones to leave us with varied length subsequences. These mimic the biological structure of a peptide, which is a motif made up of 5-15 amino acids that commonly make up proteins.

Specific peptides tend to have similar properties, and so it follows that they could reside in the same subcellular location. To test this we search for the subsequences in SwissProt, a part of the UniProt database that consists only of verified and peer-reviewed protein entries with annotations. Our search returns every entry that contains the relevant subsequence, as well as the position it occurs.

From these entries, we find the total number of entries that have the same subcellular location as the original protein sequence the subsequence was extracted from. Because a peptide's function and properties are dependent on where it is located in the sequence, we weight this ratio by how close the position of the subsequence in the entry is to its position in the original protein. This weighted ratio is the *hit rate* of the subsequence, and to get the hit rate of the original sequence we average the hit rates of all of the extracted subsequences. Below is the formula:

Given protein P with location L_P that contains N extracted subsequences s with starting positions s_{start} and external protein entries E_{s_i} with locations $L_{E_{s_i}}$ that contain subsequence s_i starting at position $E_{s_{\text{start},i}}$

$$\text{hitrate}(s) = \frac{\text{Number of entries where } L_{E_{s_i}} == L_P}{\text{Number of total entries containing subsequence } s} \times \beta(E_{s_{\text{start},i}}, s_{\text{start},i}) \tag{4}$$

where the positional weighting β is defined as

$$\beta(i,j) = \begin{cases} 1 & i = j \\ \frac{1}{i-j} & i \neq j \end{cases} \tag{5}$$

In *model driven approach*, 3.1, we perform tests motivated by interpretability literature in NL domain to determine whether attribution obtained using attention weights can be used to provide explanations of the model's decision. The *hit rate* is a domain specific metric that can be used for PSL models to directly assess whether the explanations provided by the model are valid. A higher *hit rate* suggests that the attributions scores obtained are more reliable.

4 Results

4.1 Diagnostic Attention

Experiment I: BiLSTM	F1 micro	F1 macro
Attention (baseline)	**0.673**	0.568
Uniform	0.666	0.564
Experiment II: MLP	F1 micro	F1 macro
Uniform	0.445	0.317
Trained	**0.468**	0.334
Pre-set	0.452	0.317
Experiment III DeepLoc:	F1 micro	F1 macro
3,5 size conv filters	0.667	0.556
3,5 size conv filters with attention	0.650	0.541
3,5,7,11 size conv filters	0.666	0.551
3,5,7,11 size conv filters with attention	**0.670**	0.560

Table 1: Classification Accuracy of Various Models

The results above do not appear promising. In experiment I, if attention is a necessary component for good performance, then we expect a huge drop between baseline and uniform attention model. But we see that the attention layer appears to provide little improvement on the prediction power.

In experiment II, the first result is that using pre-set LSTM attention weights and trained attention weights will improve the model's F1 score slightly. However, the F1 macro of pre-set is equal to F1 macro of uniform distribution which suggests improvement is a reflection of class imbalance. Compared to experiment I, we see that uniform attention BiLSTM outperform the MLP trained with pre-set attention weights, suggesting that the attention mechanism is less important than BiLSTM architecture for our dataset. These findings are different from expected results in [6].

In experiment III, the attention does not show much benefit. With the smaller filters, the attention weights leads to a worse performance compare with the baseline, while in the larger CNN-LSTM the attention weights have little influence on the performance.

