8-21-2024 10:30 AM

Understanding Protein Deep Learning Models through Explainability

Zahra Fazel, *The University of Western Ontario*

Supervisor: Ilie, Lucian, *The University of Western Ontario*

A thesis submitted in partial fulfillment of the requirements for the Master of Science degree in Computer Science

Abstract

This thesis investigates the application of Explainable Artificial Intelligence (XAI) techniques to deep learning models in the field of protein analysis, specifically targeting protein language models and protein interaction site prediction models. Despite the increasing adoption of these sophisticated deep learning models in bioinformatics, their intrinsic complexity often results in a black-box nature, obscuring the understanding of their decision-making processes.

This research represents a thorough effort to integrate explanation methods within this context. We analyze the resulting interpretations using biological-specific statistical tests to enhance the transparency and interpretability of the models. This work evaluates the efficacy of current XAI methods applied to protein analysis through a comprehensive set of experiments.

Keywords: Deep Learning, Explainability, Protein Language Models

Summary for Lay Audience

Deep learning models have demonstrated impressive success across various fields, and as their performance improves in different applications, their architecture becomes more complex. Despite their marvellous performance, they are often referred to as ”black boxes” due to their opaque decision-making processes. As their complexity increases, their interpretability tends to decrease even as their performance increases. This lack of transparency becomes particularly critical in fields such as bioinformatics, where the consequences of their decision are significant.

This necessity drives our research into interpretation methods specifically for proteins. We have adapted existing interpretation techniques to our protein language models and protein interaction site prediction problems to understand better what the model learned. This work aims to ensure that the deep learning models used in bioinformatics are not only robust but also comprehensible and trustworthy, helping experts make better decisions in their research. This thesis presents a detailed examination of how effective current explainable artificial intelligence methods are in making these sophisticated models more transparent and interpretable.

Acknowlegements

I would like to express my deep gratitude to my supervisor, Dr. Lucian Ilie, for his continuous and unwavering guidance, support, and patience throughout this thesis. I am truly fortunate to have had the opportunity to work under his guidance.

I also want to thank my family for being my pillars of strength and my driving force throughout this journey and for their unconditional support and continuous encouragement to trust myself and believe in what I am doing and my research, even from thousands of kilometres away.

I extend my appreciation to my internship supervisor, Dr. Thomas Markovich, for his invaluable support and mentorship that not only helped me during my internship but also made me a better researcher.

Eventually, I want to thank my friends, both those who are close and those who are kilometres away scattered around the world. Your presence in my life is a gift.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The field of computational biology has experienced significant transformations with the advent of deep learning models. In particular, these models have revolutionized protein analysis, providing unprecedented insights into protein structure prediction, function annotation, and interaction dynamics. However, proteins are inherently complex molecules critical to almost all biological processes, and errors in these models' predictions of protein behaviours could lead to incorrect conclusions and potentially harmful implications in applications such as drug design.
Current deep learning models act as black boxes, transforming inputs into outputs with little insight into the intervening processes. This black-box nature makes it difficult for researchers to verify models' predictions against known biological principles or to refine these models. Therefore, explainability has emerged as a critical factor for researchers aiming to understand what these models truly learn. This is even more challenging in biological contexts, where data often consist of complex, imbalanced, and numerical datasets that even experts find challenging to interpret.
This thesis focuses on this challenge, discussing the interpretation methods employed for protein-related deep learning models. Additionally, we will explore the relations between these interpretations and amino acid properties.

1.1 Motivation

In recent years, deep learning has integrated with biology and revolutionized the field, including understanding protein structures and functions. Due to their ability to learn complex patterns from large datasets, these models have shown unprecedented success in predicting protein folding, interactions, and functionality. However, deep learning models lack transparency and interpretability because of their black-box nature. This prevents building trust with professionals and limits the potential for broader applications and innovations. This thesis has employed explainable artificial intelligence (XAI) methods, specifically on deep learning models for proteins, to fill the gap between these tools and users.

1.2 Problem Statement

We conduct an in-depth exploration of interpretations derived from applying existing explainable artificial intelligence (XAI) methods to protein language models and protein interaction site prediction models. This analysis seeks to uncover the underlying mechanisms driving model predictions, particularly in relation to known biochemical and structural characteristics of amino acids. By examining how different XAI approaches interpret the predictions, we aim to gain insights into the consistency and reliability of these models. Our ultimate goal is to understand how the choice of model architecture and XAI technique influences the interpretability of predictions, potentially leading to more robust and trustworthy applications in protein science.

1.3 Thesis Contribution

The significant contribution of this thesis is applying explainable artificial intelligence methods to protein-related deep learning models and analyzing their relation to amino acid properties. This research focuses on understanding protein-related deep learning models to provide deeper insights into how they work through comprehensive experiments. Such findings impact the way computational biologists and chemists approach protein-related problems.

1.4 Thesis Outline

This thesis, comprising five chapters, begins with an introduction to the problem, setting the stage for our research.
Chapter 2 is dedicated to providing the essential background knowledge, a prerequisite for comprehending the concepts employed in this project.
These concepts include amino acids and proteins, deep learning, embedding models, protein interaction site prediction models, and explainable artificial intelligence methods.
In Chapter 3, we discuss how we applied explainable artificial intelligence methods to models and the techniques and tests used to analyze them. Chapter 4 contains the experiments, their results, and their analyses.
Finally, in Chapter 5, we summarize our work and present some possible research that can be done in this field to improve this research domain.

Chapter 2

Background

2.1 Proteins and Amino acids

2.1.1 Amino acids

Amino acids are building blocks of proteins that are linked by peptide bonds to form polypeptide chains. There are twenty different amino acids present in proteins as shown in Fig. 2.1: Phenylalanine(F), Tryptophan(W), Isoleucine(I), Leucine(L), Methionine(M), Valine(V), Cysteine(C), Alanine(A), Tyrosine(Y), Histidine(H), Proline(P), Glycine(G), Threonine(T), Serine(S), Lysine(K), Arginine(R), Glutamic acid(E), Aspartic acid(D), Glutamine(Q) and Asparagine(N). Amino acids have various properties [1]. Some of these properties are:

- Aromaticity: Aromaticity in amino acids refers to the presence of a specific type of cyclic side chain. Aromatic side chains are often involved in interactions with proteins. Phenylalanine(F), Tyrosine(Y), and Tryptophan(W) are aromatic amino acids [1].

- Acidity and Basicity: Acidity in amino acids refers to the presence and behaviour of the acidic and basic groups within their structures. Aspartic acid(D) and Glutamic acid(E) are acidic amino acids, while Histidine(H), Lysine(K), and Arginine(R) are basic amino acids [1].

- Hydrophobicity: Hydrophobicity in amino acids refers to the tendency of certain amino acids to avoid interaction with water and instead prefer interaction with other nonpolar substances [1].

- Molecular Mass: Molecular mass, or molecular weight, of amino acids, refers to the sum of the atomic masses of all the atoms in a single molecule of an amino acid [53].

- Van Der Waal Volume: Van der Waals volume, or molecular volume, measures the occupied space of an atom or molecule, including the area influenced by its electron cloud [53].

- Dipole Moment: The dipole moment of an amino acid is a vector quantity that describes the magnitude and direction of the separation of charge within the molecule [53].

Quantitative values of these properties are shown in Table. 2.1

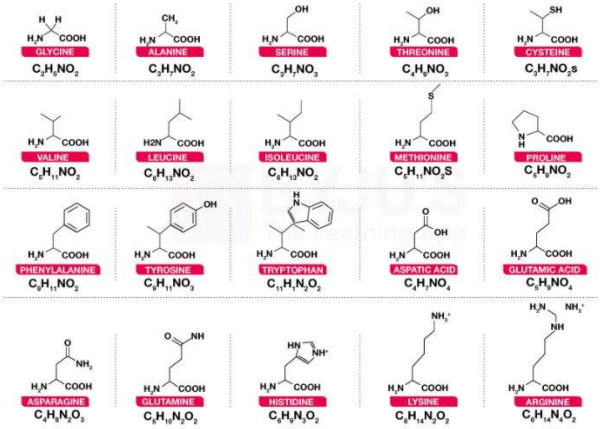

Figure 2.1: Amino acids and their structures [7]

Table 2.1: Quantitive values of amino acids' properties.

Amino acid	Hydrophobicity	Molecular Mass	Van der Waals Volume	Dipole Moment
Glycine (G)	-0.4	57	48	0.0
Alanine (A)	1.8	71	67	5.937
Serine (S)	-0.8	87	73	9.836
Proline (P)	-1.6	97	90	7.916
Valine (V)	4.2	99	105	2.692
Threonine (T)	-0.7	101	93	9.304
Cysteine (C)	2.5	103	86	10.74
Isoleucine (I)	4.5	113	124	3.371
Leucine (L)	3.8	113	124	3.782
Asparagine (N)	-3.5	114	96	18.89
Aspartic acid (D)	-3.5	115	91	29.49
Glutamine (Q)	-3.5	128	114	39.89
Lysine (K)	-3.9	128	135	50.02
Glutamic acid (E)	-3.5	129	109	42.52
Methionine (M)	1.9	131	124	8.589
Histidine (H)	-3.2	137	118	20.44
Phenylalanine (F)	2.8	147	135	5.98
Arginine (R)	-4.5	156	148	37.5
Tyrosine (Y)	-1.3	163	141	10.41
Tryptophan (W)	-0.9	186	163	10.73

2.1.2 Proteins

Proteins are large complex molecules that play many critical roles in the body. They are made up of amino acids, which are attached to one another in long chains, which determine each protein's unique 3-dimensional structure and its specific function [1].

2.1.3 Proteins Interactions

Protein interactions refer to the ways in which proteins communicate and bind with other molecules within the cell. These interactions can include interactions with other proteins, nucleic acids, lipids, and small molecules. These interactions are crucial for virtually all biological processes, from cellular signalling and structural support to immune responses and enzymatic activity [1].

2.2 Machine Learning

Machine learning (ML) is a subfield of artificial intelligence that focuses on the development of pattern recognition algorithms, also known as models, that learn from data without requiring explicit programming [5]. Machine learning has been widely used for various tasks in different fields, including biology [47, 52, 6]. Machine Learning systems can be classified into three categories according to the amount and type of supervision they get during training.

2.2.1 Supervised Learning

In supervised learning, shown in Fig. 2.2, the model is provided with the desired solutions, called labels, during training. In this framework, the dataset consists of input features and corresponding target output. The goal of the learning algorithm is to map inputs to outputs by minimizing the error between its predictions and the real labels so that it can predict correct outputs on unseen data. Classification and regression are the two most common supervised learning tasks [13].

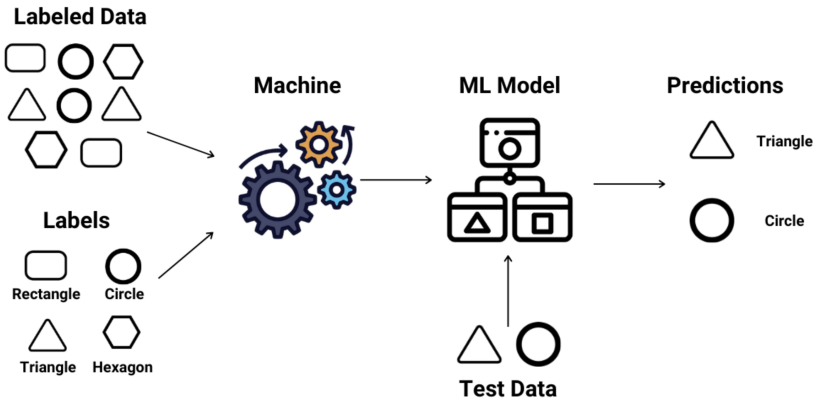

Figure 2.2: Supervised learning example [35]

2.2.2 Unsupervised Learning

Unsupervised learning, shown in Fig. 2.3, focuses on finding hidden patterns within data using only features without explicit feedback or guidance of target outputs. The goal is to learn about the structure and properties of the data itself. Clustering, anomaly detection, and dimensionality reduction are some of the applications of unsupervised learning [13].

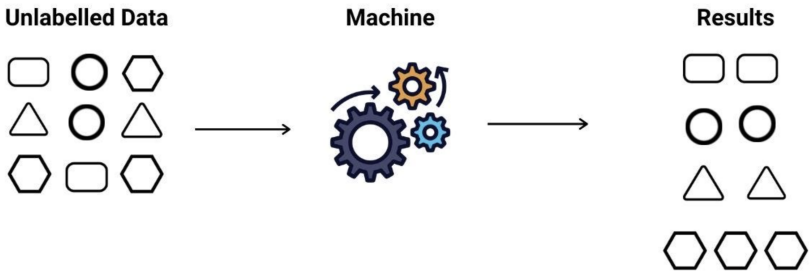

Figure 2.3: Unsupervised learning example [35]

2.2.3 Semi-supervised Learning

In semi-supervised learning, shown in Fig. 2.4, the model takes advantage of both supervised and unsupervised learning. This approach provides the model with a large unlabelled and a small labelled dataset. The objective is to leverage the pattern presented in unlabelled data to improve performance on labelled data. Semi-supervised techniques are used when gathering a large labelled dataset is inefficient [13].

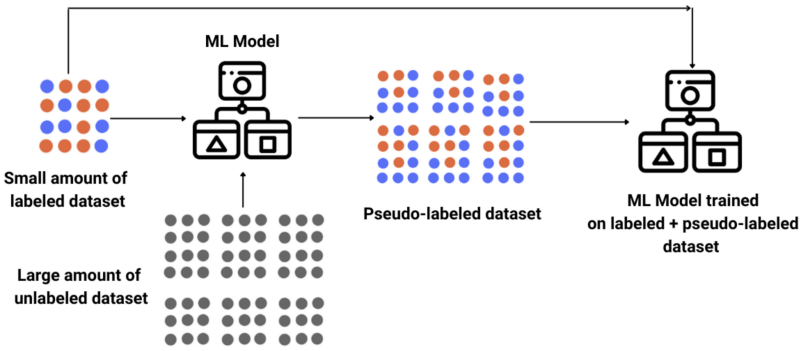

Figure 2.4: Semi-supervised learning example [35]

2.3 Deep Learning

Deep learning is a subset of machine learning that uses artificial neural networks to learn complex data patterns from raw, unstructured data.

2.3.1 Artificial Neural Networks

Inspired by the human brain and biological neural networks, artificial neural networks (ANN) are built of layers of connected artificial neurons. Each neuron receives inputs, processes them, and returns an output for another neuron [13].

2.3.2 Perceptron

Perceptron is the simplest ANN architecture, composed of threshold logic units (TLU).
The TLU unit gets a vector $x = [x_1, x_2, \ldots, x_n]$ as input and associates each input x_i with a weight w_i to compute the weighted sum $z = w_1 x_1 + \ldots + w_n x_n$, then apple a step function to that sum and outputs the result. The architecture is shown in Fig. 2.5.
A Perceptron is simply composed of a single layer of TLUs, with each TLU connected to all the inputs as shown in Fig. 2.6 [13].

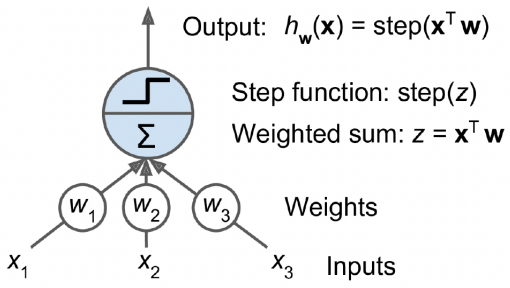

Figure 2.5: Threshold logic unit structure [13]

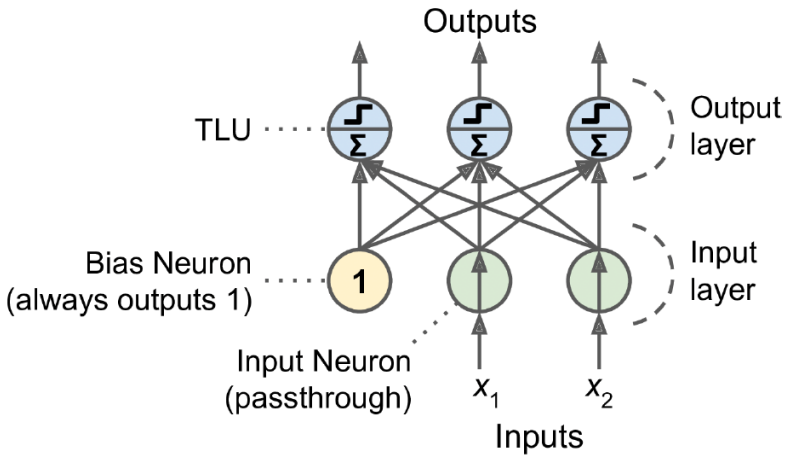

Figure 2.6: Perceptron structure [13]

2.3.3 Activation Functions

Activation functions are non-linear functions used in TLUs as step functions. The most common choices for activation functions are sigmoid, hyperbolic tangent (tanh), rectified linear unit (ReLU), and softmax.

2.3.4 Multi-layer Perceptron

A Multi-layer Perceptron (MLP), shown in Fig. 2.7, is composed of an input layer, one or more layers of TLUs, called hidden layers, and a final layer of TLUs called the output layer, where

the output of each layer is used as the input for the next layer [13].

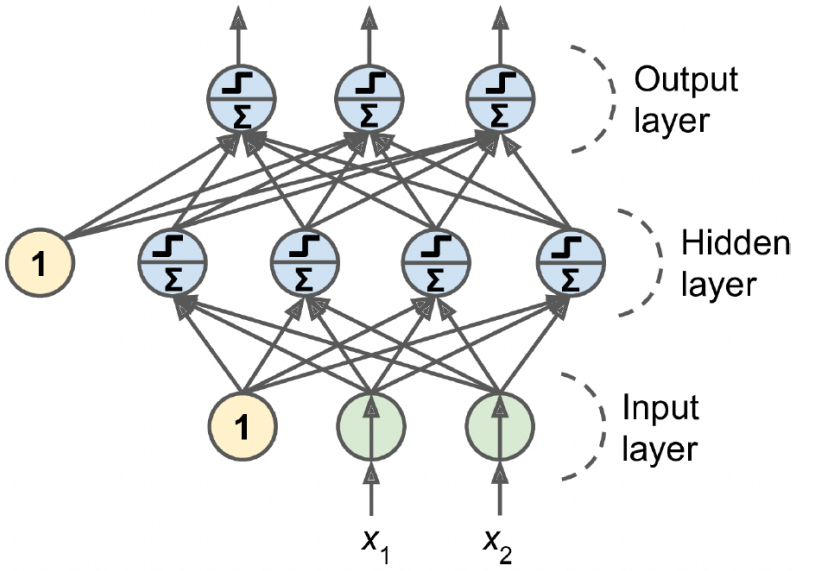

Figure 2.7: Multi-layer Perceptron structure [13]

2.3.5 Learning Algorithm

The learning algorithm, shown in Fig. 2.8, is as follows:

1. **Weight Initialization.** The weights of each TLU are randomly initialized. This provides the model with a starting point for learning.

2. **Forward Propagation.** Input data is passed through the network from the input layer to the output layer.

3. **Loss Calculation.** The error between the network's output and the desired output is calculated using a loss function. Usually, Cross Entropy is used for classification, and Mean Square Error is used for regression.

4. **Backward Propagation.** The error calculated in the previous step is propagated back through the network using the Backpropagation algorithm. This algorithm starts from the output layer to the input layer and uses the chain rule to compute the gradient of the loss function with respect to each weight in the network.

5. **Weight Updating.** Using the gradients calculated during backpropagation, the weights are then adjusted to minimize the loss using an optimization algorithm. Optimizers are algorithms used to minimize the loss. Optimizers have a learning rate parameter, which indicates how big the steps that gradient descent should take should be in the direction of the local minimum. Adam and Stochastic Gradient Descent (SGD) are the most common optimizers used in deep learning.

6. **Iteration.** Steps 1 - 4 are repeated for a certain number of rounds or until loss converges.

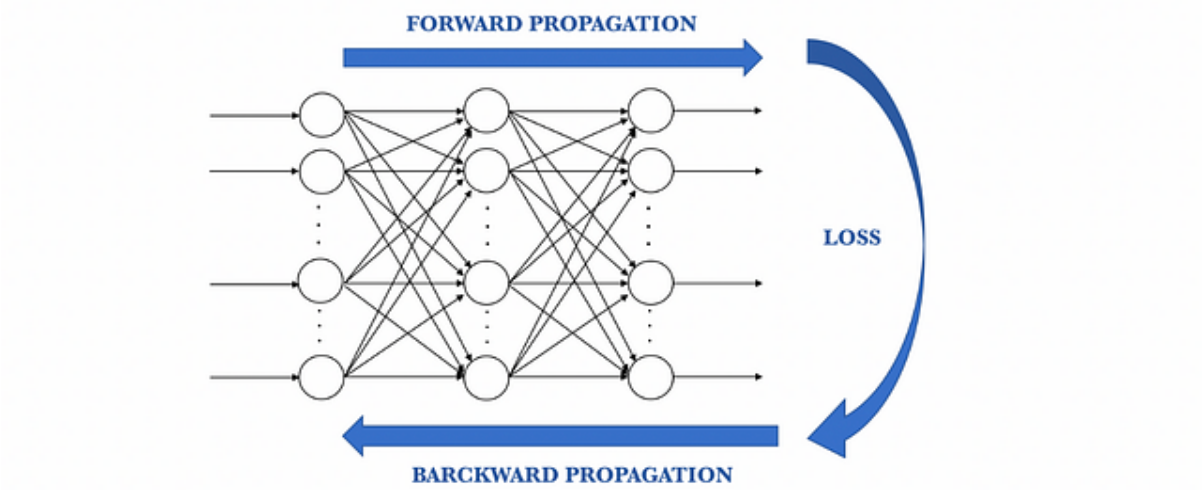

Figure 2.8: The learning process [48]

2.3.6 Convolutional Neural Networks

Convolutional neural networks (CNN) are a type of neural network designed to handle grid-like data, including images, videos, and spatial data. The architecture of CNNs, shown in Fig. 2.9, is characterized by the sequential arrangement of convolutional layers, pooling layers, and fully connected layers. This setup enables the model to extract and learn features at various levels of abstraction. In lower layers, the model tends to identify low-level features, like edges in images. As the data progresses through the network, subsequent layers focus on more complex, high-level features, allowing the network to understand broader aspects of the input data. CNNs are widely employed in vision tasks like image classification, object detection, image segmentation, etc. [13, 59].

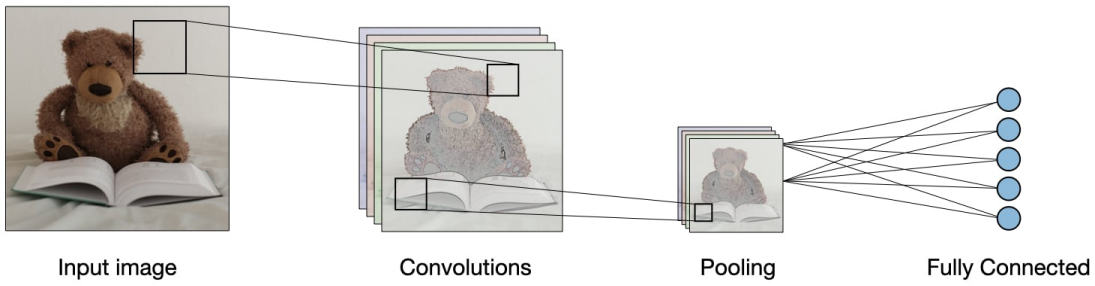

Figure 2.9: Convolutional neural network architecture [44]

1. **Convolutional Layer.** These layers are the most prominent part of CNNs. Each convolutional layer uses several small filters, called kernels, that go over the input. These filters perform a weighted sum between their values and subgrids of data. This process results in a feature map highlighting patterns the filters have identified [59]. An example of a convolutional layer is shown in Fig. 2.10.

