NATIONAL AND KAPODISTRIAN UNIVERSITY OF ATHENS

SCHOOL OF SCIENCE
DEPARTMENT OF INFORMATICS AND TELECOMMUNICATION

BSc THESIS

Deep Learning on Point Clouds for 3D Protein Classification Based on Secondary Structure

Alexandros G. Kalimeris

Supervisor: **Ioannis Emiris,** Professor

ATHENS

SEPTEMBER 2019

ΕΘΝΙΚΟ ΚΑΙ ΚΑΠΟΔΙΣΤΡΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Deep Learning on Point Clouds for 3D Protein Classification Based on Secondary Structure

Αλέξανδρος Γ. Καλημέρης

Επιβλέπων: **Ιωάννης Εμίρης,** Καθηγητής

ΑΘΗΝΑ

ΣΕΠΤΕΜΒΡΙΟΣ 2019

BSc THESIS

Deep Learning on Point Clouds for 3D Protein Classification Based on Secondary Structure

Alexandros G. Kalimeris

S.N.: 1115201400056

SUPERVISOR: **Ioannis Emirs,** Professor

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Deep Learning on Point Clouds for 3D Protein Classification Based on Secondary Structure

Αλέξανδρος Γ. Καλημέρης

Α.Μ.: 1115201400056

ΕΠΙΒΛΕΠΩΝ: **Ιωάννης Εμίρης,** Καθηγητής

ABSTRACT

Proteins are macromolecules that regulate a vast amount of biological processes. The spatial structure of proteins is the main determinant of their biological function. Consequently, discovering and researching new efficient methods for classifying proteins based on their 3D shape is an important task with applications in many scientific fields. In the context of this thesis, we explore and examine the capabilities of Deep Neural Networks in performing classification tasks on the complex 3D shapes of proteins. For these purposes, we analyze existing deep learning architectures that showed promising results. Additionally, we test the effectiveness of these architectures by performing a series of protein classification experiments. In our experiments, we represent the geometric 3D shape of proteins as point clouds, a flexible geometric data representation. Also since proteins have different sizes and the deep learning architectures we explore do not consume dynamic size input, we test ways of normalizing the proteins into the same constant size. Finally, we comprehensively present and evaluate the results of our work.

SUBJECT AREA: Deep Neural Networks

KEYWORDS: Protein Classification, Deep Neural Networks, Point Clouds, Secondary Protein Structure, 3D geometrical shape recognition

ΠΕΡΙΛΗΨΗ

Οι πρωτεΐνες είναι μακρομόρια που ρυθμίζουν πληθώρα βιολογικών διεργασιών. Η χωρική δομή των πρωτεϊνών είναι ο κύριος καθοριστικός παράγοντας της βιολογικής λειτουργίας τους. Συνεπώς, η εύρεση νέων αποτελεσματικών μεθόδων ταξινόμησης πρωτεϊνών με βάση την τρισδιάστατη δομή τους είναι ένα σημαντικό έργο με εφαρμογές σε πολλά επιστημονικά πεδία. Στο πλαίσιο αυτής της πτυχιακής εργασίας, εξερευνούμε και εξετάζουμε τις δυνατότητες των Βαθιών Νευρωνικών Δικτύων όσον αφορά την εκτέλεση εργασιών ταξινόμησης σε σύνθετα τρισδιάστατα σχήματα πρωτεϊνών. Για τους σκοπούς αυτούς, αναλύσαμε υπάρχουσες αρχιτεκτονικές βαθιάς μάθησης που έδειξαν πολλά υποσχόμενα αποτελέσματα. Επιπλέον, δοκιμάζουμε την αποτελεσματικότητα αυτών των αρχιτεκτονικών με την εκτέλεση μιας σειράς πειραμάτων ταξινόμησης πρωτεϊνών. Στα πειράματά μας, αντιπροσωπεύουμε το γεωμετρικό τρισδιάστατο σχήμα των πρωτεϊνών ως νέφη σημείων, μια ευέλικτη γεωμετρική αναπαράσταση δεδομένων. Ακόμα, λόγω του ότι οι πρωτεΐνες διαφέρουν μεταξύ τους ως προς το μέγεθος και οι αρχιτεκτονικές βαθιάς μάθησης που εξερευνούμε δεν μπορούν να δεχθούν είσοδο δυναμικού μεγέθους, δοκιμάζουμε τρόπους για την κανονικοποίηση των πρωτεϊνών σε ένα κοινό σταθερό μέγεθος. Τέλος, παρουσιάζουμε ολοκληρωμένα τα αποτελέσματα της δουλειάς μας και τα αξιολογούμε.

ΘΕΜΑΤΙΚΗ ΠΕΡΙΟΧΗ: Βαθιά Νευρωνικά Δίκτυα

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ: Κατηγοριοποίηση Πρωτεϊνών, Βαθιά Νευρωνικά Δίκτυα, Νέφη Σημείων, Δευτερεύουσα Δομή Πρωτεϊνών, Αναγνώριση τρισδιάστατων γεωμετρικών σχημάτων

AKNOWLEDGMENTS

I would like to thank my supervisor Professor Ioannis Emiris for the valuable guidance and feedback he provided during the composition of this thesis. Also, I would like to thank Emmanouil Christoforou for his constant assistance and insight.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

PREFACE

This thesis was written as a part of the BSc program of studies at the Department of Informatics and Telecommunications of the National and Kapodistrian University of Athens.

1. INTRODUCTION

In this thesis, we explore and experiment with Deep Learning architectures that are capable of working directly on 3D Point Clouds in order to handle the problem of Protein Classification based on their secondary structures. In this section, we will briefly explain the problem and provide basic knowledge of the methods used.

1.1 Protein Classification

Proteins are highly complex macro-molecular molecules with a significant role in molecular mechanisms and many biological processes. The 3D spatial structure of a protein determines its biological function.

Classification of proteins based on their 3D spatial structure can prove to be crucial for providing knowledge about protein function and their interaction with other proteins. The classification of proteins remains a difficult and time consuming task. Manual classification is no longer a viable option due to the massive volume of the available data. Considering the recent evolution of Machine Learning and especially Neural Networks and their success on classification tasks, the utilization of those methods can be a valuable and efficient alternative for protein classification.

1.2 Artificial Neural Networks

Artificial Neural Networks (ANN) are computational systems that their architecture is inspired by the biological neural networks that constitute animal brains. ANNs are systems that progressively improve their ability to perform a certain task by analyzing examples without the need of a traditional rule-based algorithm, making them an appealing model for applications that are difficult to express with a traditional algorithm.

ANNs consist of connected nodes (artificial neurons) that receive an input and produce an output that is related to the input, the activation function and the weights. Usually, ANNs are graphically represented as directed graphs with the vertices representing the neurons and the edges the connections between them (Figure 1.1). Typically the neurons are organized in layers.

For ANNs to progressively learn to perform a task, a process of training is needed. During training ANNs use an optimization algorithm to minimize or maximize an objective function. Input is forward propagated through the neuron layers until it reaches the final output layer, then the output is compared with the desired result with the help of a loss function. Afterwards, the output is backpropagated again through the network, from the output layer back to the input layer, using the error values to calculate the gradient of the loss function and update the weights. This method is called gradient descent.

The original goal of ANNs was to solve problems in the same way the human brain would. As of today ANNs are used in a variety of tasks such as image recognition, speech recognition, bioinformatics and others.

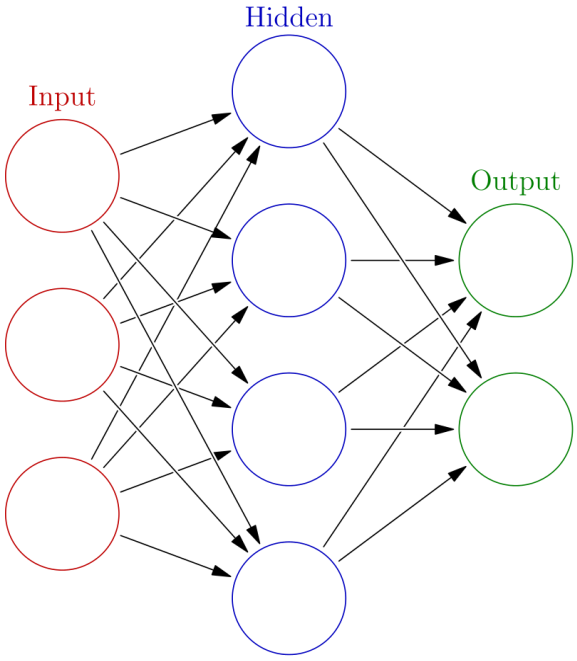

Figure 1.1: graphical representation of a simple Artificial Neural Network as a directed graph. Source: Wikipedia

1.3 Deep Neural Networks

Deep Neural Networks (DNN) are ANNs with multiple hidden layers between the input and the output layer. DNNs can model complex non-linear relationships and can provide better abstraction and feature extraction.

In DNNs each layer learns to transform its input into a more abstract and composite representation. For example, in an image recognition task, the initial input may be a matrix of pixels; the first layer may abstract the pixels and encode edges; the second layer may compose and encode arrangements of edges; the third layer may encode edge arrangements into a nose or an eye; the fourth layer may recognize if the image contains a human face. The extra layers enable composition of features from the previous layers, enabling a DNN to potentially model complex data with fewer units than an ANN.

As a result of extensive research during the last decades, DNNs have been successful and widely used in a variety of tasks such as image and object recognition, natural language processing, automated drug discovery and toxicology and others. The models we explore in this thesis are DNNs.

1.4 Point Clouds

A Point Cloud is a set of data points in space, each element of the set represents the 3D spatial coordinates of a point. Mathematically, it can be represented as a set of points $P = \{p_i | \, i = 1, \ldots, n\}$ where each point p_i is a three dimensional vector of the point's coordinates (x, y, z). Point clouds are an important type of geometric data representation, often being the direct output of 3D scanning processes. However, due to their irregular nature many researchers choose to transform the point clouds into more regular types of 3D data representation such as 3D voxel grids, a process that massively increases the volume of the data. In this thesis, we will experiment on DNN architectures that directly consume Point Clouds with the intention of keeping the voluminosity of the data low without compromising the effectiveness of the methods.

2. DEEP LEARNING ON 3D DATA

2.1 Background and Related work

In this section we will discuss previous and related work on point clouds, deep learning on 3D data and protein classification.

Handcrafted Point Cloud Features

Various tasks in geometric data analysis and processing (classification, semantic segmentation etc.) often use hand crafted features. Those features usually capture some certain statistical properties of the points and are designed to be invariant to certain transformations. Those features may be categorized as intrinsic [1], [2], [3] or extrinsic [4], [5], [6]. Handcrafting features for a specific task is not a trivial task.

