Protein Fold Classification using Graph Neural Network and Protein Topology Graph

Suri Dipannita Sayeed, Jan Niclas Wolf, Ina Koch, Guang Song
suri@iastate.edu

Abstract

Protein fold classification reveals key structural information about proteins that is essential for understanding their function. While numerous approaches exist in the literature that classifies protein fold from sequence data using machine learning, there is hardly any approach that classifies protein fold from the secondary or tertiary structure data using deep learning. This work proposes a novel protein fold classification technique based on graph neural network and protein topology graphs. Protein topology graphs are constructed according to definitions in the Protein Topology Graph Library from protein secondary structure level data and their contacts. To the best of our knowledge, this is the first approach that applies graph neural network for protein fold classification. We analyze the SCOPe 2.07 data set, a manually and computationally curated database that classifies known protein structures into their taxonomic hierarchy and provides predefined labels for a certain number of entries from the Protein Data Bank. We also analyze the latest version of the CATH data set. Experimental results show that the classification accuracy is at around 82% - 100% under certain settings. Due to the rapid growth of structural data, automating the structure classification process with high accuracy using structural data is much needed in the field. This work introduces a new paradigm of protein fold classification that meets this need. The implementation of the model for protein fold classification and the datasets are available here https://github.com/SuriDipannitaSayeed/ProteinFoldClassification.git

Author summary

Classification of protein structures is traditionally done using manual curation, evolutionary relationship, or sequence comparison-based methods. Applying machine learning and deep learning to protein structure classification is a comparatively new trend that holds great promises for automating the structure classification process. Advance deep learning technique like Graph Neural Network is still unexplored in this respect. SCOP and CATH are two traditional databases that provide the hierarchical taxonomic classification of protein structures. This work provides a novel computational approach that classifies protein folds in SCOP and CATH with graph neural network, performing a graph classification task.

1 Introduction

A protein fold is a 3D pattern in proteins that share the same composition in secondary structure elements as well as the same spatial arrangement and topology. Protein fold recognition is a problem of predicting protein folds from a given amino acid sequence or from a given protein structure. The problem of protein fold recognition is highly important for multiple reasons. Recognition of protein fold could be the first step to an efficient prediction of protein structures from amino acid sequences. On the other hand, protein structure classification is an important task also in structural biology, where newly determined structures are constantly needed to be assigned to their corresponding folds. Given the growing number of high resolution structure data in the Protein Data Bank (PDB) and other data sources, a fast, accurate, and fully automated approach for structure classification procedure is much needed.

Traditionally, sequence-based protein fold recognition methods can be divided into two categories. Those are: 1) sequence alignment-based methods and 2) machine learning-based ab-initio methods. In sequence alignment-based methods, a query protein sequence or sequence profile is aligned against templates with similar structures to identify the most probable fold. Machine learning models from the very beginning time such as support vector machines, kernel-based methods and ensemble classifiers also rely on sequence correlation-based features. These models work fine in recognizing folds at the family level, where sequences have enough evolutionary relationships. But the problem of fold recognition becomes significantly more challenging when sequences are highly dissimilar and there is not enough evidence of evolutionary relationships among sequences. This indicates the necessity to integrate structural features in the prediction.
The Structural Classification of Proteins (SCOP)\textsuperscript{8} and Hierarchical Classification of Protein Domain Structures (CATH)\textsuperscript{9} are two standard databases for taxonomic classification of protein domain structures. In both of these databases, protein structures are classified manually for the most part by considering their evolutionary relationships with other similar structures. SCOP classification is done mainly based on evolutionary relationship between homologous sequences. SCOP\textsuperscript{8} provides an automated pipeline for structure classification at protein family-fold level and superfamily-fold level. The basis of their classification approach is that, every protein has structural similarities with some other proteins that share a common evolutionary origin. Protein structures are grouped into homologous families based on their sequence similarity. SCOPe classification starts with the domain. The domains are then classified into the following hierarchy:

- **Species**: represents a unique protein sequence and its variants.
- **Protein**: groups together similar sequences with the same functions that either come from different biological species or from different isoforms within the same species.
- **Family**: combines proteins with similar sequences but distinct or similar functions. Due to their sequence similarity and or functional similarity, protein domains that are clustered in same family are likely to have a common evolutionary origin.
- **Superfamily**: merges together protein families with similar functions and structural features and come from a common evolutionary ancestor.
- **Fold**: is the purely structural layer of this classification. Structurally similar superfamilies are combined into folds.
- **Class**: combines folds based on their secondary structure content and their topological arrangements.

CATH\textsuperscript{9}, on the other hand, considers structural similarity over evolutionary relationships in classifying protein structures. Evolution gives rise to families of structurally similar proteins for which there is no evident evolutionary relationship as they have extremely low sequence similarities. CATH applies a semi-automatic approach for this classification. They use the standard multiple sequence alignment method and the structure comparison algorithm SSAP (Sequential Structure Alignment Program)\textsuperscript{10}. CATH has four levels in their hierarchy of classification:

- **Class**, C-level: describes the secondary structure element composition of each domain. It is the first level of the hierarchy.
- **Architecture**, A-level: describes the overall shape of the structure. This shape is revealed by the composition and orientation of the secondary structure units. At this level the connectivity among the secondary structure elements are not considered.
- **Topology** (Fold family), T-level: At this level, structures are clustered into fold families based on their overall shape and connectivity of secondary structure elements. Some fold families are overpopulated than others, and CATH organizes those by the SSAP score.
- **Homologous Superfamily**, H-level: When structurally similar proteins coming from the same T-level also have similar functions, they are supposed to share a common evolutionary ancestor. These proteins are grouped into a homologous superfamily.

The construction process of CATH is more automatic than that one in SCOPe. Both of these two databases have been considered as gold standards for benchmarking automatic pipelines of structure classification and for training machine learning-based models.

Every protein structure is shaped by the arrangement of secondary structure elements. And each structure defines a particular conformation in the conformation space. Recent revolution of deep learning has opened up opportunities for exploring more structural information, such as secondary structure composition and inter-residue contacts, in fold classification. Early deep learning-based models like DN-Fold\textsuperscript{11} that relied upon evolutionary distance between protein pairs showed promising performance in classifying protein folds at family and superfamily level but very limited performance at the fold level. Sequence-based classification techniques such as DeepSF\textsuperscript{12} that use Deep Convolutional Neural Network (DCNN) can classify protein folds with an accuracy of around 75.3%. Fold recognition method like DeepFR\textsuperscript{12} exploits inter-residue contact information using DCNN. Inter-residue contact information holds important information about the overall 3D structure of the protein. Residue contacts can be local or non-local. Local contact information holds information about the local structural motifs, whereas non-local interactions among residues give information about the overall arrangements of the motifs. Both of these information could be valuable for recognizing the overall protein fold. Classification techniques like PRO3DCNN\textsuperscript{13} also use DCNN to exploit fold-specific features. In addition to using a regular contact distance matrix, they use also one extra feature that is called persistence homology image of the protein structure. In combination with these two features, they reached an accuracy of around 95% – 98% for their dataset.

In this work, we propose a novel graph neural network (GNN)-based model to classify protein folds. GNN\textsuperscript{14} is effective in capturing both local interactions and global interactions. The intuition behind using GNN is that it captures information from both local structural motifs and non-local residual contacts to generate a better
embedding for classification. We use the open-source implementation of HGP-SL (Hierarchical Graph Pooling
with Structure Learning)\textsuperscript{15} for performing the graph classification task. The model takes protein topology
graphs as input. The input graphs are constructed according to the definitions of Protein Topology Graph
Library (PTGL)\textsuperscript{16}. We consider three node features for our classification, namely, i) node degree, ii) node
type, and iii) the number of residues in the node (each node represents a secondary structure element). Our
model has two basic layers: i) graph convolution layer and ii) hierarchical graph pooling layer. We apply a
sophisticated structure learning mechanism with the graph pooling layer. At the very end of the model there
is a softmax layer that generates a graph embedding to classify the graph against multiple classes. In our case,
the class labels are the fold labels. We will discuss our methods and results in Sections Material and Methods
and Results and Discussion, respectively.