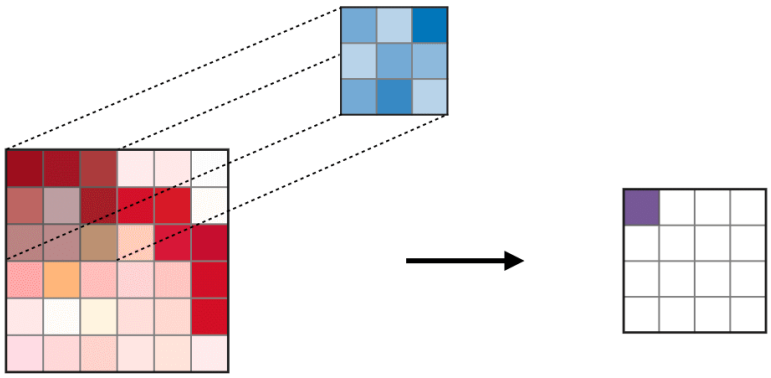

Figure 2.10: Convolutional layer [44]

2. **Pooling Layer.** These layers reduce the spatial size of feature maps, which makes the model smaller and faster to compute. This size reduction also allows the model to extract features independent of scale and orientation [59]. An example of a pooling layer is shown in Fig. 2.11.

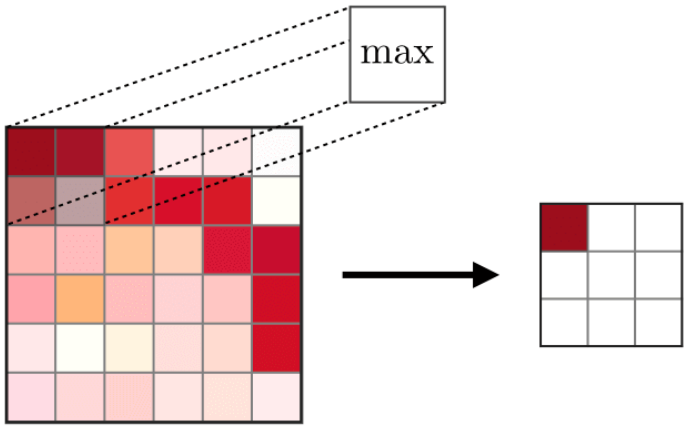

Figure 2.11: Pooling layer [44]

3. **Fully Connected Layer.** These layers have the same architecture as an MLP network, allowing the model to map extracted features to target output. An example of a fully connected layer is shown in Fig. 2.12.

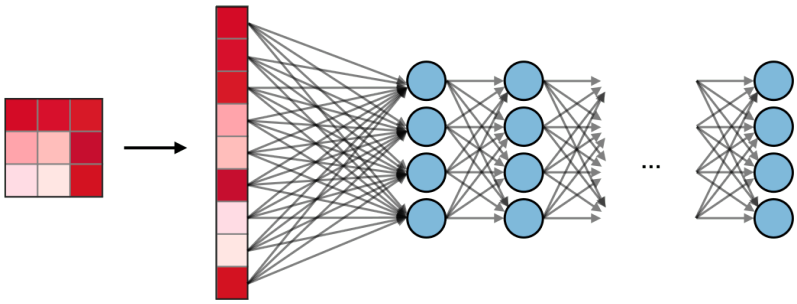

Figure 2.12: Fully connected layer [44]

2.3.7 Recurrent Neural Networks

Recurrent neural networks (RNNs) are another kind of neural network developed to process sequential data where ordering matters. Unlike feedforward neural networks such as MLPs and CNNs, RNNs have an internal memory unit that allows them to process previous inputs and successive inputs (Bidirectional RNNs) and use their information to learn current input, which makes them suitable for handling sequence-like data such as natural language and time series. There are four types of RNNs: (i) One-to-One, (ii) Many-to-One, (iii) One-to-Many, and (iv) Many-to-Many [59, 45].
Other than traditional RNNs, Gated Recurrent Units (GRUs) and Long Short-Term Memory Units (LSTMs) are two other types of RNNs.

2.3.8 Attention Mechanism

The attention mechanism allows the model to dynamically focus on different parts of the input and assign different weights depending on their relevance [27, 4, 50]. It maps a query vector Q derived from input to a pair of key and value vectors K and V. The output is a weighted sum of the value vector [50].
Self-attention is a type of attention that disambiguates words and learns semantics. A most common self-attention technique is scaled dot-product, which is shown in Fig. 2.13 and is computed as follows [50]:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_K}})V$$

Where d_K is dimension of vector K.

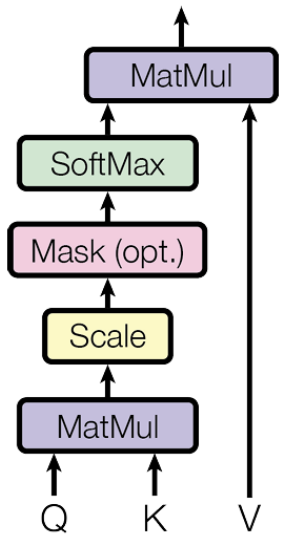

Figure 2.13: Scaled dot-product attention layer architecture [50]

2.3.9 Transformers

Transformers [50] are another type of neural network which employs a self-attention mechanism to extract features. The self-attention mechanism allows transformers to capture both dependencies between parts of data independent of their distance and based on the context. Transformers have an encoder-decoder architecture, as shown in Fig. 2.15, which allows them to process data in parallel and, therefore, use GPUs more efficiently.
The encoder maps an input sequence of symbol representations $X = (x_1, ..., xn)$ to a sequence of continuous representations $Z = (z_1, ..., z_n)$. Given Z, the decoder then generates an output sequence $Y = (y_1, ..., y_m)$ of symbols one element at a time [50].

1. **Encoder.** The encoder is built with N identical layers. Each layer combines a multi-head self-attention mechanism with a fully connected layer with residual connections. The encoder's primary objective is to transform all input sequences into a continuous representation of the learned information [50, 25].

2. **Decoder.** The decoder is also composed of N identical layers, each consisting of two multi-head self-attention layers and a fully connected layer with residual connections. The first self-attention layer is modified to employ a mask to prevent dependencies between the predictions for position i and outputs at positions greater than i. The decoder's output is passed through a softmax layer to compute probabilities of the next word in sequence [50, 25].

3. **Multi-Head Attention.** Transformers use multiple attention instead of a single attention. Multi-head attention, shown in Fig. 2.14, allows the model to learn different representations and combine their information [50, 25].

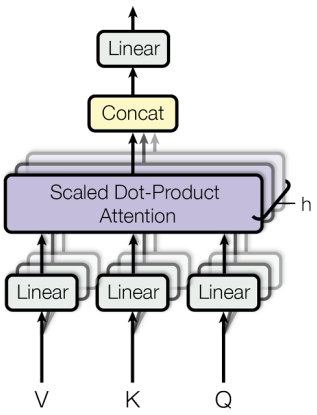

Figure 2.14: Multi-head self-attention layer architecture [50]

4. **Positional Encoding.** Since the encoder and decoder fail to encode the relative position of the tokens in a sentence, positional encoding is applied to both encode's and decoder's output to inject some information about the relative or absolute position of the tokens in the sequence. Positional encoding is two sinusoidal functions with different frequencies [50, 25].

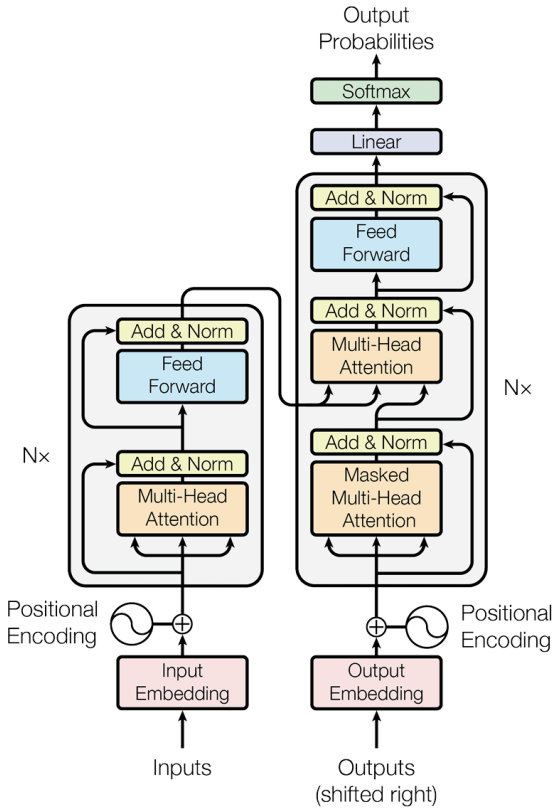

Figure 2.15: Transformer architecture [50]

2.4 Embeddings

Neural networks are unable to understand texts by their nature. Therefore, numbers are required to represent text data to neural networks. Word embeddings are a technique where individual words are transformed into a numerical representation of the word, where each word is mapped to one vector that captures various characteristics of that word with regard to the overall text [51].

2.4.1 Word2Vec

Word2Vec [28] is the first embedding method which uses a feedforward neural network. There are two methods that can be applied: (i) Continuous Bag of Words (CBOW) and (ii) Skip-gram. Both models slide across sentences in the data, and for each word considered a current word, a surrounding window of context words is defined. In CBOW, as shown in Fig. 2.16, a word is predicted given a set of neighbouring left and right words' distribution. In Skip-gram as shown in Fig. 2.17, given a word, the most probable context is predicted by predicting probability distributions for previous and next words [31, 38, 28].

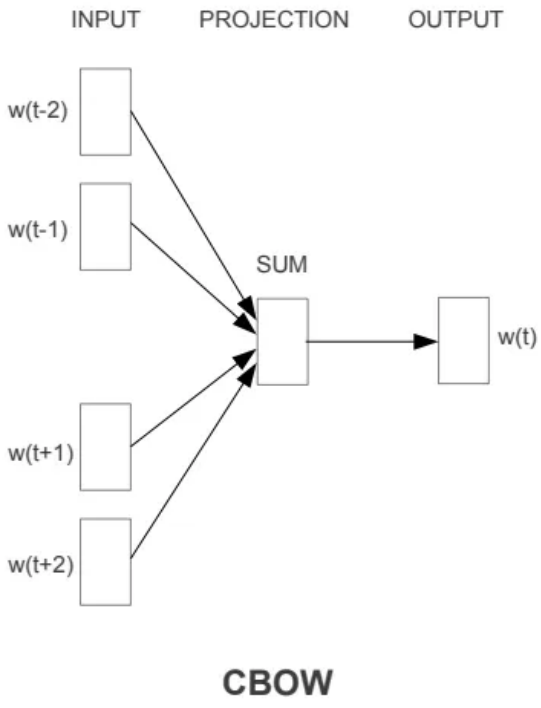

Figure 2.16: Continuous Bag of Words [51]

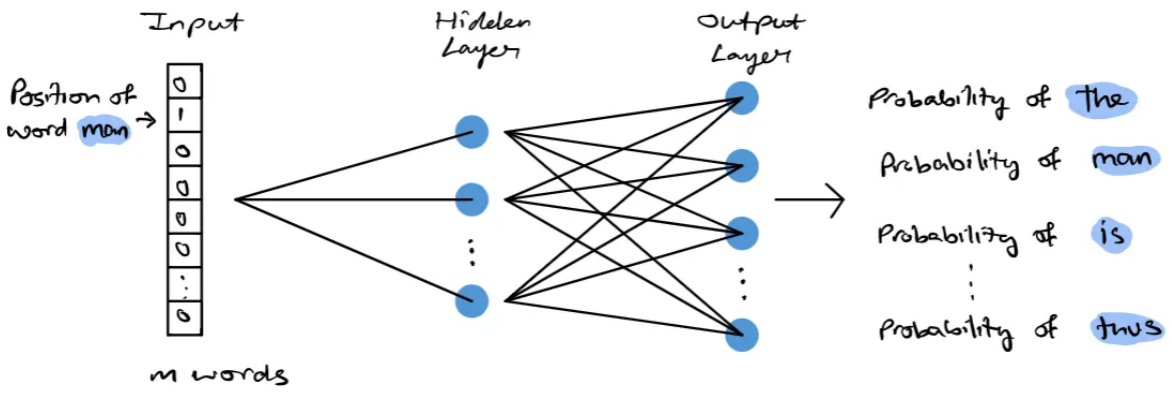

Figure 2.17: Skip-gram [51]

2.4.2 Global Vectors (GloVe)

GloVe [32] is another word embedding method. Unlike Word2Vec, GloVe takes both local and global statistics of the word. GloVe operates on the principle that the probabilities of word co-occurrences in a corpus contain potentially rich semantic information about the words and their relationships. The model efficiently leverages this information by constructing a large matrix that describes the frequency of words appearing together in a context within the corpus. This co-occurrence matrix is then factored into a lower-dimensional space using singular value decomposition (SVD) techniques. The resulting factors for each word represent its embedding vector [31, 18, 32].

2.4.3 Embeddings from Language Model (ELMo)

ELMo, introduced by [33], was developed to obtain deeper and context-dependent embeddings using bidirectional LSTMs. The process, as shown in Fig. 2.18, begins with the creation of character-based representations for each word. Then, each character is transformed into embeddings through a convolutional neural network (CNN). The sequence of these word representations is then processed by the LSTM network in both forward and backward directions, enabling the model to grasp the contextual relationships of each word with its surrounding words. In ELMo, the semantic and context-dependent information is captured primarily by the higher-level states of the LSTM, whereas the lower-level states predominantly handle syntactic elements [33, 31, 36].

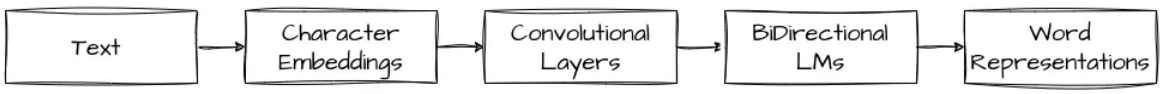

Figure 2.18: ELMo architecture [36]

2.4.4 Bidirectional Encoder Representations from Transformers (BERT)

BERT [9] is a transformer-based model trained bidirectional, allowing BERT to weigh the significance of each word based on its context, both preceding and succeeding. This context-awareness imbues BERT with the ability to generate contextualized word embeddings, which are representations of words considering their meanings within sentences [31]. Instead of predicting the next word in a sequence, BERT makes use of the Masked Language Model (MLM), where it randomly masks words in the sentence and then tries to predict them. In the training process, to understand the relationship between two sentences, BERT also receives pairs of sentences as input and learns to predict if the second sentence in the pair is the subsequent sentence in the original document [15, 21, 39]. BERT's architecture is shown in Fig. 2.19.

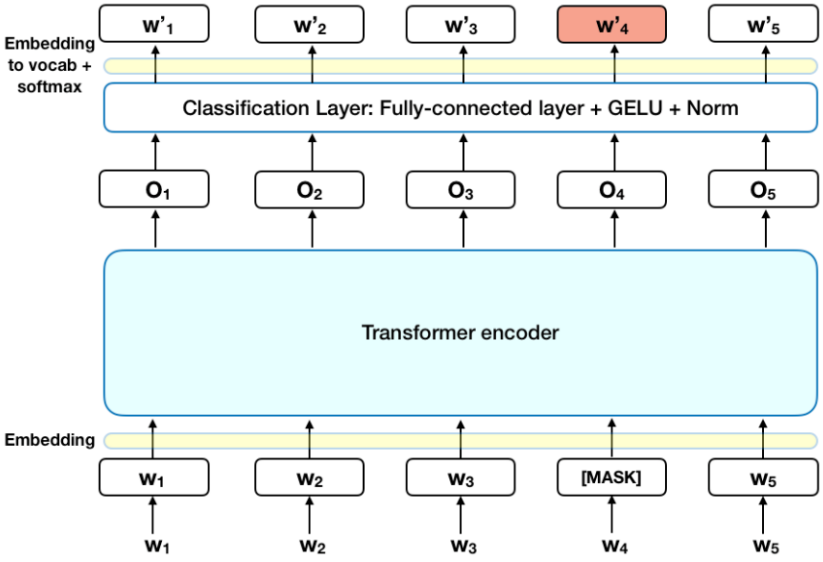

Figure 2.19: BERT architecture [15]

2.4.5 Text-to-Text Transfer Transformer (T5)

Unlike traditional models that are designed for specific natural language processing (NLP) tasks, T5 [34] reframes all NLP tasks to a text-to-text problem, allowing it to be applicable to a wide range of problems with a single model architecture. T5 uses a standard encoder-decoder transformer with minor changes to positional embedding [8, 29, 49].

2.5 Protein Embeddings

Inspired by embeddings and their performance in NLP tasks, protein embedding models represent proteins in a continuous vector space, capturing biological functional and structural relationships between amino acid sequences in proteins. This is done by considering each amino acid as a word. Several NLP language models are adapted for protein embeddings [11, 10].

2.5.1 Protein Transformers (ProtTrans)

ProtTrans [11] proposed several transformer-based models for protein embeddings, including ProtBERT and ProtT5, which are BERT and T5 trained on large protein datasets, respectively. Fig. 2.20 shows how ProtTrans models are used to derive embeddings.

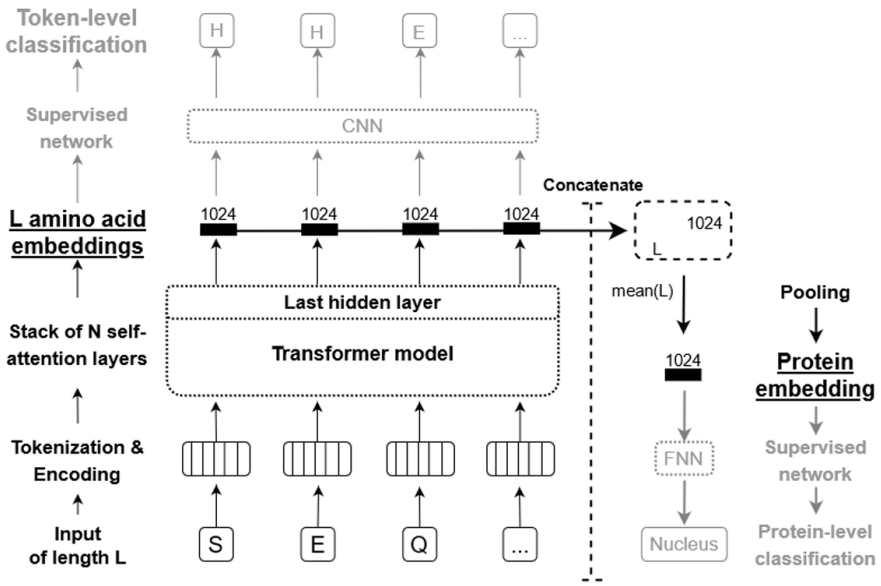

Figure 2.20: ProtTrans architecture [11]

2.5.2 Ankh

Ankh [10], the latest protein language model, uses the architecture shown in Fig. 2.21, which its performance improved using protein-specific optimizations.

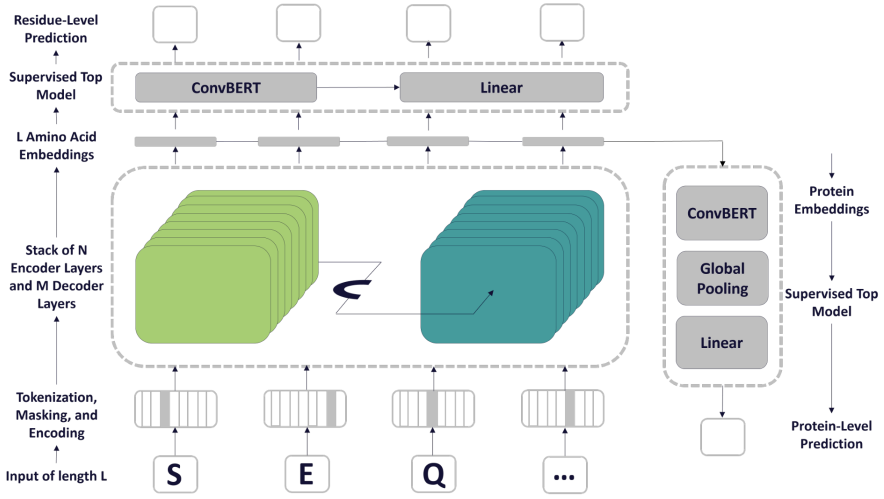

Figure 2.21: Ankh architecture [10]

2.6 Protein Interactions Site Prediction Models

Protein interaction site prediction (PISP) is the task of predicting whether each amino acid in a protein sequence interacts with any amino acid or other protein. The input for these models is a protein sequence of length n, and the output is a binary sequence of the same length with ith number showing whether ith amino acids have interactions. PISP models are either sequence-based or structure-based [56, 58, 60, 23, 17, 16].

2.6.1 Seq-InSite

Seq-InSite is the state-of-the-art PISP model that is sequence-based and has outperformed structure-based models [16]. Seq-InSite as shown in Fig. 2.22 is an ensemble of MLP and LSTM components that get a symmetric window of size 4 of ProtTrans and MSA (Multiple Sequence Alignment) embeddings as input.

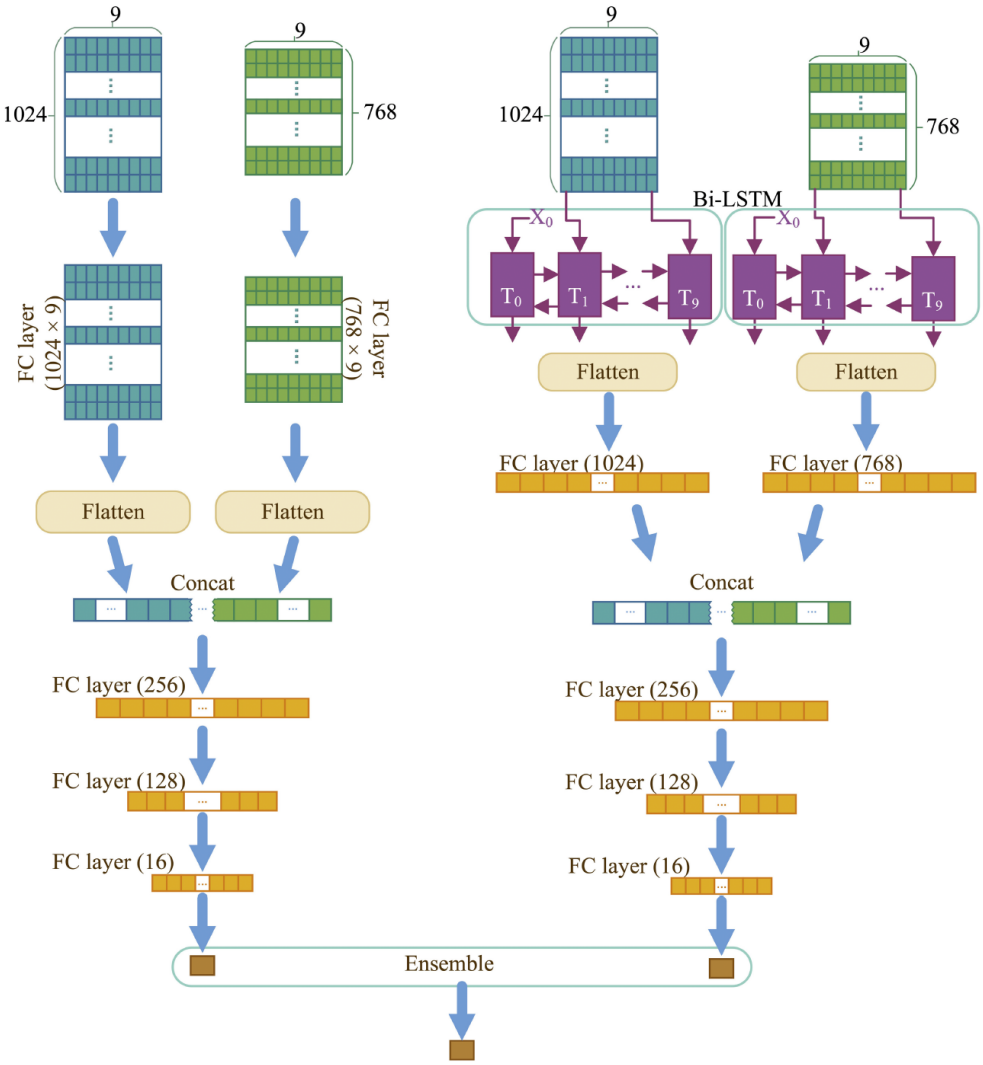

Figure 2.22: Seq-InSite architecture [16]

2.7 Explainability

Explainability is the ability to which a human can understand an artificial intelligence model, how it makes decisions, and its biases. It involves a set of methods that allow users to comprehend and, therefore, trust the machine's decision-making process. It helps build trust in artificial intelligence, transparency, debugging, improvements, and knowledge discovery from artificial black-box models [19].

