DRUG-TARGET AFFINITY PREDICTION USING A GRAPH-BASED APPROACH ENRICHED WITH MOLECULE WORDS

by

Cansu Damla Yılmaz

B.S., Computer Engineering, Koç University, 2020

B.A., Economics, Koç University, 2020

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Computer Engineering

Boğaziçi University

2023

To my grandparents Hanife & Recep Aras...

ACKNOWLEDGEMENTS

First of all, I would like to thank my advisor Assoc. Prof. Arzucan Özgür, for her support and advices during the preparation of my thesis.

I express my gratitude to our advisors in KimDil Project, Prof. Nilgün Karalı, Assoc. Prof. Arzucan Özgür, Assoc. Prof. Elif Özkırımlı, and Prof. Kutlu Ülgen, and to project members Berk Atıl, Can Koban, Taha Koulani, Selen Parlar Özçelik, Rıza Özçelik, Nural Özel, Asu Büşra Temizer and Gökçe Uludoğan for valuable discussions we made in our weekly meetings. Additionally, I thank Asu Büşra Temizer and Taha Koulani for the analysis they did for Section 6.2.

I would also like to thank Sümeyye Ağaç, Büşra Oğuzoğlu and Pınar Süngü for their partnership in lecture projects and their support in my master studies.

I thank TUBITAK (Scientific and Technological Research Council of Turkey) for their computational resource support in the scope of KimDil Project with grant number 119E133.

I show my gratitude to Prof. Tunga Güngör and Prof. Özlem Keskin Özkaya for accepting to be in my thesis committee and evaluating my work.

And finally, I am deeply grateful to my family who never stopped encouraging me in every condition.

ABSTRACT

DRUG-TARGET AFFINITY PREDICTION USING A GRAPH-BASED APPROACH ENRICHED WITH MOLECULE WORDS

Wet-lab experiments to predict the affinity of drugs for their targets are costly and time consuming. Computational methods can provide an alternative to early stage experiments and guide the research process. Recently, the use of natural language processing techniques to represent molecules has become popular and has led to successful results. In our work, we assume that proteins and ligands, like human languages, have their own languages and that these languages consist of meaningful smaller parts that we call words. We identify protein and ligand words based on their 1D sequences using a subword tokenization method and represent protein-ligand interactions with a heterogeneous graph consisting of four different node types corresponding to proteins, ligands, protein words, and ligand words. A graph-based approach is used to learn embeddings for the nodes in the graph. These embeddings are fed into a deep learning model for predicting protein-ligand binding affinity. We show that using their word embeddings to represent novel proteins and/or ligands not present in the training set improves the results compared to the case where no words are used. Using pre-trained word embeddings for previously unknown molecules is also efficient in terms of complexity, as we do not need to re-train the input graph to learn the embeddings for these new molecules.

ÖZET

MOLEKÜL KELİMELERİYLE ZENGİNLEŞTİRİLMİŞ AĞ ÇİZGESİ YAKLAŞIMLI İLAÇ-PROTEİN ETKİLEŞİMİ TAHMİNİ

İlaç-protein etkileşimi tahmini için yapılan fiziksel deneyler pahalı olduğundan ve önemli ölçüde zaman gerektirdiğinden hesaplamalı yöntemler deneylerin ön aşamaları için alternatif olabilir ve araştırmanın gidişatına kılavuzluk edebilirler. Son zamanlarda moleküllerin temsilini bulmak için doğal dil işleme yöntemlerinin kullanılması yaygınlaştı ve bu yöntemlerle başarılı sonuçlar elde edildi. Biz bu çalışmada protein ve ligantların doğal diller gibi kendilerine özgü dillerinin olduğunu varsayıp, kelime olarak adlandırdığımız küçük anlamlı parçalarının da olduğunu öne sürüyoruz. Protein ve ligantların kelimelerini bir alt kelime belirleme yöntemi yardımıyla protein ve ligantların tek boyutlu dizilimlerini kullanarak elde ediyoruz. Girdiyi heterojen ağ çizgesi olarak gösterip, ağ çizgesindeki dört farklı çeşite (protein, ligant, protein kelimesi, ligant kelimesi düğümleri) sahip her düğümün temsilini öğreniyoruz. Protein ve ligantlar arasındaki bağlanma kuvvetini tahmin etmek için öğrendiğimiz temsilleri tahmin modeline besliyoruz. Sonuç olarak, bilinmeyen protein ve/veya ligantların temsilleri için onların kelimelerinin temsillerinin kullanılmasının, kelimelerin kullanılmadığı duruma göre daha iyi sonuçlar verdiğini gösteriyoruz. Yeni moleküllerin temsillerini öğrenmek için girdi ağ çizgesini tekrar eğitmediğimizden, önceden eğitilmiş kelime temsillerinin daha önce görülmemiş moleküller için kullanılması hesaplama karmaşıklığı açısından da verimlidir.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

IC_{50}	Half Maximal Inhibitory Concentration
K_d	Dissociation Constant
K_i	Inhibition Constant
r^2	The Coefficient of Determination

LIST OF ACRONYMS/ABBREVIATIONS

1D	One Dimensional
3D	Three Dimensional
ADRB2	Beta-2 Adrenergic Receptor
Bi-LSTM	Bidirectional Long-Short Term Memory
CA	Carbonic Anhydrase
CI	Concordance Index
COPD	Chronic Obstructive Pulmonary Disease
COVID-19	Coronavirus Disease
DTA	Drug-Target Affinity
GAT	Graph Attention Network
GCN	Graph Convolutional Network
GIN	Graph Isomorphism Network
HetGNN	Heterogeneous Graph Neural Network
IDF	Inverse Document Frequency
MSE	Mean Squared Error
NLP	Natural Language Processing
RMSE	Root Mean Squared Error
RSS	Residual Sum of Squares
SMILES	Simplified Molecular-Input Line-Entry System
TF	Term Frequency
TF-IDF	Term Frequency-Inverse Document Frequency
TSS	Total Sum of Squares

1. INTRODUCTION

Drug discovery is very expensive task and requires a significant amount of time and effort. Computational methods are important since they fasten the process of early stage experiments. Some interactions that have been shown to have lower probability with the help of computational techniques can be eliminated during physical experiments. In some cases such as COVID-19, getting correct results fastly is essential. Reducing the number of pysical experiments and offering new candidate solutions can be achieved by computational methods. Thus, computational approaches can play role in the invention of new drugs and help to uncover unknown use cases for existing drugs. In this work also, our aim is to fasten early stage experiments by narrowing the research space.