2 Contribution of This Method

We claim that for protein structure classification, a highly accurate machine learning-based model that learns
pattern from structural data provides two major benefits. We discuss them in details below.

2.1 Automating the Structure Classification Process

We focus particularly on classifying the structural layer that is the fold of the protein structure taxonomy. Due
to the abundance of structural data, it seems that time is ripe for fully automating the process using deep
learning.

- **Abundance of structure data solved at high-resolution.** Due to the advancement of technology
  in both hardware and software, the number of protein structures solved at high-resolution is increasing
  rapidly. Over the past 60 years X-ray crystallography has been the primary method for structure determi-
nation. A major limitation of X-ray crystallography is that it requires molecules to be crystallized, which
  is not always possible. Figure 1 shows the overall statistics of the growth in the number of structures
  in the Protein Data Bank (PDB). The data in PDB are generally collected from X-ray crystallography,
  NMR spectroscopy, or cryo-electron microscopy. Nuclear magnetic resonance (NMR) technique deter-
mines the 3-D structures of macromolecules in solution. This technique is limited mostly to small to
  medium macromolecules.

- Apart from PDB, Electron Microscopy Data Bank (EMDB) has also become a great resource for high-
  resolution protein structure data. Currently, cryo-EM is one of the most prominent approaches for
determining high-resolution structures of small to large macromolecular complexes (Figure 2).

- Apart from these experimental approaches, artificial intelligence(AI)-based systems such as AlphaFold\textsuperscript{17}
is also making a big leap towards generating highly accurate protein structure data at high throughput.
AlphaFold has generated accurate protein structure predictions since its inception and has now achieved
an accuracy that is comparable to experimental methods. AlphaFold database\textsuperscript{18} is a fast growing resource
of predicted protein structures. Currently, it contains protein structures from human proteome and 20
other key organisms. These structures are available to the scientific community and can be accessed
through the UniProt accession ID. One important aspect about the newly predicted structures is that they
are unclassified. With the ever increasing computationally-predicted structures, a deep learning-based
automated method that can accurately classify them into their respective folds seems most consistent and
natural.
Figure 1: PDB Statistics: overall growth of released structures per year. https://www.rcsb.org/stats/growth/growth-released-structures

Figure 2: EMDB Statistics: overall growth of archived electron microscopy density maps at different resolution ranges between 2002 and 2020. https://ars.els-cdn.com/content/image/1-s2.0-S0021925821003380-gr6.jpg
2.2 Classifying Non-homologous Proteins

Sequence similarity-based fold classification techniques perform poorly when sequence similarity is low. Due to events like gene duplication and gene speciation, many sequences diverge from their homologs but their structures and functions remain mostly intact. Sequence-based models rely mainly on sequence homologs. There are many groups of proteins whose sequence similarity is lower than the required threshold but share significant structural similarity. Such structural similarities in the absence of any evident evolutionary relationship can be explained only by the similarities among the packing arrangement of the secondary structure elements. Folds like the globin-like family from the all-alpha architecture contain protein pairs whose sequence identity is less than 15% but share similar structures and functions. Figure 3 shows the similarities between two globin-like structures, and Figure 4 displays their similarities in terms of topology graphs. The two graphs share a high degree of similarity in terms of the number of secondary structure elements and their contacts.

![PyMOL structures of a similar-fold protein pair with sequence identity less than 15%](image1)

Figure 3: PyMOL structures of a similar-fold protein pair with sequence identity less than 15%

![Topology graphs of a similar-fold protein pair with sequence identity less than 15%](image2)

Figure 4: Topology graphs of a similar-fold protein pair with sequence identity less than 15%

The Tim-barrel fold/topology from the alpha-beta class, barrel architecture also displays a high degree of structural similarity in the absence of any evident sequence similarity. This fold is also challenging to classify using sequence similarity-based machine learning models. Figure 5 shows the PyMOL structures of two examples from the Tim-barrel fold. Their topology graphs are given in Figure 6.
The above examples show that a simple and intuitive approach based on protein topology graphs can potentially solve the problem with non-homologous protein pairs for protein fold recognition.

2.3 Classifying Against a Large Number of Classes

Classifying protein fold using only secondary structure elements and their topological connections is a more abstract but robust way of classification. In the cases of sequence-based approaches, when more classes are included in the data set, it becomes increasingly challenging to classify fold patterns accurately for a query protein sequence. Classifying against a large number of classes is challenging also for traditional classification models like Support Vector Machine (SVM) and Deep Neural Network (DNN). Our GNN-based approach can easily classify against multiple classes. For example, under Orthogonal Bundle architecture in mainly alpha class of the CATH dataset\(^7\), there are 290 topologies. Our model can classify against these 290 classes as well as it does against a smaller number of classes.

In summary, these points underline the potential of our proposed method for automatic structural classification.

3 Results and Discussion

To corroborate our proposed method we first analyze small data sets like DD, EDD and TG\(^2\). We find our graph-based learning model outperforms existing sequence based learning models on these benchmark data sets regarding the protein fold classification task. To further expand our analysis on small data sets, we train and test our model also on large-scale protein structure classification data sets: SCOP\(^8\) and CATH\(^7\).

3.1 Results on Benchmark Data Set

In this section, a combination of ab-initio machine learning models are compared on three benchmark data sets; DD, EDD and TG\(^2\). These models learn patterns mainly based on sequence profile-based features and some structural and chemical features. As our approach is graph learning-based, graphs are collected for the respective protein entries in these three data sets. After that it is trained and tested on the converted graph data sets. Ten-fold cross validation is applied on DD, EDD and TG data sets. The DD data set is also evaluated using the independent test set that comes along with the original data set. The classification accuracy is reported in Table\(^2\).
### Table 1: Comparison of results from different methods on three benchmark data sets

| Method       | ACCFold\textsuperscript{23} | TAXFOLD\textsuperscript{22} | HGP-SL (This Article) |
|--------------|-----------------------------|-----------------------------|-----------------------|
|              | 0.701                       | 0.715                       | 0.827\textsuperscript{b} |
|              | 0.825                       | 0.965                       |                       |
|              | 0.664                       | NA                          |                       |
|              | 0.96                         | 1.0                         |                       |

\(a\) from ten-fold cross validation accuracy, \(b\) from independent data set

The DD data set was analyzed using one-vs-all classification scheme to compare the fold prediction accuracy of each individual class. Results are compared with two other methods; DeepFrag-K and ProFold. The results of DeepFrag-k and ProFold are collected from Ref. 26. Our method secures a uniformly high classification accuracy across different folds. There is a huge performance gap in classification accuracy for 18 out of 27 fold classes for other methods. For example, for folds such as SH3-like barrel, trypsin-like serine proteases, flavodoxin-like, P-loop containing NTH and ubiquitin-like, Profold demonstrates a low accuracy, whereas DeepFrag-K shows a significantly higher accuracy. HGP-SL shows 100\% accuracy for all 27 folds. The evaluation of each class is also done using the independent test set available for the DD data set.