2.7.1 Gradient-based Methods

Various interpretability methods are designed to ascertain the gradient of a prediction or classification score in correlation with the input features. These techniques, collectively known as gradient-based approaches, encompass an extensive array of methodologies. Each method's unique aspect lies in how it calculates this gradient. These approaches vary not only in their computational strategies but also in their application contexts and the precision with which they can pinpoint the influence of individual input features on the final prediction or classification outcome. By employing these gradient-based methods, one can gain deeper insights into how and why specific predictions or classifications are made, enhancing the transparency and understanding of complex predictive models [30].

Saliency (Vanilla Gradients)

Initially proposed by Simonyan and others in 2013 [42], this technique is an early example of pixel-attribution methods applied in convolutional neural networks (ConvNets) for image categorization.
To understand it, envision a trained ConvNet designed for classification, where a specific class is labelled as c. When an image, I, is fed into the ConvNet, the classification layer generates a score, $S_c(I)$, indicating the likelihood of the image belonging to class c. The objective here is to discover an L_2-regularised image that achieves a high S_c value for that image [42].
While $S_c(I)$ behaves as a highly non-linear function in deep ConvNets, we can approximate it locally with a linear function around a specific image, I_0, using a first-order Taylor expansion:

$$S_c(I) \approx w^T I + b$$

Where w is the derivative of S_c with respect to the image I at the point I_0:

$$w = \frac{\partial S_c}{\partial I}\Big|_{I_0}$$

The magnitude of the class score derivative provides a sensitivity measure. Pixels with higher values are the most sensitive ones, meaning minimal changes in their intensity can trigger significant swings in the score for a specific class [42].
The algorithm for this method is as follows:

1. A forward pass of image I_0 is performed.
2. A backward pass of image I_0 is performed to obtain w. If activation of the neurons in a layer below during backpropagation is negative, Vanilla Gradients sets the gradients to zero.

3. Visualize w as the explanation map.

The backward pass can cause a saturation problem when the activation is negative, and the activation function is ReLU. In this case, the negative activations of neurons are capped at zero and will not change anymore. Therefore, the Saliency map will say that these neurons are not important [30]. Examples of Saliency map explanations are shown in Fig. 2.23.

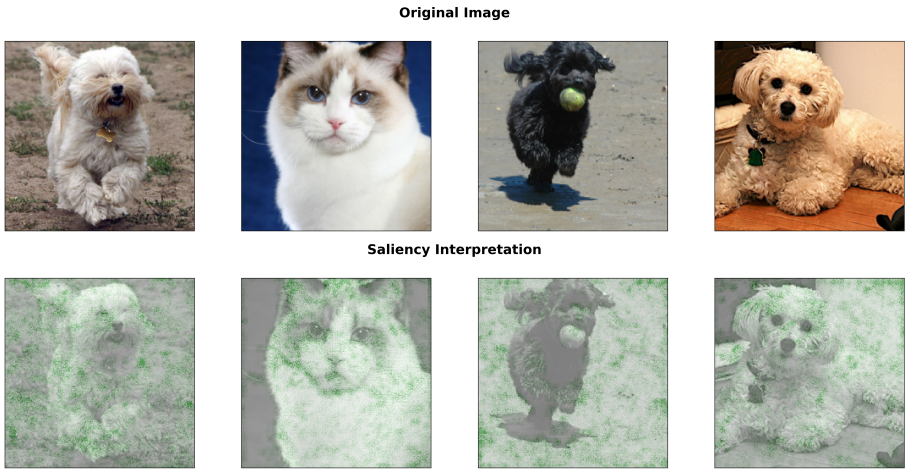

Figure 2.23: Examples of Saliency map explanation for a CNN classification model. Green pixels are pixels with positive influence in predicting the label of the pictures (Dogs and Cats)

Deconvolution Network (DeconvNet)

The DeconvNet model, introduced by Zeiler and Fergus (2014) [57], closely resembles the Vanilla Gradient technique. A DeconvNet can be conceptualized as a ConvNet operating in reverse. While a standard ConvNet translates pixels into features using components like filtering and pooling, a DeconvNet does the reverse: it maps features back to pixels [30].
To analyze a ConvNet, a DeconvNet is integrated with each layer of the ConvNet, as depicted in Fig. 2.24. This integration creates a seamless path from the deep layers of the network back to the original image pixels. The process begins by feeding an input image into the ConvNet, which computes features across various layers. To focus on a specific activation within the ConvNet, we isolate it by nullifying all other activations in the same layer. This isolated feature map is then used as the input for the corresponding DeconvNet layer [57].
The DeconvNet layer then undertakes a three-step reconstruction process: (1) 'unpooling' to reverse the pooling operation, (2) 'rectifying' to apply non-linear transformations, and (3) 'filtering' to revert the convolution operation. This procedure is iteratively performed layer by layer, retracing the steps back to the input pixel space. Through this method, an in-depth understanding of how specific features and activations within the ConvNet contribute to the final output can be gained [57]. Examples of DeconvNet explanation are shown in Fig. 2.25.

- **Unpooling.** While the max pooling operation is non-invertible, it is possible to record the locations of the maxima within each pool region and obtain an approximate inverse.

- **Rectification.** The reconstructed signal is passed through a ReLU non-linearity to obtain valid feature reconstructions at each layer.

- **Filtering.** To approximately reverse convolutional filters, transposed versions of the same filters are applied on the rectified maps.

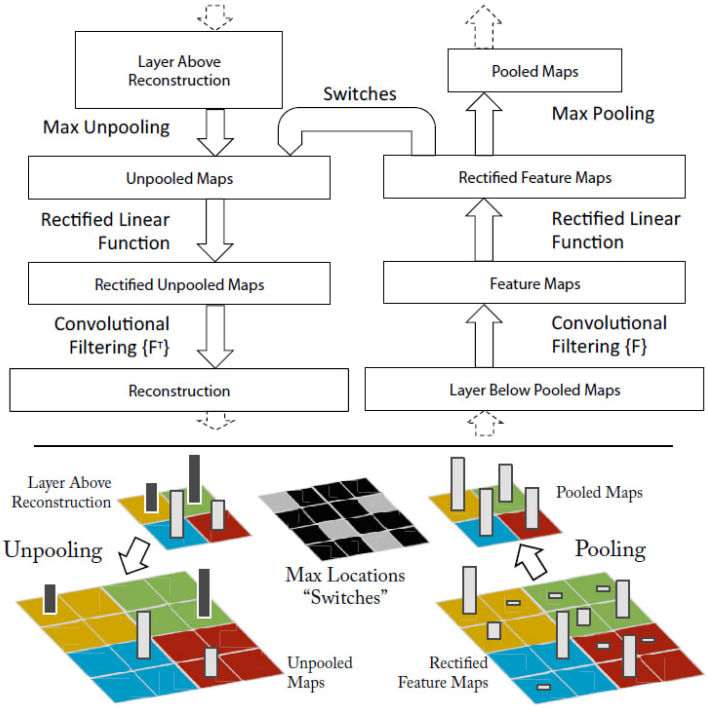

Figure 2.24: A DeconNet layer attached to a ConvNet layer. [57]

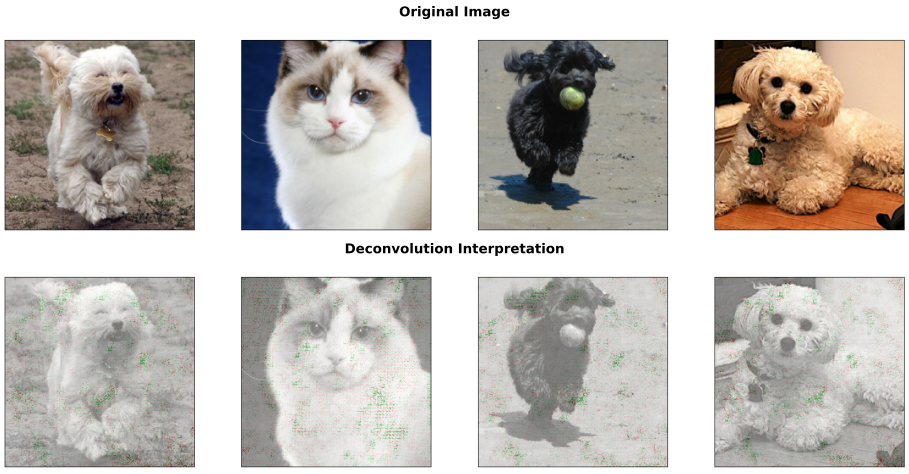

Figure 2.25: Examples of DeconvNet explanation for a CNN classification model. Green pixels are pixels with positive influence and red ones are pixels with negative impact in predicting the label of the pictures (Dogs and Cats).

Guided Backpropagation

Guided Backpropagation is built upon the ideas of saliency maps and deconvolution. While traditional backpropagation optimizes network parameters based on the overall loss function and is valuable for optimization, it does not directly answer our question: "Which pixels in the image matter most for the predicted class?." Guided Backpropagation is an extension of the basic backpropagation algorithm. It modifies the backpropagation process to propagate the gradients back through the network and selectively filter these gradients [12]. The process is as follows [43]:

1. A forward pass of a desired image is performed, and class scores for all potential classes are obtained.

2. The saliency map for the target class is computed.

3. A guided layer-wise deconvolution is computed by applying deconvolution to each intermediate layer using the element-wise product of the saliency map and the ReLU activations from the corresponding forward pass.

4. These steps are repeated for all layers to reconstruct the image iteratively.

Despite its efficacy, Guided Backpropagation has limitations. Sensitivity to network hyperparameters and initialization conditions can introduce noise and artifacts into the reconstructed image. Furthermore, its focus on individual class explanations necessitates caution when extrapolating insights to the broader model behaviour [12]. Examples of Guided Backpropagation explanations are shown in Fig. 2.26.

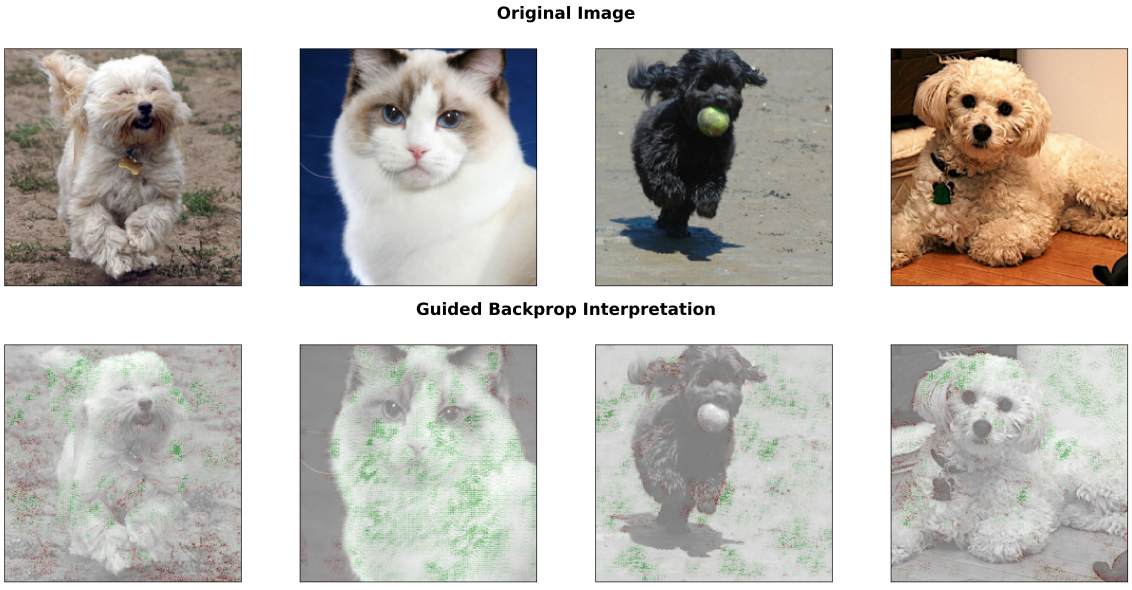

Figure 2.26: Examples of Guided Backpropagation explanation for a CNN classification model. Green pixels are pixels with positive influence and red ones are pixels with negative impact in predicting the label of the pictures (Dogs and Cats).

Input X Gradient

The technique introduced by Shrikumar et al. (2016) [41] enhances the clarity of attribution maps. In this approach, the attributions are calculated by first determining the partial derivatives of the output in relation to the input and then multiplying these derivatives with the input itself [2]. Examples of Input X Gradient explanation are shown in Fig. 2.27.

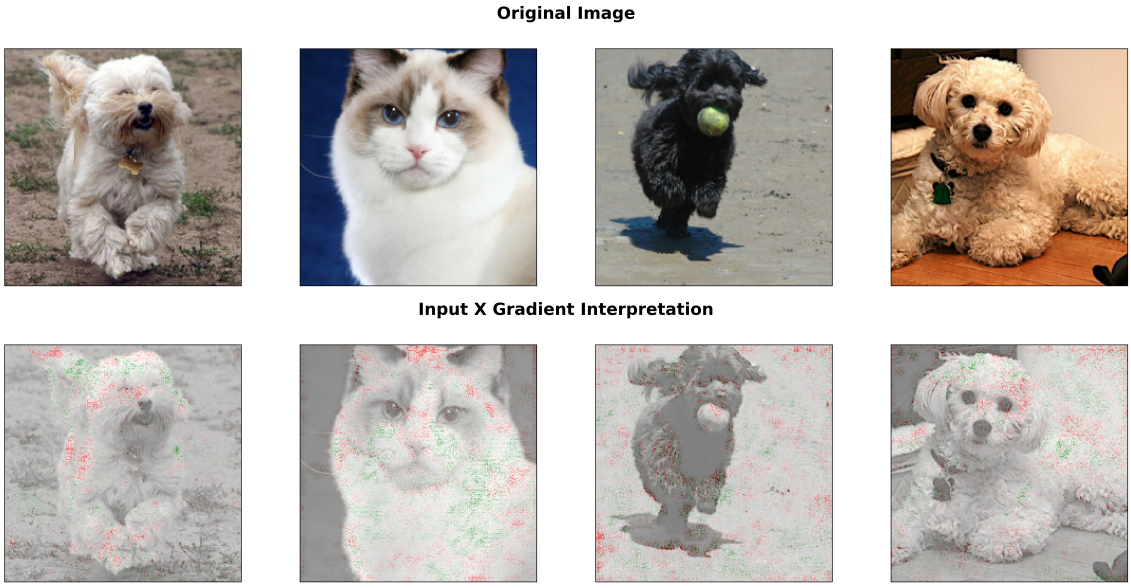

Figure 2.27: Examples of Input X Gradient explanation for a CNN classification model. Green pixels are pixels with positive influence and red ones are pixels with negative impact in predicting the label of the pictures (Dogs and Cats).

2.7.2 Path-attribution Methods

Path-attribution methods break down the impact of an input on a model's prediction by comparing it to a reference point. They take the difference in the original and reference input predictions and then spread it across all the original input features. This shows how much each feature contributed to the overall prediction [30].

DeepLIFT

DeepLIFT is a path-attribution backpropagation-based method that can propagate an importance signal even in situations where the gradient is zero.
Assuming t represents some target output neuron of interest, $x_1, x_2, \ldots, x_n$ represent some neurons in some intermediate layer or set of layers that are necessary and sufficient to compute t, and t^0 represent the reference activation of t, then $\Delta t = t - t^0$ is the difference-from-reference. DeepLIFT assigns contribution scores $C_{\Delta x_i \Delta t}$ to Δx_i such that [40]:

$$\Sigma_{i=1}^{n} C_{\Delta x_i \Delta t} = \Delta t$$

A neuron can have contributions to the output even if its gradient is zeros. However, $C_{\Delta x_i \Delta t}$ can be non-zero even if the gradient of t is zero, meaning DeepLIFT overcomes the shortcoming of gradient-based methods. Moreover, difference-from-reference is continuous, allowing DeepLIFT to address the issue of sudden jumps in gradients that are caused by discontinuity [40].
Contribution scores are assigned based on the following rules [40]:

1. **Linear Rule.** This rule applies to dense and convolutional layers. Let y be a linear function of its inputs x^i such that $y = b + \Sigma_i w_i x_i$, then $\Delta y = \Sigma_i w_i \Delta x_i$. The positive and negative parts of Δy are defined as:

$$\Delta y^+ = \Sigma_i 1\{w_i \Delta x_i > 0\} w_i \Delta x_i$$

$$\Delta y^- = \Sigma_i 1\{w_i \Delta x_i < 0\} w_i \Delta x_i$$

   This leads to the following choice for contributions:

$$\begin{aligned} C_{\Delta x_i^+ \Delta y^+} &= 1\{w_i \Delta x_i > 0\} w_i \Delta x_i^+ \\ C_{\Delta x_i^- \Delta y^+} &= 1\{w_i \Delta x_i > 0\} w_i \Delta x_i^- \\ C_{\Delta x_i^+ \Delta y^-} &= 1\{w_i \Delta x_i < 0\} w_i \Delta x_i^+ \\ C_{\Delta x_i^- \Delta y^-} &= 1\{w_i \Delta x_i < 0\} w_i \Delta x_i^- \end{aligned}$$

2. **Rescale Rule.** This rule applies to non-linear transformations. Let y be a non-linear transformation of its inputs x^i such that $y = f(x)$, then $\Delta y = C_{\Delta x \Delta y}$. The positive and negative parts of Δy are defined as:

$$\Delta y^+ = \frac{\Delta y}{\Delta x} \Delta x^+ = C_{\Delta x^+ \Delta y^+}$$

$$\Delta y^- = \frac{\Delta y}{\Delta x} \Delta x^- = C_{\Delta x^- \Delta y^-}$$

Examples of DeepLIFT explanation are shown in Fig. 2.28.

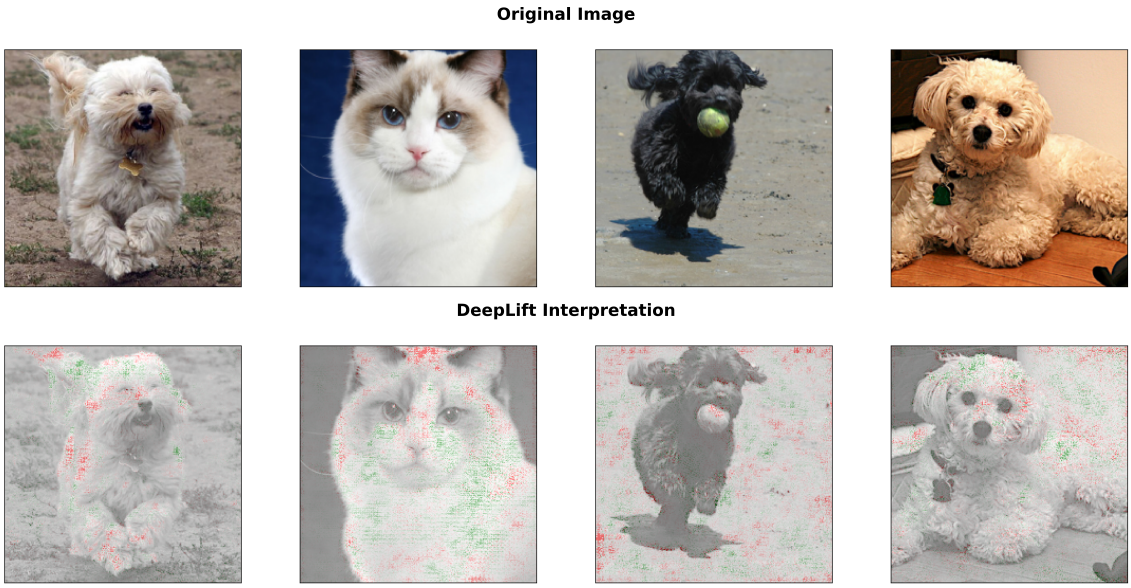

Figure 2.28: Examples of DeepLIFT explanation for a CNN classification model. Green pixels are pixels with positive influence and red ones are pixels with negative impact in predicting the label of the pictures (Dogs and Cats).

Integrated Gradients

This method was designed based on two fundamental axioms that every attribution method should satisfy [46]:

1. **Sensitivity.**: An attribution method satisfies this axiom if, for every input and baseline that differ in one feature but have different predictions, the differing feature is given a non-zero attribution. Also, if the function implemented by the deep neural network does not mathematically depend on some variable, then the attribution to that variable should always be zero. DeconvNet, Guided Backpropagation, and Saliency violate this axiom.

2. **Implementation Invariance.** The attributions for two functionally equivalent networks should always be identical. DeepLift breaks this axiom.

Given a function $F : R^n \rightarrow [0, 1]$ that represents a deep neural network, let $x \in R^n$ and $x^{'} \in R^n$ be an input and the baseline input, respectively. The Integrated Gradients method calculates the gradients at all points along the path from baseline $x^{'}$ to the input x. Integrated gradients are defined as the path integral of those gradients along the path from the baseline to the input. Specifically, the integrated gradient along the i^{th} dimension for an input x and baseline $x^{'}$ is defined as [46]:

$$IntegratedGrads_i(x) ::= (x_i - x_i^{'}) \times \int_{\alpha=0}^{1} \frac{\partial F(x^{'} + \alpha \times (x - x^{'})}{\partial x_i} d\alpha$$

The integral part of the integrated gradient is computed by the Riemman approximation of the integral [46].

Integrated Gradients satisfies a third axiom called *Completeness*. Based on this axiom, the attributions add up to the difference between the output of F at the input x and the baseline $x^{'}$, which is formalized by the following proposition [46]:

Proposition 2.7.1 *If $F : R^n \rightarrow R$ is differentiable almost everywhere, meaning it is continuous everywhere, and the partial derivative of F along each input dimension satisfies Lebesgue's integrability condition, then:*

$$\Sigma_{i=1}^{n} IntegratedGrads_i(x) = F(x) - F(x^{'})$$

Examples of Integrated Gradients explanation are shown in Fig. 2.29.

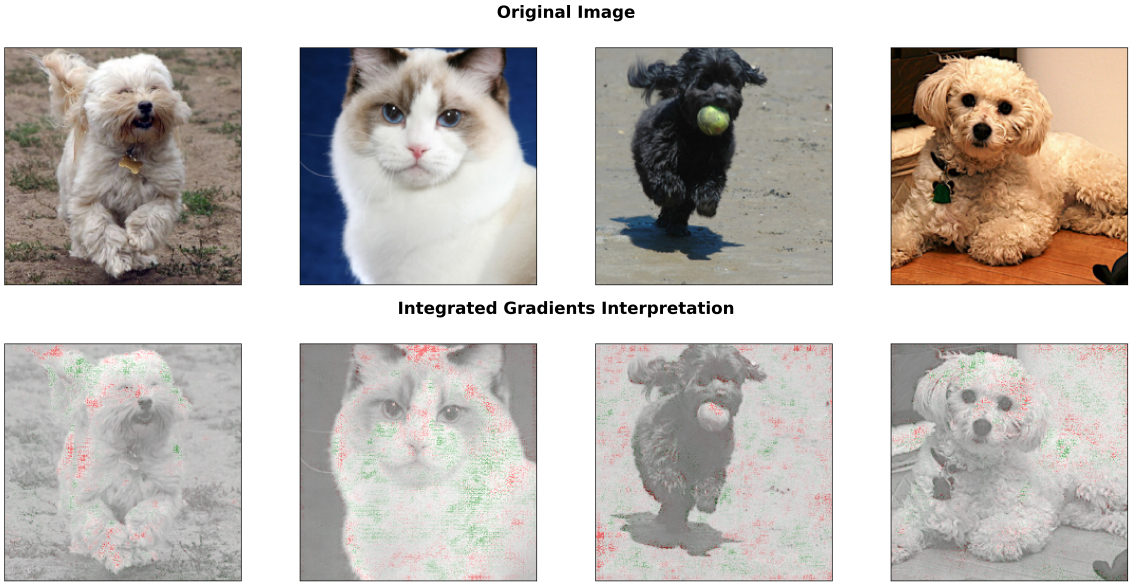

Figure 2.29: Examples of Integrated Gradients explanation for a CNN classification model. Green pixels are pixels with positive influence and red ones are pixels with negative impact in predicting the label of the pictures (Dogs and Cats).