4.2 Hit rate

Integrated Gradients	LSTM	LSTM-Att	LSTM-uniform Att
Correct Prediction	0.462	**0.500**	**0.460**
Wrong Prediction, predicted label	0.083	0.104	0.136
Wrong Prediction, true label	**0.476**	**0.500**	0.446
Integrated Gradients	big CNN-LSTM	big CNN-LSTM-Att	
Correct Prediction	**0.498**	**0.461**	
Wrong Prediction, predicted label	0.153	0.128	
Wrong Prediction, true label	0.313	0.377	

Table 2: LSTM and CNN-LSTM Hit Rates from Integrated Gradients Subsequences

For a baseline, we provide hit rate results above for correct and incorrect classifications from subsequences derived by integrated gradients. The results show that there is some grounding in the metric, as correct predictions for true labels are universally higher than incorrect predictions. When hit rate results are applied to the attention models we can see the benefit of attention more clearly on the table on the next page. Hit rates from attention-derived subsequences are better than those derived from integrated gradients across the board. And it is with this metric that we can observe the utility of attention, as those subsequences have higher hit rates than permuted or randomized attention weights.

4.3 Key Findings

According to preliminary findings looking solely at classification accuracy, attention seems to not be explanation. The more a model grows in complexity, the smaller the accuracy boost from attention

Attention Weights	LSTM-Att	big CNN-LSTM-Att
Correct Prediction	**0.559**	**0.617**
Wrong Prediction, predicted label	0.097	0.101
Wrong Prediction, true label	0.533	0.489
Permuted Attention		
Correct Prediction	0.445	0.369
Wrong Prediction, predicted label	0.129	0.130
Wrong Prediction, true label	0.393	0.347
Randomized Attention		
Correct Prediction	0.384	0.402
Wrong Prediction, predicted label	0.127	0.138
Wrong Prediction, true label	0.394	0.373

Table 3: Attention Hit Rates across Various Models

it receives. This implied that attention was not helping at all. We also noted from the classification accuracy that the class imbalance was causing predictions to appear suboptimal. From this viewpoint, it appeared that subsequences highlighted by attention would be inconsequential. However, the model driven approach was not sufficient in giving us a complete view of attention.

When looking within the biological domain, attention was actually very useful on the subsequence level. With our hit rate metric we were able to see the beneficial effect of attention across all models. Despite the neural networks having no prior biological knowledge, subsequences extracted from attention weights were grounded in biological ground truths as verified by the peer-reviewed database. Extracted subsequences derived from attention were more accurate than the integrated gradients approach, which implies that it is attending to more relevant subsequences.

Subsequences from correct predictions have higher hit rates with the true location than subsequences from wrong predictions, which means a hit rate can be used as a proxy for confidence. The higher the hit rate, the more likely the model's prediction is correct. Even when the model makes an incorrect prediction, we can observe that the derived subsequences are biologically grounded. This is because the hit rate of these "incorrect" subsequences had a higher hit rate with the true location than the wrong prediction. Therefore applying the hit rate methodology across multiple classes can act as a check if there is skepticism towards a model prediction. Through this metric, attention visualization allows for higher interpretability and trust without directly improving the classification accuracy.

5 Conclusion

We have presented findings that affirms the utility of attention and a new methodology to verify predictions and their reasoning against an external knowledge base. We found that attention focused on relevant peptides that are biologically grounded, especially in LSTMs and CNN-LSTMs. The hit rate methodology paired with subsequence visualization allows DeepLoc to be transparent, interpretable, and therefore trustworthy.

There are limitations to our work however. Proteins and subsequences can legitimately be located in multiple places, but our dataset and architectures do not handle this case. This can be remedied by framing this task as a multilabel classification problem with a more robust dataset. We may also consider more subcellular locations beyond the 10 coarsely defined categories, which may help alleviate the class imbalance in the dataset. More work needs to be done here to further validate our results. The API calls required to search each subsequence in the UniProt database were a bottleneck to our approach. Future work may involve curating a larger dataset for PSL localization that can be used to quickly perform hit rate calculations.

We were able to replicate similar experiments for the task of protein subcellular localization and observe similar behavior. However, using a new metric that we call *hit rate*, we were able to show evidence that although the attention is not able to provide any additional predictive power, it is capturing important information that can help us explain which subsequences in each protein are responsible for the classification of protein's location. While we only developed this methodology and examined these architectures in the niche of protein subcellular localization, the general paradigm can be extended to any problem where subsequences are relevant and known ground truths exist. As

[4] has shown, attention is useful for tasks such as pair sequence or language generation where the interpretability of such attention models is important.