NLP techniques are widely use in computational biology. In the literature, textual representations of molecules are utilized to get information about compounds. In this work, we assume proteins and ligands have some common features with natural languages and we use their 1D textual representations to predict affinities. For proteins, we use their amino acid sequences and for ligands, we use SMILES representations. 1D representations are simple, easy to process and has less computational complexity compared to 3D representations and also possess valuable information in themselves.

We propose a graph-based model to learn the representations of proteins and ligands. Treating them as natural languages, we assume that they have some meaningful subparts which we call words. We form an heterogeneous graph including protein, ligand, protein word and ligand word information. Since our aim is to see the effect of protein and ligand words, we use two input graphs, one is without and the other is with words. Protein and ligand graph returns the representations of proteins and ligands. The graph with words returns protein, ligand, protein word and ligand word representations. We fed the learned embeddings to the prediction part consisting of three fully connected layers. We compare the results of including words into the graph

with using only protein and ligand information. In addition, we compare different representation methods for proteins and ligands as pretrained BERT embeddings, embeddings learned from the graph or representing proteins and ligands using average of their word embeddings. As the evaluation metrics we use CI, r^2, MSE and RMSE. We observe that using word embeddings improve the results for unknown compounds or unknown pairs.

With this work, we introduce a new approach to predict drug-target affinity using word information of proteins and ligands. Passing through the common word nodes of molecules with previously known and unknown interactions, we successfully obtain information about the unknown interactions. Thus, in our work, we conclude that the subparts of molecules are meaningful and contain valuable information that plays role in binding. We show that when an unknown drug-target pair comes, we can use the average of their previously learned word embeddings to predict the strength of their interaction.

2. RELATED WORK

2.1. Drug-Target Affinity Prediction

Drug-target affinity prediction has been studied for a while and it becomes popular as the time passes. KronRLS is proposed by Pahikkala et al. in 2015, representing the task as a regression model to obtain more realistic results [1]. SimBoost method developed by He et al. makes predictions for unobserved drug-target pairs' affinities as we propose in this thesis. They use gradient boosting to train affinities of known pairs [2]. They also propose SimBoostQuant in the same work which returns a range for predicted affinity values. Öztürk et al. introduced DeepDTA, which we take as baseline in our work [3]. We use the same settings with DeepDTA in our prediction model, three fully connected layers with dropout between layers, to be able to make comparison. Our method differs in the calculation of embeddings which will be explained in the following sections. DeepDTA uses three convolutional layers followed by max pooling operation for both drugs and targets to obtain embeddings. DeepDTA model outperforms previously explained KronRLS and SimBoost methods.

In ChemBoost, Özçelik et al. assume SMILES as a language as we do in our work and they propose finding the representations of proteins using their sequence features and the words of ligands they are connected [4]. They apply eXtreme Gradient Boosting to predict protein-ligand affinities [5] using KIBA and BindingDB [6] datasets. They represent ligands using SMILESVec [7] and identify chemical words using BPE algorithm introduced in [8]. Similarly, we apply BPE to find protein and ligand words for our proposed model.

Similar to DeepDTA and our model, Shim et al. use also CNNs for the prediction of drug-target affinity and propose SimCNN-DTA. Instead of learning embeddings, they use similarity matrices of drugs and targets. Outer product of drug's and target's similarity vectors is given to CNNs as input [9].

2.2. Graph-Based Methods

2.2.1. Heterogeneous Graph Learning Methods

Metapath2Vec algorithm is a method to learn representation of heterogeneous graphs proposed by Dong et al. [10]. The idea is to do random walks on the graph using predetermined metapaths and identify neighbors of the nodes. The algorithm applies skip-gram model to find node representations, meaning that it uses node in the center to predict the surrounding nodes [11]. In our work we define metapaths according to node types in our heterogeneous graph and use metapath2vec to learn node embeddings as explained in Section 4.2 in detail.

Zhang et al. propose HetGNN, where after learning embeddings in content aggregation stage, attention mechanism is also added to combine embeddings of different node types [12]. In our case, we concatenate protein and ligand node embeddings to transfer to the prediction model. Differently from ours, they also use Bi-LSTM to hold position information. HetGNN is a general method to predict relations in heterogeneous graphs and use academic graph with author, paper and venue nodes and review graph consisting of user and item nodes. However, it is a good example for heterogeneous graph learning and the idea can be adapted to any task that uses heterogeneous graphs such as ours.

The attention mechanism is used in HAN model [13]. Node-level attention is used to learn the importance of features of neighbors and emantic-level attention mechanism is utilized to learn the importance of different metapaths. In our proposed method, we learn the important words not by attention mechanism, but using NLP techniques as explained in Section 4.4.

2.2.2. Graph Learning In Drug-Target Affinity Prediction

MHGNN model, introduced recently, uses metapaths to define semantic relations between nodes of heterogeneous graph for the task of drug-target interaction prediction [14]. However, we learn semantics using words of proteins and chemical compounds and show that models that use word information perform better than model that only use sequence information for unknown protein-ligand interactions' affinity prediction.

In GraphDTA drugs are represented in the form of graph and their representations are learned using a deep learning algorithm among GCN, GAT, GIN and GAT-GCN [15]. Protein embeddings are learned in a similar way to DeepDTA, using three convolutional layers. The representations for drugs and proteins are concatenated and fed to the fully connected layers as in our proposed model.

Yang et. al. use multiscale graph neural network and multiscale convolutional neural network to find the representations of drugs and targets respectively in MGraph-DTA [16]. They also propose a method to interpret the effect and importance of each neuron for the affinity prediction. We compare our model with GraphDTA and MGraphDTA in Section 5.2.

2.3. Datasets

In this section, we explain common datasets used in drug-target affinity prediction task. Davis dataset contains interaction of 72 kinase inhibitors with 442 kinases [17] and it is used in DeepDTA, SimBoost and SimCNN-DTA models [2,3,9]. Metz dataset consisting of 156 targets and 1421 drugs [18] is used to evaluate SimBoost model [2].