2.7.3 Local Model-agnostic Methods

Local Interpretable Model-agnostic Explanations (LIME)

LIME locally explains the predictions of any deep neural network using an interpretable model by treating the deep neural network as a complete black box [30].
Assume that $f : R^d \rightarrow R$ is the model being explained, where $f(x)$ is the probability of x belonging to a particular class. Let $\pi_x(z)$ denote a neighbourhood radius where an instance z is defined as locally around x. Moreover, let $g \in G$ be an interpretable model from a class of interpretable models G. Finally, assume $\mathcal{L}(f, g, \pi_x)$ to be a measure of the unfaithfulness of g in approximating f in the locality defined by π_x. Then, the explanation produced by LIME is [30, 37]:

$$\xi(x) = \underset{g \in G}{argmin} \mathcal{L}(f, g, \pi_x) + \Omega(g) \tag{2.1}$$

where $\Omega(g)$ is the complexity of the interpretable model g.
The algorithm for producing the explanation is as follows [30]:

1. Get the black box prediction of the desired instance.
2. Get black box predictions for the perturbed dataset.
3. Calculate the proximity of these new samples to the instance of interest and assign weight to them based on their proximity.
4. Train a weighted interpretable model on these new samples.
5. The interpretation of the local model is the explanation for the prediction of the instance of interest.

Examples of LIME explanation are shown in Fig. 2.30. The explanations in these examples are all near zero; however, there are a few green and red pixels that are visible when zoomed.

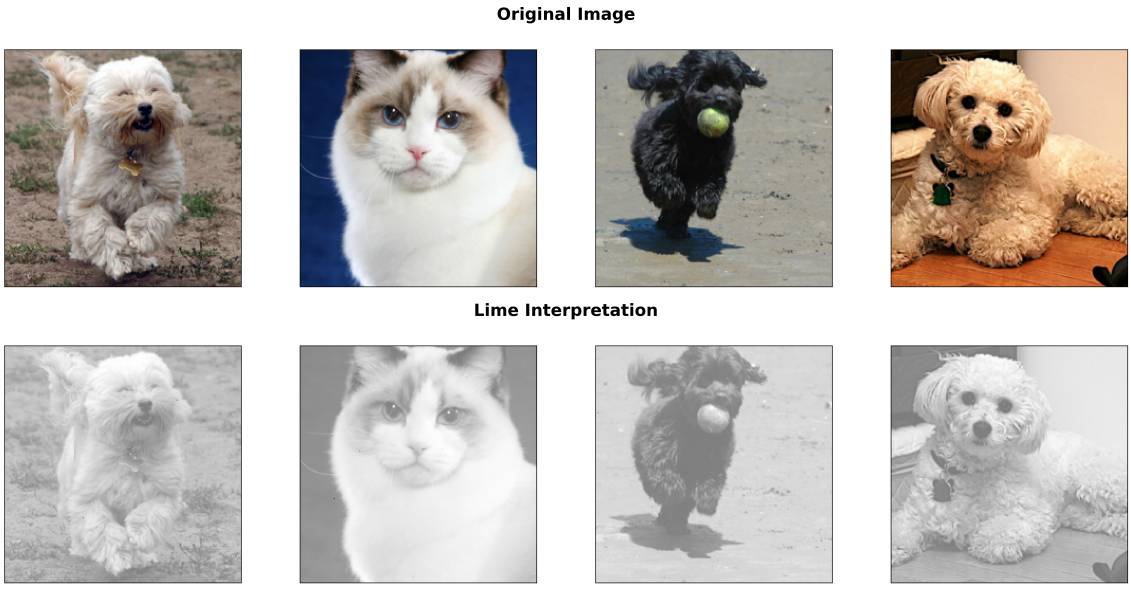

Figure 2.30: Examples of LIME explanation for a CNN classification model. Green pixels are pixels with positive influence and red ones are pixels with negative impact in predicting the label of the pictures (Dogs and Cats).

Shapley Additive Explanations (SHAP)

Lundberg and Lee (2017) [24] introduced SHAP (SHapley Additive exPlanations), a framework for explaining individual predictions of machine learning models based on Shapley values from coalitional game theory. In this context, the feature values of a specific data instance are analogous to players in a coalition. The Shapley values provide a mechanism for equitably allocating the ”payout”, which is the prediction, among the features. These players can be either single feature values or groups of feature values [30].
Let f be the black box model, g be the explanation model, and $x^{'}$ be a simplified version of x

produced using a mapping function h_x. Local methods try to ensure $g(z') \approx f(h_x(z')$ whenever $z' \approx x'$ [24].

Definition 2.7.1 (Additive feature attribution methods) *These methods have an explanation model that is a linear function of binary variables [24]:*

$$g(z') = \phi_0 + \Sigma_{i=1}^{M} \phi_i z_i'$$

where M is the number of simplified input features, $z' \in \{0,1\}^M$*, and* $\phi_i \in \mathbb{R}$*.*

Methods with explanation models satisfying the definition 2.7.1 attribute an effect ϕ to each feature, and summing the effects of all feature attributions approximates the output $f(x)$ of the original model [30].

Property 2.7.1 (Local accuracy) *The explanation model matches the original model when* $x = h_x(x')$*, where* $\phi_0 = f(h_x(0))$ *represents the model output with all simplified inputs toggled off [24].*

$$f(X) = g(x') = \phi_0 + \Sigma_{i=1}^{M} \phi_i x_i'$$

Property 2.7.2 (Missingness) *Missingness constrains features where* $x_i' = 0$ *to have no attribute impact [24].*

$$x_i' = 0 \implies \phi_i = 0$$

Property 2.7.3 (Consistency) *Let* $f_x(z') = f(h_x(z'))$ *and* z' *i denote setting* $z_i' = 0$*. for any two models f and* f' *[24]:*

$$(\forall z' \in \{0,1\}^M : f_x'(z') - f_x'(z'i) \geq f_x(z') - f_x(z'i)) \implies \phi_i(f', x) \geq \phi_i(f, x)$$

Theorem 2.7.1 *Only one possible explanation model g follows definition 2.7.1 and satisfies all properties 2.7.1 to 2.7.3:*

$$\phi_i(f, x) = \Sigma_{z' \subset x'} \frac{|z'|!(M - |z'| - 1)!}{M!} |f_x'(z') - f_x'(z'i)|$$

where $|z'|$ *is the number of non-zero entries in* z' *[24].*

KernelSHAP

KernelSHAP is an alternative kernel-based estimation approach for SHAP that combines LIME with a linear explanation model and Shapely values. Under definition 2.7.1 the specific forms of $\pi_{x'}$, $\mathcal{L}$ and Ω that make the solution of equation 2.1 consistent with all properties 2.7.1 to 2.7.3 are [24]:

$$\Omega(g) = 0$$

$$\pi_{x'z'} = \frac{M - 1}{\binom{M}{|z'|} |z'|(M - |z'|)}$$

$$\mathcal{L}(f, g, \pi_{x'}) = \Sigma_{z' \in Z} [f(h_x(z')) - g(z')]^2 \pi_{x'}(z')$$

Examples of KernelSHAP explanation are shown in Fig. 2.31.

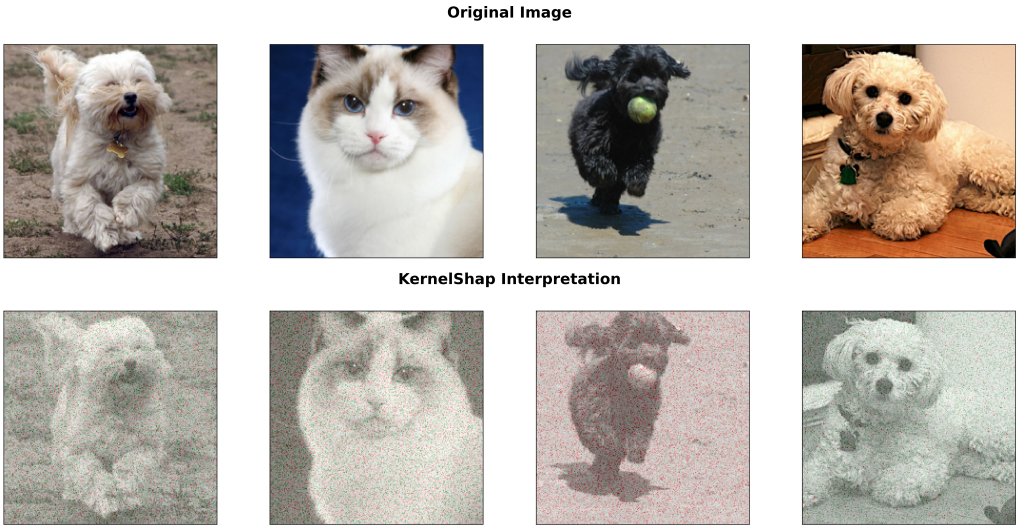

Figure 2.31: Examples of KernelSHAP explanation for a CNN classification model. Green pixels are pixels with positive influence and red ones are pixels with negative impact in predicting the label of the pictures (Dogs and Cats).

GradientSHAP

GradientSHAP, another alternative estimation approach for SHAP, assumes that the input features are independent and that the explanation model is linear, meaning that the explanations are modelled through the additive composition of feature effects. Under those assumptions, SHAP values can be approximated as the expectation of gradients [22]. Examples of GradientSHAP explanation are shown in Fig. 2.32.

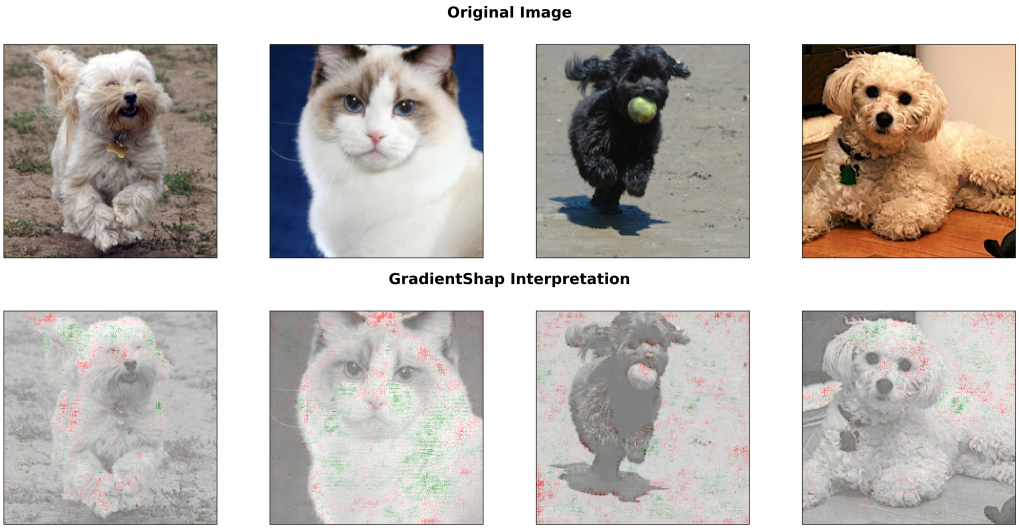

Figure 2.32: Examples of GradientSHAP explanation for a CNN classification model. Green pixels are pixels with positive influence and red ones are pixels with negative impact in predicting the label of the pictures (Dogs and Cats).

2.7.4 Evaluation

As shown in Fig. 2.33, explainability methods return highly different interpretations. They are also different in space complexity, model passes they use, number of the passes, as shown in Table. 2.2. Since the quality of explanations is crucial for understanding and trusting machine learning models, evaluating these interpretations is highly important. There are two classes of explanation evaluation measures: objective measures and subjective measures.
Subjective measures have been predominantly employed in evaluating explanations, reflecting the inherently human-centric nature of explanatory concepts.
While subjective measures are crucial, objective measures are also required to improve explanations by optimizing them. Two quantitative measures have been introduced by Yeh et al. [55]: Infidelity and Sensitivity.
The concept of infidelity assesses the quality of an explanation by evaluating how well it reflects the predictor function's response to important changes or disturbances in the input. Sensitivity is derived from local Lipschitz continuity around the input.

Definition 2.7.2 (Infidelity) *Given a black-box function f, explanation function Φ, a random variable $\mathbf{I} \in \mathbb{R}^d$ with probability measure $\mu_{\mathbf{I}}$, which represents meaningful perturbations of interest, the explanation infidelity of Φ is defined as [55]:*

$$INFD(\Phi, f, x) = \mathbb{E}_{\mathbf{I}\sim\mu_{\mathbf{I}}}[(\mathbf{I}^T\Phi(f, x) - (f(x) - f(x - \mathbf{I})))^2] \tag{2.2}$$

Definition 2.7.3 (Sensitivity) *Given a black-box function f, explanation function Φ, and a given input neighborhood radius r, the max-sensitivity for explanation is defined as[55]:*

$$SENS_{MAX}(\Phi, f, x, r) = \max_{\|y-x\|\leq r} \|\Phi(f, y) - \Phi(f, x)\|$$

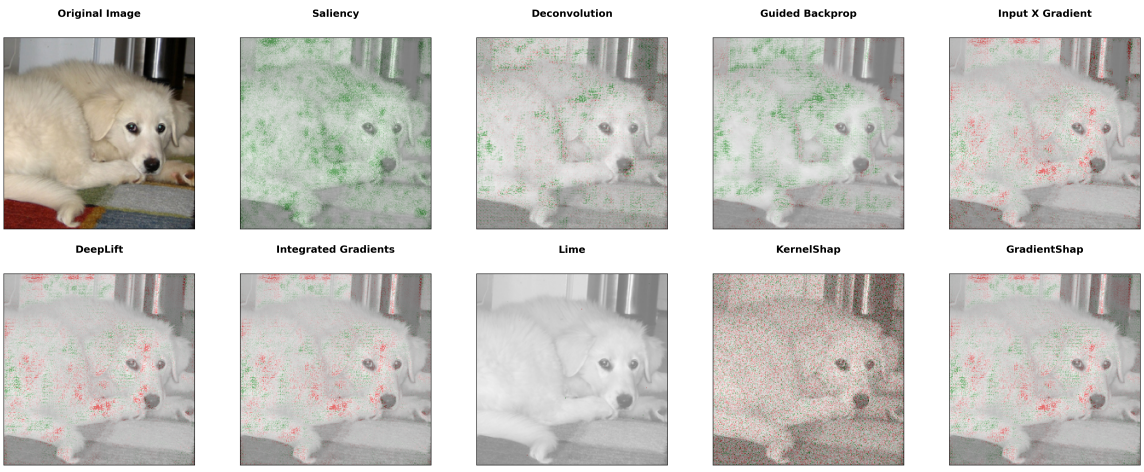

Figure 2.33: Output of explainability methods for a CNN classification model for the same image. Green pixels are pixels with positive influence and red ones are pixels with negative impact in predicting the label of the picture.

Table 2.2: Space complexity and number of passes of explainability methods.

	Space Complexity	Model Passes	Number of Passes
Saliency Map	$O(\#features)$	Forward and Backward	1
Deconvolution Network	$O(\#features)$	Forward and Backward	1
Guided Backpropagation	$O(\#features)$	Forward and Backward	1
Input X Gradient	$O(\#features)$	Forward and Backward	1
DeepLIFT	$O(\#features)$	Forward and Backward	1
Integrated Gradients	$O(\#steps \times \#features)$	Forward and Backward	$\#steps$
LIME	$O(\#samples * \#features)$	Forward	$\#samples$
KernelSHAP	$O(\#features \times \#perturbations)$	Forward	$\#features^2$
GradientSHAP	$O(\#samples \times \#features + \#baselines \times \#features)$	Forward and Backward	$\#samples$

Chapter 3

Methodology

In this chapter, we explain how we have used explainability methods on protein language models and the Seq-InSite model and how we analyzed the results.

3.1 Explainability of Protein Embeddings

A protein language model gets a protein sequence of length n and outputs a $n \times \ell$ matrix E where ℓ is the length of embedding vectors and row E_i is the embedding vector for i^{th} residue in the protein sequence. The first layer in every protein language model is an Embedding layer. This layer is a look-up table that outputs a $n \times \ell$ matrix T, and it is not trainable. Thus, it does not have a gradient, and it is not interpretable. Since this layer is set at the beginning of training and is constant during training, we can calculate the interpretation of the protein language model with respect to the output of this embedding layer.
As shown in Fig. 3.1, when we input element E_{ji} of the protein language model's output matrix to the explanation method, the output is a $n \times \ell$ matrix A, where A_{kp} shows the influence of T_{kp} on predicting the value of E_{ji}. We iterate over E to obtain the whole explanation for the sequence. The result is a 4-D matrix X^E of size $n \times \ell \times n \times \ell$. To get the effect of residue on residue, we convert this matrix to a $n \times n$ array by calculating the sum along the second and fourth dimensions as follows:

$$Exp^E_{ij} = \sum_{k=1}^{\ell} \sum_{p=1}^{\ell} X^E_{ikjp}$$

The array Exp^E measures the impact of each residue in computing the embedding vector of any other residue. The element $Exp^E[i, j]$ gives the effect of the j^{th} residue in computing the embedding vector of the i^{th} residue. Fig. 3.2 shows an example of the output of this process for a protein using ProtT5 as the embedding method and DeepLIFT as the explanation method. Exp^E is normalized so that elements are between -1 and 1.

3.2 Explainability of Interaction Prediction

Seq-InSite gets a matrix of size $(2w + 1) \times \ell$ of the embedding and outputs a single number, where w is the size of the model's window. Running explanation methods on this number, we

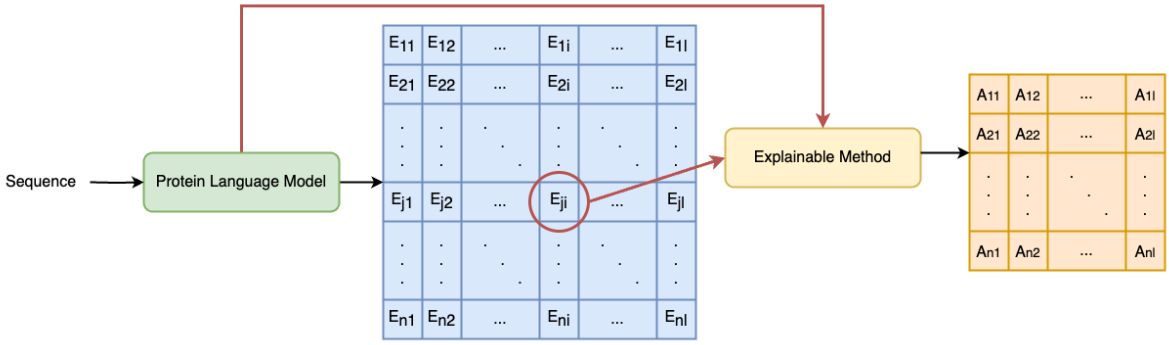

Figure 3.1: Process of calculating interpretation for an embedding element

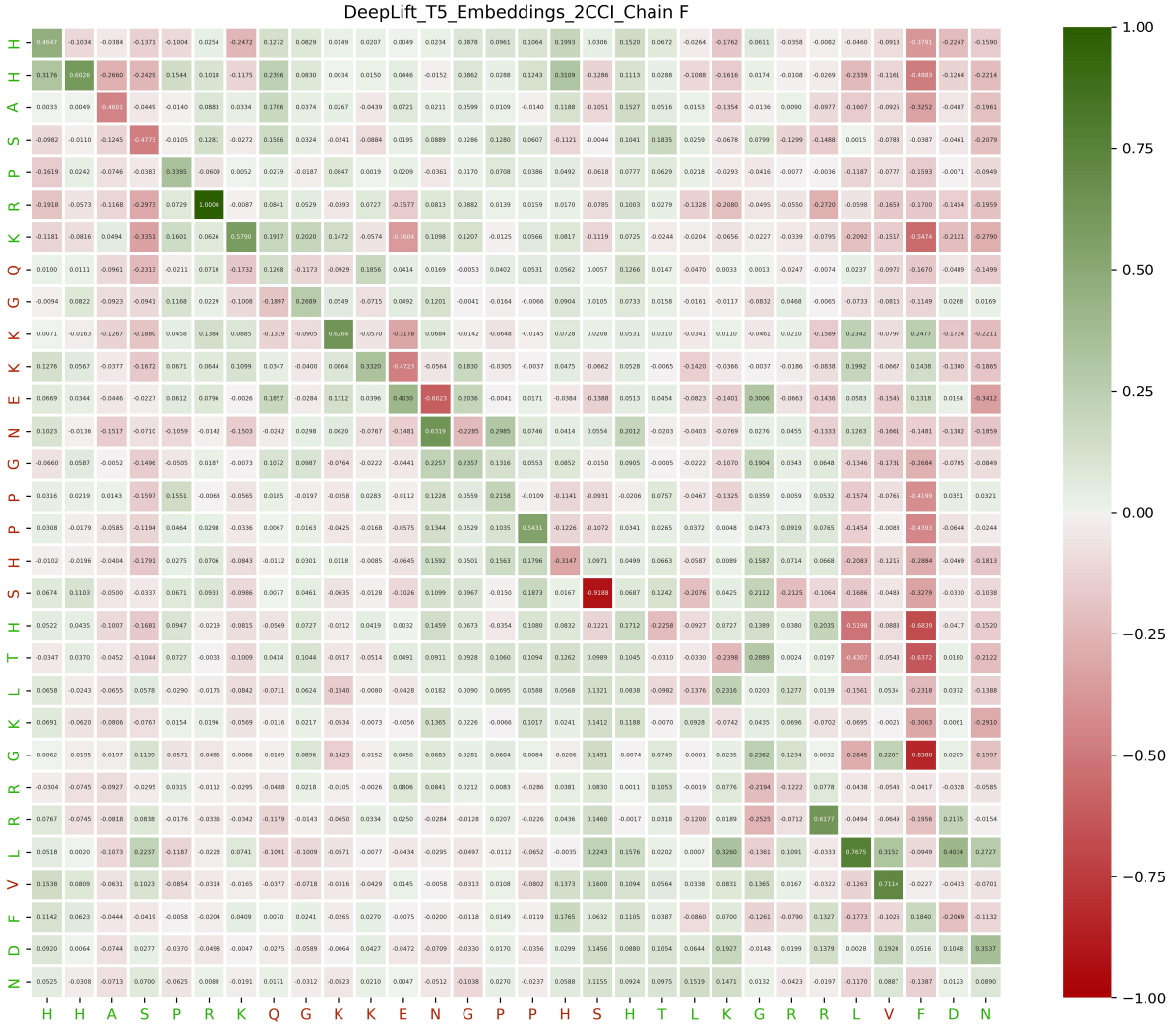

Figure 3.2: Example of embedding interpretation for a protein using ProtT5 as the embedding method and DeepLIFT as the explanation method.

get a $(2w + 1) \times \ell$ matrix B, where B_{kp} shows the impact of p^{th} element of the embedding vector for k^{th} residue in the window on the interaction prediction. If we sum over the second dimension, we will have the impact of each residue in the window on the prediction.
We iterate over all windows to get interpretations of Seq-InSite for all residues. This results in a 3-D $n \times (2 \times w + 1) \times \ell$ matrix X^S. Assuming Exp^P is a $n \times n$ matrix that shows the impact of each residue on other residues, then Exp^P is calculated as follows:

$$Exp^P_{ij} = \sum_{k=1}^{(2w+1)} \sum_{p=1}^{\ell} X^S_{ikp} \times (\sum_{r=1}^{\ell} X^E_{(i+k-w-1)pjr})$$

The element $Exp^S[i, j]$ gives the effect of the j^{th} residue in computing the interaction prediction of the i^{th} residue. Fig. 3.3 shows an example of the output of this process for a protein using ProtT5 as the embedding method and DeepLIFT as the explanation method. Exp^P is normalized so that elements are between -1 and 1.

