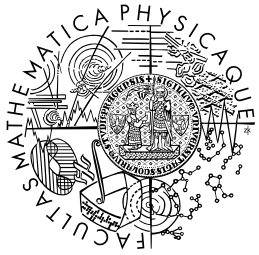


FACULTY
OF MATHEMATICS
AND PHYSICS
Charles University

BACHELOR THESIS

Ondrej Lopuch

Detection of protein-ligand binding sites using 3D Vision Transformers

Department of Software Engineering

Supervisor of the bachelor thesis: doc. RNDr. David Hoksza, Ph.D.

Study programme: Computer Science

Study branch: IPP5

Prague 2023

I dedicate this thesis to my family and my friends who have supported me throughout my studies. I give special thanks to my partner who introduced me to the field of bioinformatics. I would also like to thank my supervisor doc. RNDr. David Hoksza, Ph.D. for his guidance. Last but not least, I would like to thank everyone, who helped me with this thesis such as RNDr. Milan Straka, Ph.D. for his great insights about model architecture and RNDr. Jakub Yaghob, Ph.D. for his help with faculty's GPU cluster.

Title: Detection of protein-ligand binding sites using 3D Vision Transformers

Author: Ondrej Lopuch

Department: Department of Software Engineering

Supervisor: doc. RNDr. David Hoksza, Ph.D., Department of Software Engineering

Abstract: Detection of protein-ligand binding sites plays key role not only in understanding of protein function but it also can be used for computer-aided drug design. Improving these detection can lead to faster drug development. In recent years, many machine learning methods were proposed for this task. Nowadays, transformer architecture became one of the most prominent one for non-biological data. Its extension for images, vision transformer, became comparable to state-of-the-art algorithms. Moreover, this vision transformer can be expanded into 3D space. The goal of this thesis is to evaluate possibilities of extending transformers into 3D, for biological data, specifically for protein-ligand binding site detection, exploiting the qualities of attention mechanism.

Keywords: 3D vision transformers, protein-ligand complexes, computer-aided drug design, binding sites

Contents

Introduction

Since its introduction, the transformer architecture has become the state-of-the-art model for many machine learning tasks ranging from machine translation (Vaswani et al. [2017]), question answering (Brown et al. [2020]), to object detection(Carion et al. [2020]) and video classification (Wang et al. [2018]).

Another broad field, where transformers may be applicable now and in the future, is bioinformatics. With clever data engineering as a first step in the process, we can easily transform various types of biological data into datasets suitable for machine learning tasks. One of the key focus points at the moment is proteomics, an area of biology covering the proteins. Jumper et al. [2021]

Proteins are essential to life. Chemically, they are polypeptides - they consist of amino acids connected by peptide bonds. They interact with other molecules, and these interactions determine their biological properties. The substance that is bound by protein, the ligand, can be an ion, a small molecule, or even a macromolecule such as another protein or DNA. These protein-ligand complexes serve a fundamental function within a cell, playing a pivotal role in providing structural integrity, defining cellular shape, and executing a wide array of functions (Bruce Alberts [2019]). Ligands usually bind with protein at a highly specific *binding sites*, changing conformation, 3D structure, and activity of the protein.

The ligand-binding process is crucial in drug design. Protein-ligand complexes have a key role in cellular processes e.g. in replication or translation. When using specific ligands, this activity can be modulated in different ways. This is one of the key mechanisms in drug design. However, determining the ideal ligand for a specific modification by conducting laboratory experiments may be very costly. Therefore, many computer-aided methods for predicting ligand binding sites were recently proposed. Many of them are machine learning based tools, such as random forest classifiers (Krivák and Hoksza [2018]), graph neural networks (Sergei A. Evteev and Ivanenkov [2023]), 3D convolutional networks (Jimenez et al. [2017]) and deep residual neural networks (Kandel et al. [2021]).

Since transformers have proven their quality in many areas, even with biological data (Jumper et al. [2021]), we present a 3D vision transformer model for detecting binding sites in protein-ligand complexes. By exploiting its attention mechanism, we believe that 3D vision transformers have a great potential for prediction from 3D structure of the complex.

1. Biological background

1.1 Proteins

Proteins consists of long unbranched chains of amino acids, each of them linked to its neighbor through a covalent peptide bond. Proteins are therefore also known as polypeptides. Each protein has a unique sequence of amino acids, and there are thousands of different proteins in a cell (Bruce Alberts [2017]). Amino acids serve as the fundamental building blocks of proteins.

1.1.1 Amino acids

Amino acids are organic compounds characterized by the presence of a carboxylic acid group[1] and an amino group. They share a fundamental structural element: a central carbon atom (C_α) linked to a hydrogen atom, an amino group (NH_2), and a carboxyl group (COOH) (Branden and Tooze [1999]). The distinguishing factor among amino acids lies in the uniqueness of its side chain attached to the carbon atom (Buxbaum [2015]). The side chain determines the physiochemical properties of the amino acid. Our genetic code encodes 20 different amino acids; however, rare cases may involve other distinct types of side chains. These amino acids have different kind of properties resulting from their composition, such as hydrophobicity [2], aliphaticity [3], whether they are amides[4], positively charged etc. These properties will be later used in our model as features.

With the exception of glycine, all 20 common amino acids are chiral molecules, meaning they are not symmetrical to reflections. Consequently, they may exist in two distinct forms: L-form and D-form. In biological systems, elaborate molecular recognition processes are reliant on distinguishing between these chiral forms. The translation process responsible for protein synthesis has evolved to exclusively employ the L-form of amino acids. As a result, the amino acids found in proteins adopt only the L-form. These 20 common amino acids are typically represented using either a three-letter or one-letter code (see fig. 1.1).

1.1.2 Protein structure

In the study of proteins, four levels of organization in their structure are recognized: primary, secondary, tertiary, and quaternary (fig. 1.2).

The primary structure refers to the linear sequence of amino acids connected by peptide bonds. These sequences can vary in length, ranging from just a few amino acids to thousands of them.

The secondary structure involves the formation of three-dimensional structural elements, such as α helices and β strands. They exhibit repetitive folding patterns in three-dimensional space, and they are stabilized by hydrogen bonds.

[1] containing COOH group
[2] repelled by a mass of water
[3] pertaining to peptides
[4] derived from ammonia

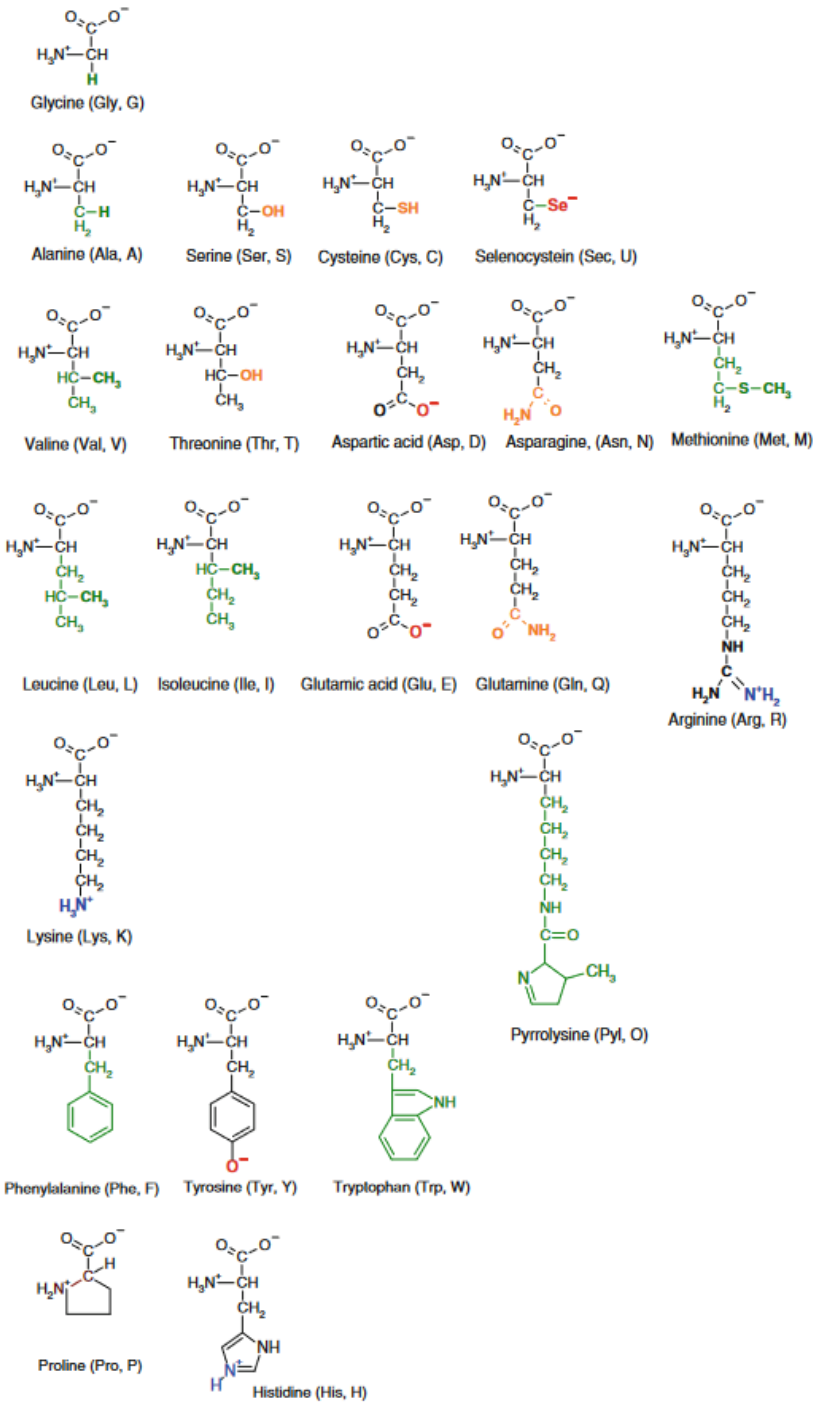
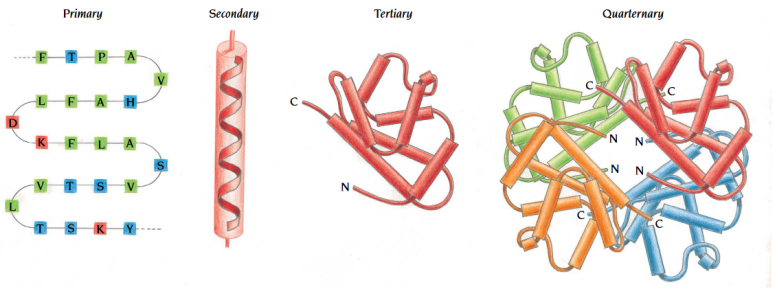