| #  | Fold ID | Fold Name                       | DeepFrag-k Accuracy | ProFold Accuracy | HGP-SL Accuracy |
|----|---------|---------------------------------|---------------------|------------------|-----------------|
| 1  | a.1     | Globin-like                     | 95.0                | 100.0            | 100.0           |
| 2  | a.3     | Cytochrome c                    | 95.0                | 100.0            | 100.0           |
| 3  | a.4     | DNA/RNA-binding 3-helical bundle| 85.9                | 60.0             | 100.0           |
| 4  | a.24    | 4-Helical up-and-down bundle    | 91.5                | 87.5             | 100.0           |
| 5  | a.26    | 4-Helical cytokines             | 98.9                | 88.9             | 100.0           |
| 6  | a.39    | EF hand-like                    | 90.8                | 77.8             | 100.0           |
| 7  | b.1     | Immunoglobulin-like beta sandwich| 91.10              | 84.1             | 100.0           |
| 8  | b.6     | Cupredoxin-like                 | 78.7                | 66.7             | 100.0           |
| 9  | b.121   | Nucleiplasmin-like, NP          | 91.3                | 92.3             | 100.0           |
| 10 | b.29    | CpnA-like lectins/glucanases    | 76.7                | 66.7             | 100.0           |
| 11 | b.34    | SH3-like barrel                 | 78.0                | 50.0             | 100.0           |
| 12 | b.40    | OB-Fold                         | 80.4                | 68.4             | 100.0           |
| 13 | b.32    | Beta-Trefoil                    | 89.0                | 100.0            | 100.0           |
| 14 | b.47    | Trypsin-like serine proteases   | 75.0                | 50.0             | 100.0           |
| 15 | b.60    | Locians                        | 90.5                | 100.0            | 100.0           |
| 16 | c.1     | TIM beta/alpha barrel           | 93.8                | 93.8             | 100.0           |
| 17 | c.2     | FAD/NAD(P)-binding domain       | 89.7                | 91.7             | 100.0           |
| 18 | c.3     | flavodoxin-like                 | 60.2                | 46.2             | 100.0           |
| 19 | c.23    | NAD(P)-binding Rossmann         | 90.2                | 85.2             | 100.0           |
| 20 | c.37    | P-loop containing NTH           | 79.5                | 50.0             | 100.0           |
| 21 | c.45    | Thioredoxin-fold                | 97.5                | 87.5             | 100.0           |
| 22 | c.55    | Ribonuclease II-like motif       | 75.3                | 58.3             | 100.0           |
| 23 | c.69    | alpha/bet-Hydrolases            | 78.4                | 71.4             | 100.0           |
| 24 | c.93    | Periplasmic binding protein-like| 92.0                | 100.0            | 100.0           |
| 25 | d.15    | beta-Grasp(ubiquitin-like)      | 69.4                | 25.0             | 100.0           |
| 26 | d.58    | Ferrodoxin-like                 | 76.8                | 59.3             | 100.0           |
| 27 | g.3     | Knottins(small inhibitors, toxins, lectins) | 88.2 | 96.3 | 100.0 |

Table 2: One-vs-all classification on DD data set

### 3.2 Results on SCOPe 2.07 Data Set

Protein Structures are grouped into four major classes in the SCOPe database\textsuperscript{8}, namely, alpha, beta, alpha/beta, and alpha+beta. These structural classes define the composition of secondary structure elements in
SCOPe domains. The next level in the SCOPe structure classification hierarchy is fold. There are 289 folds in the all-alpha class, and 178, 148, and 388 folds in the all-beta, alpha/beta, and alpha+beta classes, respectively. In this work, we focus our classification problem at the fold level and aim to classify all protein domains under every structural class into their corresponding folds.

First, we collect topology graph data from every structural class without applying any filtering. For the all-alpha data set, we have 49,670 protein domain graphs. There are 86,430, 96,716, and 84,473 graphs for all-beta, alpha/beta, and alpha+beta classes, respectively. These data sets reflect the diversity in the original PDB data source since every unique protein chain in every structural class is included. The data sets are trained for 5,500 epochs using Adam optimizer and cross-entropy loss.

The ten-fold cross validation results on the data sets are shown in Table 3. From the results, it is seen that the average accuracy of fold classification by our model is around 90%. The accuracy of fold classification for the all-alpha class is slightly lower, at around 88%.

| Class                  | No of fold groups | No of graphs | Accuracy(%) |
|------------------------|-------------------|--------------|-------------|
| All-alpha proteins     | 289               | 49,670       | 88%         |
| All-beta proteins      | 178               | 86,430       | 93%         |
| Alpha/beta proteins    | 148               | 96,716       | 91%         |
| Alpha+beta proteins    | 396               | 84,473       | 92%         |

Table 3: Prediction accuracies across different structural classes of SCOPe domains using the unfiltered data set

Many sequences in the PDB are highly similar, or even identical, to one another. Sequences that share high sequence identity usually have the same fold. To investigate how sequence similarity in the data set influences the performance of our method, we examine in the following the dependency of classification accuracy on sequence similarity. To this end, we cluster structures at different sequence similarity levels. We use the service provided by PDB sequence clusters data. The data, updated weekly, provides clusterings of all the protein chains in the PDB by MMseqs2 at sequence identity levels of 30%, 40%, 50%, 70%, 90%, 95%, and 100%, respectively. In MMseqs2, protein sequences are clustered based on sequence similarity thresholds. Each protein chain is assigned to one and only one specific cluster, with sequences within the cluster share a sequence similarity equal to or greater than the threshold. As the threshold increases from 30% to 100%, the number of clusters increases. To keep the sequence similarity at a certain identity level, only one chain is randomly selected from each cluster for classification. The folds are classified under all four structural classes, and at different sequence identity levels. The model is trained for 3,500 epochs with Adam optimizer and cross-entropy loss. The ten-fold cross validation results are given in Table 4. Note specifically that when the sequence identity threshold is at 100% (the bottom row of Table 4), the numbers of graphs are still greatly reduced when compared to those in Table 3 since identical sequences are filtered out from the data sets.

In constructing topology graphs of protein domains, domain boundaries are considered. Whether a protein chain is multi-domain or single-domain, folds are classified at the domain level.

| Class                  | Alpha     | Beta      | Alpha+Beta | Alpha/Beta |
|------------------------|-----------|-----------|------------|------------|
| Seq. Sim. level        | #F #G Acc | #F #G Acc | #F #G Acc  | #F #G Acc  |
| 30%                    | 289 3818 83% | 178 3964 98% | 396 4921 94% | 148 4582 98% |
| 40%                    | 289 4295 82% | 178 4840 97% | 396 5832 93% | 148 6238 96% |
| 50%                    | 289 4808 93% | 178 5531 92% | 396 6617 92% | 148 7709 96% |
| 70%                    | 289 5683 88% | 178 6814 97% | 396 7955 91% | 148 9788 95% |
| 90%                    | 289 6384 89% | 178 9081 97% | 396 9019 90% | 148 10973 94% |
| 95%                    | 289 6663 92% | 178 10463 97% | 396 9414 90% | 148 11337 94% |
| 100%                   | 289 10044 91% | 178 17145 96% | 396 14831 88% | 148 18329 94% |

Table 4: Classification accuracy at different sequence identity levels for all-alpha, all-beta, alpha+beta, alpha/beta classes. #F and #G denote the number of folds and the number of protein domain graphs, respectively, while Acc denotes accuracy.
Table 4 presents the results for SCOPe 2.07 dataset at different similarity levels. Remarkably, while as the sequence similarity level is reduced from 100% to 30% and the number of graphs decreases, the classification accuracy increases and reaches even 98% for Beta and Alpha/Beta classes. This implies the removal of redundant sequences actually helps training and improves the performance. For one, the removal of redundant sequences renders structures in the dataset more distinct from one another and reduces pattern overlaps between folds. For another, the removal of redundant sequences greatly reduces the unbalancedness in the dataset. Some proteins such as myoglobin have been much more extensively studied than others for historical or scientific reasons and consequently there are many more structures for them in the PDB than for others. As a result, the numbers of structures in different folds are vastly unbalanced (see the red curve in Fig 7). This hinders training. At 30% sequence similarity level, the unbalancedness is reduced by about an order (Fig 7).