References

[1] Annette Höglund, Pierre Dönnes, Torsten Blum, Hans-Werner Adolph, and Oliver Kohlbacher. MultiLoc: prediction of protein subcellular localization using N-terminal targeting sequences, sequence motifs and amino acid composition. *Bioinformatics*, 22(10):1158–1165, 01 2006.

[2] Søren Kaae Sønderby, Casper Kaae Sønderby, Henrik Nielsen, and Ole Winther. Convolutional lstm networks for subcellular localization of proteins. In Adrian-Horia Dediu, Francisco Hernández-Quiroz, Carlos Martín-Vide, and David A. Rosenblueth, editors, *Algorithms for Computational Biology*, pages 68–80, Cham, 2015. Springer International Publishing.

[3] José Juan Almagro Armenteros, Casper Kaae Sønderby, Søren Kaae Sønderby, Henrik Nielsen, and Ole Winther. DeepLoc: prediction of protein subcellular localization using deep learning. *Bioinformatics*, 33(21):3387–3395, 07 2017.

[4] Shikhar Vashishth, Shyam Upadhyay, Gaurav Singh Tomar, and Manaal Faruqui. Attention interpretability across nlp tasks, 2019.

[5] Sofia Serrano and Noah A. Smith. Is attention interpretable?, 2019.

[6] Sarah Wiegreffe and Yuval Pinter. Attention is not not explanation. *arXiv preprint arXiv:1908.04626*, 2019.

[7] Long Pang, Junjie Wang, Lingling Zhao, Chunyu Wang, and Hui Zhan. A novel protein subcellular localization method with cnn-xgboost model for alzheimer's disease. *Frontiers in Genetics*, 9:751, 2019.

[8] Paul Horton, Keun-Joon Park, Takeshi Obayashi, Naoya Fujita, Hajime Harada, C. J. Adams-Collier, and Kenta Nakai. Wolf psort: protein localization predictor. *Nucleic acids research*, 35(Web Server issue):W585–W587, Jul 2007. 17517783[pmid].

[9] Martin Wattenberg, Fernanda Viégas, and Ian Johnson. How to use t-sne effectively. *Distill*, 2016.

[10] Sarthak Jain and Byron C. Wallace. Attention is not explanation, 2019.

[11] The UniProt Consortium. UniProt: the universal protein knowledgebase. *Nucleic Acids Research*, 45(D1):D158–D169, 11 2016.

[12] Sylvain Poux, Cecilia N Arighi, Michele Magrane, Alex Bateman, Chih-Hsuan Wei, Zhiyong Lu, Emmanuel Boutet, Hema Bye-A-Jee, Maria Livia Famiglietti, Bernd Roechert, and The UniProt Consortium. On expert curation and scalability: UniProtKB/Swiss-Prot as a case study. *Bioinformatics*, 33(21):3454–3460, 07 2017.

[13] Chuming Chen, Zhiwen Li, Hongzhan Huang, Baris E. Suzek, Cathy H. Wu, and UniProt Consortium. A fast Peptide Match service for UniProt Knowledgebase. *Bioinformatics*, 29(21):2808–2809, 08 2013.

[14] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate, 2014.

[15] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 3319–3328. JMLR.org, 2017.

[16] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489, San Diego, California, June 2016. Association for Computational Linguistics.