Fig. 1.1. *The 22 amino acids encoded by genes. Once incorporated into proteins, amino acids may be further modified. Pyrrolysine has been found only in bacteria; it is encoded by the stop codon UAG. Selenocysteine is encoded by the UGA stop codon. Acidic groups are marked red, basic groups blue, polar groups orange, and hydrophobic groups green. Source: Buxbaum [2015]*



Source: Branden and Tooze [1999]

Fig. 1.2. *The amino acid sequence of a protein's polypeptide chain is called its primary structure. Different regions of the sequence form local regular secondary structures, such as alpha (a) helices or beta (b) strands. The tertiary structure is formed by packing such structural elements into one or several compact globular units called domains. The final protein may contain several polypeptide chains arranged in a quaternary structure. By formation of such tertiary and quaternary structure amino acids far apart in the sequence are brought close together in three dimensions to form a functional region, an active site*

The tertiary structure describes the overall conformation of the polypeptide chain in three-dimensional space. The final folded structure typically represents the conformation that minimizes the protein's free energy. The tertiary structure is determined by various interactions, including hydrophobic interactions, Van der Waals interactions, hydrogen bonds, salt bridges, and disulfide bonds (Buxbaum [2015]). This three-dimensional arrangement plays a crucial role in determining the biological function of the protein.

The quaternary structure describes the manner in which multiple polypeptide chains, in this case known as subunits, come together to form a functional protein complex. This level of organization highlights the interactions and arrangement of these subunits within the overall protein structure.

1.1.3 Functions of proteins

Proteins serve a fundamental function within cells, playing a pivotal role in providing structural integrity, defining cellular shape, and executing a wide array of functions (Bruce Alberts [2019]). The biological properties of a protein molecule are determined by its physical interactions with other molecules (Bruce Alberts [2017]). Their functions encompass an extensive range, including enzymatic reactions that facilitate covalent bond breakage, acting as structural proteins that offer mechanical support to cells and tissues, functioning as transport proteins responsible for conveying small molecules or ions, such as hemoglobin in the bloodstream, and participating as hormones and growth factors, among numerous other roles (Bruce Alberts [2019]). This remarkable versatility of proteins arises from their ability to adopt a wide range of distinct conformations, allowing for the execution of diverse cellular functions.

1.1.4 Protein structure determination

There are many experimental methods to determine the protein structure. In this section, we will cover three of them that are commonly used: X-ray crystallography, NMR (nuclear magnetic resonance) spectroscopy and cryogenic electron microscopy (3D cryo-EM).

X-ray crystallography (XRC) is an experimental method for determination of atomic and molecular structure of a crystal using X-ray. With the help of the X-ray diffraction patterns, we can measure the angles and intensities of the diffracted beams. (Ilari and Savino [2008]). This method produces a 3D image of electron densities, and it is able to depict different chemical bonds and other key attributes. A great advantage of this method is that the structure can be arbitrarily large, the only condition being the ability of the protein to form a crystal.

NMR spectroscopy is a newer approach which uses quantum-mechanical properties of the atom nuclei. These properties provide us a guidance of how atoms are linked chemically, how close they are in space and how rapidly they move with respect to each other (Wüthrich [1990]). It uses powerful magnets and radio frequencies, sending them through the sample in a water solution, and measuring the absorption. It is a non-destructive technique and generally it can provide more information than X-ray crystallography. Moreover, it may help determine the structure of non-crystallic molecules, such as intrinsically disordered proteins. The main disadvantages are cost and applicability only on small molecules.

Cryogenic electron microscopy (3D cryo-EM) is a novel, alternative experimental method to NMR and XRC. In this case, samples in an aqueous solution are rapidly cooled to cryogenic temperatures. This process, also known as vitrification, is so fast, that the water does not have time to crystallize. The sample is applied to a grid-mesh and frozen in liquid ethane (Tivol et al. [2008]) and bombarded with electrons. Thin specimen then scatters the electrons. Interference between scattered and unscattered electrons produces a phase contrast image. The 3D structure can be obtained from a set of views at different orientations. This method is very popular in the last years, as it is able to produce 3D structures in the best resolution possible, also thanks to better electron microscopy hardware (Yip et al. [2020]).

1.2 Protein-ligand complexes

As mentioned above, proteins interact with other molecules and these interactions determine their biological properties. The substance that is bound by protein can be an ion, a small molecule or even a macromolecule such as another protein or DNA (Bruce Alberts [2019]). We refer to this substance as the ligand. In this thesis, we will focus only on small molecules.

Protein-ligand complex is molecular recognition[5] between the protein and the ligand. Usually, this interaction is not a process by itself, but part of functionally important mechanism as self-replication, metabolism and information processing. This interaction is reversible and the bonds are non-covalent. This process always shows a very high specificity, each protein can usually bind just few molecules out

[5]specific interaction between two or more molecules

of thousands or millions of possible types. Whether the ligand binds with protein is determined by its affinity. Affinity is the strength of the binding forces such as hydrogen bonds, electrostatic attractions and van der Waals forces, measured by equilibirum dissociation constant (KD). Binding of protein with ligand usually leads to change of conformation of protein.

1.2.1 Binding sites

Binding site is a region of a protein that associates with a ligand (Bruce Alberts [2019]). It usually consists of a cavity in the protein surface, formed by a particular arrangement of amino acids. Separate regions of the protein surface generally provide binding sites for different ligands, allowing the protein's activity to be regulated in many ways (Bruce Alberts [2017]). The highest affinity of the complex is achieved when molecule has a perfect mirror image of the shape of the target surface together with a charge distribution that complements the target surface perfectly (1.3).

1.3 Small molecule discovery

Traditional approach to drug design relies on trial-and-error testing of chemical substances and matching apparent effects to treatments. However, this approach may be very costly, that is the main reason for using so-called computer-aided drug design.

Small molecule discovery via computer-aided methods can be divided into three main categories. The first category is the exploration of new ligands. In this approach, new ligands are built step by step with constraints provided by binding pocket (Schneuing et al. [2023]). This method can lead to finding novel structures, that have yet to be in any database. A second method is the optimization of known ligands, where a software usually suggests small changes on ligand to improve its affinity (Prihoda et al. [2022]). The last method is so-called virtual screening. It consists of searching databases of 3D structures for potential ligands that fit the binding pocket of given receptor, and then using docking programs to predict affinity.

Our goal is predict residues in the complexes, which are part of binding sites. Ideally, the computational methods will be able to predict affinity before a compound is synthesised and therefore saving time and cost. In reality, there is usually few iterations before finding the ideal compound.

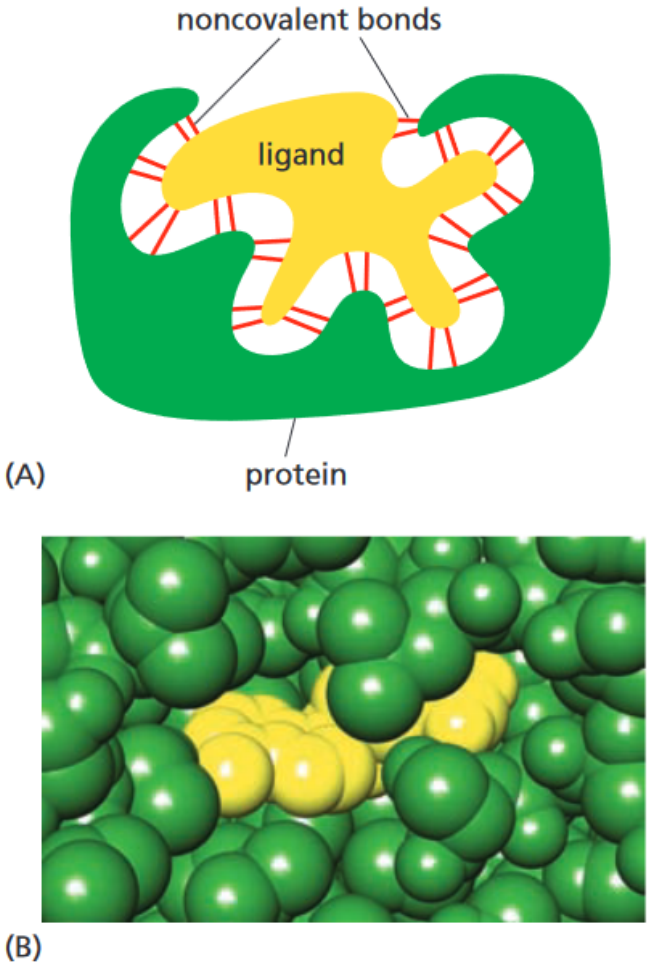
1.3.1 Binding sites predictions

Identifying binding sites is the first step in structure-based drug design. Currently there are two types of drugs on the market: orthosteric and allosteric (Nussinov and Tsai [2012]). Orthosteric bind at the active binding sites, competing with natural substrates or ligands. If their affinity is high, they will bind and block the site for other molecules. Unfortunately, these drugs usually have side effects which occur by drug binding to homologous proteins sharing a similar binding site. Hence, they need high affinity to the target. On the other hand, allosteric drugs bind elsewhere on the protein surface and allosterically change the conformation

of the protein binding site. This allows them to be far less invasive than their orthosteric counterparts resulting in lower number of side effects(Nussinov and Tsai [2012]).