The only exception to the above observation is the Alpha class: the accuracy drops to at 82-83% when the the sequence similarity level is reduced to 30-40%. A possible explanation is that compared to the Beta class, the Alpha class has significantly more folds but a similar number of graphs, while when compared to the Alpha/Beta or Alpha+Beta classes, the Alpha class has only one type of secondary structure and thus less diversity in its graphs, which makes it more difficult to train.

![Figure 7](image)

Figure 7: The number of graphs in each fold of the Beta class at different sequence similarity levels: 100% (in red, sorted in the ascending order) and 30% (in blue). The extent of variations in the number of graphs is greatly reduced at 30% sequence similarity (in blue) compared to that at 100% (in red). The graph data with different fold labels are more balanced at a lower sequence similarity level (30%).

### 3.3 Results on CATH Data Set

In addition to the SCOPe 2.07 data set, we have also analyzed the latest release of the CATH data set. CATH has five levels in their classification hierarchy, namely:

- Class, C-level;
- Architecture, A-level;
- Topology (Fold family), T-level;
- Homologous Superfamily, H-level;
- Sequence families, S-level;

with the counts of Architectures, Topologies, and Super-families in three major classes given in Table 5.
Table 5: The counts of Architectures, Topologies, and Superfamilies in three major classes of the CATH dataset.

CATH topologies are structurally equivalent to SCOP folds. Assuming that the class and architecture assignments are given/known, we classify structures at the topology level for the CATH dataset. In our first setup, as we did with the SCOPe analysis, we choose not to filter structures by sequence similarity and generate a comprehensive data set that captures the original diversity of the CATH database. The result is given in Table 6. The main difference from the SCOPe analysis is that here we classify topologies under two layers of abstraction, namely Class and Architecture, which makes the classification task comparatively easier. We again apply ten-fold cross validation and train the network for 3,500 epochs for every dataset under every structural class. The results are given in Table 6. It is seen the prediction accuracy is higher than SCOPe, reaching even 100% for some classes/architectures.

Table 6: Prediction accuracies across different topologies of the CATH dataset.

Table 7 presents the results of fold classification of the CATH dataset at 30% sequence similarity level. A comparison between the results from Table 7 and Table 6 reveals again the effect of removing redundant sequences on classification accuracy. At 30% sequence similarity level, the dataset becomes more balanced (Fig 7), which may a the major contributing factor to the higher accuracy seen for most of the architectures, except for Orthogonal Bundle and Up-down Bundle, in Table 7). The significant reduction in the numbers of graphs at 30% sequence similarity level (compared to those at 100% in Table 6) may have further facilitated the convergence of the model during training. Compared to the SCOPe dataset (Table 4), the classification accuracy is relatively higher, probably due to the extra Architecture layer in the CATH classification hierarchy.
| Class | Architecture          | No# of Folds | No# of Domains | Accuracy(%)  |
|-------|-----------------------|--------------|----------------|--------------|
| 1 (Mainly Alpha) | 10 (Orthogonal Bundle) | 272          | 3691           | 70%          |
| 1 (Mainly Alpha) | 20 (Up-down Bundle)   | 102          | 1652           | 74%          |
| 1 (Mainly Alpha) | 25 (Alpha Horseshoe)  | 6            | 419            | 100%         |
| 2 (Mainly Beta) | 10 (Ribbon)           | 26           | 243            | 92%          |
| 2 (Mainly Beta) | 20 (Single Sheet)     | 20           | 165            | 100%         |
| 2 (Mainly Beta) | 30 (Roll)             | 40           | 841            | 100%         |
| 2 (Mainly Beta) | 40 (Beta Barrel)      | 46           | 1125           | 97%          |
| 2 (Mainly Beta) | 60 (Sandwich)         | 42           | 2051           | 90%          |
| 2 (Mainly Beta) | 70 (Distorted Sandwich)| 18           | 99             | 100%         |
| 3 (Mainly Alpha Beta) | 10 (Roll)       | 60           | 1141           | 98%          |
| 3 (Mainly Alpha Beta) | 20 (Alpha-Beta Barrel)| 18         | 839            | 100%         |
| 3 (Mainly Alpha Beta) | 30 (2-Layer Sandwich)| 211          | 3546           | 97%          |
| 3 (Mainly Alpha Beta) | 40 (3-Layer(aba) Sandwich)| 116      | 4810           | 96%          |
| 3 (Mainly Alpha Beta) | 50 (3-Layer(bba) Sandwich)| 11          | 211            | 98%          |
| 3 (Mainly Alpha Beta) | 55 (3-Layer(bab) Sandwich)| 6           | 33             | 100%         |
| 3 (Mainly Alpha Beta) | 60 (4-Layer Sandwich) | 16           | 255            | 98%          |

Table 7: Prediction accuracy across different topologies of the CATH dataset at 30% sequence identity.

3.4 Results on Independent Test Data Set

In addition to $k$-fold cross-validation, we performed independent tests on some CATH topologies to assess the classification accuracy of our model on independent test data sets. For this experiment, we drew training samples from three different architectures of three different classes and five topologies under every architecture. A full description of the data set and the results are given in Table 8. We made sure that the training and testing data sets are completely separate and independent and only structures with sequence identity $< 30\%$ were selected.

| Class | Architecture          | No# of Training | No# of Testing | No# misclassification | Accuracy(%)  |
|-------|-----------------------|-----------------|----------------|------------------------|--------------|
| 1 (Mainly Alpha) | 10 (Orthogonal Bundle) | 353             | 56             | 8                      | 48/56 = 86%  |
| 2 (Mainly Beta) | 70 (Distorted Sandwich)| 335             | 45             | 6                      | 39/45 = 87%  |
| 3 (Mainly Alpha Beta) | 40 (3 layer(aba) Sandwich)| 416        | 65             | 4                      | 61/65 = 93%  |

Table 8: Results of independent data sets: classification accuracy across three different topologies at 30% sequence identity.

This is a multi-class classification problem and we have done all-vs-all classification here. In this analysis, we computed the accuracy of the classifier using the standard $Q$ percentage accuracy\(^{28}\), with $Q = C/N$, where $Q$ is the total accuracy, $C$ is number of correct classification, and $N$ is total number of test samples combining all the classes.

3.5 Comparison with Structure Alignment Methods

In this section, we compare our graph neural network model for fold classification with the state-of-the-art structure alignment approaches such as TMalign\(^{29}\) and mTMalign\(^{30}\). mTMalign produces multiple structure alignment based on TM-score of pairwise structure alignments. Both mTMalign and TMalign rely on pairwise sequence similarity between structures. A recent study by Holm\(^{31}\) using 140 query domain structures and
the SCOPe dataset showed that structure alignment-based approaches are rather poor for fold detection when structures are distantly related, i.e., when the database contains no structures of the same family or super-family as the query domain structure.

Since the focus of the present work is fold classification, which is slightly different from fold detection, we use Holm’s results in the following way for fold prediction/assignment. For each query domain, we examine the top 10 structural matching predictions from mTMalign/TMalign, excluding matches that are from the same superfamily as the query domain structure. Based on the top 10 matches and their corresponding SCOPe fold assignments, we perform majority voting to determine the fold assignment for the query domain. Doing this allows us to assess how well structural matching approaches can predict folds when only distant superfamilies are available (for the query protein). We then compare the results with the fold predictions by our graph neural network based approach. To have a fair comparison between our model and structure-matching methods, we exclude from our training set all structures that belong the same superfamilies as the query domains. We then test our trained model with the query domains to check the number of true positives. For example, query domain d1cuka1 from all alpha class belongs to long alpha-hairpin fold. This fold spreads into 20 superfamilies. In our training set, we exclude the superfamily of d1cuka1 and include only the remaining 19 superfamilies in the training.