Deep Learning on Geometric 3D Data

Due to the variety of popular representations of 3D data there are multiple different approaches from researchers. Following the breakthrough of Convolutional Neural Networks (CNN) in computer vision [7] there has been strong interest in adapting similar methods on geometric 3D tasks, [8], [9], [10] are the pioneers of using CNNs on voxelized 3D shapes. However that approach is gated by its resolution due to the data sparsity and also by the computational cost of applying 3D convolution. There have been attempts to deal with the sparsity problem such as FPNN [11] and Vote3D [12] but processing very large 3D point clouds was still challenging. Recently, PointNet [13] exemplified a deep learning method directly on point clouds, which is one of the models we will thoroughly go over in this thesis. Following that work, there have been efforts to capture local features in 3D shapes in PointNet++ [14] and in Dynamic Graph CNN [15] which will also be analyzed further in this thesis.

Protein Classification Methods

There have been previous efforts to use neural networks for protein classification tasks. In some approaches such as [16] and [17] a graph representation of each 3D protein shape was used. More recently, a deep learning model [18] that worked on 3D voxel grids representation of proteins showed promising results.

2.2 PointNet

PointNet is a DNN designed to directly consume 3D point clouds while respecting the permutation invariance of the points in the set. The key to achieving this is the usage of a single symmetric max pooling function. Their idea is to approximate a general function defined on the point set by applying a symmetric function on transformed elements in the set :

$$\boldsymbol{f}(\{\boldsymbol{x_1},\dots,\boldsymbol{x_n}\}) \approx \boldsymbol{g}(\boldsymbol{h}(\boldsymbol{x_1}),\dots,\boldsymbol{h}(\boldsymbol{x_n}))$$

where $f : 2^{R^N} \to R$, $h : R^K \to R^K$ and $g : \underbrace{R^K \times \dots \times R^K}_{n} \to R$ is a symmetric function.

They approximate h by a multi-layer perceptron network and g by a composition of a single variable function and a max pooling function and showcase that it is working well through their experiments.

Exploring the architecture of the classification network of PointNet (Figure 2.1) we can see that it consumes a set of n points as an input. An alignment of the input in a canonical space is achieved by predicting an affine transformation matrix with the use of a mini-network called T-net (Figure 2.1). After that, the data is passed to a Multi Layer Perceptron (MLP), the output of the MLP is going through another T-net for a similar transformation in order to predict a feature transformation matrix to align features from different point clouds. The output is then passed through another MLP, the output of this MLP is then fed through a max pooling layer, the symmetrical function, that aggregates the point features and creates a global descriptor which in turn is fed through a final MLP that outputs k classification scores where k is the number of classes.

Considering that the approach of directly working and manipulating on point clouds was pioneered by PointNet and that the code is available online we think that it is a good starting point for testing how well DNNs that handle point clouds would perform on point clouds of highly complex 3D shapes such as proteins.

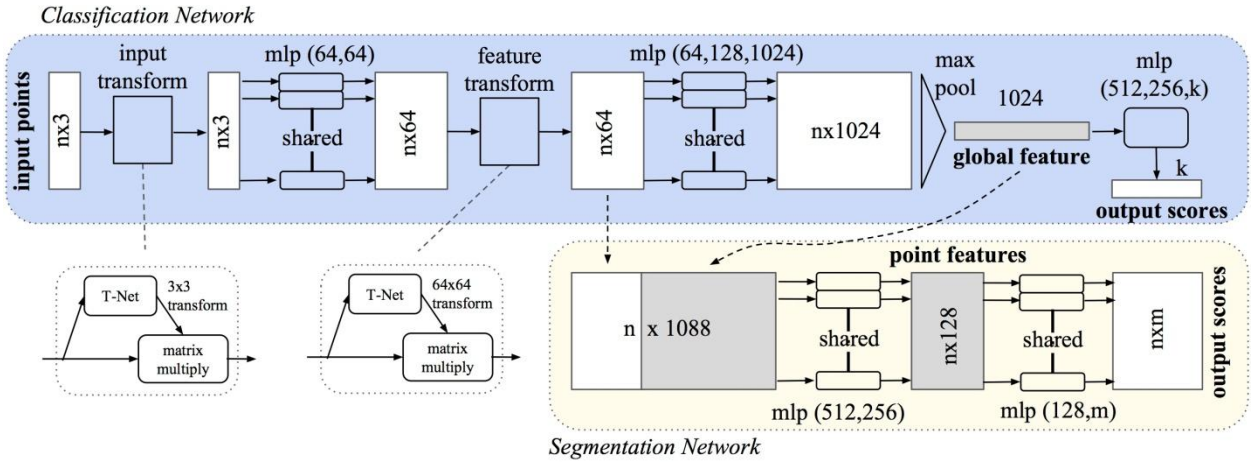

Figure 2.1: The architecture of PointNet. Source: PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation

2.3 Dynamic Graph CNN

Dynamic Graph CNN (DGCNN) is another DNN that works directly on Point Clouds. They propose a new approach heavily inspired by PointNet and convolution operations. Instead of learning on completely individual points like PointNet, they try to take advantage of local geometry structures by forming a local neighborhood graph in order to apply convolution like operations on it, an approach mainly based graph neural networks. For achieving this, they propose a novel convolution like operation called EdgeConv which has permutation invariance, translation invariance and non-locality properties. Unlike other graph Convolutional Neural Networks (CNN) their graph is not static but updated after each layer, as the k nearest neighbors of a point can change

after each layer as the proximity in input space can differ from the proximity in feature space.

Considering a point cloud $P = \{p_i | \, i = 1, \ldots, n\} \subseteq R^N$ where each point p_i is a three dimensional vector of the point's coordinates (x, y, z), they compute a directed graph $G = (V, E)$ representing local point structure where $V = \{1, \ldots, n\}$ and $E \subseteq V \times V$ are the vertices and edges respectively. The graph G is the k nearest neighbor graph of P in R^N, the graph also contains self loops, meaning each node can point to itself. They define edge features as $e_{ij} = h_\Theta(p_i, p_j)$ where $h_\Theta : R^N \times R^N \to R^K$ is a nonlinear function with a set of learnable parameters Θ. Finally they define the EdgeConv operation as:

$$\acute{p_i} = \square_{j:(i,j) \in E} \; h_\Theta(p_i, p_j)$$

where $\square$ is a symmetric aggregation operation like $\sum$ or max on the edge features associated with all the edges emanating from each.

Delving into the architecture of DGCNN (Figure 2.2), we can see that the proposed model takes an input of n points that after an initial spatial transform layer they are alternately fed through EdgeConv and simple fully connected layers. The outputs of the last EdgeConv layer are globally aggregated to form an one dimensional global descriptor which is then fed through a max pooling layer and an MLP in order to produce c classification scores.

DGCNN achieves state-of-the-art performance on several benchmarks such as ModelNet40 [24] for classification and also provide their code online. In this paper, we will use the code in order to experiment on proteins that have more complex 3D shapes. We also believe that taking advantage of local geometry will be beneficial for more accurately classifying those shapes.

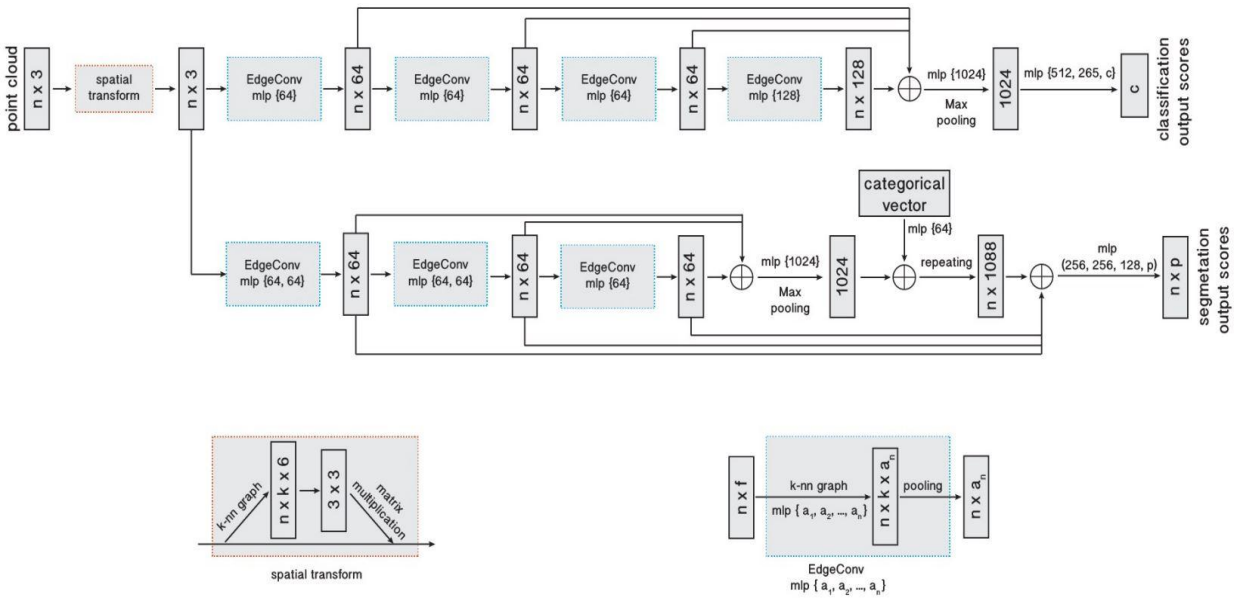

Figure 2.2: The architecture of Dynamic Graph CNN. Source: Dynamic Graph CNN for Learning on Point Clouds

3. EXPERIMENTS

3.1 Datasets for Structural Classification of Proteins

3.1.1 The Protein Data Bank Format

There is a wide variety of extensively developed public databases containing structural information of proteins such as Protein Data Bank (PDB) [19]. The file format initially used by the PDB was called the PDB file format and is now adopted almost universally by the other protein structure related databases. The PDB files contain atomic coordinates, citations, primary and secondary structure information, crystallographic structure and other experimental data. The PDB file format documentation can be found at [20].

For the experiments conducted as a part of this thesis, we will use the ATOM entries of the PDB files that provide the 3D spatial coordinates of the atoms of a protein in order to create point clouds that can be directly processed by the deep learning architectures we are exploring.

3.1.2 Protein Structure Classification Databases

There are many databases that use protein structures deposited in the PDB. The most useful for the purposes of this thesis are protein structure classification databases such as CATH [21] and SCOP [22]. The datasets we used are based on the CATH hierarchical classification specifically on the Class level, which is determined by the overall secondary structure of the domain. There are four main structural classes of proteins according to CATH:

1. mainly a (mainly consisting of a-helices),
2. mainly b (mainly consisting of b-sheets),
3. a/b (a-helixes and b-sheets alternating along the protein backbone),
4. a+b (a-helices and b-sheets occur separately along the protein backbone).