Existing approaches

There are many methods for predicting ligand binding sites, and they can be categorized based on their main algorithm strategy into geometric, such as Fpocket, (Guilloux et al. [2009]) which is based on Voronoi tessellation and alha spheres, energetic, such as SITEHOUND, (Hernandez et al. [2009]) where molecular interactions are used for prediction, and last but not least, conservation-based and template-based methods. In recent years, many machine learning based tools were proposed, such as random forest classifiers (Krivák and Hoksza [2018]), graph neural networks (Sergei A. Evteev and Ivanenkov [2023]), 3D convolutional networks (Jimenez et al. [2017]) and deep residual neural networks (Kandel et al. [2021]). Some of these methods, proved to be quite efficient in mapping short distance dependencies. However, with biological data, there are many long dependencies between atoms, such as allosteric interactions. Therefore, we believe that transformer architecture with its self-attention can better map these long dependencies.



Source: Bruce Alberts [2017]

Fig. 1.3. *The selective binding of a protein to another molecule. Many weak bonds are needed to enable a protein to bind tightly to a second molecule, or ligand. A ligand must therefore fit precisely into a protein's binding site, like a hand into a glove, so that a large number of noncovalent bonds form between the protein and the ligand. (A) Schematic; (B) space-filling model.*

2. Transformers

A new deep learning model called the transformer emerged in 2017 (Vaswani et al. [2017]). Following its release, the transformer swiftly established itself as the leading architecture in NLP systems. Today, variants of the transformer, commonly referred to as transformers, find extensive application across various domains. They are utilized in diverse tasks such as audio processing, NLP tasks including translation and text generation (Brown et al. [2020]), as well as computer vision tasks like image generation, image classification, and object detection. (Dosovitskiy et al. [2021])

2.1 Architecture

Original transformer from Vaswani et al. [2017] was used for machine translation from English to German. Its architecture consists of an encoder and a decoder(2.1).

2.1.1 Encoder

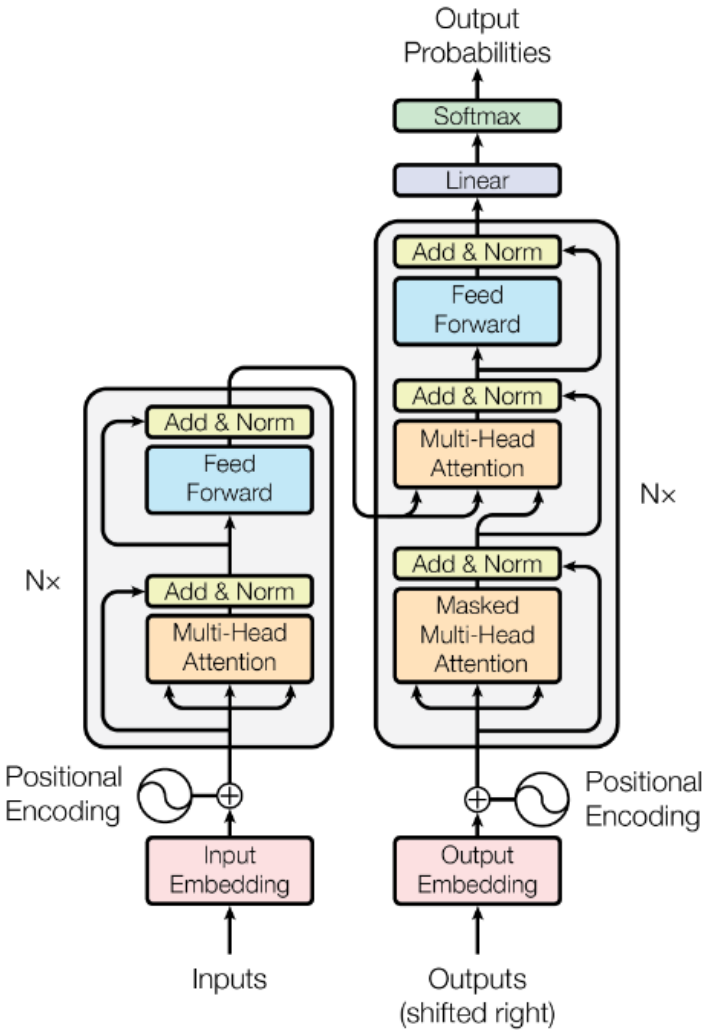
The encoder is made up of a stack of identical layers, each having two sublayers. The first sublayer is a multi-head attention layer, while the second sublayer is a feed-forward neural network. Both sublayers employ residual connections[1] and layer normalization [2]. The encoder takes an embedded sentence combined with positional encoding as its input. The embeddings and positional encoding are explained in section 2.1.5.

2.1.2 Decoder

The decoder bears strong resemblance to the encoder. It uses a stack of layers, each of the layers featuring three sublayers. The first sublayer is akin to the one found in the encoder, utilizing multi-head attention on embedded output tokens combined with positional encoding, along with a residual connection and layer normalization. The second sublayer is a multi-head attention layer that calculates attention over the encoder's output. The third sublayer mirrors the encoder's feed-forward network. Furthermore, the decoder incorporates a modified multi-head attention mechanism with a mask, which conceals subsequent positions of the input. Following the third sublayer, there is a linear transformation and a softmax operation that convert the decoder's output into predicted probabilities for the next token.

[1] connection between the output of one earlier layer to the input of another future layer several layers later

[2] all neurons in a particular layer have the same distribution across all features for a given input



Source: Vaswani et al. [2017]

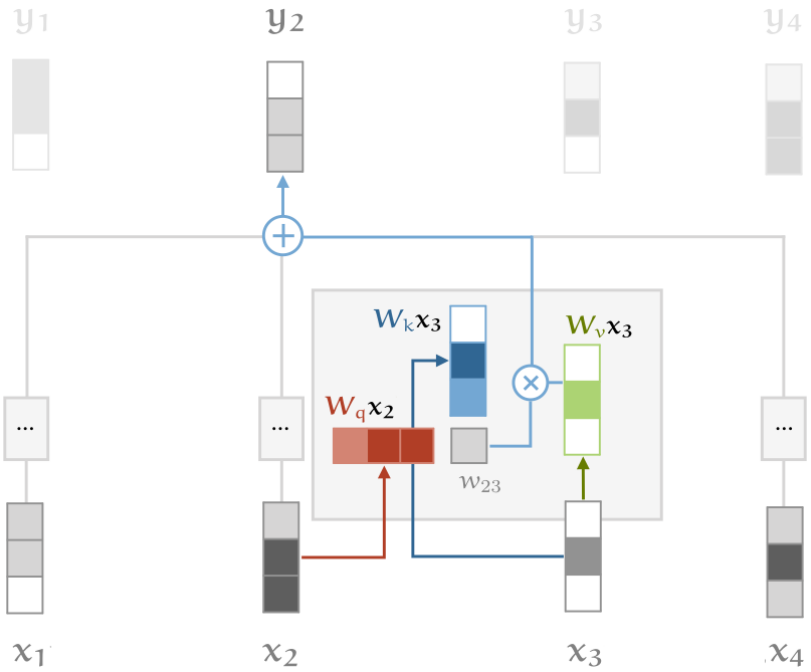
Fig. 2.1. *Transformer architecture*

2.1.3 Attention

The attention is a straightforward process that involves mapping following three elements - query and key-value pairs - to produce an output (Vaswani et al. [2017]). The input vector is used in different ways to produce the desired output. In the query, it is compared to other vectors to get its own output y_i, from the key we obtain the j-th output ($j \neq i$) and we use the value to compute the final output vector once the weights have been established (2.2).