3.3 Evaluation of Explanations

As explained in Chapter 2, we can evaluate explanations using both subjective and objective measurements. For qualitative evaluation, we defined several tests based on our expectations from models and explanation methods. We only employed Explanation Infidelity for the objective evaluation, due to computation limitations that made Explanation Sensitivity infeasible.

3.3.1 Qualitative Evaluations

Comparison with Random Matrices

To prove that resulting explanations are not random and contain information, we trained an SVM classifier using RBF kernel on 80% of the explanations and tested that on the remaining 20%. The classifier distinguishes between explanation maps and random matrices with 100% accuracy. This means that there is a hyperplane that separates these explanations from random matrices with a margin.

Distance Test

From a biological point of view, for each amino acid, those amino acids that are situated close to it within the protein sequence have more effect on its interaction than farther ones. Therefore, for each explanation method and for both embeddings and predictions, we calculated the average impact score for amino acids that have i amino acids distance between them, where $0 \leq i \leq 20$, and plot these scores by the distance. We expect that these plots have a decreasing trend.

Mann–Whitney U Test

The Mann-Whitney U-Test determines whether there is a difference between two sample groups, even when the data is not normally distributed [26]. Mann-Whitney U test is a non-parametric

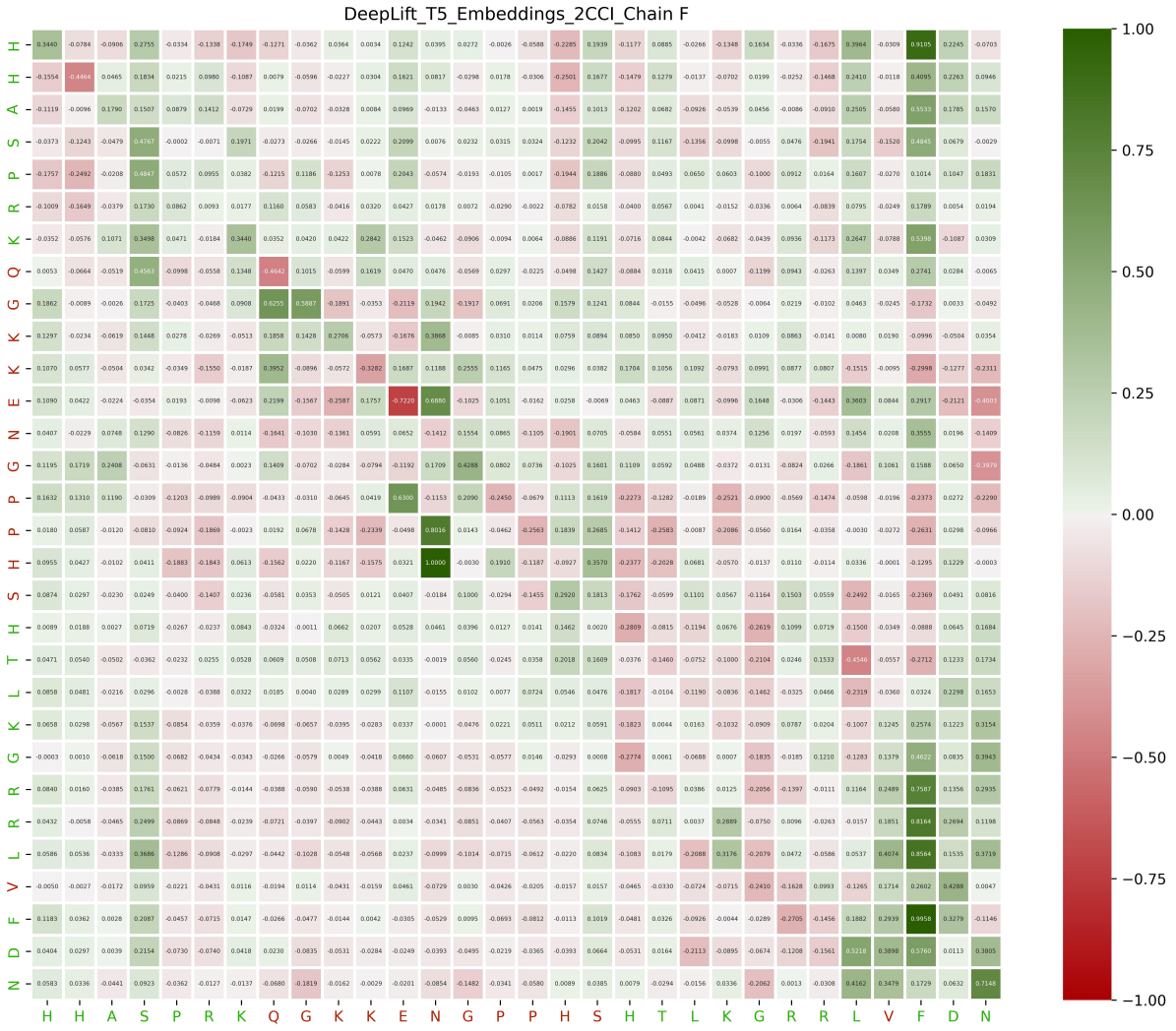

Figure 3.3: Example of interpretation for interaction predictions for a protein using ProtT5 as the embedding method and DeepLIFT as the explanation method.

statistical test on the distribution of two random variables X and Y. The hypothesis is as follows:

- Null Hypothesis: Distributions of X and Y are equal.
- Alternative Hypothesis: Distributions of X and Y are not equal.

For each explanation method, we calculated the average of impacting amino acids, called source, and being impacted amino acids, called target, scores of the following three groups for both embeddings and predictions:

1. Interacting vs. Non-interacting amino acids
2. Aromatic vs. Non-aromatic amino acids
3. Acidic vs. Basic amino acids

We then applied the Mann-Whitney U test between two groups in each of the above cases. If models have captured these properties, there will be a difference between the distributions of two populations in each group.

Kendall's τ Test

Kendall's τ test is a non-parametric test that assesses the correlation between two variables and how statistically significant it is. Assume that $(x_1, y_1), \ldots, (x_1, y_1)$ are observations of random variables X and Y. A pair (x_i, y_i) and (x_i, y_i) where $i < j$ called concordant if either both $x_i > x_j$ and $y_i > y_j$ or $x_i < x_j$ and $y_i < y_j$ holds; otherwise they are discordant. Let C be the set of concordant pairs and D be the set of discordant pairs. Then Kendall's τ coefficient correlation [20] is:

$$\tau = \frac{|C| - |D|}{|C| + |D|}$$

Kendall's τ test is a non-parametric statistical test based on the τ's value [20].

- Null Hypothesis: There is no correlation between X and Y. ($\tau = 0$)
- Alternative Hypothesis: There is a correlation between X and Y. ($\tau \neq 0$)

For each explanation method, we calculated the average of both the source and target score of each amino acid in all proteins for both embeddings and predictions, which resulted in four vectors of size 20. Then, for each of these four vectors, we used Kendall's τ test between the vector and the hydrophobicity, molecular mass, Van Der Waal volume, and dipole moment values for amino acids. We expected that there should be a correlation between them if the model captures the property.

3.3.2 Objective Evaluations

Explanation Infidelity

To get infidelity scores for protein embeddings, we have to use Eq. 2.2 on each 2-D matrix $X^E[i, j, :, :]$ for $1 \leq i \leq n, 1 \leq j \leq \ell$, which give us an infidelity score for it. This results in a $n \times \ell$ matrix I^E, where I^E_{ij} is the infidelity score for the interpretation of element E_{ij} in the embeddings. The infidelity score for the whole interpretation of embeddings is defined as the mean of elements of I^E.

$$INFD^E = \frac{\sum_{k=1}^{n} \sum_{p=1}^{\ell} I^E_{ij}}{n \times \ell}$$

To calculate infidelity scores for Seq-InSite, for each $1 \leq i \leq n$, we get infidelity score for $X^S[i, :, :]$ using 2.2. This results in a vector I^S of length n.
We denote the infidelity score for the prediction interpretations as $INFD^P$ and define it as follows:

$$S = (\sum_{i=w+1}^{n-w} \sum_{j=1}^{e} I^E_{ij} + \sum_{i=w+1}^{n-w} I^P_i) \times (2 \times w + 1) + \sum_{i=1}^{w}((\sum_{j=1}^{e} I^E_{ij} + \sum_{j=1}^{e} I^e_{(n-i+1)j} + I^p_i + I^p_{n-i+1}) \times (w + i))$$

$$INFD^P = \frac{S}{(n - 2 * w) \times (2 \times w + 1) + (\sum_{i=w+1}^{2\times w} i)} = \frac{S}{(n - 2 * w) \times (2 \times w + 1) + (w \times (3 \times w + 1))}$$

Basically, this is a weighted average of embeddings and Seq-InSite infidelity scores, where weights are the number of times each amino acid is used in predictions. For the window size of w in Seq-InSite, each amino acid is used at least $w + 1$ and at most $2w + 1$ times during prediction.

Chapter 4

Results

4.1 Experimental Details

We explain and analyze results based on the protein language model that was used for embeddings. This study investigates the comparative performance of ProtBert, ProtT5, and Ankh embeddings in protein interaction prediction using Seq-InSite. Our methodology employs state-of-the-art computational tools, including Python 3.10.0, Huggingface transformers 4.31.0 [54], PyTorch 2.0.1 [3], and Captum 0.6.0 [22], to calculate and analyze protein embeddings. All computations are done on the Graham cluster of the Digital Research Alliance of Canada using T4 GPUs. Since calculating explanations takes between 1 to 5 days (depending on the protein sequence, embedding model, and the method), we only computed them once and stored them to further be used in the tests. This has resulted in 3.3 TB of data.
We utilize a modified dataset from previous research done in Seq-InSite paper [16]. The seq-InSite paper took all proteins from the 2019 version of PiSite [14], removed all sequences with no interactions and sequences with any similarity above 25%, and split them into training (99%) and validation (1%) sets. Current implementations [22] of explanation methods require the model to be on a single GPU and do not allow for model-parallel computations. Moreover, computing embedding interpretations for one amino acid can take up to 40 minutes, depending on the method. Due to both the high space and time complexity of these methods, our analysis is limited to proteins with a maximum length of 44 amino acids. Since benchmark test datasets do not contain short proteins, we have excluded 34 proteins, as shown in Table. 4.1, from the training dataset for testing purposes.
We have performed the Distance test, seven statistical tests, and explanation infidelity, which are explained in details in Chapter 3. In all statistical tests, the P-value threshold for rejecting the null hypothesis is 0.05. Tests with rejected null hypotheses are shown in bold.
We conducted seven biology-specific statistical tests for each of the following properties, which we defined in Chapter 2, on each explanation method, considering these as subjective evaluations based on our hypotheses regarding the expected behaviour of explanations.

1. Interacting vs. non-interacting using Mann-Whitney U test
2. Aromatic vs. non-aromatic using Mann-Whitney U test
3. Acidic vs. basic using Mann-Whitney U test

4. Hydrophobicity using Kendall's τ test

5. Molecular Mass using Kendall's τ test

6. Van der Waals Volume using Kendall's τ test

7. Dipole Moment using Kendall's τ test

There are 9 tables for each embedding method: 2 tables for Mann-Whitney U tests results for embedding explanations, 2 tables for Mann-Whitney U tests results for explanations of the Seq-InSite trained on the embedding, 2 tables for Kendall's τ tests for embedding explanations, 2 tables for Kendall's τ tests results for explanations of the Seq-InSite trained on the embedding, and a table comparing results of subjective and objective evaluations.

Table 4.1: IDs and lengths of proteins in the test dataset

Protein ID	Chain	Length	Protein ID	Chain	Length
2CCI	F	30	1SGH	B	39
1MZW	B	31	6F4U	D	40
1OQE	K	31	2L9U	A	40
5KQ1	C	31	5OM2	B	40
5JPO	E	32	2XZE	R	40
2L34	A	33	4LZX	B	40
6B7G	B	33	5TUV	C	41
3MJH	B	34	2XA6	A	41
4NAW	D	34	2MOF	A	42
3DXC	B	35	2K9J	B	43
2XJY	B	35	2F9D	P	43
2BE6	D	37	4GDO	A	43
1IK9	C	37	6GNY	B	43
5XJL	M	37	6AU8	C	43
5FV8	E	38	2KS1	A	44
4UED	B	38	3HRO	A	44
5FV8	A	38	2L2T	A	44

4.2 ProtBERT

Table. 4.2 provides a view of each method's performance across ProtBERT embeddings and Seq-InSite trained on them. Methods passing more than half of the tests are highlighted in green, indicating superior performance, while those performing poorly are marked in red.

We calculated the mean explanation infidelity for each method. This objective measure complements our subjective evaluations, with the lowest infidelities (best performance) highlighted in green and the highest in red.

For ProtBERT, our analysis reveals that Saliency Map, Integrated Gradients, and KernelSHAP emerge as the top-performing methods based on statistical tests. However, an interesting discrepancy emerges when considering explanation infidelity. While Integrated Gradients and KernelSHAP maintain strong performance, the Saliency Map shows relatively high explanation infidelity, ranking second worst for ProtBERT explanations.

The interpretability of Seq-InSite trained on ProtBERT embeddings shows notable similarities to ProtBERT embeddings' interpretations. Like the embeddings' interpretations, Saliency Map, Integrated Gradients, and KernelSHAP are the best performers in statistical tests for Seq-InSite trained on these embeddings. There is a similar situation with infidelity scores, where the Saliency Map shows higher infidelity for Seq-InSite explanations than the other two methods.

In the case of both ProtBERT and Seq-InSite trained on ProtBERT, LIME and Deconvolution Network are the worst methods based on statistical tests and infidelity, respectively.

These findings demonstrate significant consistency between the interpretability of ProtBERT embeddings and Seq-InSite trained on these embeddings, suggesting a transfer of interpretability features.

Delving deeper into specific biological properties, our analysis of Tables. 4.3, 4.4, 4.5, and 4.6 reveals that ProtBERT captures all tested properties to some degree, with at least one explanation method passing each statistical test, except for Kendall's τ test for Van der Waals volume. However, we infer from Table. 4.10 that this property is indeed learned by the model but missed by the explanation methods since this test is passed by one method. This conclusion is based on the fact that Seq-InSite's only input about the protein comes through the embeddings, and one of its explanations passes Kendall's τ test for Van der Waals volume, which means that this information is in the embeddings. Tables. 4.7, 4.8, 4.9, and 4.10 show that at least one of the interpretations of Seq-InSite trained on ProtBERT passes each of the seven statistical tests. This further shows the consistency between the interpretability of ProtBERT embeddings and Seq-InSite trained on these embeddings.

Fig. 4.2 and Fig. 4.3 show examples of interpretations of ProtBERT and Seq-InSite trained on ProtBERT embeddings for each method. Fig. 4.1 shows the results of the distance test for them. We can see that KernelSHAP and Saliency Map have almost flat graphs and are able to capture the importance scores even for distant amino acids, while for methods like GradientSHAP, the graph immediately drops after the amino acid itself (distance = 0).

Table 4.2: Comparison of a number of passed statistical tests (Mann-Whiteney U tests and Kendall's τ tests) and explanation infidelity scores for ProtBERT embeddings and predictions interpretations. The best methods are shown in green, and the worst method is shown in red.

	Number of Passed Statistical Tests		Explanation Infidelity	
	ProtBERT Embeddings	Seq-InSite trained on ProtBERT Embeddings	ProtBERT Embeddings	Seq-InSite trained on ProtBERT Embeddings
Saliency Map	**8**	**7**	6.981E-08	5.027E-05
Deconvolution Network	3	4	**7.031E-08**	**5.311E-05**
Guided Backpropagation	3	3	6.949E-08	4.931E-05
Input X Gradient	2	4	5.151E-08	3.733E-05
DeepLIFT	3	3	4.608E-08	3.316E-05
Integrated Gradients	**7**	**9**	**4.273E-08**	3.215E-05
LIME	**0**	**2**	4.400E-08	3.216E-05
KernelSHAP	**7**	**7**	4.468E-08	3.246E-05
GradientSHAP	2	4	4.555E-08	**3.032E-05**

Table 4.3: Resulted P-Values of Mann-Whitney U test on target scores of ProtBERT embeddings interpretations.

	Interacting vs. Non-Interacting	Aromatic vs. Non-Aromatic	Acidic vs. Basic
Saliency	**6.09E-28**	**1.87E-82**	**8.98E-31**
Deconvolution Network	5.05E-01	2.83E-01	2.79E-01
Guided Backpropagation	5.05E-01	2.83E-01	2.79E-01
Input X Gradient	6.16E-02	6.45E-01	6.98E-01
DeepLIFT	4.56E-01	3.48E-01	**1.47E-02**
Integrated Gradients	5.09E-01	9.09E-02	**1.20E-08**
LIME	3.82E-01	5.47E-01	3.93E-01
KernelShap	**0.00E+00**	**3.74E-126**	**2.41E-20**
GradientShap	2.79E-01	9.32E-01	**8.37E-06**

Table 4.4: Resulted P-Values of Mann-Whitney U test on source scores of ProtBERT embeddings interpretations.

	Interacting vs. Non-Interacting	Aromatic vs. Non-Aromatic	Acidic vs. Basic
Saliency	**1.88E-21**	**1.21E-124**	**7.03E-22**
Deconvolution Network	**4.71E-02**	**1.53E-04**	2.55E-01
Guided Backpropagation	**4.71E-02**	**1.53E-04**	2.55E-01
Input X Gradient	**2.41E-02**	**1.86E-02**	1.60E-01
DeepLIFT	3.28E-01	**1.62E-03**	**2.15E-02**
Integrated Gradients	**2.28E-19**	**1.09E-05**	**2.92E-19**
LIME	6.28E-01	3.55E-01	2.24E-01
KernelShap	**3.95E-33**	**7.09E-17**	**1.01E-12**
GradientShap	6.14E-01	**2.35E-02**	**7.30E-01**

Table 4.5: Resulted correlation and P-Values of Kendall's τ test on target scores of ProtBERT embeddings interpretations.

	Hydrophobicity		Molecular Mass		Van Der Waals Volume		DipoleMoment	
	Correlation	PValue	Correlation	PValue	Correlation	PValue	Correlation	PValue
Saliency	0.2995	0.0680	-0.2540	0.1190	-0.0213	0.8965	**-0.4421**	**0.0059**
Deconvolution Network	-0.0642	0.6957	**0.3915**	**0.0162**	0.2873	0.0790	0.0947	0.5859
Guided Backpropagation	-0.0642	0.6957	**0.3915**	**0.0162**	0.2873	0.0790	0.0947	0.5859
Input X Gradient	0.2246	0.1710	0.0741	0.6493	0.1383	0.3978	-0.1474	0.3859
DeepLIFT	0.1818	0.2678	-0.2011	0.2171	-0.1702	0.2980	-0.2842	0.0855
Integrated Gradients	**-0.4279**	**0.0091**	**0.3915**	**0.0162**	0.2660	0.1039	**0.5263**	**0.0008**
LIME	0.2567	0.1177	-0.1799	0.2695	-0.2873	0.0790	-0.2000	0.2333
KernelShap	-0.2032	0.2155	0.1164	0.4749	0.1064	0.5154	0.1895	0.2598
GradientShap	-0.1711	0.2970	0.2434	0.1352	0.1915	0.2416	0.2211	0.1859

Table 4.6: Resulted correlation and P-Values of Kendall's τ test on source scores of ProtBERT embeddings interpretations.

	Hydrophobicity		Molecular Mass		Van Der Waals Volume		DipoleMoment	
	Correlation	PValue	Correlation	PValue	Correlation	PValue	Correlation	PValue
Saliency	0.2781	0.0901	-0.2540	0.1190	-0.0426	0.7947	**-0.3789**	**0.0198**
Deconvolution Network	0.1177	0.4733	-0.0106	0.9482	-0.1064	0.5154	0.1368	0.4223
Guided Backpropagation	0.1177	0.4733	-0.0106	0.9482	-0.1064	0.5154	-0.1368	0.4223
Input X Gradient	0.0000	1.0000	-0.0317	0.8455	-0.1064	0.5154	-0.0632	0.7246
DeepLIFT	0.1711	0.2970	-0.2011	0.2171	-0.1809	0.2688	-0.1789	0.2884
Integrated Gradients	-0.1818	0.2678	-0.0847	0.6033	-0.2660	0.1039	0.2316	0.1650
LIME	-0.0856	0.6020	0.0423	0.7950	0.1489	0.3625	0.1158	0.5006
KernelShap	0.2888	0.0784	-0.1693	0.2987	0.0638	0.6963	**-0.3789**	**0.0198**
GradientShap	-0.0107	0.9480	-0.1693	0.2987	-0.1064	0.5154	0.1263	0.4605

Table 4.7: Resulted P-Values of Mann-Whitney U test on target scores for interpretations of Seq-InSite trained on ProtBERT embeddings.

	Interacting vs. Non-Interacting	Aromatic vs. Non-Aromatic	Acidic vs. Basic
Saliency	**6.20E-52**	**1.04E-170**	**4.58E-38**
Deconvolution Network	**1.65E-02**	**1.29E-06**	2.63E-01
Guided Backpropagation	**6.56E-03**	**9.32E-01**	4.12E-01
Input X Gradient	9.01E-01	8.19E-01	7.53E-01
DeepLIFT	**7.11E-13**	4.81E-01	3.41E-01
Integrated Gradients	**6.86E-108**	4.81E-01	**4.23E-20**
LIME	2.06E-01	7.92E-01	1.67E-01
KernelShap	**3.96E-236**	**4.68E-157**	**1.12E-20**
GradientShap	2.92E-01	**2.87E-02**	**4.41E-05**

Table 4.8: Resulted P-Values of Mann-Whitney U test on source scores for interpretations of Seq-InSite trained on ProtBERT embeddings.

	Interacting vs. Non-Interacting	Aromatic vs. Non-Aromatic	Acidic vs. Basic
Saliency	7.43E-02	**3.49E-219**	**1.90E-32**
Deconvolution Network	7.41E-01	**6.66E-09**	**8.29E-09**
Guided Backpropagation	2.53E-01	**3.52E-03**	3.53E-01
Input X Gradient	2.88E-01	**1.46E-02**	4.90E-01
DeepLIFT	**6.81E-06**	**4.63E-04**	7.06E-02
Integrated Gradients	**1.51E-02**	**9.82E-07**	**2.76E-07**
LIME	4.45E-01	3.82E-01	1.91E-01
KernelShap	**2.49E-29**	**2.97E-12**	**1.34E-07**
GradientShap	9.66E-01	**2.82E-02**	8.91E-01

Table 4.9: Resulted correlation and P-Values of Kendall's τ test on target scores for interpretations of Seq-InSite trained on ProtBERT embeddings.