For the rest of this thesis we will be focusing on the classification of proteins between the mainly alpha and mainly beta classes. We use the Cath-Dataset-Non-Redundant-S20 found at CATH website. This dataset contains the PDB files of a non-redundant subset of CATH protein domains. In total, there are 3987 mainly alpha domains and 3159 mainly beta domains in the dataset. Also, we use a manually downloaded dataset from RCSB [23]. We downloaded 2500 mainly alpha and 2500 mainly beta proteins according to CATH classification with the help of the advanced search interface, that RCSB provides, that integrates CATH classification. This dataset contains PDB files from RCSB that include all entries (not only non-redundant domains), as well as the header metadata.

3.1.3 **Dataset Generation**

The data we acquired from CATH and RCSB required some additional processing. PointNet, as well as, DGCNN receive their input into HDF5[1] file format and for that reason we used some simple python scripts to transfer the atomic coordinates from the PDB files into appropriately formatted HDF5 files.

The number of atoms of each protein is different, while the deep learning architectures we explore take a fixed size point cloud as their input. In order to overcome this problem we needed to make all the point clouds sampled from PDB files have the same size. Given a fixed input size num_points, if a protein contains more atoms than that threshold we randomly sample num_points atoms from the protein. On the other hand, if a protein contains less atoms than num_points we "pad" the point clouds using two approaches (Figure 3.1). We either add zero vectors until the point cloud reaches the size of num_points (zero padding) or we copy an initial segment of the molecule (copy padding). We observed a better result with the copy padding method as we hypothesize it reinforces the local features, so that will be the most used padding method in this thesis.

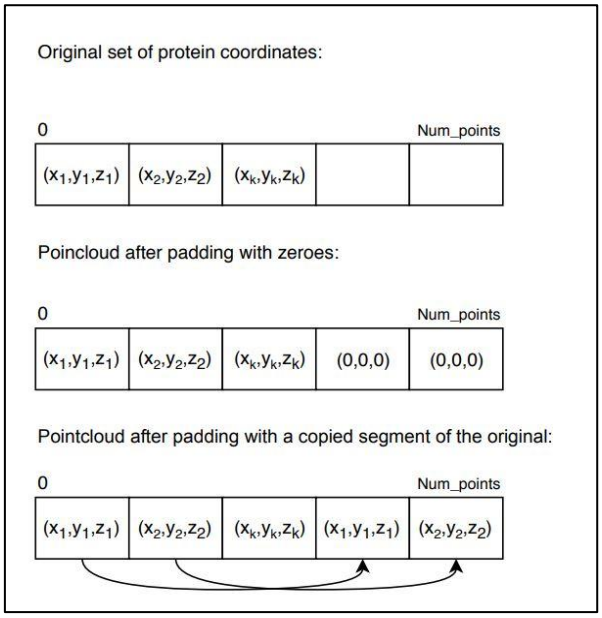

Figure 3.1: Graphical representation of the padding methods

The protein backbone is what holds the protein together and mainly determines its structure, for that reason we also generated more separate datasets, in which we only take into consideration atoms that form the backbone (C, CA, N, O) from each PDB file. The PDB files from RCSB include the full header and metadata. With the help of that metadata, namely the secondary structure section, we also prepared other datasets that only keep points that belong in the secondary structures of a protein, the a-helices and the b-sheets.

[1] www.hdfgroup.org [Accessed: 07/09/2019]

Lastly, we also wanted to test the deep learning architectures in a less complex classification task, classifying a single secondary structure into either an a-helix or a b-sheet, thus we generated another dataset that contains only a-helices or b-sheets as separate entities to be classified.

3.1.4 Datasets based on RCSB PDB files

In this section we will describe the datasets we generated based on the PDB files we obtained from RCSB. Following every dataset we will present histograms that show the distribution of proteins based on the number of atoms they contain or the number of alpha carbons. We should also note that regarding the other backbone atoms (C, N, O) proteins follow the same distribution as the one regarding CA.

Proteins consisting any number atoms.

We use point clouds formed from 5000 manually downloaded proteins from RCSB that 2500 are mainly alpha and 2500 are mainly beta. For the training procedure, the proteins were separated into 2250 mainly alpha and 2250 mainly beta for training and 250 mainly alpha and 250 mainly beta for testing. We used the copy method for padding.

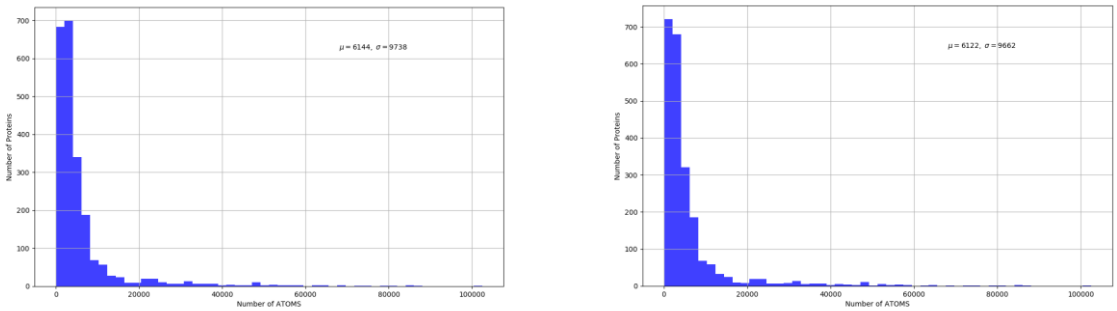

Figure 3.2: Mainly alpha proteins (left) and mainly beta proteins (right) based on their number of atoms.

Proteins consisting any number atoms (backbone atoms only)

The same as the previous dataset except that we only keep backbone atoms.

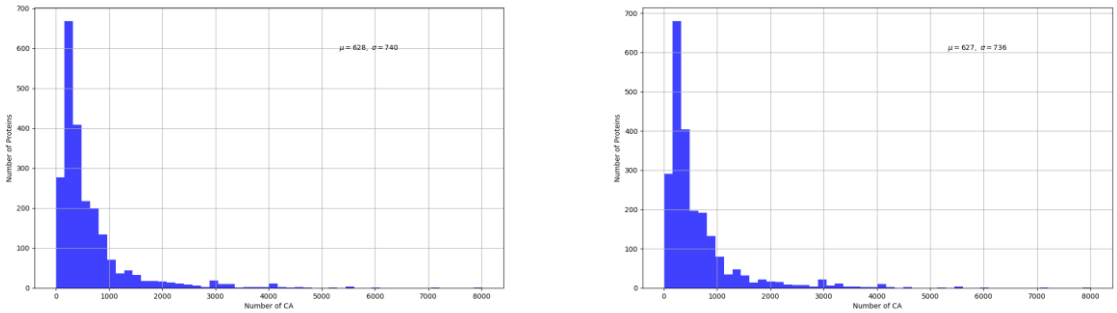

Figure 3.3: Mainly alpha proteins (left) and mainly beta proteins (right) based on their number of alpha carbons.

Isolated Secondary Structures

We isolate the secondary structures of the 5000 proteins downloaded from RCSB, with the help of the secondary structure section of the PDB files. There are 2250 point clouds formed from the secondary structures of mainly alpha proteins and 2250 from the secondary structure of mainly beta proteins used for training and 250 mainly alpha and 250 mainly beta used for testing. We used the copy method for padding.

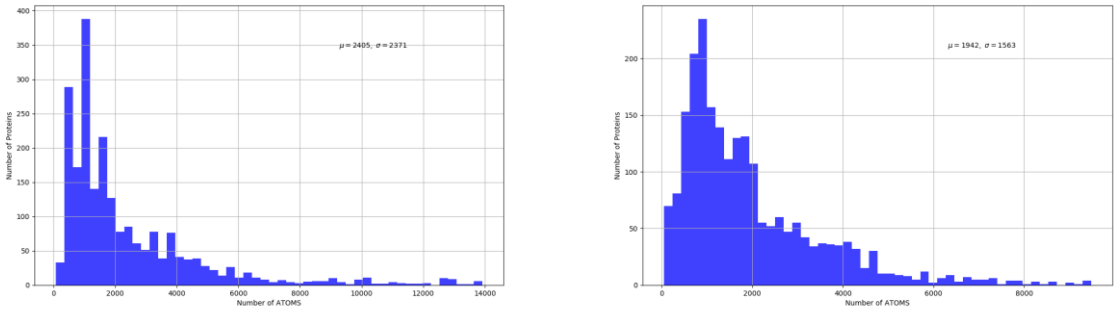

Figure 3.4: Mainly alpha proteins (left) and mainly beta proteins (right) based on the number of atoms that belong into their secondary structures.

Isolated Secondary Structures (backbone atoms only)

The same as the previous dataset except that we only keep backbone atoms.

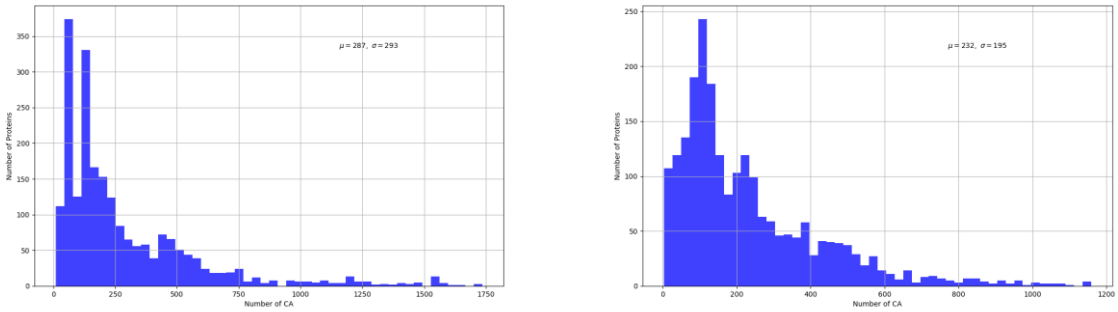

Figure 3.5: Mainly alpha proteins (left) and mainly beta proteins (right) based on the number of alpha carbons that belong into their secondary structures.

Helix and Sheet dataset

From all the 5000 proteins downloaded from RCSB we isolate 15000 a-helices and 15000 b-sheets. For the training procedure, we separate the data into 13500 a-helices and 13500 b-sheets for training and 1500 a-helices and 1500 b-sheets for testing. We used the copy method for padding.