KIBA dataset is formed from multiple databases and used in DTA prediction methods to compare results with other datasets [19]. 229 targets and 2111 drugs are used from KIBA dataset in DeepDTA model [3]. He et al., Özçelik et al. and Shim et al. also utilize KIBA dataset for the evaluation of SimBoost, ChemBoost and SimCNN-

DTA models [2, 4, 9]. KIBA dataset includes information on the K_i, K_d and IC_{50} metrics that show the strenght of binding affinity. Low IC_{50} and K_i values indicate high binding affinity.

BindingDB dataset is also commonly used for the task of DTA prediction [6]. As of April 2023, BindingDB database contains 476365 compounds, 2484 targets and 993608 binding scores [20]. BindingDB dataset is used in ChemBoost and DebiasedDTA [4,21]. In our work, we form ligands' data from BindingDB in metapath2vec part and use the BindingDB dataset from DebiasedDTA in the prediction part.

3. BACKGROUND

Our task is drug-target affinity prediction. Target is molecular structure that interacts with chemicals which we call drugs [22]. In our case targets are the proteins. Ligand is a substance that binds to target and forms a complex. Finally, the affinity is a measure of how strong the binding is.

We represent proteins and ligands as 1D strings using their amino acid sequences and SMILES representations respectively. SMILES is a notation language for chemical compounds proposed in [23]. Even if it has some limitations such as not representing 3D structure of molecules, it is known for its simplicity and efficacity to be used in computational methods.

Each protein and ligand is matched with an unique id. UniprotIds are used for proteins [24] and PubchemCIDs are used to represent ligands [25]. Table 3.1 and Table 3.2 show examples from ligand and protein datasets respectively.

Table 3.1. A part of ligand sequences dataset.

PubchemCID	**SMILES**
10936212	CC(=O)c1ccc(OCc2ccc(CN3CCCCC3)cc2)cc1
9929127	COc1c(Cl)cc2c3ccncc3[nH]c2c1NC(=O)c1cccnc1C
10035301	O=c1cc(C2CCNCC2)s[nH]1

Table 3.2. A part of protein sequences dataset.

Uniprot ID	**Sequence**
P32305	MMDVNSSGRPDLYGHLRSLILPEVGRGLQDLSPDG GAHPVVSSWMPHLLSGFLEVTASPAPTWDAPPDNV SGCGEQINYGRVEKVVIGSILTLITLLTIAGNCLVV ISVCFVKKLRQPSNYLIVSLALADLSVAVAVMPFV SVTDLIGGKWIFGHFFCNVFIAMDVMCCTASIMTL CVISIDRYLGITRPLTYPVRQNGKCMAKMILSVWL LSASITLPPLFGWAQNVNDDKVCLISQDFGYTIYS TAVAFYIPMSVMLFMYYQIYKAARKSAAKHKFPGF PRVQPESVISLNGVVKLQKEVEECANLSRLLKHER KNISIFKREQKAATTLGIIVGAFTVCWLPFFLLST ARPFICGTSCSCIPLWVERTCLWLGYANSLINPFI YAFFNRDLRTTYRSLLQCQYRNINRKLSAAGMHEA LKLAERPERSEFVLQNSDHCGKKGHDT
P28074	MALASVLERPLPVNQRGFFGLGGRADLLDL GPGSLSDGLSLAAPGWGVPEEPGIEMLHGT TTLAFKFRHGVIVAADSRATAGAYIASQTV KKVIEINPYLLGTMAGGAADCSFWERLLAR QCRIYELRNKERISVAAASKLLANMVYQYK GMGLSMGTMICGWDKRGPGLYYVDSEGNRI SGATFSVGSGSVYAYGVMDRGYSYDLEVEQ AYDLARRAIYQATYRDAYSGGAVNLYHVRE DGWIRVSSDNVADLHEKYSGSTP
P0A6K3	MSVLQVLHIPDERLRKVAKPVEEVNAEIQRIVDDM FETMYAEEGIGLAATQVDIHQRIIVIDVSENRDER LVLINPELLEKSGETGIEEGCLSIPEQRALVPRAE KVKIRALDRDGKPFELEADGLLAICIQHEMDHLVG KLFMDYLSPLKQQRIRQKVEKLDRLKARA

4. MATERIALS AND METHODS

4.1. Identifying Molecular Words: Byte Pair Encoding

We use Byte Pair Encoding (BPE) introduced in [8] to tokenize 1D representations of proteins (amino acid sequences) and ligands (SMILES strings) into words. BPE is commonly used for natural languages and gives successful results. BPE relies on replacing most common bytes (2 characters) in a document with a character that does not occur in that document. The algorithm continues to check for most common bytes iteratively, until there is no more pairs left which appear more than once in the document. An example of tokenizing a protein sequence can be found in Appendix A. To identify protein and ligand words, we use tokenizers library from huggingface with default parameters [26] and obtain protein and ligand tokenizers. We use 1036474 SMILES representation from BindingDB [6] and 480651 amino acid sequence from Uniprot Swissprot [24] to learn the words. Table 4.1 shows train, validation and test splits of Uniprot Swissprot and BindingDB Pubchem datasets. As a result, we obtain 30000 protein words and 30000 ligand words. Examples of ligand and protein words are shown in Table 4.2 and Table 4.3 respectively.

Table 4.1. Train, validation and test splits in terms of number of data points.

	Train	**Validation**	**Test**
Uniprot Swissprot	432651	24000	24000
BindingDB	934474	51000	51000

4.2. Determining Node Representations: Metapath2vec

The graphs with single node type is called homogeneous graph and the graph with multiple node types is called heterogeneous graph. In our case, the input is a

heterogeneous graph with four node types being protein, ligand, protein word and ligand word. The number of unique nodes of each type can be found in Table 4.4. We obtain protein and ligand words using BPE algorithm as explained in Section 4.1. We define edges of the input graph using the following rules:

- A protein is connected to a ligand if there is interaction between them.
- A protein is connected to its words obtained tokenizing its amino acid sequence.
- A ligand is connected to its words obtained tokenizing its SMILES representation.