	Hydrophobicity		Molecular Mass		Van Der Waals Volume		DipoleMoment	
	Correlation	PValue	Correlation	PValue	Correlation	PValue	Correlation	PValue
Saliency	0.3102	0.0587	-0.2116	0.1939	0.0000	1.0000	**-0.4316**	**0.0073**
Deconvolution Network	-0.0535	0.7445	0.0635	0.6967	-0.1489	0.3625	0.1474	0.3859
Guided Backpropagation	-0.2995	0.0680	0.0423	0.7950	-0.1383	0.3978	**0.4421**	**0.0059**
Input X Gradient	**0.4386**	**0.0075**	-0.0847	0.6033	-0.0532	0.7450	**-0.4316**	**0.0073**
DeepLIFT	0.2032	0.2155	-0.2434	0.1352	-0.1596	0.3292	-0.2632	0.1126
Integrated Gradients	-0.1498	0.3614	0.1905	0.2423	-0.0426	0.7947	**0.3368**	**0.0398**
LIME	**0.5027**	**0.0022**	-0.2011	0.2171	-0.1702	0.2980	**-0.4526**	**0.0047**
KernelShap	0.1391	0.3968	-0.0847	0.6033	0.0000	1.0000	-0.2947	0.0740
GradientShap	0.0856	0.6020	-0.2434	0.1352	-0.0745	0.6489	-0.0842	0.6308

Table 4.10: Resulted correlations and P-Values of Kendall's τ test on source scores for interpretations of Seq-InSite trained on ProtBERT embeddings.

	Hydrophobicity		Molecular Mass		Van Der Waals Volume		DipoleMoment	
	Correlation	PValue	Correlation	PValue	Correlation	PValue	Correlation	PValue
Saliency	0.2674	0.1032	-0.2646	0.1044	-0.0532	0.7450	**-0.3895**	**0.0164**
Deconvolution Network	0.0107	0.9480	0.1270	0.4357	0.0319	0.8453	0.0211	0.9235
Guided Backpropagation	-0.1177	0.4733	0.0106	0.9482	0.0106	0.9481	0.0737	0.6771
Input X Gradient	-0.0963	0.5574	-0.2540	0.1190	**-0.3830**	**0.0192**	-0.0316	0.8728
DeepLIFT	0.2781	0.0901	-0.2540	0.1190	-0.2873	0.0790	-0.2842	0.0855
Integrated Gradients	**-0.5241**	**0.0014**	**0.3598**	**0.0272**	0.2128	0.1933	**0.4947**	**0.0018**
LIME	-0.2781	0.0901	-0.1270	0.4357	-0.1702	0.2980	0.1368	0.4223
KernelShap	0.2888	0.0784	-0.1799	0.2695	0.0532	0.7450	**-0.3895**	**0.0164**
GradientShap	0.3209	0.0505	-0.1905	0.2423	0.0106	0.9481	**-0.5053**	**0.0014**

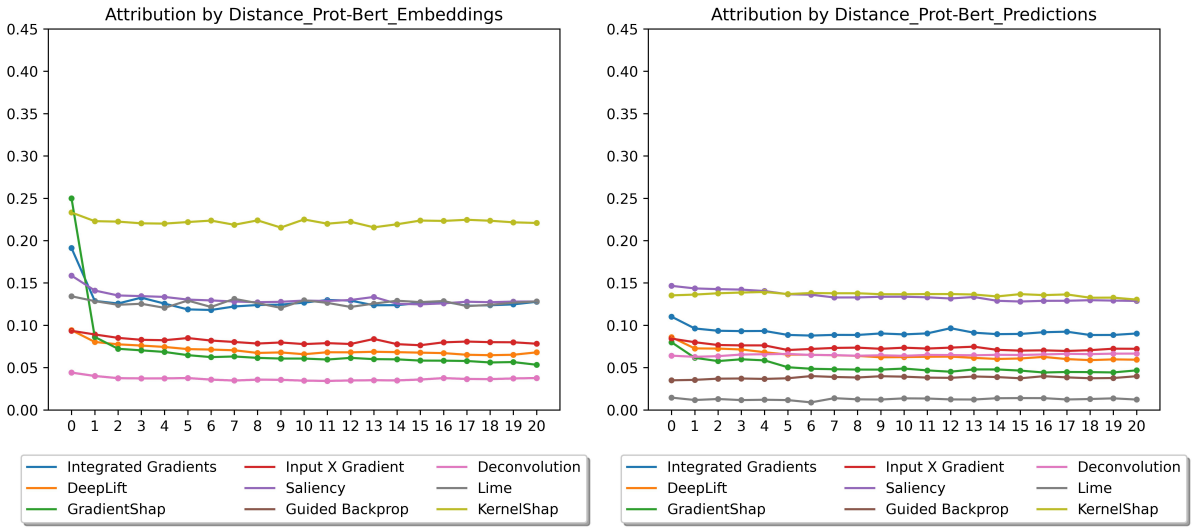

Figure 4.1: Left: Results of distance test for ProtBERT embeddings interpretations, Right: Results of distance test for interpretations of Seq-InSite trained on ProtT5 embeddings.

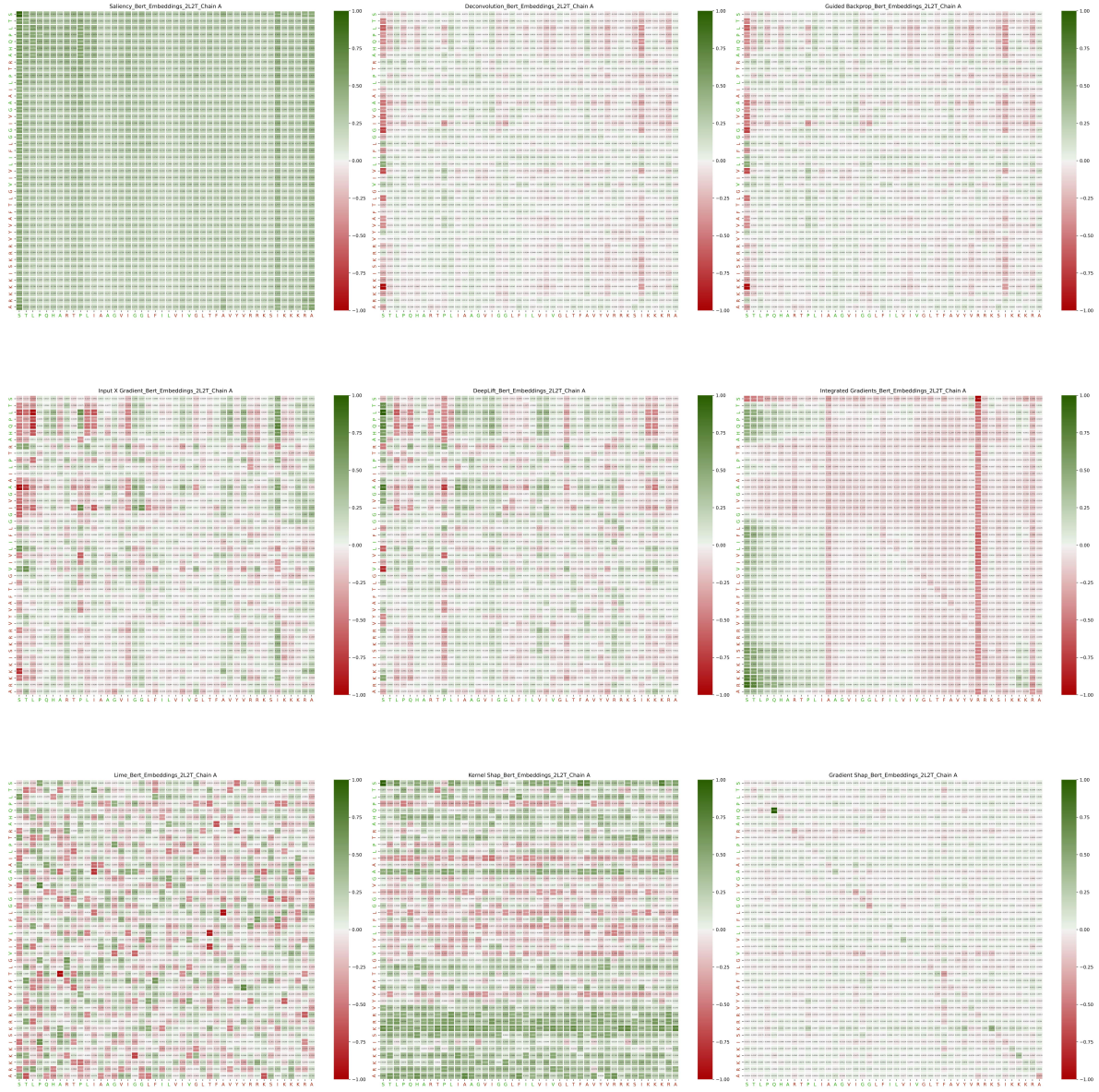

Figure 4.2: Examples of ProtBERT embeddings interpretations for 2L2T protein chain A.

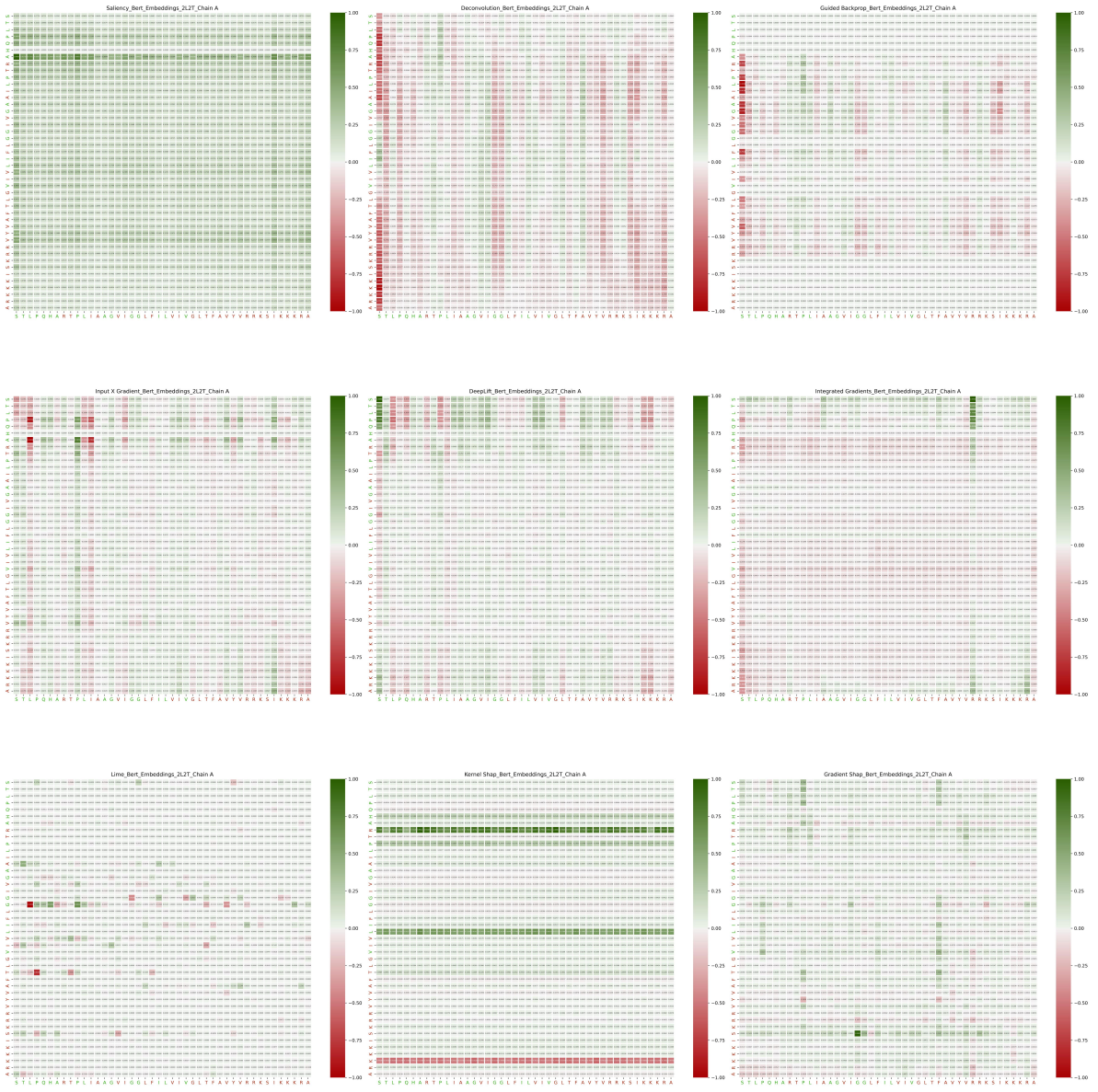

Figure 4.3: Examples of interpretations of Seq-InSite trained on ProtBERT embeddings for 2L2T protein chain A.

4.3 ProtT5

Fig. 4.6 and Fig. 4.5 illustrate interpretations of ProtT5 and Seq-InSite (trained on ProtT5 embeddings) for various methods. Fig. 4.4 presents the distance test results. KernelSHAP and Deconvolution Network demonstrate consistently high importance scores across amino acid distances. In contrast, methods like GradientSHAP, Integrated Gradients, and DeepLIFT show a sharp decline in importance scores for amino acids beyond the target position (distance > 0).

An overview of each method's performance on ProtT5 embeddings and Seq-InSite models trained on them is shown in Table. 4.11.

Our analysis reveals that ProtT5, Saliency Map, Deconvolution Network, Guided Backpropagation, and KernelSHAP are the top-performing methods based on statistical tests. However, while the Saliency Map performs well based on the infidelity score, KernelSHAP shows relatively high explanation infidelity. This gets even more interesting as the Deconvolution Network and Guided Backpropagation are the worst methods based on the infidelity score.

The interpretability of Seq-InSite models trained on ProtT5 embeddings exhibits significant similarities to the interpretations of the ProtT5 embeddings themselves. Like the embeddings' interpretations, Deconvolution Network, Guided Backpropagation, and KernelSHAP are the best performers in statistical tests for Seq-InSite models trained on these embeddings. A similar pattern is observed with infidelity scores, where the Deconvolution Network and Guided Backpropagation have the two highest infidelity for Seq-InSite explanations compared to the other methods.

In a deeper analysis of specific biological properties, Tables. 4.12, 4.13, 4.14, and 4.15 show that ProtT5 captures all tested properties to some extent, with at least one explanation method passing each statistical test. However, Tables. 4.18 and 4.19 shows that Seq-InSite trained on ProtT5 embeddings does not pass Kendall's τ test for Van der Waals volume. Since we know that this information was learned by the embedding model, failing these tests shows that explanation methods fail to capture this information in the Seq-InSite model. Tables. 4.16, 4.17, 4.18, and 4.19 also highlights the consistency between the interpretability of ProtT5 embeddings and Seq-InSite models trained on these embeddings.

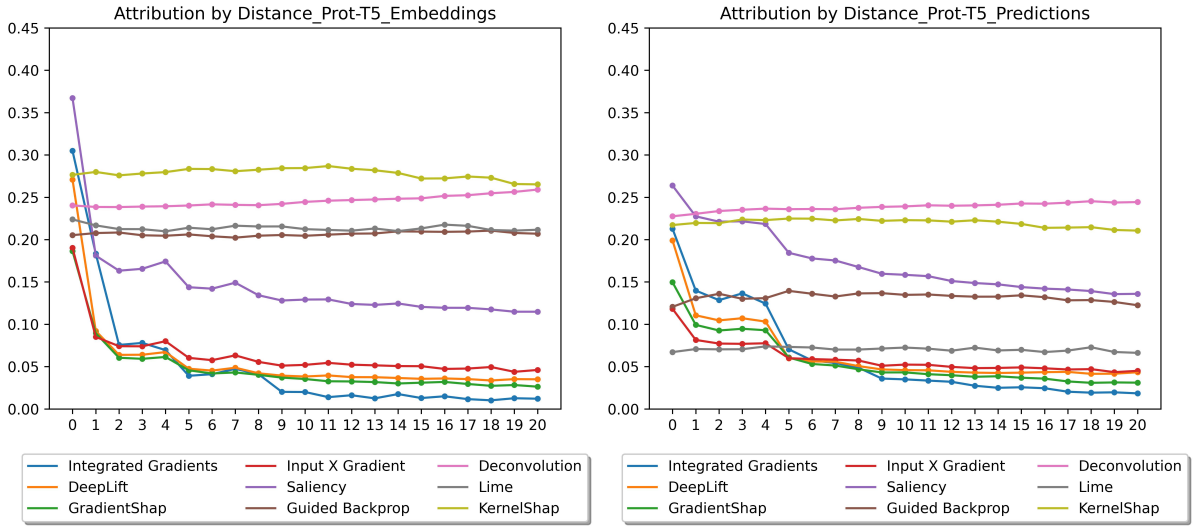

Figure 4.4: Left: Results of distance test for ProtT5 embeddings interpretations, Right: Results of distance test for interpretations of Seq-InSite trained on ProtT5 embeddings.

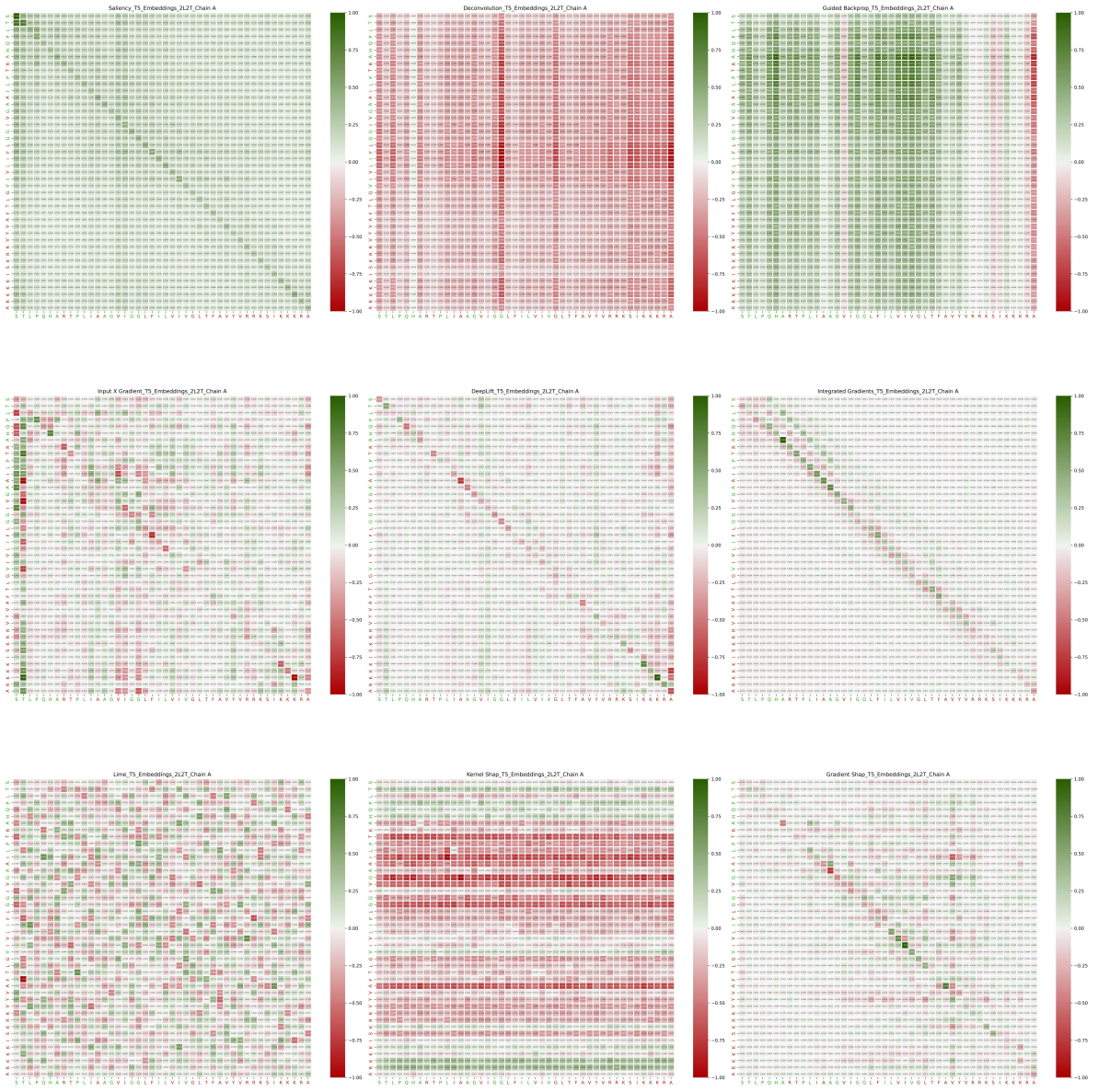

Figure 4.5: Examples of ProtT5 embeddings interpretations for 2L2T protein chain A.

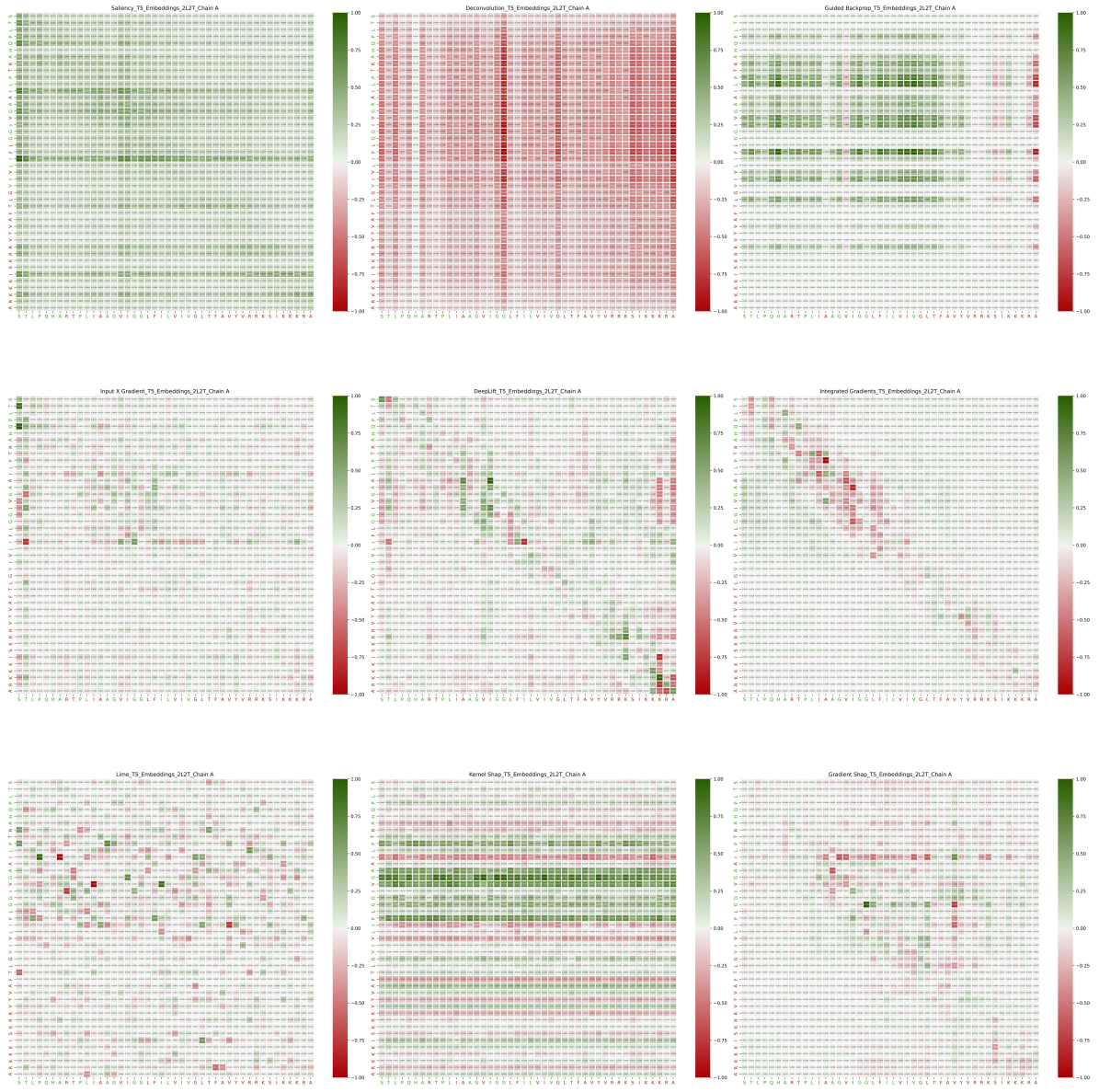

Figure 4.6: Examples of interpretations of Seq-InSite trained on ProtT5 embeddings for 2L2T protein chain A.