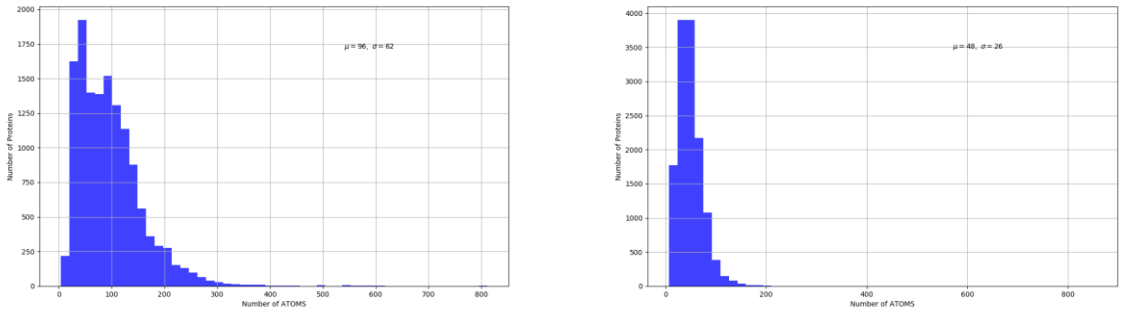

Figure 3.6: Alpha helices (right) and beta sheets (left) based on the number of atoms

3.1.5 Datasets based on CATH-Dataset-Non-Redundant-S20

In this section, in the same way, we will describe the datasets we generated based on the PDB files of the CATH-Dataset-Non-Redundant-S20. Following every dataset we will present histograms that show the distribution of proteins based on the number of atoms they contain or the number of alpha carbons. We should also note that regarding the other backbone atoms (C, N, O) proteins follow the same distribution as the one regarding CA.

Proteins consisting of more than 1024 atoms

In the original dataset we downloaded from CATH there were 1699 mainly alpha and 1503 mainly beta protein domains that had more than 1024 atom entries. We randomly separated those into 1529 mainly alpha and 1353 mainly beta for training and 170 mainly alpha and 150 mainly beta for testing.

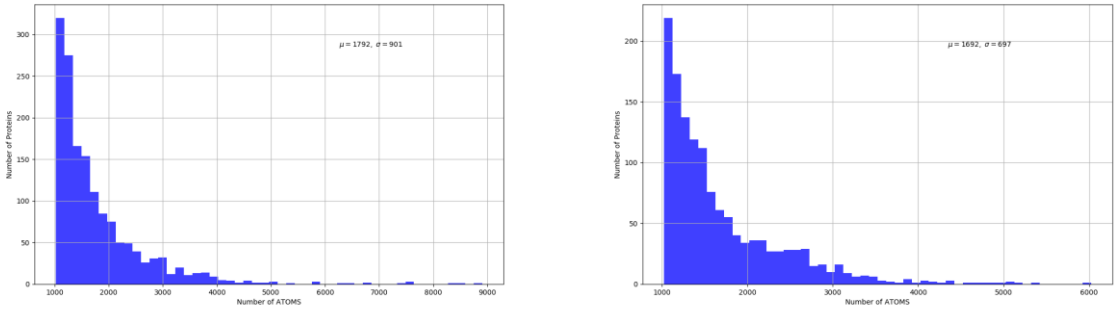

Figure 3.7: Mainly alpha proteins (left) and mainly beta proteins (right) with more than 1024 atoms based on their number of atoms.

Proteins consisting of more than 1024 atoms (backbone atoms only)

From the aforementioned dataset we take only the backbone atoms in consideration and used both zero padding and copy padding when needed.

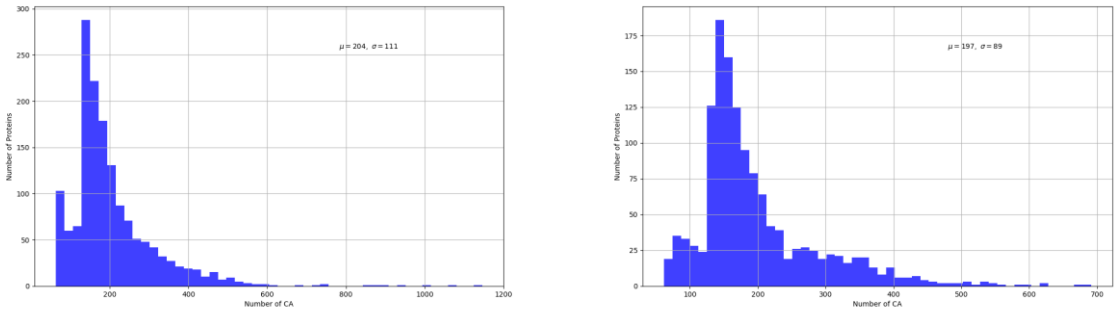

Figure 3.8: Mainly alpha proteins (left) and mainly beta proteins (right) with more than 1024 atoms based on their number of alpha carbons.

Proteins consisting of any number of atoms

We use 5000 proteins from Cath-Dataset-Non-Redundant-S20, 2500 mainly alpha and 2500 mainly beta. The proteins are separated into 2250 mainly alpha and 2250 mainly beta for training and 250 mainly alpha and 250 mainly beta for testing. We used the copy method for padding.

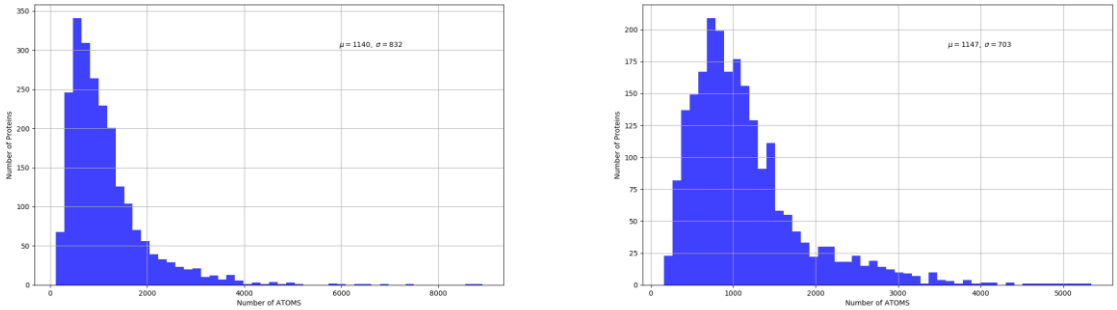

Figure 3.9: Mainly alpha proteins (left) and mainly beta proteins (right) with more than 1024 atoms based on their number of atoms.

Proteins consisting of any number of atoms (backbone atoms only)

The same as the previous dataset except that we only keep backbone atoms.

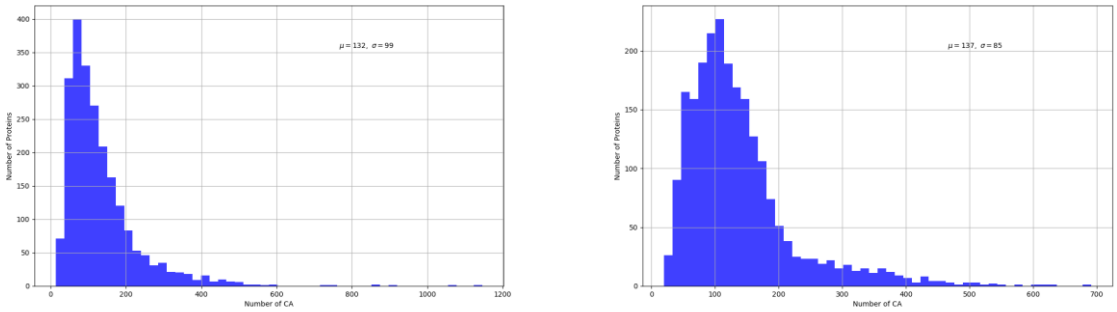

Figure 3.10: Mainly alpha proteins (left) and mainly beta proteins (right) based on their number of alpha carbons.

3.2 Evaluation Metrics

For the experiments conducted as a part of this thesis, we used a variety of metrics such as Accuracy, Precision and Recall, for the purpose of evaluating the effectiveness of the classification task performed by the DNN models we used.

Accuracy

Accuracy is defined as the ratio of all the correct predictions to the total number of predictions. It is a helpful metric that can capture the general effectiveness of a classification model, when the number of samples of each class is about equal.

$$Accuracy = \frac{Correct\ Predictions}{Total\ Predictions}$$

Precision and Recall

The results of a binary classification experiment can be summarized using a Confusion Matrix (Figure 3.11).

	Predicted Positives	Predicted Negatives
Actual Positives	True Positives	False Negatives
Actual Negatives	False Positives	True Negatives

Figure 3.11: Confusion Matrix

Consequently, Precision and Recall can be defined as follows:

$$Precision = \frac{True\ Positives}{True\ positives + False\ Positives}$$

$$Recall = \frac{True\ Positives}{True\ Positives + False\ Negatives}$$

Precision represents the fraction of relevant instances among the retrieved instances, while Recall represents the fraction of relevant instances that have been retrieved over the total amount of relevant instances. Precision and Recall can offer valuable insight on the exactness and the quality of a classification experiment. Considering the protein class “mainly alpha” as the positive outcome of a prediction, we are able to similarly use the aforementioned metrics in our experiments.

3.3 Training Procedure and parameters

All the training procedures for both PointNet and DGCNN have been carried out on Google's cloud based data science platform called Google Colab[2], with the usage of a Nvidia Tesla T4 GPU provided by the platform. The duration of the training procedure for each experiment was 200 – 300 epochs due to the demanding and time consuming nature of the task and took about 4-5 hours to finish for 200 epochs and 6-7 hours for 300 epochs respectively.

PointNet and DGCNN, as any other Deep Neural Network, have a set of parameters that need to be initialized before the training process begins. DGCNN code is heavily inspired from PointNet; as a result both models have a similar set of parameters that need to be initialized. For the experiments conducted as a part of this thesis, we used the following parameter values:

- $Batch\ size\ =\ 25$
- $Decay\ rate\ =\ 0.7$
- $Decay\ step\ =\ 200000$
- $Learning\ rate\ =\ 0.001$
- $Optimizer\ =\ adam$
- $Num\ points\ =\ 128/1024/2048$ (depending on the dataset of the experiment)

Both networks use dropout layers on the final MLP with the purpose of avoiding overfitting. The keep probability value is 0.7 and 0.5 for PointNet and DGCNN respectively.

We should also note that in the case of DGCNN a graph of the k nearest neighbors is calculated between each layer. For all the following experiments $k = 20$.

3.4 Result Visualization

The experiment results will be visualized by graphs that present the Accuracy and Loss value of the test set on each epoch throughout the course of each experiment. We will also present a graph of the loss value of the training set on each epoch. Each graph consists of one "smoothened" curve, heavily inspired by the smoothen function of TensorBoard[3], that attempts to capture a more intuitive pattern of the value progression throughout the experiment. Additionally, another more transparent curve is shown on each graph that represents the raw measurement values.

[2] colab.research.google.com [Accessed: 07/09/2019]

[3] www.tensorflow.org/tensorboard [Accessed: 07/09/2019]

3.5 Experiments

In this section we will present our experiments and comprehensively go over the results. We used the implementations of PointNet[4] and DGCNN[5] that are available online, with slight changes to fit our number of classes.