The results, in terms of the number of true positives, are given in Table 9. For example, for mTMalign, there are 4 true positive cases out of the 11 for the Alpha class, and 2 out of 14, 4 out of 12, and 0 out of 5 for the remaining three classes, respectively.

| Class (# of queries) | Alpha (11) | Beta (14) | Alpha+Beta (12) | Alpha/Beta (5) |
|----------------------|------------|-----------|-----------------|----------------|
| our approach         | 5/11       | 4/14      | 3/12            | 2/5            |
| mTMalign             | 4/11       | 2/14      | 4/12            | 0/5            |
| TMalign              | 2/11       | 2/14      | 3/12            | 1/5            |

Table 9: A comparison between our ML-based approach and two structure matching-based approaches (mTMalign and TMalign) in fold prediction for query structural domains that belongs to a new superfamily. The number of query domain structures used in each class is listed in the top row. The table lists the accuracy in fold assignments for the query domains.

Next we repeat the same analysis but keep every structure in the dataset except the query domain structures. This corresponds to an easier fold prediction scenario where for the query domain, there exist in the dataset domain structures of the same family/superfamily. It was known that for such a scenario structure matching approaches perform extremely well in fold detection. Indeed, this is confirmed in the fold assignment accuracy results given in Table 10.

| Class (# of queries) | Alpha (11) | Beta (14) | Alpha+Beta (12) | Alpha/Beta (5) |
|----------------------|------------|-----------|-----------------|----------------|
| HGP-SL               | 0.85       | 0.71      | 0.81            | 0.40           |
| mTMalign             | 1.0        | 1.0       | 1.0             | 1.0            |
| TMalign              | 1.0        | 1.0       | 1.0             | 1.0            |

Table 10: A comparison between our ML-based approach and two structure matching-based approaches (mTMalign and TMalign) in fold prediction for query structural domains that belong to an existing family or superfamily. The number of query domain structures used in each class is listed in the top row. The table lists the accuracy in fold assignments for the query domains.

The results show that for structure matching methods, the accuracy in fold assignment for a domain that belongs to an existing family or superfamily increases dramatically. As we consider the top 10 hits for every query domain, we see that structures of the same family or superfamily are nearly always selected by structure matching methods (namely TMalign and mTMalign) and thus produces an accuracy of 100%. For our neural network-based model, the improvement in accuracy is not as dramatic. It is possible, however, when we include more node level features, further improvement in accuracy could be achieved. One important observation here is that the inclusion of structures of the same family/superfamily certainly improves the training and fold assignments. Compared with structure matching based methods, our deep neural network model is less dependent on the existence of structures of the same family or superfamily. This makes our approach more suited for fold assignments for query domains of a novel superfamily. Our approach may also be adapted to identify a new fold.
3.6 Summary

There are a few interesting observations after analyzing both CATH and SCOPe 2.07 data sets.

- For both CATH and SCOPe data set, we can see that the all alpha class got lower accuracy compared to others. At different sequence similarity levels their accuracy ranges from 88% – 96% compared to other classes like beta and alpha/beta. One reason could be that alpha fold patterns are less distinct from one another, which is reflected on their overall graph representations. This makes their classification more challenging than other classes.

- The SCOPe data set is most difficult when the data set is large and there are many different folds in the data set. SCOPe groups structures into a few structural classes and diverse superfamilies may be placed under the same fold. This makes it harder to train and optimize, especially when the data set is large.

- For the CATH data set, there is an additional level in the hierarchy between the class and topology levels, namely the architecture level. Under one architecture, structures may share a common overall shape but different topologies. Although CATH topology is the counterpart of the SCOPe fold, the topologies in CATH are more finely categorized by their architectures, making it easier to learn patterns for generalization from CATH. This is seen from the results in Table 7 even when a certain architecture/class contains a large number of topologies and consequently a large graph data set, the classification accuracy remains high. Whereas for SCOPe, the classification accuracy is always affected by the size of the data set - the more graphs are in the data set, the lower is the accuracy.

3.7 Misclassification Analyses

Results on the CATH and SCOPe data set show that certain topologies get a lower accuracy than others. To analyze the patterns of misclassification, we examine closely some misclassification cases and compared them with their actual classification by visualizing both their structures and their topology graphs. We visually examine these structures in PyMol to see if there is any obvious reason for the misclassification. Figure 8 shows the protein graphs of two structures from the mainly-alpha-beta roll architecture. PDB entries are 1yrw chain A (or 1yrwA) of CATH topology 25 (Figure 8(A)), and 2he7 chain A (or 2he7A) of CATH topology 20 (Figure 8(B)). In this case, 1yrwA is misclassified as topology 20. We explored all the alpha beta graphs from topology 20 and found out that PDB entry 2he7A of topology 20 has a similar topological arrangement like 1yrwA of topology 25 (see Figure 9). Both entries have ten alpha helices and 11 and 12 beta strands. This similarity explains why 1yrwA is misclassified into the topology of 2he7A.

![Figure 8: Topology graphs of two alpha-beta-roll proteins](image)

![Figure 9: PyMOL structures of two alpha-beta-roll proteins](image)
4 Materials and Methods

4.1 Protein Structure Classification

Every protein structure has an evolutionary origin and can be classified into their taxonomic hierarchy. Also to better understand the structural relationship with other proteins, their taxonomic hierarchical classification gives important information. SCOPe and CATH are two databases that classify protein structure topologies into their taxonomic hierarchy. Although these traditional approaches are present, automated approaches are in need due to the rapid growth in the availability of new structures. Specifically, machine learning-based approaches for structure classification are becoming increasingly attractive. SCOPe is a mainly manually curated database that classifies protein structures in their fold hierarchy considering their evolutionary relationship. At first, SCOPe partitions proteins into domains. These domains are then classified into four levels. The highest level of this hierarchy is class. There are seven classes in SCOPe, namely, (a) alpha proteins, (b) beta proteins, (c) alpha/beta proteins, (d) alpha+beta proteins, (e) multi-domain proteins, (f) membrane and cell surface proteins and peptides, and (g) small proteins. Class describes the content of secondary structure elements in protein domains. The next level in this classification hierarchy corresponds to protein fold. Fold groups structures having similar number and order of secondary structure elements with similar connectivity patterns. There are 1232 protein folds all together in SCOPe. Fold groups together structurally similar superfamilies that might not have any evolutionary relationship. Family contains proteins that share similar sequences and close evolutionary relationships. Superfamily combines proteins with similar functional and structural features that descend from same ancestor. The primary concern of this work is to classify protein structures into their respective folds. In summary it can be said that, the evolutionary relationship among known protein structures is the basis of classification for SCOPe. On the other side, CATH classification relies on structural
rearrangements and structural properties of proteins. It applies a structure comparison-based method for classifying unknown structures. CATH has three major classes in their first layer. Those are: a) mainly alpha, b) mainly beta and c) mainly alpha-beta. In the second layer, CATH classifies the structures based on the common overall shape of the topologies. This level is called the architecture level. Combining the three major classes there are around 40 architectures in CATH. The next level is the topology/fold level. Inside topology level, structures are clustered based on their architecture and connectivity pattern of the secondary structure elements. Combining the 3 major classes and 40 architectures, there are around 1282 topologies in CATH. One interesting observation here is, although CATH classifies much more domains than SCOPe, the numbers of fold/topology in both databases are almost equal. It reminds that the number of folds in nature is limited although the structure and sequences are ever increasing. Finally, after the topology layer, CATH classifies the structures into homologous superfamilies. There are around 5660 superfamilies in the CATH database, combining all the structural classes, architectures and folds. CATH superfamily is a mixture of SCOPe superfamily and family and the number of superfamilies in CATH reflects the sum of both.

4.2 Protein Topology Graph Library

The PTGL comprises protein topologies based on unique graph-theoretic definitions at different levels of abstraction. Protein topology graphs were defined on the secondary structure level mainly for protein structure prediction\(^{32}\), analysis\(^{33}\) and comparison of protein structures\(^{34,35}\). PTGL\(^{19,16,36}\) reads the atom coordinates from a macromolecular crystallographic information file (mmCIF) of the PDB\(^{37}\) and the assignments of amino acids to secondary structure elements from a DSSP file\(^{38}\), all per chain. All types of helices are treated as one helix type. Single beta bridges that occur next to a beta sheet are treated as beta sheet. Secondary structure elements that are interrupted by one differing classification, e.g., loop, are merged. Finally, PTGL neglects secondary structure elements of less than three amino acids.