Table 4.5 shows the number of each edge type in the input graph derived applying the rules above. Each node has an id. For protein nodes we use their Uniprot IDs and for ligand nodes we use their corresponding PubchemCIDs. For each protein word, we define an unique id starting with "pw". Thus, for 30000 protein words we have ids from pw0 to pw29999. Similarly, for each ligand word we define ids from dw0 to dw29999. Samples from protein word and ligand word data are shown in Table 4.6 and Table 4.7 respectively. Protein sequences and ligands' SMILES representations are matched with their unique ids as explained in Section 3 and shown in Table 3.2 and Table 3.1. The interactions between ligands and proteins are identified by matching the ids of pairs that interact (see Table 4.8). Finally, we form ligand-ligand word and protein-protein word edges pairing their ids as in Table 4.9 and Table 4.10 respectively.

For protein and ligand nodes, we have two subcategories being warm and cold nodes. As pointed out in Figure 4.1:

- A cold ligand is a ligand whose we do not know other interactions but we know the other interactions of the protein it interacts (see ligand 4 in Figure 4.1a).
- Similarly, a cold protein is a protein whose we do not know other interactions but we know the other interactions of the ligand it interacts (see protein 4 in Figure 4.1b).
- Cold both is protein and ligand pair whose we do not know other interactions (see protein 4 and ligand 4 in Figure 4.1c)

- Warm is protein and ligand pair whose we know other interactions (see proteins 1, 2, 3 and ligands 1, 2, 3 in Figure 4.1c)

The split of cold and warm nodes in the dataset is shown in Table 4.11.

Table 4.2. Ligand with PubchemCID 49843538 tokenized into words.

49843538	NS(=O)(=O)c1cc(ccc1Cl)C(=O)CSc1nc2cc3OCCOc3cc2[nH]1
Words of 49843538	NS, (=, O,)(=, O,), c1cc, (, ccc1Cl,), C, (=, O,), CSc1n, c2cc3OCCOc3cc2, [, nH,], 1

Table 4.3. Protein with UniprotId P30810 tokenized into words.

P30810	LPSLSADAEAPSKIDTAYYNGTKTAPVY QYQFGVGIELTYVFKGSQYQDIESPTAYQSK
Words of P30810	L, P, SLS, ADAE, APS, KID, TAY, Y, NGT, KT, APVY, QY QFGVG, IEL, TY, VF, KGS, QY, QD, IES, PT, AYQS, K

Table 4.4. Total number of nodes for different node types.

Node Type	**Total Number of Nodes**
Ligand nodes	924298
Protein nodes	5892
Ligand word nodes	30000
Protein word nodes	30000

We calculate embeddings for each node of each node type using metapath2vec algorithm that executes random walks through a heterogeneous network and returns embeddings of each node using skip-gram model [10]. We identify metapaths as follows:

(i) (“ligandWord”, “ligandWord is in ligand”, “ligand”),

(ii) (“ligand”, “rev protein interacts with ligand”, “protein”),

(iii) (“protein”, “rev proteinWord is in protein”, “proteinWord”),

(iv) (“proteinWord”, “proteinWord is in protein”, “protein”),

(v) (“protein”, “protein interacts with ligand”, “ligand”),

(vi) (“ligand”, “rev ligandWord is in ligand”, “ligandWord”).

Here, “ligandWord”, “ligand”, “protein” and “proteinWord” are node types, “ligandWord is in ligand”, “proteinWord is in protein” and “protein interacts with ligand” are edge types and “rev protein interacts with ligand”, “rev proteinWord is in protein”, and “rev ligandWord is in ligand” are reverse edges.

To see the effect of protein and ligand words, we construct two graphs as input and train metapath2vec algorithm separately for each graph. The first graph contains only protein and ligand nodes, whereas the second graph contains protein, ligand, protein word and ligand word nodes. Figure 4.2 represents node and edge types for each input graph. Training losses for each graph are plotted in Appendix B Figure B.1, decreasing losses showing that the models are learning well during training.

Table 4.5. Number of edges.

Edge Type	**Number of Edges**
Protein-ligand edges	1376753
Protein-protein word edges	1152297
Ligand-ligand word edges	15020697

Table 4.6. A part of protein words dataset.

Protein Word ID	**Protein Word**
pw0	[UNK]
pw5	A
pw30	LL
pw403	VGG
pw1568	NNNNNNNN
pw16868	HGIQVERDKL

Table 4.7. A part of ligand words dataset.

Ligand Word ID	**Ligand Word**
dw0	[UNK]
dw7	(
dw64	cc
dw65	CC
dw87	N1
dw94	c1ccccc1
dw17302	NCCC1CN
dw17660	COc1ccccc1Oc1ccccc1CN1CCC2
dw20040	CCc1cccc2C

Table 4.8. A part of protein-ligand interaction dataset.

Uniprot ID	**PubchemCID**
P32305	17654
P32305	19219
P32305	4595
P28074	90655049
P08908	53310757

Table 4.9. A part of ligand-ligand words dataset.

Ligand Word ID	**PubchemCID**
dw65	10936212
dw67	10936212
dw529	9929127
dw87	44280298
dw94	57336561
dw11183	9929127

Table 4.10. A part of protein-protein words dataset.

Protein Word ID	**Uniprot ID**
pw3144	P32305
pw102	P32305
pw363	P28074
pw257	P08684
pw23826	P49662
pw3643	Q9NNX6
pw16687	O60346
pw197	P15539