3.5.1 Experiments on datasets based on RCSB PDB files

Classification of Proteins consisting of any number of atoms

In this experiment, we trained both PointNet and DGCNN with 2250 point clouds sampled from mainly alpha protein PDB files and 2250 sampled from mainly beta protein PDB files. The test set consisted of 250 mainly alpha point clouds and 250 mainly beta. The number of points each network processed (num_points parameter) was 2048 for PointNet and 1024 for DGCN. DGCNN training with 2048 points was reaching the memory limit on the training platform and could not complete the execution. For proteins with atoms less than the num_points parameters, we used the copy padding method. Below (Figure 3.12, Figure 3.13), we show the results of the experiments for every network.

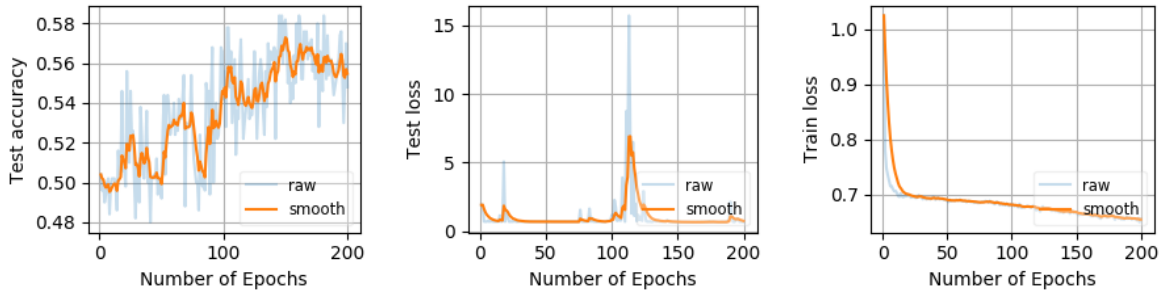

Figure 3.12: PointNet: Accuracy (left) and Test Set Loss (middle) and Train Set Loss (right) over 200 epochs

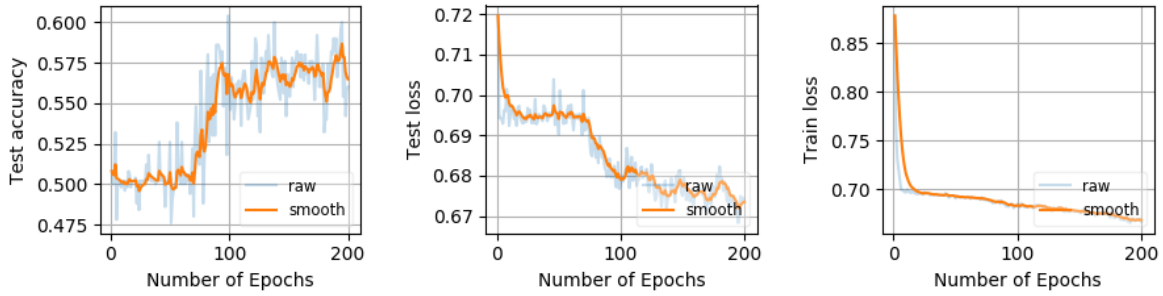

Figure 3.13: DGCNN: Accuracy (left) and Test Set Loss (middle) and Train Set Loss (right) over 200 epochs

After 200 epochs of training, PointNet achieved an accuracy of 0.548 and DGCNN 0.560 respectively. In addition, for DGCNN we measured the Precision value at 0.562

[4] github.com/charlesq34/pointnet [Accessed: 07/09/2019]

[5] github.com/WangYueFt/dgcnn [Accessed: 07/09/2019]

and the Recall value at 0.522. We observed slightly better results from DGCNN even though it processed half the amount of points compared to PointNet, suggesting that the consideration of local features is improving the classification results.

Classification of Proteins consisting of any of number of atoms (backbone)

We performed the same experiment as above, except this time we only sample atoms from each protein backbone. Training parameters and padding methods are the same.

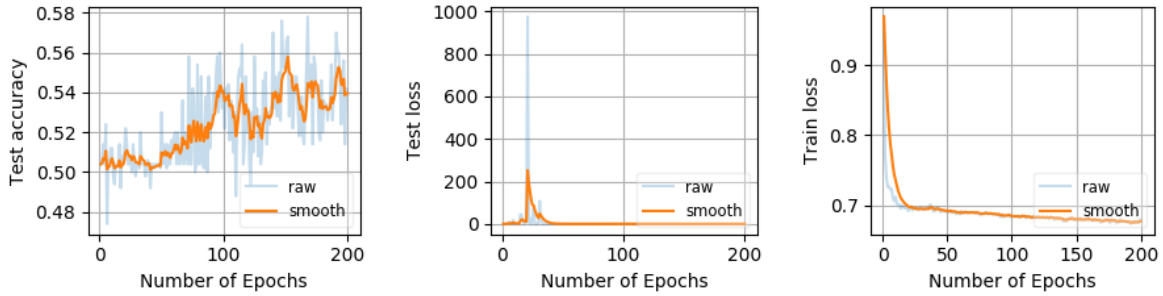

Figure 3.14: PointNet: Accuracy (left) and Test Set Loss (middle) and Train Set Loss (right) over 200 epochs

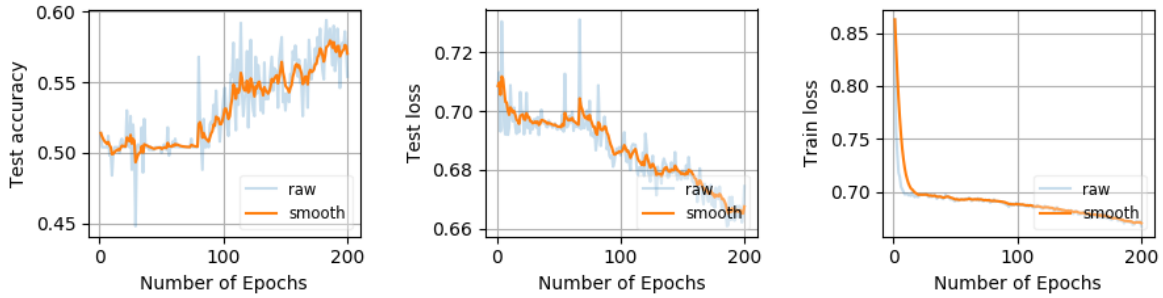

Figure 3.15: DGCNN: Accuracy (left) and Test Set Loss (middle) and Train Set Loss (right) over 200 epochs

In this experiment, after 200 epochs of training, PointNet achieved an accuracy of 0.542 and DGCNN 0.554 respectively. In addition, for DGCNN we measured the Precision value at 0.558 and the Recall value at 0.479. These results show no significant change. However, we believe that DGCNN would achieve better results if we could sample more points to capture a more complete version of the backbone.

Classification of Proteins based on Isolated Secondary Structures

We conducted a classification experiment similar to the previous ones. We trained PointNet and DGCNN with 2250 point clouds sampled from mainly alpha protein PDB files and 2250 sampled from mainly beta protein PDB files. However, this time we only sampled atoms that belong to the secondary structures of each protein. The test set consisted of 250 mainly alpha and 250 mainly beta point clouds that were sampled in the same way. The number of points each network processed (num_points parameter) was 2048 for PointNet and 1024 for DGCNN, for DGCNN training with 2048 points was reaching the memory limit on the training platform and could not complete the

execution. For proteins with atoms less than the num_points parameters, we used the copy padding method.

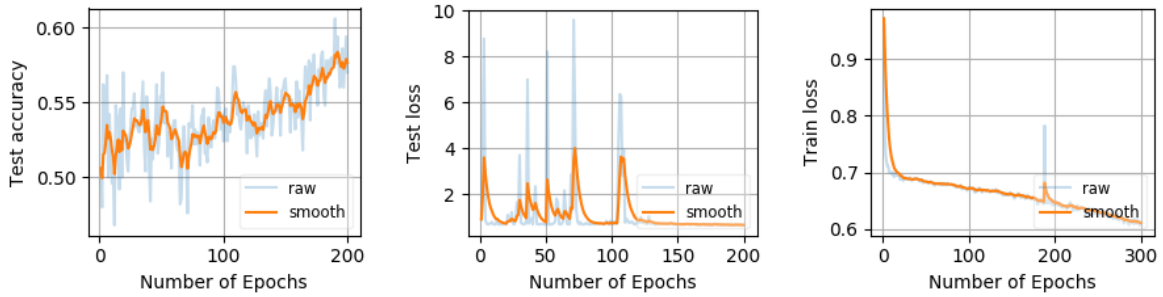

Figure 3.16: PointNet: Accuracy (left) and Test Set Loss (middle) and Train Set Loss (right) over 300 epochs

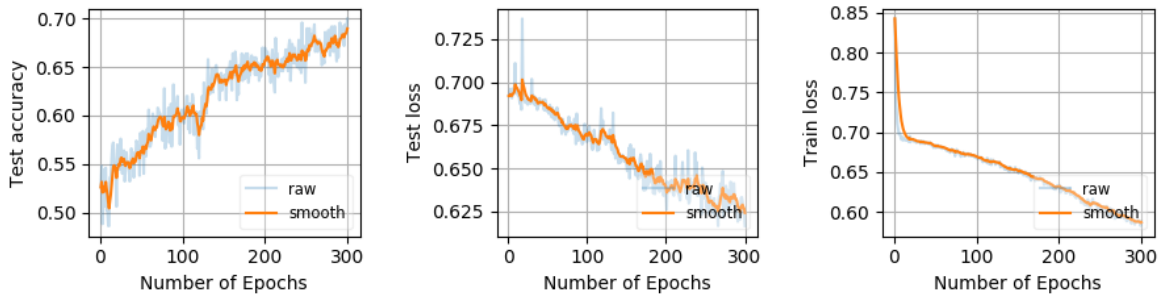

Figure 3.17: DGCNN: Accuracy (left) and Test Set Loss (middle) and Train Set Loss (right) over 300 epochs

As we can see in the plots showcasing the results (Figure 3.16, Figure 3.17), after 300 epochs of training, PointNet achieved an accuracy of 0.624 and DGCNN 0.700 respectively. In addition, for DGCNN we measured the Precision value at 0.732 and the Recall value at 0.643. These results show a slight improvement for the PointNet network and at the same time a significant improvement for the DGCNN network, indicating that sampling points that only belong in the protein secondary structure hold significantly more information that can be further exploited by local feature capturing.

Classification of Proteins based on Isolated Secondary Structures (backbone)

We performed the same experiment as above, except this time we only sample backbone atoms that belong in the secondary structures of each protein. Training parameters and padding methods are the same.