PTGL computes all atom-atom contacts applying a Euclidean distance less than 4 Å. Depending on the involved atoms, the contacts are classified as backbone-backbone (BB), backbone-sidechain (BC) or sidechain-sidechain (CC). A contact between secondary structure elements is defined in dependence of the type of involved secondary structure elements, i.e., helix or strand, and rules for the involved atom contacts, i.e., BB, BC or CC (see Table 11).

Table 11: Required residue-residue contacts for assignment of a contact between secondary structure elements. Residue-residue contacts are differentiated by the type of the atoms involved in the contact as backbone-backbone (BB), backbone-sidechain (BC) and sidechain-sidechain (CC) contacts.

| Type of SSE 1   | Type of SSE 2   | Required residue-residue contacts |
|----------------|----------------|-----------------------------------|
| beta strand    | beta strand    | BB > 1 or BC > 2                  |
| beta strand    | helix          | BB > 1 or BC > 3 or CC > 3        |
| helix          | helix          | BC > 3 or CC > 3                  |

A direction of secondary structure elements can be inferred from N to C terminus. PTGL creates a vector in three-dimensional space for each SSE. The initial point and terminal point are the geometric mean of the atom positions of the first and last four residues, respectively. For shorter secondary structure elements of length \(n\), the first and last \(n - 1\) residues are used instead. PTGL computes for each secondary structure element contact the angle between the secondary structure elements and assigns a parallel, mixed or antiparallel orientation of the secondary structure elements to each other for angles up to a degree of 65, 115 and 180, respectively.

We define a Protein Graph (PG) of a protein chain, where the vertices correspond to secondary structure elements and the edges to spatial neighborhoods of the vertices (see Figure 12a). The vertices are labeled by their SSE type, i.e., strand or helix. The edges are labeled by the orientation of the participating secondary structure elements to each other, i.e., parallel, mixed or antiparallel. A Folding Graph (FG) is a connected component of a PG (see Fig. 12, vertices 2 - 4). An alpha, beta or alpha-beta (albe) FG or FG contains only helices, only strands or both helices and strands, respectively (see Fig. 12).
Figure 12: A (a) alpha-beta, (b) alpha and (c) beta Protein Graph. The vertices are horizontally aligned from N to C terminus. Helices and strands are drawn as red circles and black boxes, respectively. Edges represent spatial neighborhoods and are colored red, blue or green for parallel, antiparallel and mixed orientation of the involved secondary structure elements to each other, respectively. The vertices are labeled with increasing IDs by occurrence in the graph (PG) and occurrence in the sequence (SQ).

4.3 Graph Classification

Graph classification is the problem of learning a function from a set of attributed graphs to predict their class labels. Given a set of attributed graphs $G = \{(G_1, y_1), (G_2, y_2), ..., (G_n, y_n)\}$, the goal is to learn a function $f: G \rightarrow Y$ to approximate the graph labels. Here, $G$ is the input space of the graphs, and $Y$ is the set of all graph labels.

4.4 Graph Processing and Feature Extraction

We identified the PTGL graph in JSON format corresponding to every entry in every data set and then parsed that graph to convert into a pyTorch geometric graph. The JSON-formatted graph contains all the relevant information regarding the graph structure, such as node labels, edge labels, adjacency relations etc. We parsed and processed these information to capture the node features and construct the graph in pyTorch format from the adjacency relation. In this work, we considered only node features for every pyTorch graph. We have selected three node features to learn patterns from the graph. Those are the node types; (alpha helix or beta strand), node degrees and the number of residues or length of secondary structure element element in each node.

4.5 HGP-SL for Protein Fold Classification

The HGP-SL (Hierarchical Graph Pooling with Structure Learning) work was developed by Zhang et al. A summary of HGP-SL is given below. Readers are referred to the original work for details.

4.5.1 Problem Formulation

Given a set of labelled graph data $G = \{G_1, G_2, ..., G_n\}$, the goal is to learn patterns from this data set and predict the label for unknown graphs. Each graph could be represented as $G_i = (V_i, E_i, A_i, X_i, Z_i, Y_i)$. $V_i$ and $E_i$ represent the node and edge set of the graph, where $n_i$ and $e_i$ correspond to the number of nodes and edges, respectively, in the graph. $A_i \in R^{n_i \times n_i}$ represents the adjacency matrix which contains all the edge information of the graph. $X_i \in R^{n_i \times d}$ is the node feature matrix where $d$ represents the dimension of the node attributes. $Z_i^{f \times f}$ is the edge features matrix where $f$ is the dimension of the edge attributes. $Y \in R^{n \times s}$ is the label matrix that corresponds to the full graph data set. Here, $s$ is the length of the label set, and $n$ is the number of graphs in the data set. $Y_i$ contains the label for an individual graph $i$. During the graph pooling operation, the number of nodes and graph structure changes at each subsequent layer. The $i$th graph trained on $k$th layer could be represented as $G_i^k$ having nodes $n_i^k$. The adjacency matrix and hidden representation matrix for the $i$th graph at $k$th layer could be represented as $A_i^k \in R^{n_i^k \times n_i^k}$ and $H_i^k \in R^{n_i^k \times d}$. 


4.5.2 Graph Convolutional Neural Network

Graph Convolutional Neural Networks (GCN) are a variant of GNN. The graph convolution operation can be categorized into two branches. Those are spatial and spectral approaches. The spectral-based approaches define graph convolution operation by parameterized filters from the perspective of graph spectral theory. However, due to the computational flexibility and efficiency, spatial-based approaches gained much popularity in recent years. Spatial-based approaches are analogues to conventional convolution operation on images. Graph convolution works by utilizing node’s spatial relation. By directly aggregating node’s neighborhood information, node-level representations are generated in every subsequent convolution layer, which in turn contributes to generate the graph-level representation. For generating a representation for $k$th layer, GCN takes as input graph $G$’s adjacency matrix and $k - 1$th layer’s hidden representation matrix. It generates the representation for $k$th layer as follows:

$$ H_k = \sigma(\tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}H_{k-1}W_k), $$

where $\sigma$ is the non-linear activation function. $\tilde{A} = A + I$ is the adjacency matrix with self-connections. $W_k \in \mathbb{R}^{d_{k-1} \times d_k}$ is the trainable weight matrix for layer $k$, and $\tilde{D}$ is the diagonal degree matrix of $\tilde{A}$.

![Hierarchical Graph pooling with Structure Learning for Graph Classification](https://github.com/cszhangzhen/HGP-SL)

Figure 13: Hierarchical Graph pooling with Structure Learning for Graph Classification. https://github.com/cszhangzhen/HGP-SL

4.5.3 HGP-SL

HGP-SL is a novel graph classification approach with sophisticated pooling operation. This method utilizes the node features and the entire graph’s topological information for predicting labels associated with the graphs. The common approach for graph classification is to summarize all node information to create a global representation of the graph. This approach neglects the entire graph’s structural information, and therefore, the graph level representation that is generated this way is not very intense. On the other hand, GNN cannot aggregate node information in a hierarchical way. Moreover, to create a meaningful representation of the graph, sometimes it is required to put a special significance to graph substructures, as different substructure may contribute differently to the whole graph representation. That’s why, to learn from both graph local information and global structure and to generate hierarchical representation, a graph pooling operation is required. HGP-SL provides such a pooling process to meet this requirement. It provides a sophisticated pooling layer in addition with a graph convolution layer to be used with different graph neural network architectures. In this subsection, we will describe the two major components of HGP-SL for graph classification.