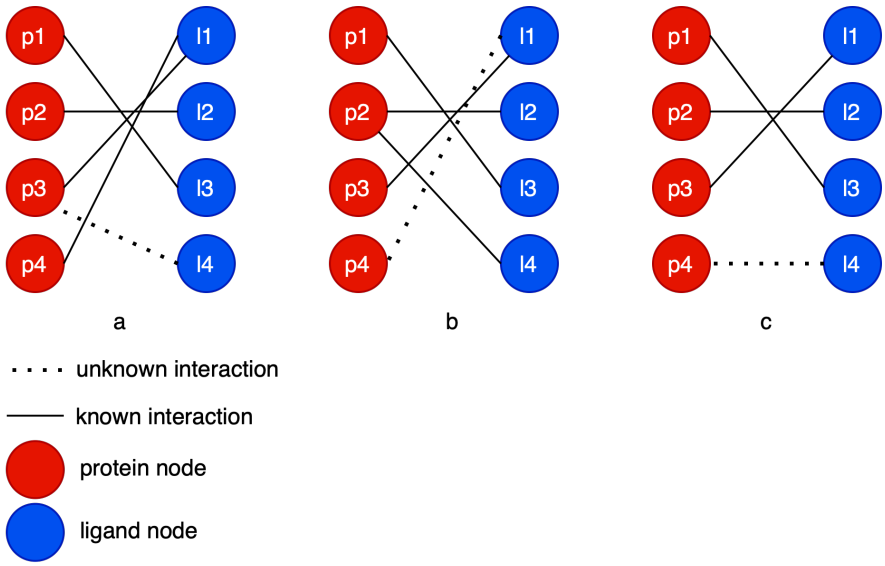

Figure 4.1. Example of (a) cold ligand (l4), (b) cold protein (p4), (c) cold both (l4 and p4) node.

Table 4.11. Split of cold and warm nodes.

Node Type	Number of Cold Nodes	Number of Warm Nodes
Ligand nodes	363	923935
Protein nodes	193	5699

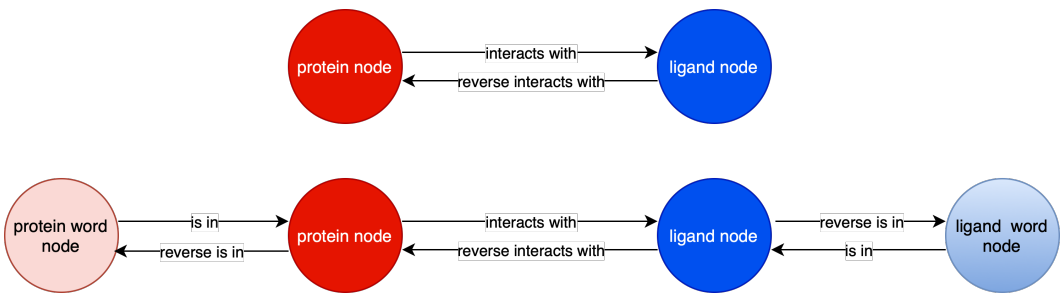

Figure 4.2. Relations between nodes that are used to define metapaths.

We fed the embeddings learned from the input graphs to the prediction model which is explained in detail in Section 4.3.

4.3. Proposed Method

Figure 4.3 represents the pipeline of the proposed model. To be able to get information about the effect and the importance of molecule words in the task of drug-target affinity prediction, we run the experiments for two different graphs, one without words and one with words. For each input graph, the usage of the metapath2vec algorithm changes as follows:

- Protein-Ligand Graph: Cold proteins and cold drugs are not included in the graph. Embeddings of warm nodes are calculated using metapath2vec algorithm. For cold nodes, BERT embeddings are used directly (seyonec/ PubChem10M_SMILES_BPE_450k[1] and Rostlab/prot_bert_embeddings[2] for cold ligands and cold proteins respectively [27, 28]).
- Protein-Ligand-Protein Words-Ligand Words Graph
  (i) All proteins and ligands in the dataset are included in the graph. Embeddings of warm and cold nodes are calculated using metapath2vec algorithm.
  (ii) All proteins and ligands in the dataset are included in the graph. Embeddings of warm nodes are calculated using metapath2vec algorithm. For the embeddings of cold nodes, average of their word embeddings are used. Word embeddings are also computed with the help of metapath2vec. For example, to calculate embedding of a cold protein, since we have already tokenized it into its words while forming the input graph, we average the embeddings of its words. We do the similar process also for the cold ligands.
  (iii) All proteins and ligands in the dataset are included in the graph. Using metapath2vec, protein, ligand, protein word and ligand word embeddings are calculated. To represent each protein and ligand, their average word embeddings are used in the prediction model.

Table 4.12 shows how the embeddings of each node type are calculated for each of the cases above, to pass to the prediction model. We concatenate protein and ligand

[1] https://huggingface.co/seyonec/PubChem10M_SMILES_BPE_450k
[2] https://huggingface.co/Rostlab/prot_bert

embeddings obtained from each case and fed the concatenated embeddings separately to the prediction part consisting of three fully connected layers with dropout. Visualization for different cases can be found in Figure 4.4.

Table 4.12. Embedding calculation for different node types to pass to the prediction model.

Graph Type	**Output of Metapath2-Vec**	**Cold Ligand Emb.**	**Cold Protein Emb.**	**Warm Ligand Emb.**	**Warm Protein Emb.**
Protein-Ligand	Ligand Emb. Protein Emb.	prot_bert emb.	PubChem10M emb.	metapath-2vec output	metapath-2vec output
Protein-Ligand-Words	Ligand Emb. Protein Emb. Ligand Word Emb. Protein Word Emb.	metapath-2vec output	metapath-2vec output	metapath-2vec output	metapath-2vec output
Protein-Ligand-Words	Ligand Emb. Protein Emb. Ligand Word Emb. Protein Word Emb.	average of word emb.	average of word emb.	metapath-2vec output	metapath-2vec output
Protein-Ligand-Words	Ligand Emb. Protein Emb. Ligand Word Emb. Protein Word Emb.	average of word emb.	average of word emb.	average of word emb.	average of word emb.