Scaled dot-product attention

The initial transformer model employs the scaled dot-product attention mechanism. This mechanism operates on input vectors consisting of queries and keys, both with a dimension of k, as well as values with a dimension of v. To calculate attention, the dot-product of queries and keys is computed, and then scaled by



Source: https://peterbloem.nl/files/transformers/key-query-value.svg

Fig. 2.2. *Illustration of the self-attention with key, query and value transformations.*

$\sqrt{k}$. The resulting scaled dot-product is processed through the softmax function[3], generating weights for the values (2.3). The attention output is obtained by taking the dot-product of these weights and the values. In practice, this operation is performed on matrices Q, K, and V, representing the query, keys, and values, respectively. Alternatively, additive attention can be used (Bahdanau et al. [2015]), but the dot-product approach allows for the utilization of matrix multiplication, which enables faster computation and greater space efficiency. Overall the computation results in:

$$Attention(Q, K, V = softmax(\frac{QK^T}{\sqrt{k}})V \tag{2.1}$$

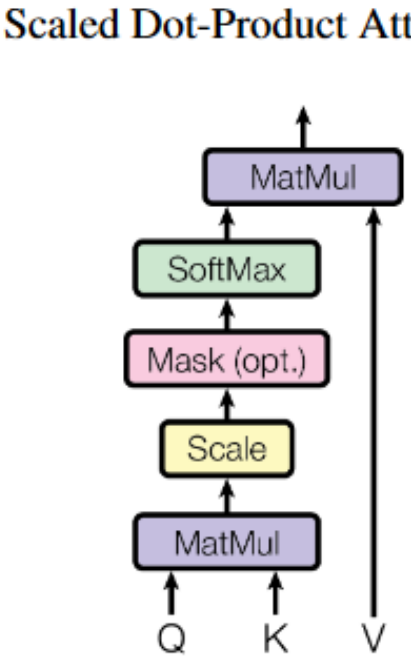
Multi-head attention

Incorporating the multi-head attention introduces significant advantages, including accelerated computation. This approach entails projecting queries, keys, and values into dimensions of k, k, and v, respectively, which is repeated h times, where h is the number of heads. By doing this, the computations can be performed in parallel. The results from the multi-head attention are then combined through concatenation and further projected to derive the final values(2.2). The Multi-head attention given as

$$MultiHead(Q, K, V) = Concat(head_1, ..., head_h)W^O \tag{2.2}$$

$$head_i = Attention(QW_i^Q, KW_i^K, VW_i^V)$$

[3] $\sigma(x)_i = \frac{e^{x_i}}{\sum_{j=1}^{K} e^{x_j}}$, where K represents number of classes



Source: Vaswani et al. [2017]

Fig. 2.3. *Scaled-dot product*

where the projection matrices are $W_i^Q \in \mathbb{R}^{d \times k}$, $W_i^K \in \mathbb{R}^{d \times k}$, $W_i^v \in \mathbb{R}^{d \times V}$, $W^O \in \mathbb{R}^{hv \times d}$ and d represents output dimension of model. Vaswani et al. [2017] uses $d = 512$ and $h = 8$.

2.1.4 Feed-forward networks

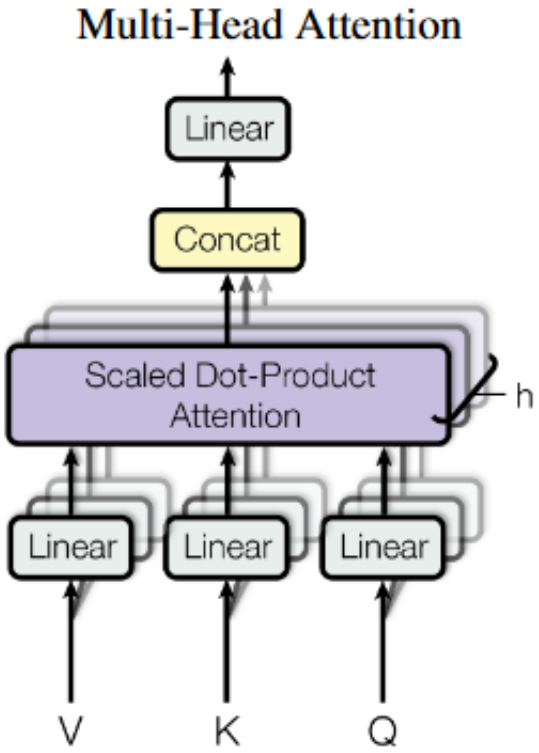
As previously stated, every layer within the encoder and decoder incorporates a fully connected feed-forward neural network. These sublayers are composed from two linear transformations with a ReLU activation[4] function applied in between (2.3).

$$FFN(x) = max(0, xW_1 + b_1)W_2 + b_2 \tag{2.3}$$

2.1.5 Embeddings and positional encodings

Nearly all sequence transduction models utilize learned embeddings to convert input tokens and output tokens into fixed-length vectors. These embeddings serve as internal representations of tokens for the model.

[4] $ReLU(x) = \max(0,x)$



Source: Vaswani et al. [2017]

Fig. 2.4. *Multi-head attention*

Since transformers do not employ recurrence or convolution, they lack inherent understanding of sequence order. To address this, positional encoding is utilized to provide information about the relative and absolute positions of tokens. Absolute position encodings are computed in the input layer and are then combined with the input token embeddings. This approach, originally introduced in the transformer model, does have a major drawback - it requires a fixed length of the input sequence and does not directly capture the relative positions between words. To overcome this limitation, several schemes involving relative position encoding have been proposed (Chen et al. [2021]).

In the original transformer paper, absolute positional encoding was implemented using sine and cosine functions with varying frequencies (2.4 , 2.5). Each dimension of the positional encoding corresponds to a sinusoidal pattern (Vaswani et al. [2017]).

$$PE_{pos,2i} = sin(pos/10000^{2i/d_{model}}) \tag{2.4}$$

$$PE_{pos,2i+1} = cos(pos/10000^{2i/d_{model}} \tag{2.5}$$

2.1.6 Pros and cons of transformers

In recent years, transformers have emerged as superior models across numerous tasks. An exemplary instance is the BERT model, which has achieved remarkable performance in various NLP tasks(Devlin et al. [2019]). Additionally, Transformers exhibit competitive results in object detection when compared to fast-RNN

(Carion et al. [2020]). One of the key reasons for the tremendous success of transformers lies in their ability to recognize long-distance relationships between tokens—a task that is challenging or computationally expensive for RNN or CNN (Lin et al. [2021]). Moreover, Transformers are highly parallelizable, making them more cost-effective to compute (Wang et al. [2019]).

However, Transformers do have a few drawbacks. They require vast amounts of training data, which, in turn, leads to extended training times (Hafiz et al. [2021]).

2.2 Vision transformers

As previously discussed, transformers are presently regarded as state-of-the-art models for NLP tasks. However, their excellence extends beyond this category alone. Transformers have also been utilized in computer vision tasks, such as object detection (Carion et al. [2020]) and video classification (Wang et al. [2018]). In 2021, researchers at Google introduced a novel model, known as vision transformers, specifically designed for computer vision tasks Dosovitskiy et al. [2021]. This model relies solely on self-attention without incorporating any convolutional operations. While traditional CNNs still outperform vision transformers on smaller datasets, the latter has proven to outshine the former on larger datasets, ranging from 14 million to 300 million images, achieving superior results across multiple image recognition benchmarks (Dosovitskiy et al. [2021]).

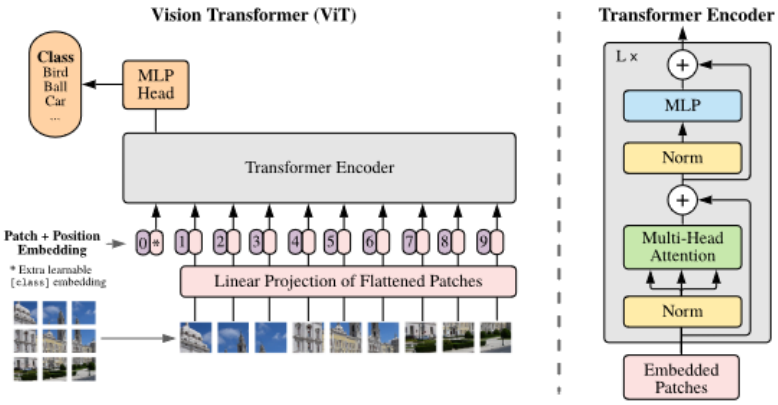
2.2.1 Architecture

The standard Transformer model operates on 1D sequences of token embeddings as input. Therefore, when dealing with an image represented as $x \in \mathbb{R}^{H \times W \times C}$, where H, W, and C denote the image's height, width, and number of channels, respectively, the image needs to be reshaped into a sequence of flattened 2D patches denoted as $x_p \in \mathbb{R}^{N \times P^2 C}$. Here, (P, P) represents the resolution of the image patch, and $N = \frac{HW}{P^2}$ represents the resulting number of patches, which also corresponds to the length of the input sequence for the transformer. Throughout all layers of the transformer, a constant vector size of D is utilized.

To retain positional information, a 1D learnable positional embedding is employed. As an encoder, the structure remains consistent with the one mentioned in 2.1.1, encompassing multi-head attention, feed-forward networks, layer normalization, and residual connections (2.5).

2.2.2 3D vision transformers

Due to the promising outcomes demonstrated by vision transformers in 2D tasks, various 3D vision approaches have incorporated transformers into their model designs (Xu [2021]) (Saining Xie [2018]) (Zhao et al. [2021]). Nonetheless, 3D tasks present additional challenges that require different architectural designs (Wang et al. [2022]).



Source: Dosovitskiy et al. [2021]

Fig. 2.5. *Architecture of proposed Vision Transformer*

3D representation

The primary distinction between 2D and 3D tasks lies in the data preprocessing. Images possess an inherent natural representation characterized by pixels on a standardized grid. However, such organized structure does not exist in 3D geometry Lahoud et al. [2022]. Consequently, there are several commonly used representations for 3D data, which enable the application of different deep learning algorithms: point clouds, meshes, and voxels (2.6).

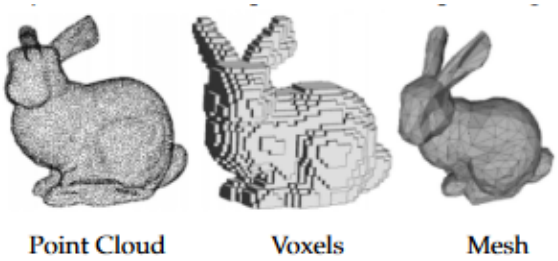
A point cloud refers to a set of vertices in 3D space, represented by their coordinates along the x, y, and z axes. Each vertex can optionally include additional information. Unlike images, point clouds represent an unordered set of data points.

In contrast, voxel representation provides information about data on a regular grid in 3D space. Voxels can be seen as the 3D equivalent of pixels. Each voxel contains its own set of features. Voxel representation can be created from point clouds through a process called voxelization, wherein all features within a voxel are grouped together for subsequent processing. This representation enables the application of certain methods similar to 2D convolution. However, voxel representation tends to be quite sparse, with many empty voxels. Handling this sparsity becomes a specific consideration during convolution operations.