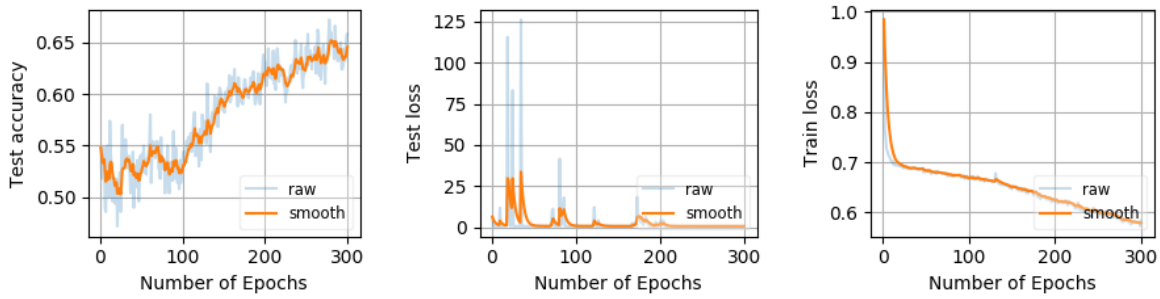

Figure 3.18: PointNet: Accuracy (left) and Test Set Loss (middle) and Train Set Loss (right) over 300 epochs

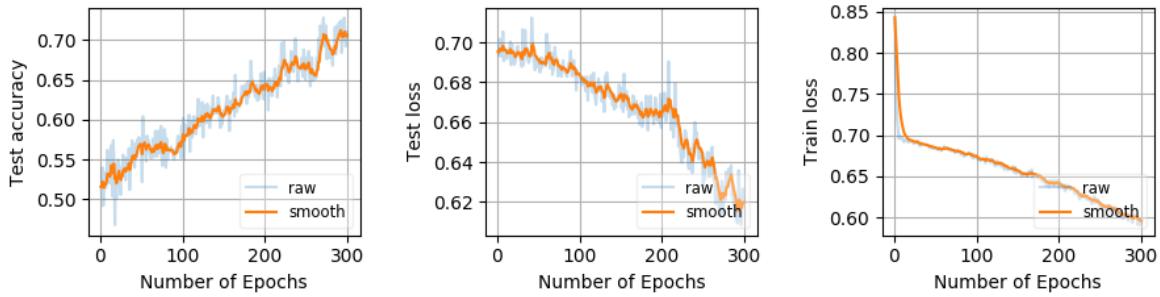

Figure 3.19: DGCNN: Accuracy (left) and Test Set Loss (middle) and Train Set Loss (right) over 300 epochs

As we can see on the results above (Figure 3.18, Figure 3.19), PointNet achieved an accuracy of 0.658 and DGCNN 0.710 respectively. In addition, for DGCNN we measured the Precision value at 0.740 and the Recall value at 0.635. The values remained relatively the same for PointNet while slightly dropping for DGCNN. Again, we believe that DGCNN would achieve better results if we could sample more points to capture a more complete version of the backbone, unfortunately we could not test that on the platform we used for training.

Classification of Alpha Helices and Beta Sheets

For this experiment we isolated alpha helices and beta sheets form the PDB files. We wanted to experiment with the effectiveness of each network, on a simpler problem. We trained both PointNet and DGCNN using 27000 point clouds each one containing the 3D spatial coordinates of the atoms that belong to a single alpha helix or a single beta sheet of a protein, 13500 of those point clouds represent alpha helices and 13500 point clouds represent beta sheets. The test set consisted of 1500 alpha helix point clouds and 1500 beta sheet point clouds. The number of points each network processed (num_points parameter) was 128 as the helices and sheets are significantly shorter (contain less atoms) than whole proteins. For helices or sheets with atoms less than the num_points parameters, we used the copy padding method.

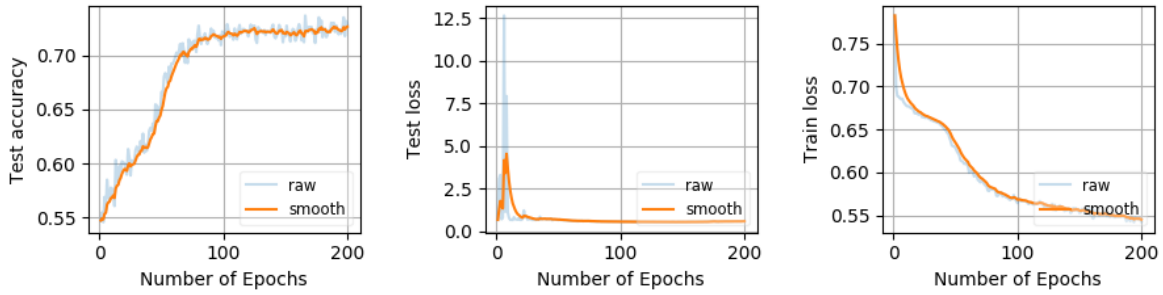

Figure 3.20: PointNet: Accuracy (left) and Test Set Loss (middle) and Train Set Loss (right) over 200 epochs

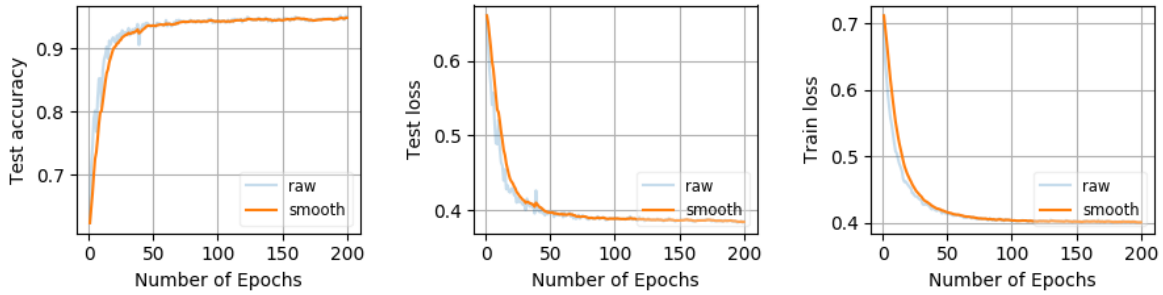

Figure 3.21: DGCNN: Accuracy (left) and Test Set Loss (middle) and Train Set Loss (right) over 200 epochs

The result plots (Figure 3.20, Figure 3.21) show that after 200 epochs PointNet achieved an accuracy of 0.728 and DGCNN 0.950 respectively on classifying an object into the alpha helix or beta sheet category. Also, for DGCNN we measure the Precision value at 0.935 and the Recall value at 0.965. Both networks show significantly better results, considering this is a less complex problem. Nevertheless, DGCNN achieves great values in all metrics suggesting that the consideration of local features by a neural network is important even in less demanding tasks.

3.5.2 Experiments on datasets based on CATH-Dataset-Non-Redundant-S20

Classification of proteins consisting of any number of atoms

For this experiment, we trained our networks, PointNet and DGCNN, with 2250 point clouds sampled from mainly alpha domains and 2250 point clouds sampled from mainly beta domains, from the CATH-Dataset-Non-Redundant-S20 dataset which contains PDB files with non-redundant domains of proteins. The test set, sampled the same dataset, consisted of 250 mainly alpha and 250 mainly beta point clouds. The number of points each network processed (num_points parameter) was 1024. For proteins with atoms less than the num_points parameters, we used the copy padding method.

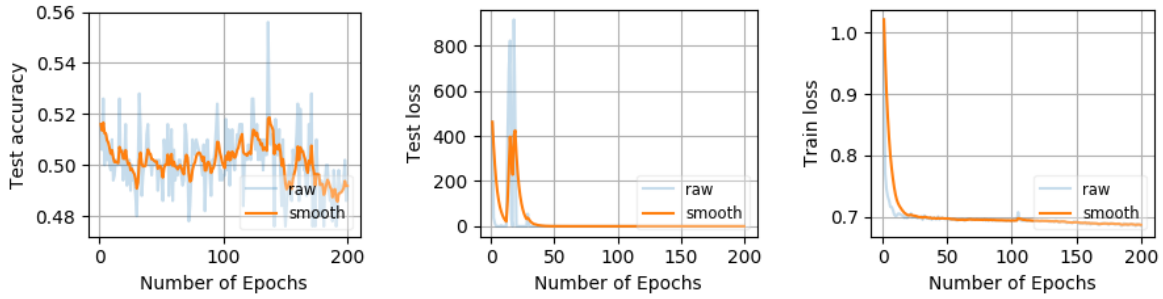

Figure 3.22: PointNet: Accuracy (left) and Test Set Loss (middle) and Train Set Loss (right) over 200 epochs

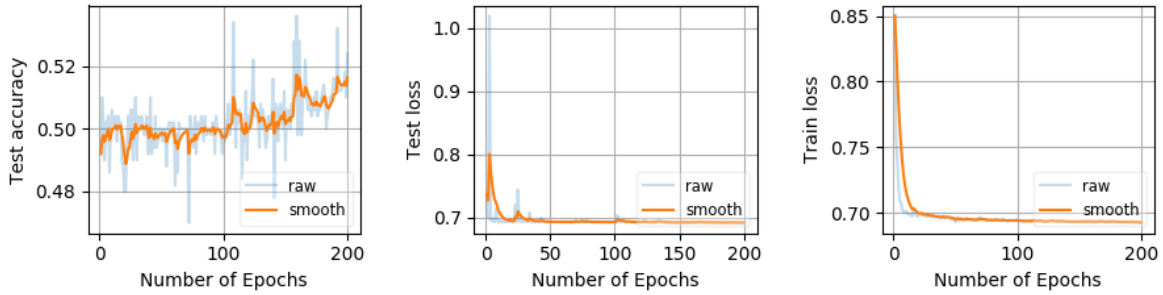

Figure 3.23: DGCNN: Accuracy (left) and Test Set Loss (middle) and Train Set Loss (right) over 200 epochs

PointNet achieved an accuracy of 0.492 and DGCNN 0.524 respectively. In addition, for DGCNN we measured the Precision value at 0.512 and the Recall value at 0.988. Those values suggest a poor classification performance, close to random. The precision and recall values suggest that DGCNN classifies almost all point clouds in one class.

Classification of proteins consisting of any number of atoms (backbone)

We performed the same experiment as above, except this time we only sample atoms from each protein domain backbone. Training parameters and padding methods are the same.