Table 4.11: Comparison of a number of passed statistical tests (Mann-Whiteney U tests and Kendall's τ tests) and explanation infidelity scores for ProtT5 embeddings and predictions interpretations. The best methods are shown in green, and the worst method is shown in red.

	Number of Passed Statistical Tests		Explanation Infidelity	
	ProtT5 Embeddings	Seq-InSite trained on ProtT5 Embeddings	ProtT5 Embeddings	Seq-InSite trained on ProtT5 Embeddings
Saliency Map	**8**	6	5.76E-09	5.495E-06
Deconvolution Network	**7**	**7**	**INF**	**INF**
Guided Backpropagation	**8**	**9**	1.14E+01	1.102E+04
Input X Gradient	1	3	6.49E-09	4.750E-06
DeepLIFT	4	4	8.42E-08	7.901E-05
Integrated Gradients	2	3	**4.095E-09**	**3.500E-06**
LIME	**0**	2	7.64E-09	6.98E-06
KernelSHAP	**10**	**9**	2.39E-07	3.51E-06
GradientSHAP	1	**0**	4.52E-08	4.209E-05

Table 4.12: Resulted P-Values of Mann-Whitney U test on target score of ProtT5 embeddings interpretations.

	Interacting vs. Non-Interacting	Aromatic vs. Non-Aromatic	Acidic vs. Basic
Saliency	**2.58E-09**	**5.67E-08**	8.32E-02
Deconvolution Network	**4.53E-15**	7.06E-01	**2.79E-02**
Guided Backpropagation	**5.25E-06**	9.70E-01	**7.20E-60**
Input X Gradient	6.85E-02	7.12E-01	4.75E-01
DeepLIFT	5.95E-01	3.73E-01	5.07E-01
Integrated Gradients	2.19E-01	8.96E-02	8.56E-01
LIME	7.64E-01	1.58E-01	7.80E-01
KernelShap	**1.17E-02**	**3.79E-67**	**5.89E-63**
GradientShap	9.30E-02	7.30E-02	8.92E-01

Table 4.13: Resulted P-Values of Mann-Whitney U test on source scores of ProtT5 embeddings interpretations.

	Interacting vs. Non-Interacting	Aromatic vs. Non-Aromatic	Acidic vs. Basic
Saliency	**2.49E-152**	**1.19E-02**	**8.39E-61**
Deconvolution Network	**3.12E-02**	**4.96E-45**	**0.00E+00**
Guided Backpropagation	**3.57E-14**	**1.35E-35**	**9.16E-139**
Input X Gradient	3.25E-01	9.18E-01	3.60E-01
DeepLIFT	**3.82E-22**	**8.98E-30**	**1.41E-21**
Integrated Gradients	**9.56E-32**	**4.23E-04**	3.23E-01
LIME	9.03E-01	4.18E-01	5.31E-01
KernelShap	**2.84E-13**	**3.61E-05**	**6.75E-03**
GradientShap	7.21E-02	1.23E-01	4.41E-01

Table 4.14: Resulted correlations and P-Values of Kendall's τ test on target scores of ProtT5 embeddings interpretations.

	Hydrophobicity		Molecular Mass		Van Der Waals Volume		DipoleMoment	
	Correlation	PValue	Correlation	PValue	Correlation	PValue	Correlation	PValue
Saliency	0.2460	0.1338	-0.1799	0.2695	0.0319	0.8453	**-0.3684**	**0.0237**
Deconvolution Network	-0.2781	0.0901	0.1693	0.2987	-0.0638	0.6963	**0.4000**	**0.0135**
Guided Backpropagation	0.2567	0.1177	-0.2328	0.1530	-0.0319	0.8453	**-0.3789**	**0.0198**
Input X Gradient	-0.0107	0.9480	-0.2751	0.0912	**-0.3830**	**0.0192**	-0.1158	0.5006
DeepLIFT	-0.2139	0.1923	0.2011	0.2171	0.0532	0.7450	0.2737	0.0983
Integrated Gradients	0.2139	0.1923	-0.0212	0.8966	-0.1702	0.2980	-0.2105	0.2086
LIME	-0.0642	0.6957	0.0635	0.6967	0.0745	0.6489	0.1474	0.3859
KernelShap	**-0.4920**	**0.0027**	**0.3704**	**0.0230**	0.2234	0.1719	**0.6947**	**0.0000**
GradientShap	-0.0107	0.9480	-0.1058	0.5160	-0.2766	0.0908	-0.0526	0.7732

Table 4.15: Resulted correlations and P-Values of Kendall's τ test on source scores of ProtT5 embeddings interpretations.

	Hydrophobicity		Molecular Mass		Van Der Waals Volume		DipoleMoment	
	Correlation	PValue	Correlation	PValue	Correlation	PValue	Correlation	PValue
Saliency	**0.3637**	**0.0267**	-0.2963	0.0689	-0.0638	0.6963	**-0.4632**	**0.0038**
Deconvolution Network	-0.1711	0.2970	**0.3386**	**0.0377**	0.1064	0.5154	0.2947	0.0740
Guided Backpropagation	**0.3423**	**0.0370**	-0.1058	0.5160	0.0213	0.8965	**-0.3895**	**0.0164**
Input X Gradient	-0.1070	0.5145	-0.0635	0.6967	-0.1064	0.5154	0.0842	0.6308
DeepLIFT	0.0000	1.0000	**0.3810**	**0.0194**	0.2766	0.0908	0.0737	0.6771
Integrated Gradients	0.1818	0.2678	0.0212	0.8966	0.0638	0.6963	-0.0632	0.7246
LIME	0.1925	0.2407	0.0635	0.6967	0.0745	0.6489	-0.0947	0.5859
KernelShap	-0.2674	0.1032	0.1799	0.2695	-0.0532	0.7450	**0.3895**	**0.0164**
GradientShap	-0.1711	0.2970	0.1799	0.2695	0.0319	0.8453	**0.3263**	**0.0468**

Table 4.16: Resulted P-Values of Mann-Whitney U test on target scores for interpretations of Seq-InSite trained on ProtT5 embeddings.

	Interacting vs. Non-Interacting	Aromatic vs. Non-Aromatic	Acidic vs. Basic
Saliency	5.61E-02	**1.29E-33**	2.52E-01
Deconvolution Network	**8.16E-69**	**5.95E-12**	6.02E-01
Guided Backpropagation	**5.26E-94**	7.31E-02	**8.84E-102**
Input X Gradient	1.44E-01	4.54E-01	7.71E-01
DeepLIFT	1.12E-01	1.94E-01	2.13E-01
Integrated Gradients	**1.09E-02**	8.95E-01	9.01E-01
LIME	8.42E-01	1.42E-01	7.55E-01
KernelShap	**1.62E-93**	**3.90E-07**	**6.22E-21**
GradientShap	5.14E-01	7.91E-01	2.08E-01

Table 4.17: Resulted P-Values of Mann-Whitney U test on source scores for interpretations of Seq-InSite trained on ProtT5 embeddings.

	Interacting vs. Non-Interacting	Aromatic vs. Non-Aromatic	Acidic vs. Basic
Saliency	**4.48E-132**	6.03E-01	**3.49E-127**
Deconvolution Network	**1.71E-14**	**9.20E-62**	0.00E+00
Guided Backpropagation	**9.26E-12**	**1.17E-61**	**9.02E-276**
Input X Gradient	9.88E-01	6.27E-01	7.30E-01
DeepLIFT	**2.27E-19**	**1.67E-22**	**2.37E-18**
Integrated Gradients	**4.77E-28**	8.94E-01	9.82E-01
LIME	6.80E-01	6.38E-01	7.21E-01
KernelShap	**3.40E-06**	**2.35E-03**	**1.77E-03**
GradientShap	7.09E-02	3.88E-01	9.76E-01

Table 4.18: Resulted correlations and P-Values of Kendall's τ test on target scores of interpretations of Seq-InSite trained on ProtT5 embeddings.

	Hydrophobicity		Molecular Mass		Van Der Waals Volume		DipoleMoment	
	Correlation	PValue	Correlation	PValue	Correlation	PValue	Correlation	PValue
Saliency	0.2139	0.1923	-0.2222	0.1725	-0.0106	0.9481	**-0.3684**	**0.0237**
Deconvolution Network	-0.2995	0.0680	0.2116	0.1939	0.0106	0.9481	**0.4421**	**0.0059**
Guided Backpropagation	**0.3958**	**0.0159**	-0.1587	0.3299	0.0106	0.9481	**-0.4632**	**0.0038**
Input X Gradient	0.2139	0.1923	-0.1693	0.2987	-0.1064	0.5154	**-0.3263**	**0.0468**
DeepLIFT	**0.3423**	**0.0370**	-0.0952	0.5588	0.0532	0.7450	-0.1895	0.2598
Integrated Gradients	-0.1818	0.2678	**0.3492**	**0.0321**	0.1596	0.3292	0.2632	0.1126
LIME	**-0.3851**	**0.0189**	0.2540	0.1190	0.1489	0.3625	**0.3368**	**0.0398**
KernelShap	0.2567	0.1177	**-0.3915**	**0.0162**	-0.2873	0.0790	**-0.4211**	**0.0091**
GradientShap	-0.0642	0.6957	0.2540	0.1190	0.0319	0.8453	0.0947	0.5859

Table 4.19: Resulted correlations and P-Values of Kendall's τ test on source scores of interpretations of Seq-InSite trained on ProtT5 embeddings.

	Hydrophobicity		Molecular Mass		Van Der Waals Volume		DipoleMoment	
	Correlation	PValue	Correlation	PValue	Correlation	PValue	Correlation	PValue
Saliency	**0.3637**	**0.0267**	-0.2963	0.0689	-0.0638	0.6963	**-0.4632**	**0.0038**
Deconvolution Network	-0.1711	0.2970	**0.3386**	**0.0377**	0.1064	0.5154	0.2947	0.0740
Guided Backpropagation	**0.3423**	**0.0370**	-0.1058	0.5160	0.0213	0.8965	**-0.3895**	**0.0164**
Input X Gradient	**0.4493**	**0.0062**	-0.2646	0.1044	-0.1702	0.2980	**-0.5789**	**0.0002**
DeepLIFT	0.0214	0.8963	-0.0212	0.8966	0.0532	0.7450	0.0000	1.0000
Integrated Gradients	-0.1177	0.4733	-0.1270	0.4357	-0.2553	0.1185	0.0842	0.6308
LIME	0.0000	1.0000	0.0423	0.7950	0.0213	0.8965	-0.1789	0.2884
KernelShap	0.2781	0.0901	-0.2222	0.1725	0.0319	0.8453	**-0.3684**	**0.0237**
GradientShap	-0.0428	0.7943	0.0635	0.6967	0.0213	0.8965	0.1158	0.5006

4.4 Ankh

Table. 4.20 provides a thorough outline of each explanation method's performance across both models.

For Ankh embeddings, our analysis identifies Integrated Gradients and KernelSHAP as top performers in statistical tests. Unlike ProtBERT and ProtT5, both leading performers in statistical tests also uphold strong performance in infidelity scores.

The interpretability of Seq-InSite models trained on Ankh embeddings shows remarkable differences from the Ankh embeddings' interpretations. Saliency Map and Guided Backpropagation emerge as the best performers in statistical tests for the trained Seq-InSite model.

Our in-depth analysis of specific biological properties is presented in Tables. 4.21, 4.22, 4.23, and 4.24, indicates that Ankh captures all tested properties to some degree, with at least one explanation method passing each statistical test, except for Kendall's τ test for molecular mass. Table. 4.27 indicates that the molecular mass property, while not consistently detected in Ankh embeddings by explanation methods, is indeed learned by the model. This conclusion is supported by Seq-InSite's ability to pass Kendall's τ test for this property, demonstrating that the information is encoded within the embeddings. These findings contribute to our understanding of the relationship between embedding model interpretability and the interpretability of downstream prediction models in protein interaction prediction. Tables 4.25, 4.26, 4.27, and 4.28 highlight the overall consistency in interpretability between Ankh embeddings and the trained Seq-InSite model.

Fig. 4.8 and Fig. 4.9 illustrate examples of interpretations from Ankh and Seq-InSite trained on Ankh embeddings for each method. Fig. 4.7 presents the results of the distance test for these methods. It is evident that KernelSHAP produces almost flat graphs and is capable of capturing importance scores even for distant amino acids. In contrast, methods like GradientSHAP, Deconvolution Network, and DeepLIFT show a steep drop in the graph immediately after the amino acid itself (distance = 0).

Table 4.20: Comparison of a number of passed statistical tests (Mann-Whiteney U tests and Kendall's τ tests) and explanation infidelity scores for Ankh embeddings and predictions interpretations. The best methods are shown in green, and the worst method is shown in red.

	Number of Passed Statistical Tests		Explanation Infidelity	
	Ankh Embeddings	Seq-InSite trained on Ankh Embeddings	Ankh Embeddings	Seq-InSite trained on Ankh Embeddings
Saliency Map	5	**7**	2.054E-11	3.577E-06
Deconvolution Network	6	6	2.033E-11	**6.172E-05**
Guided Backpropagation	6	**7**	2.021E-11	**1.626E-06**
Input X Gradient	4	**3**	4.811E-10	2.086E-06
DeepLIFT	3	4	**5.801E-10**	2.075E-06
Integrated Gradients	**7**	**3**	**1.358E-11**	1.694E-06
LIME	**1**	4	4.342E-11	1.769E-06
KernelSHAP	**7**	5	1.609E-11	1.740E-06
GradientSHAP	3	4	3.105E-10	2.025E-06

Table 4.21: Resulted P-Values of Mann-Whitney U test on target scores of Ankh embeddings interpretations.

	Interacting vs. Non-Interacting	Aromatic vs. Non-Aromatic	Acidic vs. Basic
Saliency	**5.59E-164**	**2.64E-05**	7.14E-02
Deconvolution Network	**5.00E-03**	8.79E-01	**6.77E-16**
Guided Backpropagation	**5.00E-03**	8.79E-01	**6.77E-16**
Input X Gradient	6.45E-01	**3.69E-03**	**6.14E-03**
DeepLIFT	**5.75E-08**	2.57E-01	3.58E-01
Integrated Gradients	2.50E-01	**3.29E-02**	**7.26E-08**
LIME	2.18E-01	2.40E-01	3.04E-01
KernelShap	**1.78E-21**	**3.01E-03**	**0.00E+00**
GradientShap	1.51E-01	9.62E-01	3.46E-01

Table 4.22: Resulted P-Values of Mann-Whitney U test on source scores of Ankh embeddings interpretations.

	Interacting vs. Non-Interacting	Aromatic vs. Non-Aromatic	Acidic vs. Basic
Saliency	**9.23E-101**	4.12E-01	1.81E-01
Deconvolution Network	**5.95E-22**	7.93E-01	**5.57E-07**
Guided Backpropagation	**5.95E-22**	7.93E-01	**5.57E-07**
Input X Gradient	3.01E-01	**5.36E-05**	2.88E-01
DeepLIFT	1.45E-01	**2.65E-03**	**2.80E-02**
Integrated Gradients	**1.94E-27**	**2.51E-87**	**2.12E-288**
LIME	**4.60E-02**	9.19E-02	6.05E-01
KernelShap	**1.61E-04**	**1.54E-05**	**1.04E-11**
GradientShap	**5.43E-03**	**5.03E-03**	**2.45E-05**

Table 4.23: Resulted correlations and P-Values of Kendall's τ test on target scores of Ankh embeddings interpretations.

	Hydrophobicity		Molecular Mass		Van Der Waals Volume		DipoleMoment	
	Correlation	PValue	Correlation	PValue	Correlation	PValue	Correlation	PValue
Saliency	0.2781	0.0901	-0.2116	0.1939	0.0000	1.0000	**-0.3579**	**0.0283**
Deconvolution Network	**-0.3530**	**0.0315**	0.0423	0.7950	-0.0851	0.6028	**0.3474**	**0.0336**
Guided Backpropagation	**-0.3530**	**0.0315**	0.0423	0.7950	-0.0851	0.6028	**0.3474**	**0.0336**
Input X Gradient	0.1604	0.3282	-0.1164	0.4749	-0.0106	0.9481	-0.0947	0.5859
DeepLIFT	0.2995	0.0680	-0.2540	0.1190	-0.0319	0.8453	-0.2947	0.0740
Integrated Gradients	-0.2888	0.0784	0.0212	0.8966	-0.1489	0.3625	**0.3263**	**0.0468**
LIME	-0.2139	0.1923	0.1270	0.4357	0.0426	0.7947	0.0632	0.7246
KernelShap	0.0107	0.9480	-0.1270	0.4357	-0.0213	0.8965	-0.1684	0.3189
GradientShap	-0.2032	0.2155	0.0635	0.6967	-0.0213	0.8965	0.1368	0.4223

Table 4.24: Resulted correlations and P-Values from Kendall's τ test on source scores of Ankh embeddings interpretations.

	Hydrophobicity		Molecular Mass		Van Der Waals Volume		DipoleMoment	
	Correlation	PValue	Correlation	PValue	Correlation	PValue	Correlation	PValue
Saliency	0.2246	0.1710	-0.3069	0.0596	-0.0958	0.5583	**-0.3263**	**0.0468**
Deconvolution Network	-0.1818	0.2678	0.0741	0.6493	0.0106	0.9481	0.2947	0.0740
Guided Backpropagation	-0.1818	0.2678	0.0741	0.6493	0.0106	0.9481	0.2947	0.0740
Input X Gradient	0.0214	0.8963	0.2328	0.1530	**0.3617**	**0.0270**	0.0000	1.0000
DeepLIFT	0.0749	0.6482	0.0212	0.8966	0.1809	0.2688	-0.1789	0.2884
Integrated Gradients	-0.2460	0.1338	-0.1587	0.3299	**-0.3298**	**0.0437**	0.1789	0.2884
LIME	0.1284	0.4341	-0.1693	0.2987	-0.1702	0.2980	-0.1158	0.5006
KernelShap	0.2460	0.1338	-0.2116	0.1939	0.0426	0.7947	**-0.3579**	**0.0283**
GradientShap	-0.2353	0.1515	-0.1799	0.2695	-0.2979	0.0686	0.2000	0.2333

Table 4.25: Resulted P-Values of Mann-Whitney U test on target scores of interpretations of Seq-InSite trained on Ankh embeddings.

	Interacting vs. Non-Interacting	Aromatic vs. Non-Aromatic	Acidic vs. Basic
Saliency	**0.00E+00**	**1.70E-29**	**8.82E-07**
Deconvolution Network	**3.97E-14**	**4.89E-05**	**6.77E-08**
Guided Backpropagation	**5.68E-06**	9.42E-02	**1.99E-06**
Input X Gradient	6.42E-01	6.72E-02	**1.75E-04**
DeepLIFT	**1.39E-09**	**1.93E-02**	**8.31E-07**
Integrated Gradients	1.19E-01	9.03E-01	**9.37E-03**
LIME	2.02E-01	2.04E-01	1.02E-01
KernelShap	**7.78E-11**	7.89E-01	**1.31E-213**
GradientShap	1.17E-01	8.57E-02	5.02E-01

Table 4.26: Resulted P-Values of Mann-Whitney U test on source scores of interpretations of Seq-InSite trained on Ankh embeddings.

	Interacting vs. Non-Interacting	Aromatic vs. Non-Aromatic	Acidic vs. Basic
Saliency	**2.22E-154**	1.48E-01	**7.43E-08**
Deconvolution Network	**1.59E-15**	**1.38E-12**	5.22E-01
Guided Backpropagation	**1.38E-51**	**4.74E-02**	**2.40E-04**
Input X Gradient	**4.05E-08**	**1.51E-07**	3.39E-01
DeepLIFT	7.14E-02	**8.33E-07**	9.57E-01
Integrated Gradients	**3.78E-15**	**4.92E-62**	1.30E-01
LIME	7.91E-02	9.86E-02	5.28E-01
KernelShap	**1.47E-02**	**3.41E-03**	**7.55E-09**
GradientShap	**3.98E-04**	**2.62E-09**	**4.35E-02**

Table 4.27: Resulted correlations and P-Values from Kendall's τ test on target scores for interpretations of Seq-InSite trained on Ankh embeddings.

	Hydrophobicity		Molecular Mass		Van Der Waals Volume		DipoleMoment	
	Correlation	PValue	Correlation	PValue	Correlation	PValue	Correlation	PValue
Saliency	0.2567	0.1177	-0.2116	0.1939	-0.0106	0.9481	**-0.4000**	**0.0135**
Deconvolution Network	-0.2567	0.1177	0.2011	0.2171	0.0532	0.7450	**0.3579**	**0.0283**
Guided Backpropagation	-0.2460	0.1338	0.1587	0.3299	-0.0851	0.6028	**0.3895**	**0.0164**
Input X Gradient	0.2032	0.2155	0.2011	0.2171	0.1915	0.2416	-0.0526	0.7732
DeepLIFT	0.0749	0.6482	-0.0741	0.6493	-0.1489	0.3625	0.0737	0.6771
Integrated Gradients	0.2888	0.0784	-0.0635	0.6967	-0.0638	0.6963	-0.2526	0.1284
LIME	**0.3851**	**0.0189**	**-0.4550**	**0.0052**	**-0.3830**	**0.0192**	**-0.4421**	**0.0059**
KernelShap	0.2567	0.1177	-0.0847	0.6033	-0.1809	0.2688	-0.2526	0.1284
GradientShap	0.1177	0.4733	0.1693	0.2987	0.1489	0.3625	0.0421	0.8227

Table 4.28: Resulted correlations and P-Values of Kendall's τ test on source scores for interpretations of Seq-InSite trained on Ankh embeddings.

	Hydrophobicity		Molecular Mass		Van Der Waals Volume		DipoleMoment	
	Correlation	PValue	Correlation	PValue	Correlation	PValue	Correlation	PValue
Saliency	0.2567	0.1177	-0.3175	0.0513	-0.1064	0.5154	**-0.3579**	**0.0283**
Deconvolution Network	-0.1925	0.2407	0.0529	0.7453	-0.0106	0.9481	0.2737	0.0983
Guided Backpropagation	**-0.4386**	**0.0075**	0.1587	0.3299	0.0319	0.8453	**0.5053**	**0.0014**
DeepLIFT	-0.0749	0.6482	-0.0529	0.7453	-0.2234	0.1719	0.1158	0.5006
Integrated Gradients	0.0214	0.8963	0.0847	0.6033	0.2341	0.1524	0.0000	1.0000
LIME	0.0428	0.7943	-0.1270	0.4357	-0.0426	0.7947	0.0316	0.8728
KernelShap	-0.0749	0.6482	0.0847	0.6033	-0.1277	0.4350	0.1368	0.4223
GradientShap	**0.4172**	**0.0110**	0.0423	0.7950	0.0745	0.6489	-0.1895	0.2598
Input X Gradient	-0.1604	0.3282	-0.0635	0.6967	-0.2128	0.1933	0.2421	0.1458

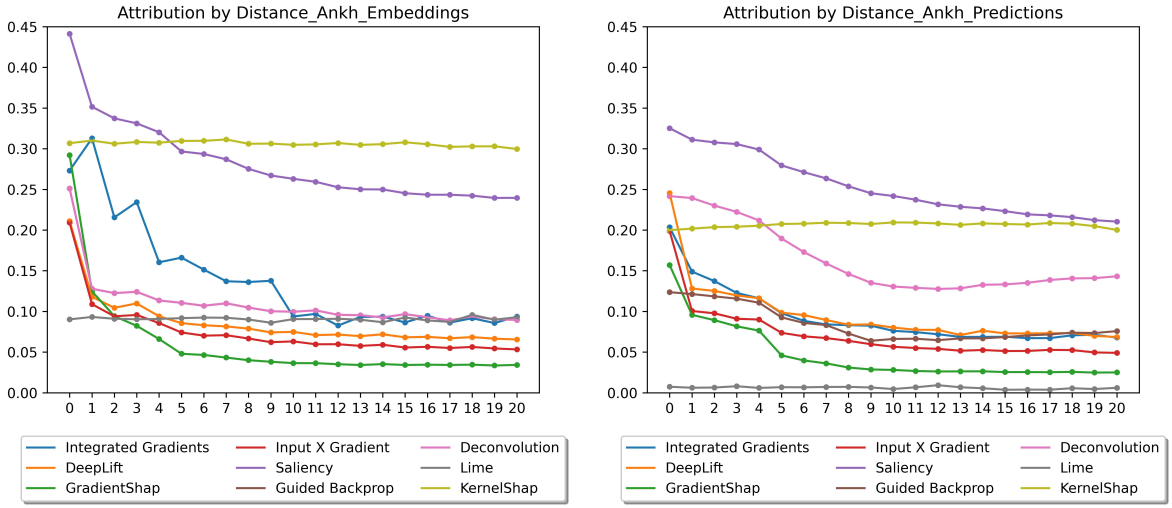

Figure 4.7: Left: Results of distance test for Ankh embeddings interpretations, Right: Results of distance test for interpretations of predictions using Ankh embeddings.