6 Appendix

6.1 Attention Visualization

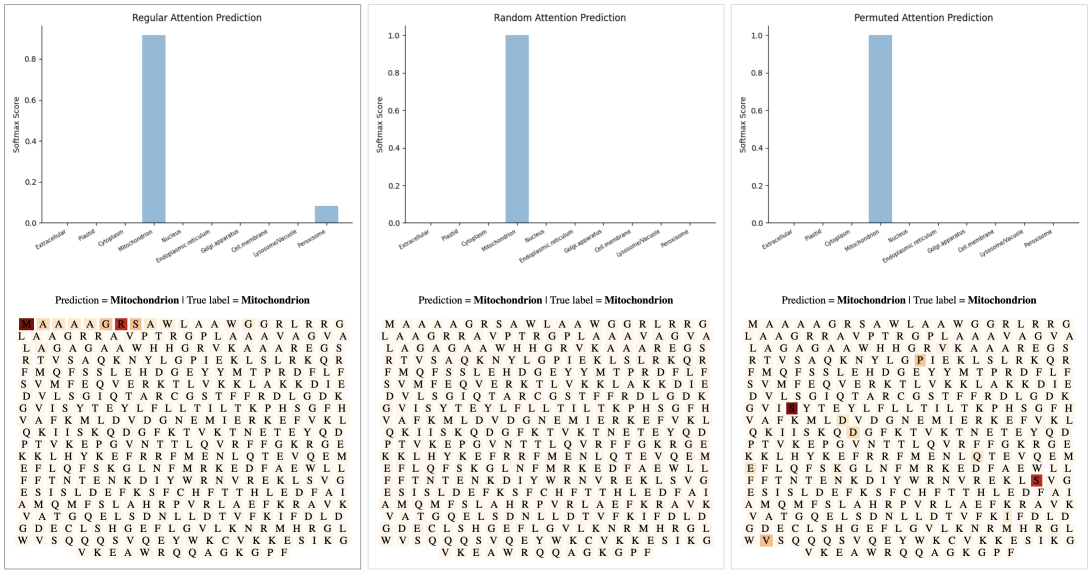


Figure 2: Attention Visualization Dashboard

The attribution scores extracted from the BiLSTM model can be visualized in 3. Darker shades of red shown represent a greater importance attributed by the attention layer. In this example, the subsequence extractor would extract the sequence "MAAAAGRS". In the middle and right images show the attributions set randomly and by shuffling attention weights. For the *hit rate* calculations in 4, the API call would search for proteins containing sequence "MAAAAGRS". The resulting entries would then be used to calculate the hit rate.

With a visualization like this, we can also see the change in model's prediction confidence when attention weights are altered.

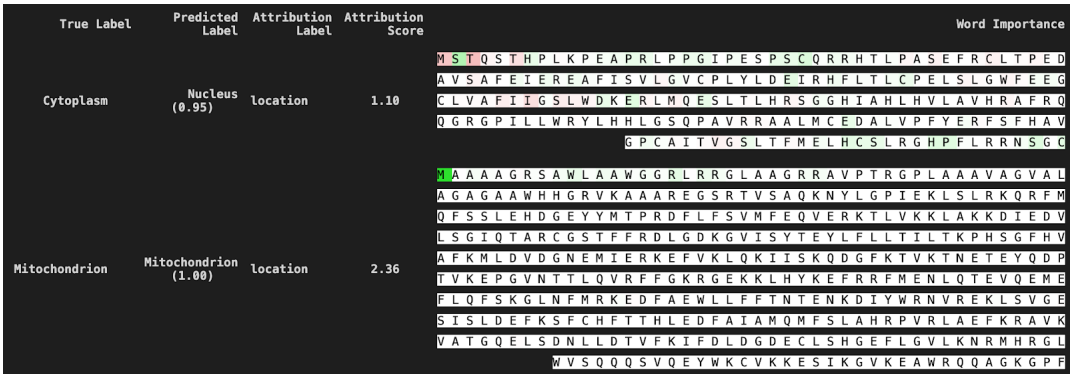


Figure 3: Integrated Gradient Visualization

Above is an attribution score visualization derived from integrated gradients of two sequences. Visualization and calculation of gradients is from the implementation in Captum.