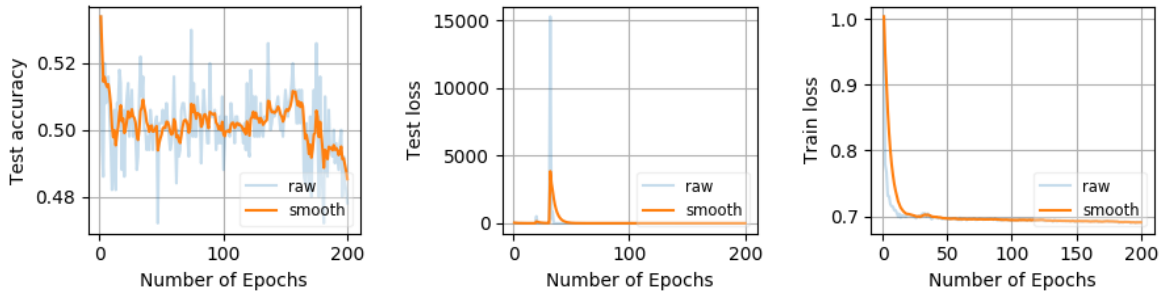

Figure 3.24: PointNet: Accuracy (left) and Test Set Loss (middle) and Train Set Loss (right) over 200 epochs

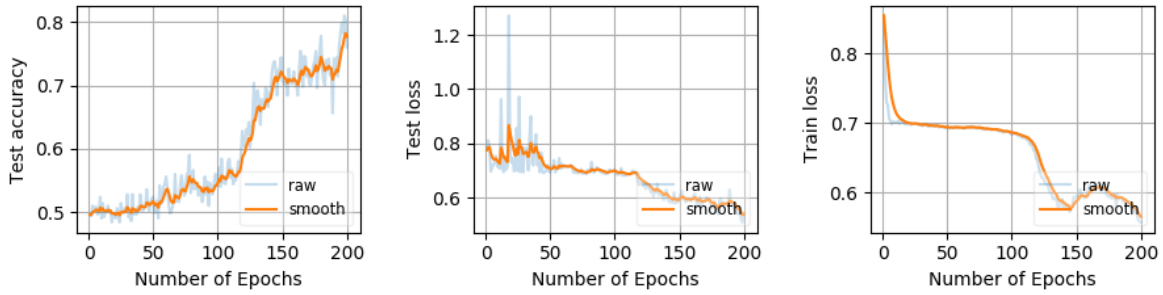

Figure 3.25: DGCNN: Accuracy (left) and Test Set Loss (middle) and Train Set Loss (right) over 200 epochs

We can see on the plots (Figure 3.24, Figure 3.25), sampling only backbone atoms from a non redundant protein domain drastically improved DGCNN effectiveness with an Accuracy value of 0.760, a Precision value of 0.710 and a Recall value of 0.864. On the other hand PointNet results remained at the same levels as it achieved an Accuracy value of 0.478. These results indicate that processing the backbone of a non-redundant protein domain offers a lot of information that can be exploited by a network capable of capturing local features.

Classification of proteins consisting of more than 1024 atoms

In the original dataset we downloaded from CATH there were 1699 mainly alpha and 1503 mainly beta protein domains that had more than 1024 atom entries. We randomly separated those into 1529 mainly alpha and 1353 mainly beta for training and 170 mainly alpha and 150 mainly beta for testing. The number of points each network processed (num_points parameter) was 1024.

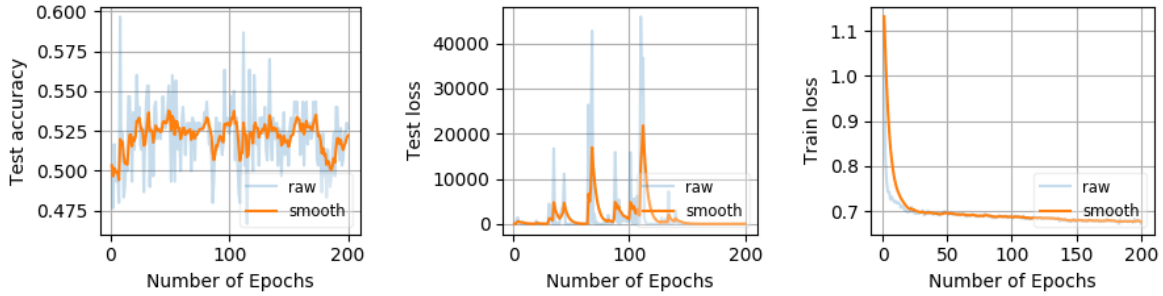

Figure 3.26: PointNet: Accuracy (left) and Test Set Loss (middle) and Train Set Loss (right) over 200 epochs

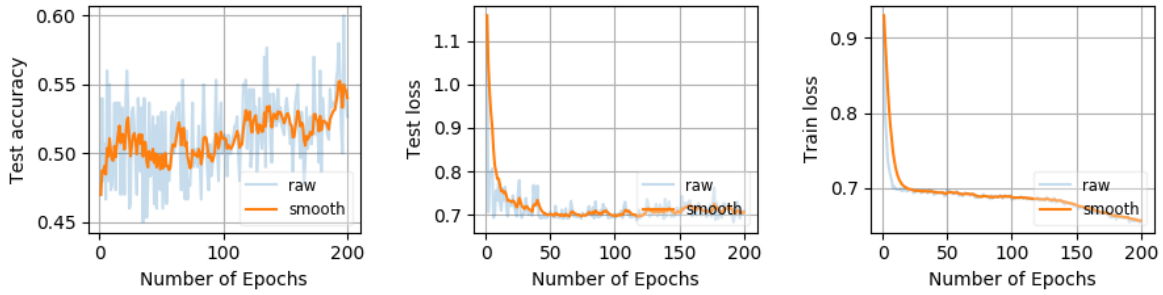

Figure 3.27: DGCNN: Accuracy (left) and Test Set Loss (middle) and Train Set Loss (right) over 200 epochs

PointNet achieved an accuracy of 0.526 and DGCNN 0.520 respectively. In addition, for DGCNN we measured the Precision value at 0.496 and the Recall value at 0.780. Those values suggest a poor classification performance, close to random.

Classification of proteins consisting of more than 1024 atoms (backbone)

We performed the same experiment as above, except this time we only sample atoms from each protein domain backbone. Training parameters are the same. For this experiment we also show the results using both padding methods we mentioned in this thesis.

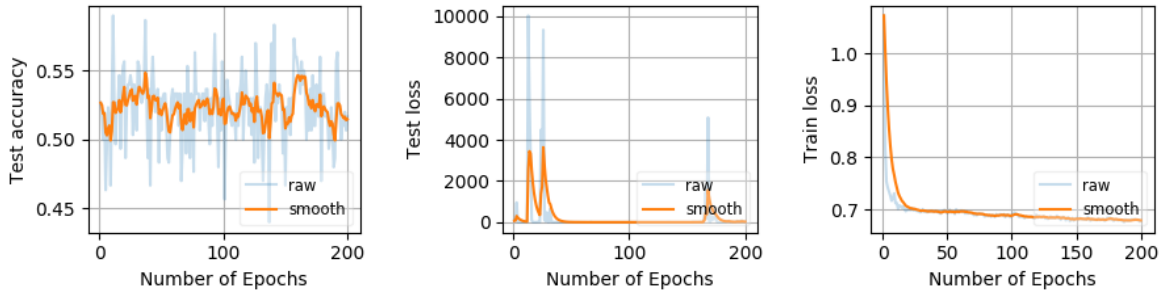

Figure 3.28: PointNet: Accuracy (left) and Test Set Loss (middle) and Train Set Loss (right) over 200 epochs (copy padding)

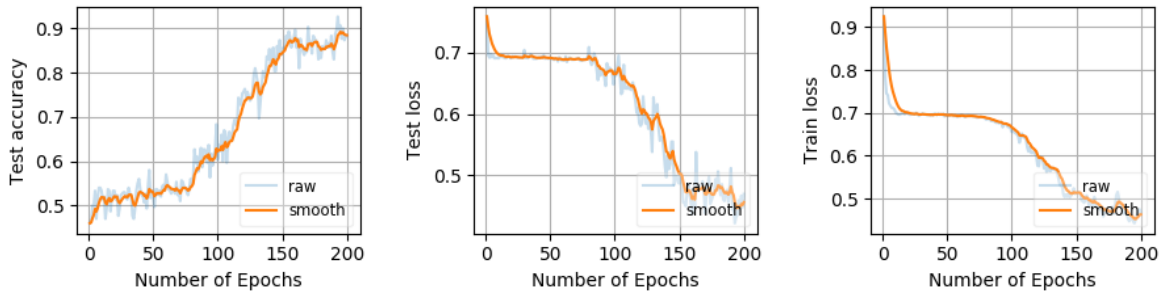

Figure 3.29: DGCNN: Accuracy (left) and Loss (right) over 200 epochs (copy padding)

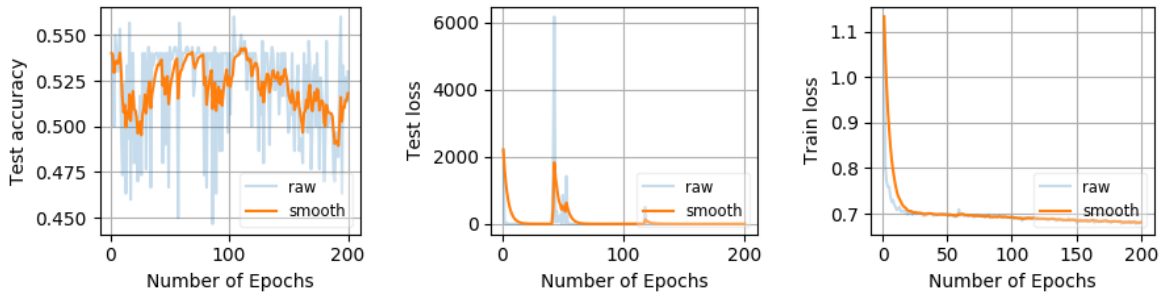

Figure 3.30: PointNet: Accuracy (left) and Test Set Loss (middle) and Train Set Loss (right) over 200 epochs (zero padding)

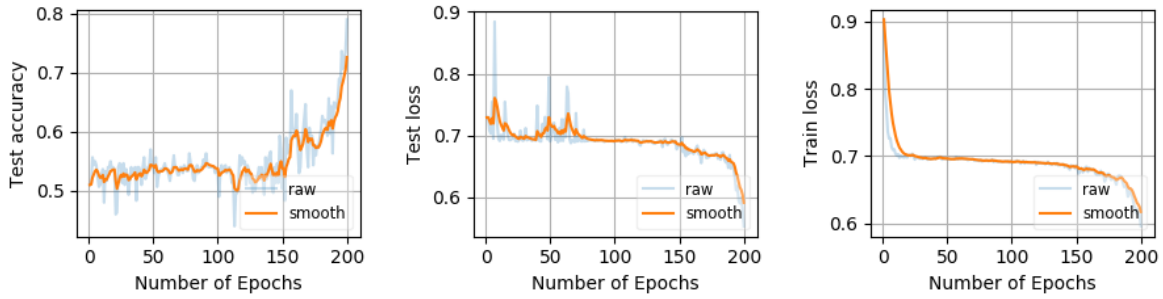

Figure 3.31: DGCNN: Accuracy (left) and Test Set Loss (middle) and Train Set Loss (right) over 200 epochs (zero padding)

Using the copy padding method (Figure 3.28, Figure 3.29), PointNet achieved an accuracy of 0.516 and DGCNN 0.883 respectively. In addition, for DGCNN we measured the Precision value at 0.867 and the Recall value at 0.885. While PointNet accuracy remains close to random, DGCNN shows a remarkable increase in all performance metrics which is consistent to our hypothesis that a complete sample of the backbone points with limited padding, processed with a network that accounts for local geometric features, is encapsulating the most useful information and provides the best results. With the usage of zero padding method (Figure 3.30, Figure 3.31), we observe a similar trend in results, with DGCNN being significantly better performing, albeit worse than the results achieved with the copy padding method.