A mesh, on the other hand, is a collection of vertices, edges, and faces. This representation shares similarities with point clouds, but meshes also include information about the object surface.

3D vision transformer design

The design of vision transformers presents a significant challenge, primarily concerning the representation and processing of input data. Transformers, as a type of model architecture, are versatile and capable of handling various data representations. The choice of data representation impacts critical factors such as data size, distribution, granularity, and structure (Lahoud et al. [2022]). Point clouds, being an unordered set with a straightforward representation, provide a simple



Source: Lahoud et al. [2022]

Fig. 2.6. *Representation of 3D object with different techniques - point cloud, voxelization and mesh*

way to utilize 3D data in transformers. To efficiently handle large point clouds while maintaining fine resolution, they can be cropped to a specific physical size. This cropping technique reduces the number of points processed and accelerates attention calculations, which can be computationally intensive for large spaces with numerous points. Another approach involves converting points into a regular grid, where the granularity level becomes a crucial consideration.

The attention block remains a key component in 3D vision transformers (Dosovitskiy et al. [2021]). As previously mentioned, attention mechanisms excel at capturing long-range dependencies that convolutional networks struggle to exploit. Transformers can be applied to sets, and point clouds naturally lend themselves to this representation. Similar to NLP tasks where sequences can have varying lengths, point clouds can have different sizes. In 2D transformers, position information is typically added to the feature information, while in 3D, the position is naturally represented by the coordinates of points within the point cloud. This positional information can be further processed using 3D positional encodings (Liu et al. [2022]).

Contextual information is crucial for efficient vision applications, encompassing both fine local details and global context. Attention is a well-suited mechanism for capturing such information; however, this often increases the computational requirements. To achieve both targets, data is commonly processed at different scales. Transformers can be applied to local neighborhoods of points to capture local shape information. Similar to the 2D domain, local pooling enables processing on a different scale with a larger receptive field. Alternatively, the entire 3D data structure can be used, eliminating the need for local neighborhood sampling as the entire point cloud is processed at once.

There are two approaches to utilizing transformers: pure and hybrid. In pure transformers, attention layers are used to extract features and generate specific outputs. In some cases, non-attention layers can be employed to encode the input or complement the attention mechanism. Hybrid applications involve integrating transformers into other deep learning architectures, either replacing extraction modules or augmenting them with non-transformer layers to extract richer information (Hafiz et al. [2021]).

Scalability is a concern with transformers, as they tend to be computationally expensive due to the generation of large attention maps, which have a quadratic complexity with respect to the input size (Vaswani et al. [2017]). Given the increased data size in 3D tasks, this can result in longer training times.

3. Implementation

In this chapter, we introduce our deep learning model utilizing a pure transformer architecture for the purpose of classifying tasks. The objective is to predict the specific residues within protein structure that constitute binding sites. This predictive capability has the potential to enhance both the cost and time efficiency of drug design, as outlined in chapter 1.

3.1 Model design

To our knowledge, the utilization of 3D vision transformers for the prediction of binding sites has not been adopted in practice. Hence, this thesis adopts an exploratory approach in investigating this novel approach.

The whole architecture is shown in picture. 3.1.

3.1.1 Programming language and libraries

The model was developed using Python 3.11.1 and relied on various external libraries. Two prominent libraries, Tensorflow and Biopython, played crucial roles in the implementation.

Tensorflow, a highly popular library, serves as an interface for expressing machine learning algorithms and provides their execution capabilities (Abadi et al. [2016]). Its versatility allows for effective training and evaluation of deep learning models, including support for GPU cards. Additionally, Tensorflow offers optimized implementations of state-of-the-art techniques such as multi-head attention.

The Biopython project, another key module utilized in this thesis, comprises open-source Python libraries specifically tailored for a wide range of bioinformatics challenges. Biopython acts as an interface, allowing seamless integration of Python with optimized C and C++ code for scientific programming (Cock et al. [2009]). In this research, Biopython was employed for dataset preprocessing tasks.

3.1.2 Input/Output

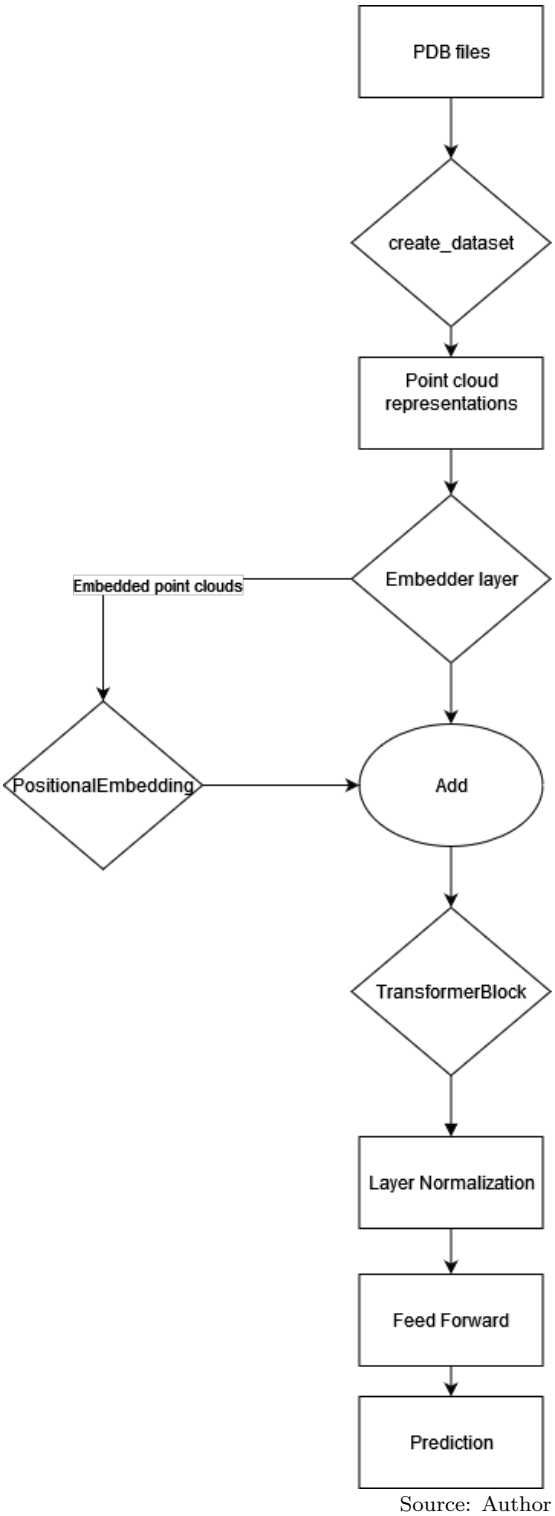
The raw input are pdb files from protein database. Every file contains header and atom-wise properties such as coordinates and atom type. Pdb file is linear representation of protein-ligand complexes (3.2). From this raw files, we extracted every atom and used it for further preprocessing.

The input for our model is numpy array, which consists of protein-ligand complexes. Every complex consists of atoms with 41 features. The full dataset is represented by 3 dimensional shape

$$(x, y, z)$$

where z represents number of complexes, y represents the length of complex and z is 41 as number of features. For variable length we used ragged tensors.

The output of our model is a straightforward classification of each atom within the complexes, categorized into two classes: those that belong to a binding site



Source: Author

Fig. 3.1. *Architecture of our 3D Vision Transformer. Rectangles represents files and Tensorflow layers. Diamond-shaped blocks represents our own implementation.*

```
ATOM      1  N   MET A   1      35.947 -54.045  75.418  1.00 22.64           N
ATOM      2  CA  MET A   1      36.467 -53.041  74.451  1.00 19.28           C
ATOM      3  C   MET A   1      35.359 -52.008  74.353  1.00 17.09           C
ATOM      4  O   MET A   1      34.201 -52.379  74.289  1.00 15.30           O
ATOM      5  CB  MET A   1      36.697 -53.726  73.106  1.00 24.67           C
ATOM      6  CG  MET A   1      37.760 -53.079  72.229  1.00 29.36           C
ATOM      7  SD  MET A   1      37.922 -53.817  70.579  1.00 27.46           S
ATOM      8  CE  MET A   1      39.530 -54.435  70.646  1.00 32.52           C
ATOM      9  N   LYS A   2      35.703 -50.729  74.414  1.00 15.61           N
ATOM     10  CA  LYS A   2      34.707 -49.667  74.353  1.00 16.77           C
ATOM     11  C   LYS A   2      35.000 -48.619  73.330  1.00 15.54           C
ATOM     12  O   LYS A   2      36.117 -48.109  73.288  1.00 16.57           O
ATOM     13  CB  LYS A   2      34.658 -48.890  75.651  1.00 17.43           C
ATOM     14  CG  LYS A   2      34.129 -49.602  76.818  1.00 21.32           C
```

Source: Author

Fig. 3.2. *Example of pdb file. Column 3 represents atom type, columns 6 to 8 represents coordinates.*

and those that do not. It is essential to highlight that, due to the inherent characteristics of the data, the negative class vastly dominates in terms of ratio of positive/negative samples. This ratio is approximately 15 negative examples to 1 positive.

3.1.3 Preproccessing

The initial objective entailed creating a dataset from Protein Data Bank (PDB) files containing protein-ligand complexes, intended for utilization as input for our model. From these files, we built point cloud representation of the complex in **point_cloud.py**. For each complex, we extracted all atoms and assigned them 41 distinct features, categorized into five groups: atom names, coordinates, categorical features, numerical features, and neighborhood features. All features are mentioned with description in table 3.1.