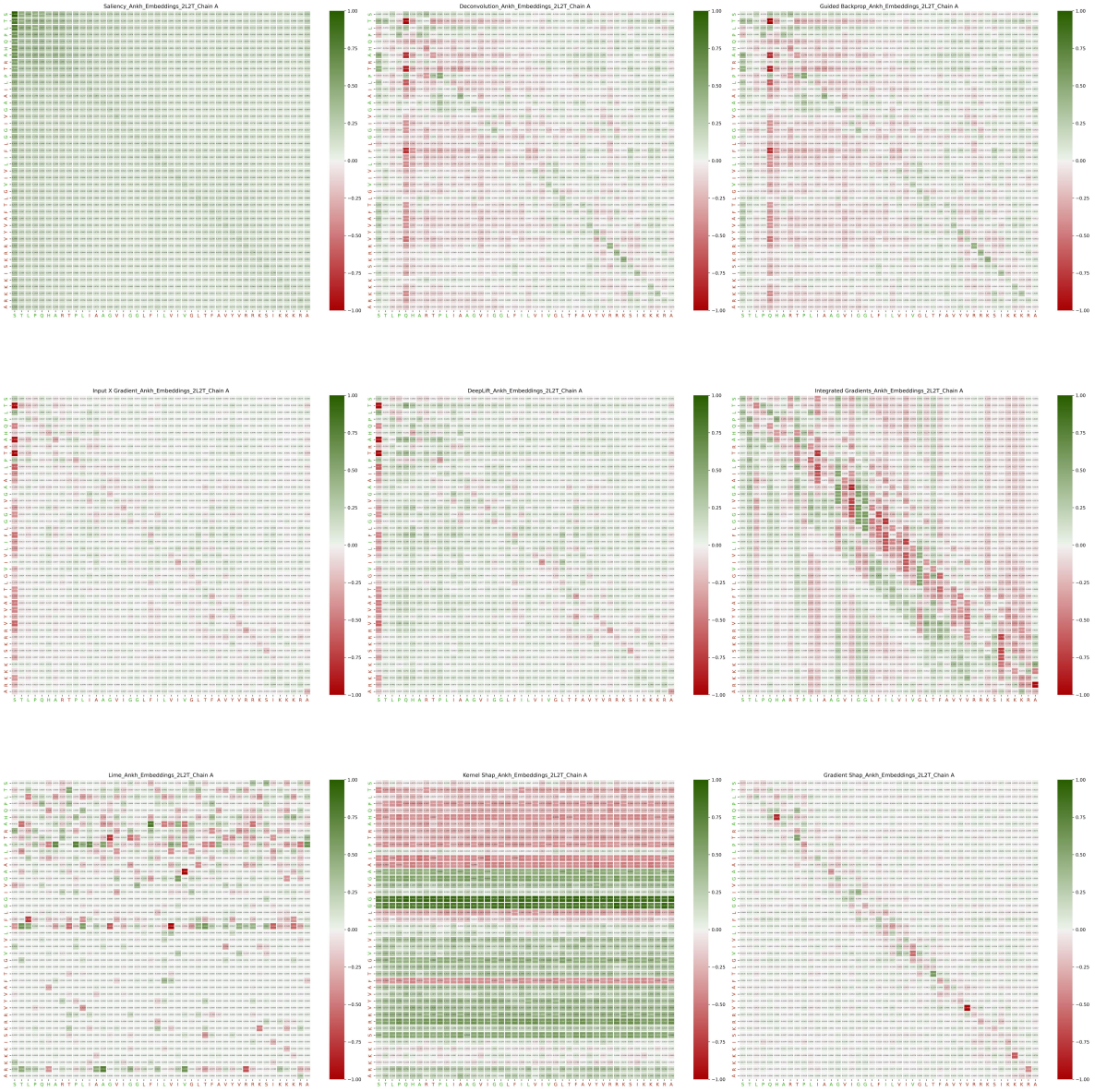

Figure 4.8: Examples of Ankh embeddings interpretations for 2L2T protein chain A.

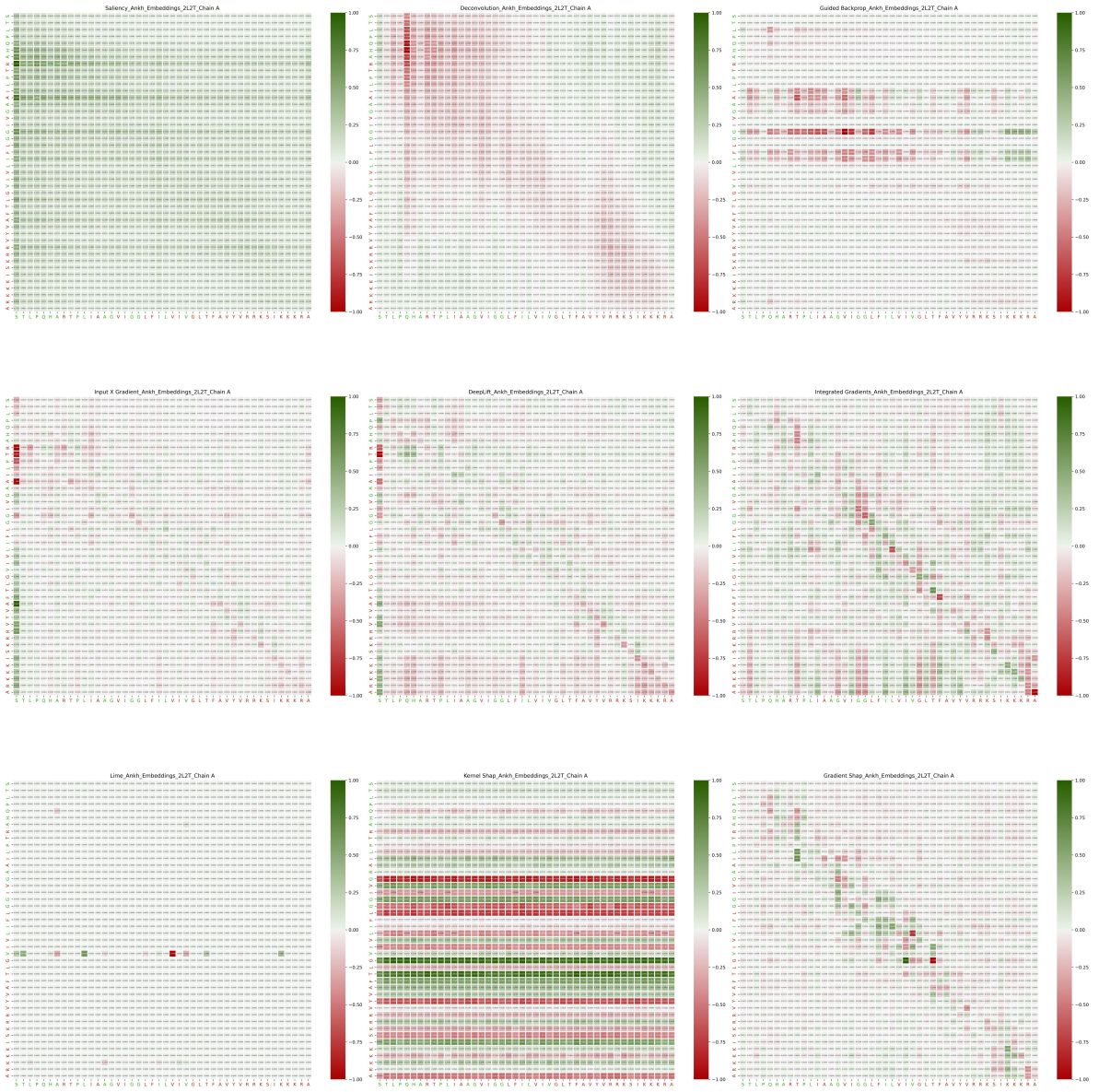

Figure 4.9: Examples of interpretations of predictions using Ankh embeddings for 2L2T protein chain A.

4.5 Discussion

Our results show that no single method consistently performs best across all models and metrics, and some methods, like Deconvolution Networks and Integrated Gradients, show high variability in performance across different models. Table. 4.29 shows the total number of passed statistical tests and mean infidelity score across all embedding models. We can see that the Deconvolution Network and Guided Backpropagation perform poorly in terms of explana-

tion infidelity despite passing many of the statistical tests. Meanwhile, despite low infidelity scores, LIME and GradientSHAP perform poorly in statistical tests. KernelSHAP and Saliency Map perform well in both statistical tests and infidelity scores.
In conclusion, choosing the best explanation method may depend on the specific protein analysis model and the preferred metric (statistical tests vs. infidelity). Some methods, like KernelSHAP and Saliency Map, show promise across multiple models and metrics. Our thorough analysis indicates very clearly that a lot of further work is needed in this area.

Table 4.29: Comparison of total number of passed statistical tests (Mann-Whiteney U tests and Kendall's τ tests) and mean explanation infidelity scores.

	Total Number of Passed Statistical Tests	Mean Explanation Infidelity
Saliency Map	41	9.903E-06
Deconvolution Network	33	INF
Guided Backpropagation	36	1.839E+03
Input X Gradients	17	7.371E-06
DeepLIFT	21	1.906E-05
Integrated Gradients	31	6.232E-06
LIME	9	6.827E-06
KernelSHAP	45	6.332E-06
GradientSHAP	14	1.242E-05

Chapter 5

Conclusion and Future Work

In this thesis, we have employed existing explanation methods on protein language models and protein interaction site prediction model, aiming to demystify their opaque, "black-box" nature. To the best of our knowledge, this integration of explainable AI techniques within these specific domains represents a pioneering approach in the existing scientific literature. We have not only utilized these methods but have also developed and introduced subjective biological-specific evaluation metrics tailored to assess the quality of the interpretation maps generated by these models.

Our evaluations have further augmented the interpretability, facilitating a clearer understanding for users to trace biological properties models have learned. Researchers and practitioners can now trace the specific biological properties and patterns that these models learn and consider significant, effectively linking high-level quantitative data analysis with biological insights. This bridging of the gap is crucial for advancing the application of machine learning in bioinformatics, providing a foundation upon which more intuitive and accessible analytical tools can be developed.

Moreover, through detailed experiments and thorough analyses, this work has not only demonstrated the current capabilities of explainable AI in protein modelling but has also illuminated potential areas for further enhancement. Our findings suggest several avenues for future research, including the refinement of explanation methods to capture more nuanced biological phenomena and the integration of these methods into a broader array of bioinformatics applications. Additionally, this thesis underscores the need for more sophisticated interpretative tools that can cater to the complex nature of protein interactions and the dynamic environments in which they function.

The promising results of this thesis pave the way for several avenues of future research in explainable AI for protein analysis:

- Further work could extend the applications of explainable AI to other areas of bioinformatics, especially downstream tasks of protein language models. This area could benefit significantly from enhanced model transparency.

- Explainable AI methods have a lot of room for improvement. As we show in the results, these methods fail to capture the full interpretation of models in many cases. Future research could focus on developing new explainable AI algorithms that provide even more precise and granular explanations.

- As mentioned in Chapter 3, current explainable AI techniques have many computational limitations that prevent them from being employed properly on large models like protein language models. Developing more efficient explainable AI methods is another important direction for future research.

By addressing these areas, future research can continue to advance the field of explainable AI in different areas, including protein modelling, ensuring that these sophisticated tools provide not only high accuracy but also the transparency needed for their widespread adoption in scientific research and clinical decision-making.

Bibliography

[1] Lizabeth A. Allison. *Fundamental molecular biology*. Wiley, Hoboken, NJ, 2nd ed. edition, 2012.

[2] Marco Ancona, Enea Ceolini, Cengiz Öztireli, and Markus Gross. Towards better understanding of gradient-based attribution methods for deep neural networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018.

[3] Jason Ansel, Edward Yang, Horace He, Natalia Gimelshein, Animesh Jain, Michael Voznesensky, Bin Bao, Peter Bell, David Berard, Evgeni Burovski, Geeta Chauhan, Anjali Chourdia, Will Constable, Alban Desmaison, Zachary DeVito, Elias Ellison, Will Feng, Jiong Gong, Michael Gschwind, Brian Hirsh, Sherlock Huang, Kshiteej Kalambarkar, Laurent Kirsch, Michael Lazos, Mario Lezcano, Yanbo Liang, Jason Liang, Yinghai Lu, C. K. Luk, Bert Maher, Yunjie Pan, Christian Puhrsch, Matthias Reso, Mark-Albert Saroufim, Marcos Yukio Siraichi, Helen Suk, Shunting Zhang, Michael Suo, Phil Tillet, Xu Zhao, Eikan Wang, Keren Zhou, Richard Zou, Xiaodong Wang, Ajit Mathews, William Wen, Gregory Chanan, Peng Wu, and Soumith Chintala. Pytorch 2: Faster machine learning through dynamic python bytecode transformation and graph compilation. *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2*, 2024.

[4] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473, 2014.

[5] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer, 1 edition, 2007.

[6] Diogo M Camacho, Katherine M Collins, Rani K Powers, James C Costello, and James J Collins. Next-generation machine learning for biological networks. *Cell*, 173(7):1581–1592, 2018.

[7] Priyam Study Centre. Amino Acids. `https://www.priyamstudycentre.com/2021/09/amino-acids.html`.

[8] Qiurui Chen. T5: a detailed explanation. `https://towardsdatascience.com/understanding-t5-model-text-to-text-transfer-transformer-model-69ce4c165023`, 2020.

[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *North American Chapter of the Association for Computational Linguistics*, 2019.

[10] Ahmed Elnaggar, Hazem Essam, Wafaa Salah-Eldin, Walid Moustafa, Mohamed Elkerdawy, Charlotte Rochereau, and Burkhard Rost. Ankh: Optimized protein language model unlocks general-purpose modelling, 01 2023.

[11] Ahmed Elnaggar, Michael Heinzinger, Christian Dallago, Ghalia Rehawi, Wang Yu, Llion Jones, Tom Gibbs, Tamas Feher, Christoph Angerer, Martin Steinegger, Debsindhu Bhowmik, and Burkhard Rost. Prottrans: Towards cracking the language of lifes code through self-supervised deep learning and high performance computing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 07 2021.

[12] Kemal Erdem. XAI Methods - Guided Backpropagation. `https://erdem.pl/2022/02/xai-methods-guided-backpropagation/`, 2022.

[13] Aurlien Gron. *Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. O'Reilly Media, Inc., 1st edition, 2017.

[14] Miho Higurashi, Takashi Ishida, and Kengo Kinoshita. Pisite: A database of protein interaction sites using multiple binding states in the pdb. *Nucleic acids research*, 37:D360–4, 11 2008.

[15] Rani Horev. BERT Explained: State of the art language model for NLP. `https://towardsdatascience.com/bert-explained-state-of-the-art-language-model-for-nlp-f8b21a9b6270`, 2018.

[16] SeyedMohsen Hosseini, G Brian Golding, and Lucian Ilie. Seq-InSite: sequence supersedes structure for protein interaction site prediction. *Bioinformatics*, 40(1):btad738, 01 2024.

[17] SeyedMohsen Hosseini and Lucian Ilie. Pithia: Protein interaction site prediction using multiple sequence alignments and attention. *International Journal of Molecular Sciences*, 23(21), 2022.

[18] Jonathan Hui. NLP — Word Embedding & GloVe. `https://jonathan-hui.medium.com/nlp-word-embedding-glove-5e7f523999f6`, 2019.

[19] IBM. What is explainable AI? `https://www.ibm.com/topics/explainable-ai#:~:text=AI%20Topic%20Updates-,What%20is%20explainable%20AI%3F,expected%20impact%20and%20potential%20biases.`

[20] M. G. KENDALL. A NEW MEASURE OF RANK CORRELATION. *Biometrika*, 30(1-2):81–93, 06 1938.

[21] Samia Khalid. BERT Explained: A Complete Guide with Theory and Tutorial. https://medium.com/@samia.khalid/bert-explained-a-complete-guide-with-theory-and-tutorial-3ac9ebc8fa7c, 2019.

[22] Narine Kokhlikyan, Vivek Miglani, Miguel Martin, Edward Wang, Bilal Alsallakh, Jonathan Reynolds, Alexander Melnikov, Natalia Kliushkina, Carlos Araya, Siqi Yan, and Orion Reblitz-Richardson. Captum: A unified and generic model interpretability library for pytorch, 2020.

[23] Yiwei Li, G Golding, and Lucian Ilie. Delphi: Accurate deep ensemble model for protein interaction sites prediction. *Bioinformatics (Oxford, England)*, 37, 08 2020.

[24] Scott M. Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *CoRR*, abs/1705.07874, 2017.

[25] Cory Maklin. Transformers Explained. https://medium.com/@corymaklin/transformers-explained-610b2f749f43, 2022.

[26] H. B. Mann and D. R. Whitney. On a Test of Whether one of Two Random Variables is Stochastically Larger than the Other. *The Annals of Mathematical Statistics*, 18(1):50 – 60, 1947.

[27] Dale Markowitz. Transformers, Explained: Understand the Model Behind GPT-3, BERT, and T5. https://daleonai.com/transformers-explained, 2021.

[28] Tomas Mikolov, Ilya Sutskever, Kai Chen, G.s Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. *Advances in Neural Information Processing Systems*, 26, 10 2013.

[29] Prakhar Mishra. Understanding T5 Model : Text to Text Transfer Transformer Model. https://towardsdatascience.com/understanding-t5-model-text-to-text-transfer-transformer-model-69ce4c165023, 2020.

[30] Christoph Molnar. *Interpretable Machine Learning*. 2 edition, 2022.

[31] Rajvardhan Patil, Sorio Boit, Venkat Gudivada, and Jagadeesh Nandigam. A survey of text representation and embedding techniques in nlp. *IEEE Access*, 11:36120–36146, 2023.

[32] Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe: Global vectors for word representation. In Alessandro Moschitti, Bo Pang, and Walter Daelemans, editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics.

[33] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In Marilyn Walker, Heng Ji, and Amanda Stent, editors, *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[34] Colin Raffel, Noam M. Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67, 2019.

[35] Ravish Raj. Supervised, Unsupervised and Semi-supervised Learning with Real-life Usecase. https://www.enjoyalgorithms.com/blogs/supervised-unsupervised-and-semisupervised-learning, 2021.

[36] Ritvik Rastogi. Papers Explained 33: ELMo. https://medium.com/dair-ai/papers-explained-33-elmo-76362a43e4#:~:text=ELMo%20is%20a%20new%20type,of%20challenging%20language%20understanding%20problems., 2023.

[37] Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should I trust you?": Explaining the predictions of any classifier. In *Proceedings of the Demonstrations Session, NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pages 97–101. The Association for Computational Linguistics, 2016.

[38] Erhan Sezerer and Selma Tekir. A survey on neural word embeddings. *ArXiv*, abs/2110.01804, 2021.

[39] Rayyan Shaikh. Mastering BERT: A Comprehensive Guide from Beginner to Advanced in Natural Language Processing. https://medium.com/@shaikhrayyan123/a-comprehensive-guide-to-understanding-bert-from-beginners-to-advanced-2379699e2b 2023.

[40] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. *CoRR*, abs/1704.02685, 2017.

[41] Avanti Shrikumar, Peyton Greenside, Anna Shcherbina, and Anshul Kundaje. Not just a black box: Learning important features through propagating activation differences. *CoRR*, abs/1605.01713, 2016.

[42] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. In Yoshua Bengio and Yann LeCun, editors, *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Workshop Track Proceedings*, 2014.

[43] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin A. Riedmiller. Striving for simplicity: The all convolutional net. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Workshop Track Proceedings*, 2015.

[44] Stanford. Convolutional Neural Networks cheatsheet. https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-convolutional-neural-networks, 2019.

[45] Stanford. Recurrent Neural Networks cheatsheet. https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-recurrent-neural-networks, 2019.

[46] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 3319–3328. PMLR, 2017.

[47] Adi L Tarca, Vincent J Carey, Xue-wen Chen, Roberto Romero, and Sorin Drăghici. Machine learning and its applications to biology. *PLoS computational biology*, 3(6):e116, 2007.

[48] Jordi Torres. Learning process of a neural network. https://resources.experfy.com/ai-ml/learning-process-of-a-neural-network/, 2019.

[49] Sik-Ho Tsang. Review — T5: Text-to-Text Transfer Transformer. https://sh-tsang.medium.com/review-t5-text-to-text-transfer-transformer-b3f0f3c07295, 2022.

[50] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6000–6010, Red Hook, NY, USA, 2017. Curran Associates Inc.

[51] Vatsal. Word2Vec Explained. https://towardsdatascience.com/word2vec-explained-49c52b4ccb71, 2021.

[52] Paul Villoutreix. What machine learning can do for developmental biology. *Development*, 148(1):dev188474, 2021.

[53] Robert Weast. *CRC Handbook of Chemistry and Physics*. CRC Press, 62nd ed. edition, 1981.

[54] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le

Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, October 2020. ACL, Association for Computational Linguistics.

[55] Chih-Kuan Yeh, Cheng-Yu Hsieh, Arun Sai Suggala, David I. Inouye, and Pradeep Ravikumar. On the (in)fidelity and sensitivity of explanations. In *Neural Information Processing Systems*, 2019.

[56] Qianmu Yuan, Jianwen Chen, Huiying Zhao, Yaoqi Zhou, and Yuedong Yang. Structure-aware protein-protein interaction site prediction using deep graph convolutional network. *Bioinformatics (Oxford, England)*, 38, 09 2021.

[57] Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In David J. Fleet, Tomás Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I*, volume 8689 of *Lecture Notes in Computer Science*, pages 818–833. Springer, 2014.

[58] Min Zeng, Fuhao Zhang, Fang-Xiang Wu, Yaohang Li, Jianxin Wang, and Min Li. Protein-protein interaction site prediction through combining local and global features with deep neural networks. *Bioinformatics (Oxford, England)*, 36, 09 2019.

[59] Aston Zhang, Zachary Lipton, Mu Li, and Alexander Smola. *Dive into Deep Learning*. 06 2021.

[60] Buzhong Zhang, Jinyan Li, Lijun Quan, Yu Chen, and Qiang Lü. Sequence-based prediction of protein-protein interaction sites by simplified long-short term memory network. *Neurocomputing*, 357, 05 2019.

Curriculum Vitae

Name:	Zahra Fazel
Post-Secondary Education and Degrees:	Sharif University of Technology Tehran, Iran 2017 - 2022 B.Sc. University of Western Ontario London, ON 2022 - 2024 M.Sc.
Honours and Awards:	Member of National Iranian Elites Foundation 2017-Present
Related Work Experience:	Graduate Teaching Assistant The University of Western Ontario 2022 - 2024 Graduate Research Assistant The University of Western Ontario 2022 - 2024 Applied Deep Learning Intern Cash App 2023 - 2024 Research Assistant École polytechnique fédérale de Lausanne 2021 - 2022 Undergraduate Teaching Assistant Sharif University of Technology 2019 - 2022 Undergraduate Research Assistant Sharif University of Technology 2020 - 2022