- **Graph Pooling Operation:** The pooling layer in HGP-SL is a non-parametric layer. The proposed pooling layer works by selecting a set of informative nodes to produce an induced sub-graph for the next layer. The selection process works as follows:
  - It first re-orders the nodes according to the node information scores.
  - The node information score can be computed by the Manhattan distance between the node representation itself and the one calculated from its neighbors:

$$ p = \gamma(G_i) = ||(I_k^i - (D_k^i)^{-1}A_k^i)H_k^i|| $$

(2)
Here, $A_i^k \in R^{n_i^k \times n_i^k}$ and $H_i^k \in R^{n_i^k \times d}$ are the adjacency and node representation matrices, respectively. $D_i^k$ is the diagonal matrix of $A_i^k$, and $I_i^k$ is the identity matrix. $p \in R^n$ encodes the information score of each node in the graph. $\gamma$ is the node score-encoding function. Here, it is the Manhattan distance.

- After calculating the node information score of every node, top-ranked nodes are selected as follows:

$$\text{idx} = \text{top}_{\text{rank}}(p, [r \ast n_i^k])$$
$$H_i^{k+1} = H_i^k(\text{idx}, :)$$
$$A_i^{k+1} = A_i^k(\text{idx}, \text{idx})$$

Here $r$ is the pooling ratio, $\text{top}_{\text{rank}}$ is the function by which top $n_i^{k+1} = \lceil r \ast n_i^k \rceil$ indices are selected. For the induced sub-graph, the node representation matrix and adjacency matrix are extracted using $H_i^k(\text{idx}, :)$ and $A_i^k(\text{idx}, \text{idx})$, respectively. So, both the node feature and graph structure information are retained this way for the next layer.

**Structure Learning Mechanism:** HGP-SL incorporates a novel structure learning mechanism to learn a refined structure from the induced sub-graph in each pooling layer. After the pooling operation, the induced sub-graph structure may result in highly correlated nodes being disconnected, which in turn may prevent message passing and effective node level representation learning. Moreover, the induced sub-graph structure might preserve important domain knowledge for the graph that originates from a specific domain-like protein structure. Considering all these issues, the structure-learning mechanism turns out to be an effective scheme to produce meaningful hierarchical representation of the graph. The basics of the structure-learning layer is a single-layer neural network parametrized by a weight vector $\bar{\alpha} \in R^{1 \times 2d}$. It takes as input graph $G_i$’s $k$th hidden layer presentation $H_i^k \in R^{n_i^k \times d}$ and structure information $A_i^k \in R^{n_i^k \times n_i^k}$ and encodes a refined graph structure by learning the pairwise relationship between each pair of nodes using an attention mechanism. The similarity score between two nodes $v_p$ and $v_q$ can be expressed by the following equation:

$$E_i(p, q) = \sigma(\bar{\alpha}[H_i^k(p,:)]H_i^k(q,:)^T) + \lambda.A_i^k(p,q)$$

(3)

Here, $\sigma$ is a non-linear activation function such as ReLU. $H_i^k(p,:) \in R^{1 \times d}$ and $H_i^k(q,:) \in R^{1 \times d}$ correspond to the $p$th and $q$th row of the hidden representation matrix $H_i^k$ that represent the nodes $v_p$ and $v_q$, respectively.

### 4.5.4 Protein Fold Classification

We formulate the protein fold classification problem as a graph classification problem according to the definition described in section 4.3. Here, we take as input, the protein graphs of PTGL for a number of PDB entries from SCOPe and CATH data sets and give them labels according to the SCOPe/CATH fold labels.

### 4.6 Data Sets

We have analyzed the SCOPe 2.07 data set for training, validation and testing. We classify four different structural classes of proteins, namely, all-alpha, all beta, all alpha and beta proteins, and alpha and beta proteins. We also examine the latest version of the CATH data set. CATH classifies the proteins in three major classes: mainly-alpha, mainly-beta and mainly-alpha-beta. Additionally, we examine three different fold recognition benchmark data sets, namely the DD data set, EDD data set, and TG data set. The DD data set consists of a training and testing data set. The sequences in this data set cover 27 protein folds, which are categorized into four protein structural classes; alpha, beta, alpha/beta, alpha+beta. The training set contains 311 protein sequences with ≤ 40% residue identity, and the testing set contains 383 sequences with ≤ 35% residue identity. Also the training and testing set have ≤ 35% sequence similarity. The EDD data set is an extended version of DD data set, which is also composed of 27 protein folds categorized into four different classes. There are 1612 protein sequences in the TG data set, which covers 30 different folds from SCOP 1.73 having ≤ 25% sequence similarity. These three benchmark data sets are classic data sets made up with sequence data, for which we constructed topology graphs for our graph classifier and compared the results with the sequence-based models.
4.6.1 Network Parameters

We trained the HGP-SL model on all the aforementioned benchmark data sets in addition to the SCOPe 2.07 data set. We tried different network settings for different data sets. A batch size of 512 and number of epoch of 3000 was mainly used. The learning rate, dropout ratio and pooling ratio were 0.005, 0.1 and 0.5 most of the time. During the training time, the model optimizes by generating minibatches of graphs, where each batch is considered as a connected graph and within this densely connected graph, each individual graph is like a connected sub-graph.

4.7 Evaluation Strategy

We applied ten-fold cross validation in our data set. We divided each data set into ten equal-sized groups and used eight out of ten for training, one for test and one for validation. We did this for every unique group. That means, every data sample got the opportunity to be used in the holdout or test set once. The model is trained ten times, and then, the overall score is computed by taking the mean of the ten scores.

5 Conclusion

In this study, domain specific protein graphs from PTGL and graph neural network (GNN) have been used for protein fold classification. The work demonstrates that graph neural network is potent for fold assignments, especially for query proteins of a novel superfamily, in which regard it outperforms even top-notch structure-matching-based methods. As a future work, we plan to extend our method to classify protein complexes.

References

[1] Helen M. Berman, Kim Henrick, and Haruki Nakamura. Announcing the worldwide Protein Data Bank. *Nature Structural Biology*, October:980–980, 2003.
[2] Catherine L. Lawson, Ardan Patwardhan, Matthew L. Baker, Corey Hryc, Eduardo Sanz Garcia, Brian P. Hudson, Ingvar Lagerstedt, Steven J. Ludtke, Grigore Pintilie, Raul Saha, John D. Westbrook, Helen M. Berman, Gerard J. Kleweweg, and Wai Chiu. EMDataBank unified data resource for 3DEM. *Nucleic Acids Research*, 44(D1):D396–D403, November 2015.
[3] James Lyons, Kulidip K. Paliwal, Abdollah Dehzangi, Rhys Hefermann, Tatsuhiro Tsumoda, and Alok Sharma. Protein fold recognition using HMM–HMM alignment and dynamic programming. *Journal of Theoretical Biology*, 393:67–74, 2016.
[4] Jiaqi Xia, Zhenling Peng, Dawei Qi, Hongbo Mu, and Jianyi Yang. An ensemble approach to protein fold classification by integration of template-based assignment and support vector machine classifier. *Bioinformatics*, 33(6):863–870, December 2016.
[5] Leyi Wei, Minghong Liao, Xing Gao, and Quan Zou. Enhanced Protein Fold Prediction Method Through a Novel Feature Extraction Technique. *IEEE Transactions on NanoBioscience*, 14(6):649–659, 2015.
[6] Alexey G. Murzin, Steven E. Brenner, Tim Hubbard, and Cyrus Chothia. SCOP: A structural classification of proteins database for the investigation of sequences and structures. *Journal of Molecular Biology*, 247(4):536–540, 1995.
[7] CA Orengo, AD Michie, S Jones, DT Jones, MB Swindells, and JM Thornton. CATH – a hierarchic classification of protein domain structures. *Structure*, 5(8):1093–1109, 1997.
[8] Naomi K. Fox, Steven E. Brenner, and John-Marc Chandonia. SCOPe: Structural Classification of Proteins—extended, integrating SCOP and ASTRAL data and classification of new structures. *Nucleic Acids Research*, 42(D1):D304–D309, December 2013.
[9] William R. Taylor and Christine A. Orenco. Protein structure alignment. *Journal of Molecular Biology*, 208(1):1–22, 1989.
[10] Taeho Jo, Jie Hou, Jesse Eickholt, and Jianlin Cheng. Improving Protein Fold Recognition by Deep Learning Networks. *Scientific Reports*, 5(1):17573, December 2015.
[11] Jie Hou, Badri Adhikari, and Jianlin Cheng. DeepSF: deep convolutional neural network for mapping protein sequences to folds. *Bioinformatics*, 34(8):1295–1303, December 2017.
[12] Jianwei Zhu, Haicang Zhang, Shuai Cheng Li, Chao Wang, Lupeng Kong, Shiwei Sun, Wei-Mou Zheng, and Dongbo Bu. Improving protein fold recognition by extracting fold-specific features from predicted residue–residue contacts. *Bioinformatics*, 33(23):3749–3757, August 2017.