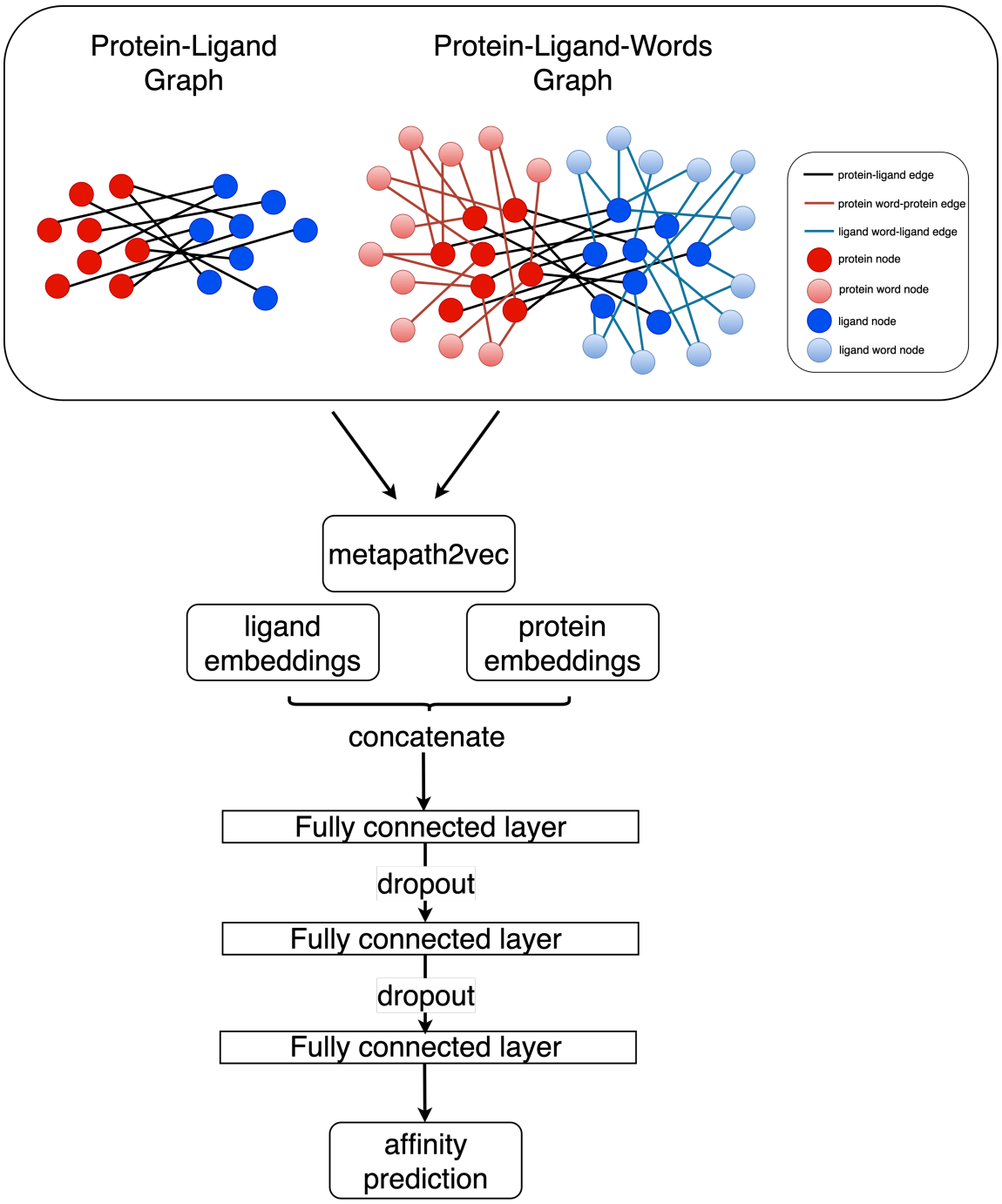

Figure 4.3. The pipeline of the proposed model.

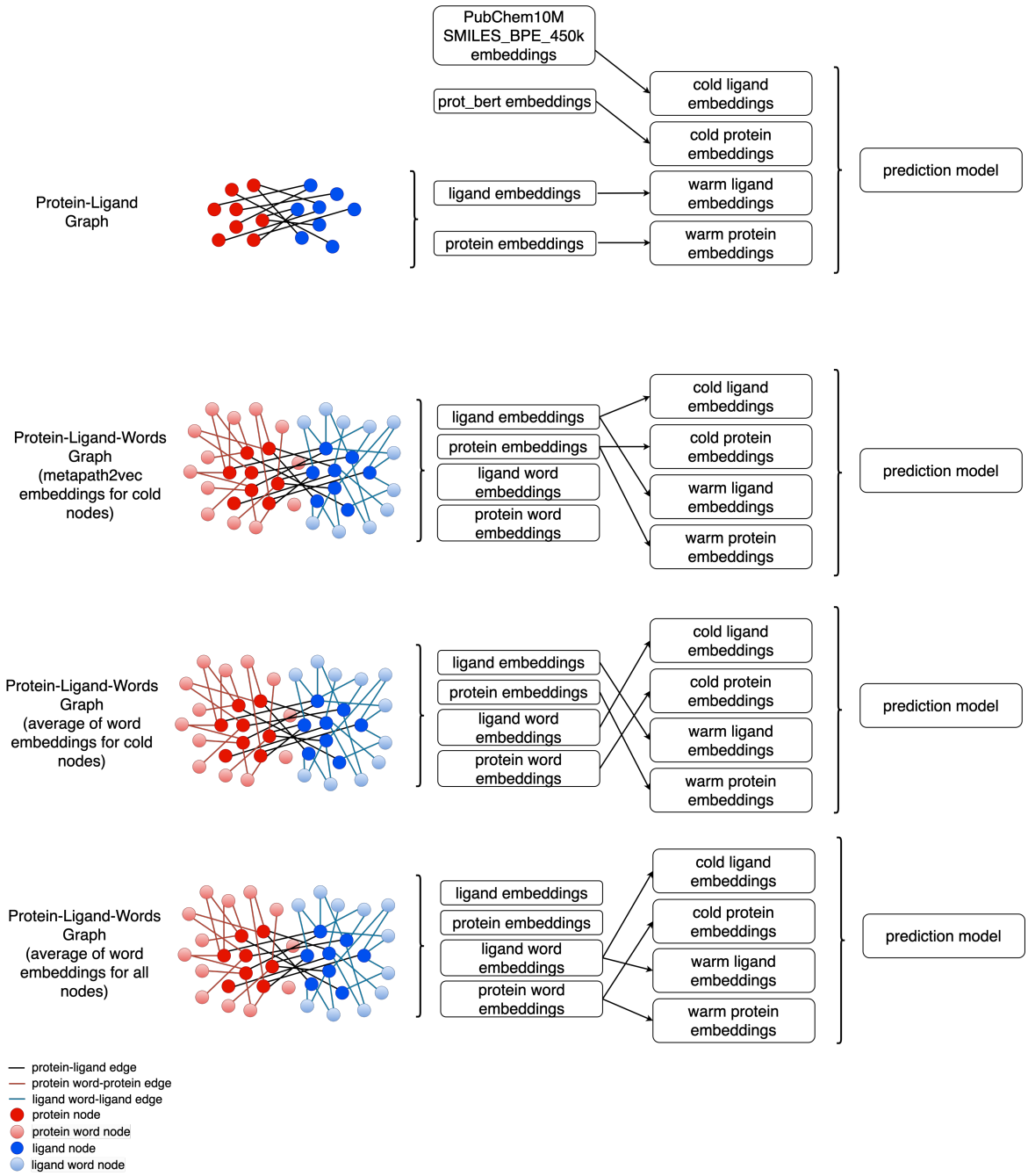

Figure 4.4. Embedding calculation for different node types.