3.6 Comparison between PointNet and DGCNN

In this section we will present tables (Table 1, Table 2) for a side by side comparison of the deep learning neural networks we explored. According to the results of the experiments, DGCNN exhibits better performance in almost any task compared to PointNet, this suggests that a network that factors the local geometric features of the shapes it processes is a more promising and effective method for classifying complex shapes. Regarding the padding methods we used, our experimental results indicate that for PointNet both padding methods do not provide any significant increase in performance, suggesting that independently treating each point, atom in our case, “confuses” the network when there are padded atoms. On the other hand, DGCNN considers the k nearest points so it can ignore points with no relevant information such as the padded zero vectors or reinforce existing features with points copied from the existing structures.

Table 1: Comparison between PointNet and DGCNN on Experiments on datasets based on RCSB PDB files

Experiments on datasets based on RCSB PDB files	Pointnet Accuracy	Dgcnn Accuracy
Classification of Proteins consisting of any number of atoms (copy padding)	0.548	0.560
Classification of Proteins consisting of any number of atoms backbone (copy padding)	0.542	0.554
Classification of Proteins based on Isolated Secondary Structures (copy padding)	0.624	0.700
Classification of Proteins based on Isolated Secondary Structures (backbone, copy padding)	0.658	0.710
Classification of Alpha Helices and Beta Sheets	0.728	0.950

Table 2: Comparison between PointNet and DGCNN on Experiments on datasets based on CATH-Dataset-Non-Redundant-S20

Experiments on datasets based on CATH-Dataset-Non-Redundant-S20	PointNet Accuracy	DGCNN Accuracy
Classification of Proteins consisting of more than 1024 atoms	0.526	0.520
Classification of Proteins consisting of more than 1024 atoms (backbone, copy padding)	0.516	0.886
Classification of Proteins consisting of more than 1024 atoms (backbone, zero padding)	0.530	0.790
Classification of Proteins consisting of any number of atoms (copy padding)	0.492	0.524
Classification of Proteins consisting of any number of atoms (backbone, copy padding)	0.478	0.760

4. CONCLUSIONS AND FUTURE WORK

In this thesis, we explore Deep Neural Networks that were recently proposed as effective models for geometrical 3D tasks that work directly on point clouds. We performed various experiments on these deep learning architectures in order to test their effectiveness on classifying highly complex 3D shapes such as proteins. For the purposes of this thesis, we performed a variety of classification tasks with the usage of PointNet and DGCNN on many differently preprocessed point cloud datasets that have been directly sampled from PDB files.

Our experiments have shown that a network that captures the local geometric features of a shape instead of only global ones, such as DGCNN, is important for improving the performance of classification tasks on protein shapes that consist of complex surfaces and folds. Furthermore, focusing only on the secondary structures and the backbone of the protein, exhibited an increase in the performance of the networks, specifically on the DGCNN, achieving accuracy values of up to 71% on DGCNN and 65.4% on PointNet.
The results of our experiments show that, taking into consideration only backbone atoms or backbone atoms that belong in the secondary structures of the protein is beneficial for increasing the accuracy of the networks. The best results were observed when we sampled backbone atoms from non-redundant protein domains and used them to train the DGCNN network achieving an accuracy value of 76% without any restriction on the length of the proteins and up to 88.6% when sampling from proteins with more than 1024 atoms. Lastly, regarding the padding methods we used, both methods showed little difference on PointNet. However, on DGCNN copy padding appeared as the more effective method, suggesting that the copied points could reinforce local features compared with the padding of zero vectors that add no useful information.

Reviewing the results of the experiments, in which we classified structures into alpha helixes or beta sheets, we observe that the neural networks we tested are significantly better on performing that classification task compared to the classification of proteins into mainly alpha or mainly beta. For PointNet, the best accuracy value we achieved for protein classification task was 65.4% while the accuracy of the single secondary structure classification into an alpha helix or beta sheet was 72.8%. In the same manner, our best accuracy value on protein classification for DGCNN was 88.6% whereas the task of classifying a single structure into alpha helix or beta sheet had an accuracy of was 95%. Those results are also consistent with our hypothesis that DGCNN will perform better on the given tasks as it considers the local geometric features.

The results presented in this thesis are encouraging and can be improved further with future work. Architectures with dynamic input size to cater for the difference in size between proteins could be researched or new and more effective methods for normalizing the proteins of different size to a constant size could be proposed. Refinement of the current architectures for more efficient capturing of data features is also an option. It should be noted that in the experiments we conducted regarding the protein backbone, we considered Oxygen (O) as a part of the backbone, however, going forward the consideration of only carbons (C), alpha carbons (CA) and nitrogen (N), as a part of the protein backbone should be tested. Finally, a platform with access to more resources could prove to be helpful in alleviating experimentation limits.

ABBREVIATIONS - ACRONYMS

ANN	Artificial Neural Network
CNN	Convolutional Neural Network
DNN	Deep Neural Network
DGCNN	Dynamic Graph CNN
MLP	Multi Layer Perceptron
PDB	Protein Data Bank

REFERENCES

[1] M. Aubry, U. Schlickewei, and D. Cremers, “The wave kernel signature: A quantum mechanical approach to shape analysis”, *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pp. 1626–1633, 2011.

[2] J. Sun, M. Ovsjanikov, and L. Guibas, “A Concise and Provably Informative Multi-Scale Signature Based on Heat Diffusion”, *Computer Graphics Forum*, vol. 28, no. 5, pp. 1383–1392, 2009.

[3] M. M. Bronstein and I. Kokkinos, “Scale-invariant heat kernel signatures for non-rigid shape recognition”, *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1704–1711 2010.

[4] R. Rusu, N. Blodow, Z. Marton, and M. Beetz, “Aligning point cloud views using persistent feature histograms”, *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp 3384–3391, 2008.

[5] D.-Y. Chen, X.-P. Tian, Y.-T. Shen, and M. Ouhyoung, “On Visual Similarity Based 3D Model Retrieval”, *Computer Graphics Forum*, vol. 22, no. 3, pp. 223–232, 2003.

[6] R. B. Rusu, N. Blodow, and M. Beetz, “Fast Point Feature Histograms (FPFH) for 3D registration”, *2009 IEEE International Conference on Robotics and Automation*, pp. 3212-3217, 2009.

[7] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks”, *Proc. NIPS*, 2012.

[8] D. Maturana and S. Scherer, “VoxNet: A 3D Convolutional Neural Network for real-time object recognition”, *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015.

[9] C. R. Qi, H. Su, M. Niebner, A. Dai, M. Yan, and L. J. Guibas, “Volumetric and Multi-view CNNs for Object Classification on 3D Data”, *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[10] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, “3D ShapeNets: A deep representation for volumetric shapes”, *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp 1912–1920, 2015.

[11] Y. Li, S. Pirk, H. Su, C. R. Qi, and L. J. Guibas. “Fpnn: Field probing neural networks for 3d data”, *arXiv*, preprint arXiv:1605.06240, 2016.

[12] D. Z. Wang and I. Posner, “Voting for Voting in Online Point Cloud Object Detection”, *Robotics: Science and Systems XI*, pp. 1317, 2015.

[13] R. Q. Charles, H. Su, M. Kaichun, and L. J. Guibas, “PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation”, *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[14] Charles R. Qi, Li Yi, Hao Su, and Leonidas J. Guibas, “PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space”, Advances *in neural information processing systems*, pp. 5099-5108, 2017.

[15] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, “Dynamic graph CNN for learning on point clouds”, *arXiv*, preprint arXiv:1801.07829, 2018.

[16] W. Dhifli and A. B. Diallo, “ProtNN: fast and accurate protein 3D-structure classification in structural and topological space”, *BioData Mining*, vol. 9, no. 1, 2016.

[17] E. D. Levy, J. B. Pereira-Leal, C. Chothia, and S. A. Teichmann, “3D Complex: a Structural Classification of Protein Complexes”, *PLoS Computational Biology*, vol. preprint, no. 2006, 2005.

[18] F. Langenfeld, A. Axenopoulos, H. Benhabiles, P. Daras, A. Giachetti, X. Han, K. Hammoudi, D. Kihara, T. M. Lai, H. Liu, M. Melkemi, S. K. Mylonas, G. Terashi, Y. Wang, F. Windal, M. Montess, “SHREC’19 Protein Shape Retrieval Contest”, 12th Eurographics Workshop on 3D Object Retrieval, 2019,

[19] H.M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T.N. Bhat, H. Weissig, I.N. Shindyalov, P.E. Bourne, “The Protein Data Bank”, *Nucleic Acids Research*, vol. 28, no. 1, pp. 235–242, 2000.

[20] “Protein Data Bank Contents Guide”, *Atomic Coordinate Entry Format Version 3.3*. [Online]. Available: http://www.wwpdb.org/documentation/file-format-content/format33/v3.3.html. [Accessed: 07/09/2019].

[21] F. M. G. Pearl, C. F. Bennett, J. E. Bray, A. P. Harrison, N. Martin, A. Shepherd, I. Sillitoe, J. Thornton, C. A. Orengo, “The CATH database: an extended protein family resource for structural and functional genomics”, *Nucleic Acids Research*, vol. 31, no. 1, pp. 452–455, 2003.

[22] Antonina Andreeva, Dave Howorth, Steven E. Brenner, Tim J. P. Hubbard, Cyrus Chothia, Alexey G. Murzin, “SCOP database in 2004: refinements integrate structure and sequence family data”, *Nucleic Acids Research*, vol. 32, no. 90001, pp.226-229, 2004.

[23] Stephen K. Burley, Helen M. Berman, et al., “RCSB Protein Data Bank: biological macromolecular structures enabling research and education in fundamental biology, biomedicine, biotechnology and energy”, *Nucleic Acids Research*, vol. 47, pp. 464–474, 2019.

[24] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, “3d shapenets: A deep representation for volumetric shapes”, *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1912–1920, 2015