Atom names encompass a straightforward encoding of atom types, occasionally accompanied by supplementary information like remoteness indicator codes and branch indicators. Coordinates are represented by three values corresponding to the x, y and z axes.

Categorical features are further divided into two subcategories: amino acid features and atomic features. Amino acid features remain constant for every atom within a specific residue and provide information regarding hydrophobicity, acidity, charge, and other relevant attributes. Atomic features consist of pharmacophoric[1] annotations of cavity grid points utilized by (Desaphy et al. [2012]). These properties include positive/negative ionizability, hydrophobicity, aromacity, among others.

Numerical values encompass various types of propensities, such as ligand binding propensities for biologically valid and invalid ligands. Invalid ligands serve no known biological function (Khazanov and Carlson [2013]). The propensity is a measure of residue over-representation to explore the binding site composition(Bartlett et al. [2002]).

[1] An ensemble of steric and electronic features, necessary to ensure the optimal supramolecular interactions with a specific biological target structure and to trigger (or to block) its biological response

Additionally, neighborhood features are considered. We calculated the absolute number of atoms within a 6 Å[2] radius of each atom, the number of protein atoms weighted by distance, and the count of different atom types such as carbon, oxygen, and nitrogen. These features were also calculated for a broader neighborhood within a 10 Å radius. Furthermore, for the smaller neighborhood, we tallied the number of hydrogen-bond donor and acceptor atoms. For this purpose, we used neighbor search, implemented in Biopython module.

3.1.4 Embedder

Initially, the raw input undergoes transformation through our custom class, known as Embedder implemented in `vit.py`.

```
class Embedder(tf.keras.layers.Layer):
    def __init__(self, dim):
        super().__init__()
        self.atom_name_embd =
            tf.keras.layers.Embedding(100, dim)
        self.numerical_embd =
            tf.keras.layers.Embedding(500,dim // 8)

    def split_inputs(self, inputs):
        names, coordinates,
            categorical, numerical, neighborhood = ...
        return names, coordinates,
            categorical, numerical, neighborhood

    def call(self, inputs):
        batch_size = tf.shape(inputs)[0]
        names, coordinates, categorical,numerical, neighborhood =
        self.split_inputs(inputs)

        embeddings = ...
        features = tf.concat([embeddings, categorical,
            numerical, coordinates],axis=-1)

        return features
```

This class partitions the input into columns using method `Embedder.split_inputs()`, which are subsequently preprocessed. The encoding of atom names and neighborhood values is achieved using a classic embedding layer from Tensorflow. Other data types are directly extracted from the dataset. Considering the potential challenges posed by protein-ligand complexes in our dataset, which could consist of over 11,000 atoms, we aimed to minimize the number of features. To address this, we combined atom name embedding with neighborhood embeddings. These embeddings were then concatenated with categorical and numerical features, as well as with coordinates.

3.1.5 3D ViT

The resulting embedding serves as the input for our 3D vision transformer block. We adopted an architecture similar to the original transformer with minimal

[2]Angstrom, $1\mathring{A} = 0,1$ nm

modifications. For each input, we calculated positional encoding within a custom layer called **PositionalEmbedding** implemented by employing sine and cosine functions.

```
class PositionalEmbedding(tf.keras.layers.Layer):
        def __init__(self, dim, *args, **kwargs):
            assert dim % 2 == 0  # The ‘dim‘ needs to be even to
    have the same number of sin&cos.
            super().__init__(*args, **kwargs)
            self.dim = dim

        def get_config(self):
            return {"dim": self.dim}

        def call(self, inputs, batch_size, max_seq_len):
            ...
            positional_embeddings = ...
            positional_embeddings_batch = tf.reshape(tf.tile(
    positional_embeddings, multiples=[batch_size, 1]), shape=[
    batch_size, max_seq_len, self.dim])

            return positional_embeddings_batch
```

A notable departure from the original paper is the omission of the decoder component. This decision was straightforward, as our transformer solely requires the output weights. Instead of a decoder, we employed a single dense layer with one unit to generate predictions.

Transformer Block

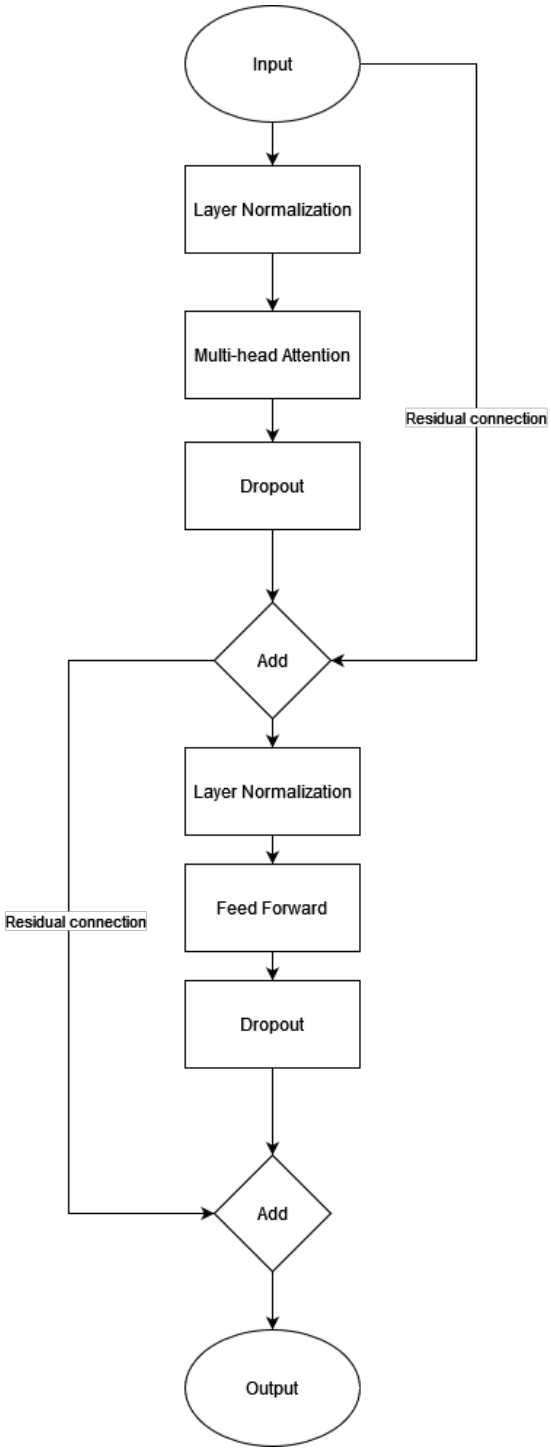
The architecture of the transformer block is relatively simple (3.3), implemented by custom layer **TransformerBlock**.

```
class TransformerBlock(tf.keras.layers.Layer):
    def __init__(self,num_heads, ffn_size, key_dim, value_dim,
    mlp_ratio=4):
        super(TransformerBlock, self).__init__()
        self.norm1 = tf.keras.layers.LayerNormalization()
        self.self_attention = tf.keras.layers.MultiHeadAttention(
    num_heads=num_heads, key_dim=key_dim, value_dim=value_dim)
        self.norm2 = tf.keras.layers.LayerNormalization()
        self.mlp = tf.keras.Sequential([
            tf.keras.layers.Dense(mlp_ratio * ffn_size),
            tf.keras.layers.ReLU(),
            tf.keras.layers.Dense(ffn_size)
        ])
        self.dropout = tf.keras.layers.Dropout(rate=args.dropout)

    def call(self, inputs):
        ...
```

It begins with a sublayer utilizing multi-head attention, followed by a feed-forward neural network sublayer consisting of two dense layers with a ReLU activation function in between. Residual connections are employed within each sublayer. While the original paper suggested the use of learning rate warmup (Vaswani et al. [2017]) to facilitate convergence in the plain transformer, we used normalization before sublayer, as proposed in a 2020 paper (Ruibin Xiong [2020]), removed

the need for learning rate warmup. Leveraging this insight, we eliminated the requirement for learning rate warmup in our approach.



Source: Author

Fig. 3.3. *Architecture of our transformer block. All used layers are from Tensorflow library*

Putting all of these elements together, we have the complete 3D vision transformer, implemented as class ViT3D:

```
class ViT3D(tf.keras.Model):
    def __init__(self, num_heads, ffn_size, num_layers, key_dim,
    value_dim, mlp_ratio=4):
        super(ViT3D,self).__init__()
        self.postional_embedding = PositionalEmbedding(ffn_size)

        self.transformer_blocks = [TransformerBlock(num_heads,
    ffn_size, mlp_ratio, key_dim, value_dim) for _ in range(
    num_layers)]

        self.normalization = tf.keras.layers.LayerNormalization()

    def get_max_length(self, tensor):
        ...

    def call(self, inputs):
        ...
```

3.2 Datasets

For our prototype model, we have decided to use the HOLO4K dataset. It was partitioned into three subsets: the train set, development set, and test set, in the ratio of 8:1:1. This dataset consists of approximately 4000 protein-ligand complexes obtained directly from the Protein Data Bank (PDB). Each amino acid within these complexes is annotated with a classification indicating its membership in a binding site. These classifications were determined experimentally, utilizing methods described in chapter 1. Notably, this dataset was first used by Schmidtke et al. [2010].