4.4. Identifying The Most Important Words

We use TF-IDF to measure the importance of each word in a document (a protein and a ligand in our problem). Term Frequency-Inverse Document Frequency (TF-IDF) is calculated by multiplying TF and IDF that are defined as

$$\begin{aligned} TF-IDF &= TF * IDF \\ &= \frac{count\ of\ w\ in\ d}{num.\ of\ words\ in\ d} * \log\left(\frac{num.\ of\ docs\ in\ c}{count\ of\ docs\ in\ c\ with\ w}\right). \end{aligned} \quad (4.1)$$

Here, w corresponds to a word, d corresponds to a document, c is the corpus and *docs* are the documents.

We have two corpora being ligands and proteins. Each ligand is a document in ligands' corpus and similarly, each protein is a document in proteins' corpus. We calculate list of TF-IDF values separately for each corpus. Examples of TF-IDF scores for proteins and ligands are shown in Table 4.13 and Table 4.14 respectively.

Table 4.13. A part of TF-IDF values of protein corpus.

	pw0	**...**	**pw7**	**...**	**pw117**	**...**	**pw29998**	**pw29999**
P32305	0	...	0.01	...	0.01	...	0	0
P28074	0	...	0	...	0	...	0	0
...	...	...	...	...	...	...	...	...
Q9NRA2	0	...	0.01	...	0.01	...	0	0
P51144	0	...	0.01	...	0	...	0	0

Table 4.14. A part of TF-IDF values of ligand corpus.

	dw0	**...**	**dw38**	**...**	**dw65**	**dw66**	**...**	**dw29999**
10936212	0	...	0.01	...	0.11	0	...	0
9929127	0	...	0.01	...	0	0	...	0
...	...	...	...	...	...	...	...	...
129140121	0	...	0.01	...	0.05	0	...	0
129140314	0	...	0	...	0.04	0	...	0

To filter the words that are most important according to TF-IDF score, we get average of TF-IDF values corresponding to each ligand and each protein without counting zero values. Then, we eliminate the words that have TF-IDF scores smaller than

the calculated average value corresponding to that document (protein or ligand). For instance, the steps of filtering for the ligand with PubchemCID 10936212 is as follows:

- Step 1: The SMILES string "CC(=O)c1ccc(OCc2ccc(CN3CCCCC3)cc2)cc1" is tokenized into its words: "(", ")", "O", "CC", "(=", "c1ccc", "cc1", "cc2", "OCc2ccc", "CN3CCCCC3"
- Step 2: After running TF-IDF algorithm using the whole corpus, the scores for each word of the ligand with PubchemCID 10936212 is calculated as: "(": 0.0065993269, ")": 0.0019136415, "O": 0.0125090249, "CC": 0.1133916242, "(=": 0.0273307470, "c1ccc": 0.1735102837, "cc1": 0.1289624629, "cc2": 0.1726810058, "OCc2ccc": 0.4901238218, "CN3CCCCC3": 0.5772570610
- Step 3: We take the average of the scores which is 0.17042789997.
- Step 4: We continue with the words whose TF-IDF scores are greater than the average. So the filtered words are "c1ccc", "cc2", "OCc2ccc" and "CN3CCCCC3".

Before filtering words using their TF-IDF scores, there were 1152297 protein-protein word edges and 15020697 ligand-ligand word edges (see Table 4.5). Table 4.15 shows the number of edges for each edge type after filtering words.

Table 4.15. Number of edges after filtering words according to TF-IDF scores.

Edge Type	**Number of Edges**
Protein-ligand edges	1376753
Protein-protein word edges	507293
Ligand-ligand word edges	6036063

4.5. Experimental Setup

We run the experiments for each case explained in Section 4.3 with hyperparameters stated in Table 4.16 and Table 4.17 for the metapath2vec part and the prediction model respectively. All of the metapath2vec and prediction models are trained on a machine with 10 core 384GB memory and 1 Nvidia V100 16GB GPU card.

Table 4.16. Metapath2vec model parameters.

Parameter	**Value**
Embedding dimension	128
Walk length	10
Context size	5
Walks per node	10
Negative samples	5
Sparse	True
Batch size	32
Optimizer	SparseAdam

Table 4.17. Parameters of the prediction model.

Parameter	**Value**
Maximum SMILES length	100
Maximum protein length	1000
Sequence embedding dimension	32
Pretrained embedding dimension	32
Learning rate	0.001
Batch size	256
Epochs	200
Number of filters	32
SMILES filter length	4
Protein filter length	6
Optimizer	Adam

4.6. Evaluation Metrics

CI, r^2, MSE and RMSE are used as evaluation metrics. The higher CI and r^2, the better it is. On the contrary, the lower MSE and RMSE, the better it is.

The concordance index (CI) ranges between 0 and 1 and it is defined as

$$CI = \frac{1}{Z} \sum_{\delta_i > \delta_j} h(b_i - b_j). \tag{4.2}$$

In this expression, Z is the normalization constant, b_i and b_j are predicted affinities, and $h(x)$ is the step function.

The mean squared error (MSE) shows how much the predicted value differs from the ground truth and it is defined as

$$MSE = \frac{1}{N} \sum_{i=1}^{N} (b_i - b_i')^2. \tag{4.3}$$

In this expression, $\mathbf{N}$ is the number of data points, $\mathbf{b}_i$ is the ground truth and $\mathbf{b}_i'$ and is the predicted value.

The root mean squared error (RMSE) is expressed as the square root of MSE

$$RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (b_i - b_i')^2}. \tag{4.4}$$

Similar to MSE, N is the number of data points, b_i is the ground truth and b_i' and is the predicted value.