[13] Yechan Hong, Yongyu Deng, Haofan Cui, Jan Segert, and Jianlin Cheng. Classifying protein structures into folds by convolutional neural networks, distance maps, and persistent homology. *bioRxiv*.

[14] Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. Graph neural networks: A review of methods and applications. *AI Open*, 1:57–81, 2020.

[15] Zhen Zhang, Jiajun Bu, Martin Ester, Jianfeng Zhang, Chengwei Yao, Zhi Yu, and Can Wang. Hierarchical Graph Pooling with Structure Learning. *arXiv CoRR*, abs/1911.05954, 2019.

[16] May, Patrick and Kreuchwig, Annika and Steinke, Thomas and Koch, Ina. PTGL: a database for secondary structure-based protein topologies. *Nucleic Acids Research*, 38(suppl 1):D326–D330, 2010.

[17] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Židek, 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, Aug 2021.

[18] Kathryn Tunyasuvunakool, Jonas Adler, Zachary Wu, Tim Green, Michal Zielinski, Augustin Židek, Alex Bridgland, Andrew Cowie, Clemens Meyer, Agata Laydon, Sameer Velankar, Gerard J. Kleywegt, Alex Bateman, Richard Evans, Alexander Pritzel, Michael Figurnov, Olaf Ronneberger, Russ Bates, Simon A. A. Kohl, Anna Potapenko, Andrew J. Ballard, Bernardino Romera-Paredes, Stanislav Nikolov, Rishub Jain, Ellen Clancy, David Reiman, Stig Petersen, Andrew W. Senior, Koray Kavukcuoglu, Ewan Birney, Pushmeet Kohli, John Jumper, and Demis Hassabis. Highly accurate protein structure prediction for the human proteome. *Nature*, 596(7873):590–596, Aug 2021.

[19] May, Patrick and Barthel, Stefan and Koch, Ina. PTGL—a web-based database application for protein topologies. *Bioinformatics*, 20(17):3277–3279, 2004.

[20] Leyi Wei and Quan Zou. Recent Progress in Machine Learning-Based Methods for Protein Fold Recognition. *International Journal of Molecular Sciences*, 17(December), 2016.

[21] Qiwen Dong, Shuigeng Zhou, and Jihong Guan. A new taxonomy-based protein fold recognition approach based on autocross-covariance transformation. *Bioinformatics*, 25(20):2655–2662, August 2009.

[22] Yuedong Yang, Eshel Faraggi, Huiying Zhao, and Yaoqi Zhou. Improving protein fold recognition and template-based modeling by employing probabilistic-based matching between predicted one-dimensional structural properties of query and corresponding native properties of templates. *Bioinformatics*, 27(15):2076–2082, June 2011.

[23] James Lyons, Abdullah Dehzangi, Rhys Heffernan, Yuedong Yang, Yaoqi Zhou, Alok Sharma, and Kuldip Paliwal*. Advancing the Accuracy of Protein Fold Recognition by Utilizing Profiles From Hidden Markov Models. *IEEE Transactions on NanoBioscience*, 14(7):761–772, 2015.

[24] Ngaam J. Cheung, Xue-Ming Ding, and Hong-Bin Shen. Protein folds recognized by an intelligent predictor based-on evolutionary and structural information. *Journal of Computational Chemistry*, 37(4):426–436, 2016.

[25] Daozheng Chen, Xiaoyu Tian, Bo Zhou, and Jun Gao. ProFold: Protein Fold Classification with Additional Structural Features and a Novel Ensemble Classifier. *BioMed Research International*, 2016:1–10, August 2016.

[26] Wessam Elhefnawy, Min Li, Jianxin Wang, and Yaohang Li. DeepFrag-k: a fragment-based deep learning approach for protein fold recognition. *BMC Bioinformatics*, 21(S6), November.

[27] Aik Choon Tan, David Gilbert, and Yves Deville. Multi-class protein fold classification using a New Ensemble Machine Learning Approach. *Genome informatics. International Conference on Genome Informatics*, 14:206–17, February 2003.

[28] Pierre Baldi, Soren Brunak, Yves Chauvin, Claus A. F. Andersen, and Henrik Nielsen. Assessing the accuracy of prediction algorithms for classification: an overview. *Bioinformatics*, 16(5):412–424, May 2000.
[29] Yang Zhang and Jeffrey Skolnick. TM-align: a protein structure alignment algorithm based on the TM-score. *Nucleic Acids Research*, 33(7):2302–2309, 01 2005.

[30] Runze Dong, Zhenling Peng, Yang Zhang, and Jianyi Yang. mTM-align: an algorithm for fast and accurate multiple protein structure alignment. *Bioinformatics*, 34(10):1719–1725, 12 2017.

[31] Liisa Holm. Benchmarking fold detection by DaliLite v.5. *Bioinformatics*, 35(24):5326–5327, 07 2019.

[32] Frieder Kaden, Ina Koch, and Joachim Selbig. Knowledge-based prediction of protein structures. *Journal of Theoretical Biology*, 147:85–100, January 1990.

[33] Ina Koch, Frieder Kaden, and Joachim Selbig. Analysis of protein sheet topologies by graph-theoretical methods. *Proteins: Structure, Function, and Genetics*, 12:314–324, 1992.

[34] Ina Koch, Thomas Lengauer, and Egon Wanke. An algorithm for finding maximal common subtopologies in a set of proteins. *Journal of Computational Biology*, 3:289–306, February 1996.

[35] Ina Koch. Fundamental study: Enumerating all connected maximal common subgraphs in two graphs. *Theoretical Computer Science*, 250:1–30, 2001.

[36] Wolf, Jan Niclas and Keßler, Marcus and Ackermann, Jürg and Koch, Ina. PTGL: extension to graph-based topologies of cryo-EM data for large protein structures. *Bioinformatics*, 37(7):1032–1034, 2021.

[37] Helen M Berman, John Westbrook, Zukang Feng, Gary Gilliland, Talapady N Bhat, Helge Weissig, Ilya N Shindyalov, and Philip E Bourne. The Protein Data Bank. *Nucleic Acids Research*, 28(1):235–242, 2000.

[38] Wouter G Touw, Coos Baakman, Jon Black, Tim AH Te Beek, Elmar Krieger, Robbie P Joosten, and Gert Vriend. A series of PDB-related databanks for everyday needs. *Nucleic Acids Research*, 43(D1):D364–D368, 2015.

[39] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann Lecun. Spectral networks and locally connected networks on graphs. December 2013.

[40] Y-h Taguchi and Michael Gromiha. Application of amino acid occurrence for discriminating different types of globular proteins. *BMC Bioinformatics*, 8:404, February 2007.