Feature Name	Description
atomName	atom name from PDB file
coordinateX	coordinate x in Angstroms
coordinateY	coordinate y in Angstroms
coordinateZ	coordinate z in Angstroms
aaHydrophobic	binary attribute, 1 for hydrophobic residues
aaHydrophilic	binary attribute, 1 for hydrophilic residues
aaAliphatic	binary attribute, 1 for aliphatic residues
aaAromatic	binary attribute, 1 for aromatic residues
aaSulfur	binary attribute, 1 for residues containing sulfur
aaHydroxyl	binary attribute, 1 for hydroxyl group containing residues
aaBasic	binary attribute, 1 for basic residues
aaAcidic	binary attribute, 1 for acidic residues
aaAmide	binary attribute, 1 for amide group containing residues
aaCharge	categorical attribute, 1 for positive charge, -1 for negative charge
aaHBondDonor	binary attribute, 1 for H-bond donor containing residues
aaHBondAcceptor	binary attribute, 1 for H-bond acceptor containing residues
aaHBondDonoAcceptor	binary attribute, 1 for residues that have H-bond donor and acceptor
aaPolar	binary attribute, 1 for polar residues
aaIonizable	binary attribute, 1 for ionizable residues
atAromatic	Volsite atomic level features
atCation	Volsite atomic level features
atAnion	Volsite atomic level features
atHydrophobic	Volsite atomic level features
atAcceptor	Volsite atomic level features
atDonor	Volsite atomic level features
bfactor	B-factor number of the atom from pdb file
aaHydrophatyIndex	side-chain hydropathy index with values range from -4,5 to 4,5
aaPropensities	binding propensity, value from 0 to 3,1
atRawValids	Ligand binding propensity for biologically valid ligands, real number from 0 to 3
atRawInvalids	Ligand binding propensity for biologially invalid ligands, real number from 0 to 3
atSasaRawValids	Ligand binding propensity for biologically valid ligands on Solvent Accessible Surface (SAS), real number from 0 to 3
atSasaRawInvalids	Ligand binding propensity for biologically invalid ligands on Solvent Accessible Surface (SAS), real number from 0 to 3
atHydrophobicity	Atom type hydrophobicity scale
atoms	absolute number of protein atoms in the neighborhood (within 6 Angstroms of the point)
atomDensity	number of protein atoms weighted by distance
atomC	number of carbon atoms in neighborhood
atomO	number of oxygen atoms in the neighborhood
atomN	number of nitrogen atoms in the neighborhood
hDonorAtom	number of H-bond donor atoms in the neighborhood
hAcceptorAtom	number of H-bond acceptor atoms in the neighborhood
protrusion	number of all protein atoms within 10 Angstrom radius of the point

Table 3.1

4. Results

We trained our 3D vision transformer on faculty's cluster, since it is computationally expensive for such huge examples. Even though we used cluster with multiple high end GPUs such as NVIDIA Tesla V100 SXM2 32 GB and Intel Xeon Gold 5218 CPU with 16 cores and 384 GB RAM we had to remove complexes with more than 5000 atoms. The bottleneck is caused mainly by self-attention due to its quadratic complexity (Vaswani et al. [2017]).This modification allowed us to run the model.

We used a batch size of 1, mainly due to attention's quadratic complexity. Our model has a large number of hyperparameters, which are listed in figure 4.1 with our chosen values. We intend to search for better values in the future, but so far, due to high space complexity, we chose most of the parameters empirically.

The one parameter that we were able to optimize is learning rate. We chose logarithmic values from 1e-2 to 1e-7. The best learning, determined by keras tuner, was 1e-6. We also chose AdamW as the optimizer (Loshchilov and Hutter [2019]).

We trained our model for 2 epochs, which was our intention for the prototype version. Since the dataset is heavily imbalanced towards negative value, we use weighted binary cross entropy as our loss function. The ratio between negative and positive targets is approximately 15:1.

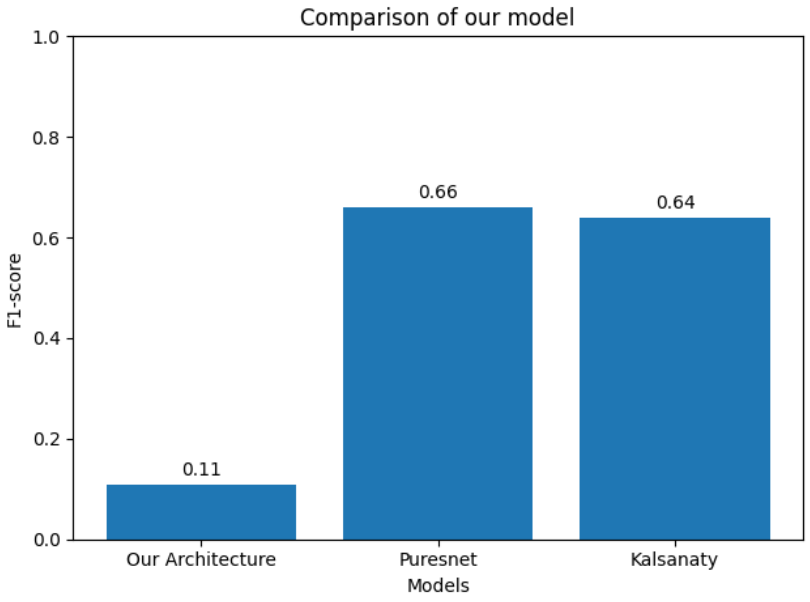
We used 9 metrics, showed in the table 4.2. We focused mostly on MCC and F1 score, since this is common metrics used by bioinformaticians for this task (Krivák and Hoksza [2018]). Unfortunately, our model underperformed during our experiment in every single metric. Our MCC -0.0050 suggests that our predictions are almost uncorrelated with real targets. It is hard to compare different algorithms for this task since, they usually use different metrics on different datasets. This can lead to misinterpretation of the results. However, our model, comparing to commonly used algorithms (Kandel et al. [2021], Stepniewska-Dziubinska et al. [2020]) shows overwhelmingly negative results with F1 score 0.1085 compared to Puresnet 0.66 and Kalsanaty 0.64 (4.1). This comparison is only illustrative, since these models where trained on different datasets, but there is clear that our model is nor performing well enough right now.

Hyperparameters	**Used value**
Embedding size	16
Number of heads for attention	8
Number of transformer blocks	3
Key and value size for attention	4
Feed forward network ratio	4
Learning rate	1e-6

Table 4.1. *Table of hyperparameters with values used in our model*

Metrics	Description
True positives (TP)	Number of correctly predicted atoms in binding sites
True negatives (TN)	Number of correctly predicted atoms outside of binding sites
False positives (FP)	Number of incorrectly predicted atoms outside of binding sites
False negatives (FN)	Number of incorrectly predicted atoms inside of binding sites
Precision (P)	P = TP / (TP + FP)
Recall (R)	R = TP / (TP + FN)
Accuracy (ACC)	ACC = (TP + TN) / (TP + TN + FP + FN)
F1-score (F1)	F1 = 2 TP / (2TP + FP + FN)
Mathews correlation coefficient (MCC)	$MCC = \frac{(TP*TN-FP*FN)}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}$

Table 4.2. *Table of measured metrics*



Source: Author

Fig. 4.1. *Comparison of our model with Puresnet and Kalsanaty.*

5. Discussion

As stated in previous chapter, our model underperformed compared to other machine learning methods for detection of protein-ligand binding sites. There are various reasons, why this happened. We were training on very small data size, only about 2000 protein-ligand complexes. Even with images, transformers are not able to compete with state-of-the-art CNN on small datasets (Dosovitskiy et al. [2021]). This disadvantage is therefore multiplied by amount of information in 3D space.

The next problem is complexity of training. Since attention mechanism has quadratic time complexity (Vaswani et al. [2017]), one epoch took around 5 hours on our faculty cluster. With combination of "data hungry" architecture, this can rapidly increase training time.

Another limitation we have encountered is the near impossibility to make prediction on big complexes with current available hardware. This is also caused mostly by attention mechanism and its high space complexity. All of these factors can significantly restrict model capacity. However, we believe that if modified, our model can produce much better results with its insights on long dependencies and long-distance interactions.

5.1 Future work

Even though this prototype did not provide good enough results, we believe that this architecture can be useful after some modifications. In this chapter, we present the next steps we aim to take in our future work.

Firstly, we would like to create dataset with smaller examples. Similarly as in original vision transformer, we would like to use patches as small inputs for model (Dosovitskiy et al. [2021]). This can be achieved in two ways: linear slices and 3D slices.

Linear slices can be created from PDB file, just simply cutting protein in arbitrary position. However, this can lead to slices that are not close together in 3D conformation of protein. The other option, 3D slices, is more complicated but it can solve this problem. We can create slices according to 3D coordinates of atoms and therefore create slices that are close together and possible binding sites. This method is similar to voxelization of the 3D space. On this slices we can calculate local attention, but we will lose the biggest advantage of attention mechanism - mapping of long-distance relations. This process can help with finding orthosteric binding sites but can have problems with finding allosteric, since those can change conformation of binding site from greater distance (Nussinov and Tsai [2012]). This slicing can also increase speed of training, since the attention is computed only locally and not through whole protein-ligand complex. Also, on smaller inputs, we can utilize bigger batch size which better aproximates gradient of loss function. The change of batch size also influences learning rate, we can use bigger value, therefore speed up training.

To solve this kind of limitation, we can look at the complexes on a different level of granularity instead of working only with patches. We can use protein-ligand complexes represented not by atoms, but by whole amino acids. The

smallest amino acid, glycine, consists of 10 atoms, therefore, it could reduced the dataset size roughly tenfold. Then we can exploit attention mechanism globally, to maximize its potential. Due to lack of input data, we would like to also scrape some well known databases for more protein-ligand complexes.

Conclusion

Computer-aided drug design became a popular method in recent years, thanks to big improvement in machine learning models. Identifying bindig sites, a region of a protein that associates with a ligand, in protein-ligand complexes is the first step for understanding protein function and therefore crucial for possible perturbation. Since transformers and vision transformers proved themselves as the state-of-the-art models for many NLP and computer vision tasks, we tried to use its attention mechanism on biological data.

In this thesis we tried to explore possibilities of using 3D vision transformer for detection of protein-ligand complexes. We designed our prototype of 3D vision transformer, inspired by original architecture Vaswani et al. [2017] and processed dataset consisting of approximately of 2000 protein-ligand complexes. In comparison with other recent techniques, our model showed results that are nowhere near state-of-the-art models. There are various reasons, why this model could not compete with other methods. Transformer's requirement for huge amount of data proved to be the biggest problem. With high computational cost, it is impossible to determine fully capacity of the model just with the prototype. Even though our test results did not show any correlation, we believe that our model can be trained to obtain better results, mainly due to few promising results during search for optimal parameters.

Bibliography

Mart´ın Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Man´e, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Vi´egas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. 2016.

Dzmitry Bahdanau, KyungHyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. 2015.

Gail J. Bartlett, Craig T. Porter, Neera Borkakoti, and Janet M. Thornton. Analysis of catalytic residues in enzyme active sites. *Journal of Molecular Biology*, 324(1):105–121, nov 2002. doi: 10.1016/s0022-2836(02)01036-7.

Carl Branden and John Tooze. *Introduction to Protein Structure*. 1999.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, , Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. 2020.

Julian Lewis David Morgan Martin Raff Keith Roberts Peter Walter Bruce Alberts, Alexander Johnson. *Molecular biology of the cell*. CRC Press, 2017.

Karen Hopkin Alexander D Johnson Alexander Johnson Julian Lewis Martin Raff Keith Roberts Peter Walter Bruce Alberts, Dennis Bray. *Essential Cell Biology*. 2019.

Engelbert Buxbaum. *Fundamentals of ProteinStructure and Function*. 2015.

Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. 2020.

Pu-Chin Chen, Henry Tsai, Srinadh Bhojanapalli, Hyung Won Chung, Yin-Wen Chang, and Chun-Sung Ferng. A simple and effective positional encoding for transformers. 2021.

Peter J. A. Cock, Tiago Antao, Jeffrey T. Chang, Brad A. Chapman, Cymon J. Cox, Andrew Dalke, Iddo Friedberg, Thomas Hamelryck, Frank Kauff, Bartek Wilczynski, and Michiel J. L. de Hoon. Biopython: freely available python

tools for computational molecular biology and bioinformatics. *Bioinformatics*, 25(11):1422–1423, mar 2009. doi: 10.1093/bioinformatics/btp163.

Jeremy Desaphy, Karima Azdimousa, Esther Kellenberger, and Didier Rognan. Comparison and druggability prediction of proteinligand bindingsites from pharmacophore-annotated cavity shapes. 2012. doi: .org/10.1021/ci300184x.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. 2019.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 wordds transformers for image recognition at scale. (advising), 2021.

Vincent Le Guilloux, Peter Schmidtke, and Pierre Tuffery. Fpocket: An open source platform for ligand pocket detection. *BMC Bioinformatics*, 10(1), jun 2009. doi: 10.1186/1471-2105-10-168.

Abdul Mueed Hafiz, Shabir Ahmad Parah, and Rouf Ul Alam Bhat. Attention mechanisms and deep learning for machine vision:a survey of the state of the art. 2021.

M. Hernandez, D. Ghersi, and R. Sanchez. Sitehound-web: a server for ligand binding site identification in protein structures. *Nucleic Acids Research*, 37 (Web Server):W413–W416, apr 2009. doi: 10.1093/nar/gkp281.

Andrea Ilari and Carmelinda Savino. *Protein structure determination by X-ray crystallography*, chapter Chapter 3. 2008.

J. Jimenez, S. Doerr, G. Martınez-Rosell, A. S. Rose, and G. De Fabritiis. Deepsite: protein-binding site predictor using3d-convolutional neural networks. 2017.

John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, Alex Bridgland, Clemens Meyer, Simon A. A. Kohl, Andrew J. Ballard, Andrew Cowie, Bernardino Romera-Paredes, Stanislav Nikolov, Rishub Jain, Jonas Adler, Trevor Back, Stig Petersen, David Reiman, Ellen Clancy, Michal Zielinski, Martin Steinegger, Michalina Pacholska, Tamas Berghammer, Sebastian Bodenstein, David Silver, Oriol Vinyals, Andrew W. Senior, Koray Kavukcuoglu, Pushmeet Kohli, and Demis Hassabis. Highly accurate protein structure prediction with AlphaFold. *Nature*, 596(7873):583–589, jul 2021. doi: 10.1038/s41586-021-03819-2.

Jeevan Kandel, Hilal Tayara, and Kil To Chong. PUResNet: prediction of protein-ligand binding sites using deep residual neural network. *Journal of Cheminformatics*, 13(1), sep 2021. doi: 10.1186/s13321-021-00547-7.

Nickolay A. Khazanov and Heather A. Carlson. Exploring the composition of protein-ligand binding sites on a large scale. *PLoS Computational Biology*, 9 (11):e1003321, nov 2013. doi: 10.1371/journal.pcbi.1003321.

Radoslav Krivák and David Hoksza. P2rank: machine learning based tool for rapid and accurate prediction of ligand binding sites from protein structure. 2018. doi: .org/10.1186/s13321-018-0285-8.

Jean Lahoud, Jiale Cao, Fahad Shahbaz Khan, Hisham Cholakkal, and Rao Muhammad Anwer. 3d vision with transformers: A survey. 2022.

Tianyang Lin, Yuxin Wang, Xiangyang Liu, and Xipeng Qiu. A survey of transformers. 2021.

Yingfei Liu, Tiancai Wang, Xiangyu Zhang, and Jian Sun. Petr: Position embedding transformation for multi-view 3d object detection. 2022.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. 2019.

Ruth Nussinov and Chung-Jung Tsai. The different ways through which specificity works in orthosteric and allosteric drugs. 2012. doi: 10.2174/138161212799436377.

David Prihoda, Jad Maamary, Andrew Waight, Veronica Juan, Laurence Fayadat-Dilman, Daniel Svozil, and Danny A. Bitton. Biophi: A platform for antibody design, humanization, and humanness evaluation based on natural anti. 2022. doi: .org/10.1080/19420862.2021.2020203.

Di He Kai Zheng Shuxin Zheng Chen Xing Huishuai Zhang Yanyan Lan Liwei Wang Tie-Yan Liu Ruibin Xiong, Yunchang Yang. On layer normalization in the transformer architecture. 2020.

Zeyu Chen Zhuowen Tu Saining Xie, Sainan Liu. Attentional shapecontextnet for point cloud recognition. 2018.

Peter Schmidtke, Catherine Souaille, Frédéric Estienne, Nicolas Baurin, and Romano T. Kroemer. Large-scale comparison of four binding site detection algorithms. *Journal of Chemical Information and Modeling*, 50(12):2191–2200, sep 2010. doi: 10.1021/ci1000289.

Arne Schneuing, Yuanqi Du, Charles Harris, Arian Jamasb, Ilia Igashov, Weitao Du, Tom Blundell, Pietro Liò, Carla Gomes, Max Welling, Michael Bronstein, and Bruno Correia. Structure-based drug design with equivariant diffusion models. 2023.

Alexey V. Ereshchenko Sergei A. Evteev and Yan A. Ivanenkov. Siteradar: Utilizing graph machine learning for precise mapping of protein–ligand-binding sites. 2023. doi: .org/10.1021/acs.jcim.2c01413.

Marta M. Stepniewska-Dziubinska, Piotr Zielenkiewicz, and Pawel Siedlecki. Improving detection of protein-ligand binding sites with 3d segmentation. 2020.

William F. Tivol, Ariane Briegel, and Grant J. Jensen. An improved cryogen for plunge freezing. 2008. doi: 10.1017/S1431927608080781.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, and Lukasz Kaiser. Attention is all you need. 2017.

Chenguang Wang, Mu Li, Alexander J. Smola, Amazon Web, and Services. Language models with transformers. 2019.

Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. 2018.

Yi Wang, Zhiwen Fan, Tianlong Chen, Hehe Fan, and Zhangyang Wang. Can we solve 3d vision tasks starting from a 2d vision transformer. 2022.

K Wüthrich. Protein structure determination in solution by nmr spectroscopy. *Journal of Biological Chemistry*, 265(36):22059–22062, dec 1990. doi: 10.1016/s0021-9258(18)45665-7.

Ali Hatamizadeh; Yucheng Tang; Vishwesh Nath; Dong Yang; Andriy Myronenko; Bennett Landman; Holger R. Roth; Daguang Xu. Unetr: Transformers for 3d medical image segmentation. 2021.

Ka Man Yip, Niels Fischer, Elham Paknia, Ashwin Chari, and Holger Stark. Atomic-resolution protein structure determination by cryo-EM. *Nature*, 587 (7832):157–161, oct 2020. doi: 10.1038/s41586-020-2833-4.

Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip Torr, and Vladlen Koltun. Point transformer. 2021.

List of Figures

List of Tables

List of Abbreviations

1D, 2D, 3D	one-, two-, three-dimensional
XRC	X-ray crystallography
NMR	nuclear magnetic resonance (spectroscopy)
3D cryo-EM	3D cryogenic electron microscopy
DNA	deoxyribonucleic acid
NLP	natural language processing
PDB	Protein data bank
XRC	X-ray crystallography

List of Symbols

C	carbon atom
N	nitrogen atom
H	hydrogen atom
O	oxygen atom
C_α	α carbon atom
COOH	carboxillic acid group
NH_2	amino group
Å	Ångstrom, unit of distance
KD	dissociation constant
$\sum$	sum
$x \in X$	x is an element of the set X