The coefficient of determination (r^2) showing the goodness of fit, is calculated as

$$\begin{aligned} r^2 &= 1 - \frac{RSS}{TSS} \\ &= 1 - \frac{\sum_{i=1}^{N} (b_i - b_i')^2}{\sum_{i=1}^{N} (b_i - \bar{b})^2}. \end{aligned} \tag{4.5}$$

Here, N is the number of data points, b_i is the ground truth, b_i' is the predicted value and $\bar{b}$ is the mean value of the sample. The highest possible r^2 score is 1, meaning that all of the predicted values match with ground truth. The model that predicts mean of the observed values has r^2 equal to 0, which is the baseline model. Negative

r^2 means that the model has worst predictions than the mean of the observed values, the baseline model. Thus, in our experiments, it is expected to see lower r^2 values in cold cases compared to warm interactions since we know less on cold nodes as shown in Section 5.

5. RESULTS

5.1. Experiment Results

Experiments are conducted for warm, cold protein, cold ligand and cold both interaction types with the input graphs explained in Section 4.3. Tables representing the experiment results are in the same form showing if the word nodes are included in the setup, if the words are filtered using TF-IDF, how the cold and warm node embeddings are calculated and the evaluations for each case (Table 5.1, Table 5.2, Table 5.3 and Table 5.4). The rows of the tables correspond to Model 1, Model 2, Model 3, Model 4, Model 5 and Model 6 respectively where Model 1 uses protein and ligand nodes to form the input graph, Models 2, 3, 4, 5 and 6 use protein, ligand, protein word and ligand word nodes to create the input graph shown in Figure 4.4. The best result of each column is emphasized in bold and standard deviations are shown in parentheses. Comparing Model 1 with Models 2 and 3, inclusion of the words yielded to a small amount of increase in CI score for warm nodes, but the results do not differ significantly as it can be seen in Table 5.1. Filtering words using TF-IDF improved the results. Model 6 has highest CI score, but overall, the best result for warm setup is achieved in Model 4.

Table 5.1. Effect of using word information on warm nodes.

	Words	**TF-IDF**	**Cold Node Emb.**	**Warm Node Emb.**	**ci**	**r2**	**rmse**	**mse**
Model 1	-	-	BERT Emb.	meta-path-2vec	0.884 (0.007)	0.770 (0.016)	0.550 (0.019)	0.303 (0.021)
Model 2	+	-	meta-path-2vec	meta-path-2vec	0.885 (0.005)	0.769 (0.022)	0.552 (0.032)	0.306 (0.036)
Model 3	+	-	avg. of word emb.	meta-path-2vec	0.886 (0.009)	0.772 (0.020)	0.548 (0.027)	0.301 (0.031)
Model 4	+	+	meta-path-2vec	meta-path-2vec	0.889 (0.007)	**0.776** (0.013)	**0.543** (0.018)	**0.296** (0.020)
Model 5	+	+	avg. of word emb.	meta-path-2vec	0.884 (0.006)	0.774 (0.018)	0.546 (0.030)	0.299 (0.034)
Model 6	+	+	avg. of word emb.	avg. of word emb.	**0.891** (0.006)	0.772 (0.019)	0.548 (0.022)	0.301 (0.025)

For cold ligands, best performance is achieved when average of word embeddings are used to represent cold nodes (Models 5 and 6 in Table 5.2). Model 5 has the best CI score, whereas Model 6 has the best r^2, MSE and RMSE. Compared to the protein-ligand graph usage of word embeddings performs better than pretrained embeddings

for cold nodes (see Table 5.2).

Table 5.2. Effect of using word information on cold ligands.

	Words	**TF-IDF**	**Cold Node Emb.**	**Warm Node Emb.**	**ci**	**r2**	**rmse**	**mse**
Model 1	-	-	BERT Emb.	metapath-2vec	0.665 (0.098)	-0.073 (0.308)	1.217 (0.386)	1.629 (1.085)
Model 2	+	-	metapath-2vec	metapath-2vec	0.675 (0.044)	-0.010 (0.147)	1.164 (0.217)	1.403 (0.539)
Model 3	+	-	avg. of word emb.	metapath-2vec	0.690 (0.065)	0.030 (0.242)	1.146 (0.301)	1.403 (0.783)
Model 4	+	+	metapath-2vec	metapath-2vec	0.648 (0.103)	-0.168 (0.330)	1.258 (0.362)	1.713 (1.072)
Model 5	+	+	avg. of word emb.	metapath-2vec	**0.705** (0.041)	-0.025 (0.164)	1.159 (0.119)	1.356 (0.277)
Model 6	+	+	avg. of word emb.	avg. of word emb.	0.694 (0.069)	**0.068** (0.154)	**1.125** (0.257)	**1.331** (0.643)

For cold proteins, using embeddings learned from metapath2vec and using average of word embeddings for cold nodes give similar results as seen in Table 5.3. In both cases, the word information is used in some way, directly or undirectly. When we

use metapath2vec embeddings for cold nodes, we only use the connection between cold nodes and its words. The embeddings of cold nodes are calculated during metapath2vec part using their edges to words. When we average word embeddings, we directly use the word embeddings learned from metapath2vec. Comparing Model 3 with Models 4 and 5, we see that applying TF-IDF decreases errors and increases r^2 and there is no significant decrease in CI score.

Table 5.3. Effect of using word information on cold proteins.

	Words	**TF-IDF**	**Cold Node Emb.**	**Warm Node Emb.**	**ci**	**r2**	**rmse**	**mse**
Model 1	-	-	BERT Emb.	meta-path-2vec	0.769 (0.007)	0.299 (0.082)	1.050 (0.079)	1.108 (0.168)
Model 2	+	-	meta-path-2vec	meta-path-2vec	0.773 (0.016)	0.330 (0.074)	1.027 (0.079)	1.061 (0.165)
Model 3	+	-	avg. of word emb.	meta-path-2vec	**0.776** (0.008)	0.325 (0.075)	1.030 (0.067)	1.065 (0.141)
Model 4	+	+	meta-path-2vec	meta-path-2vec	0.772 (0.012)	**0.342** (0.063)	1.017 (0.063)	1.039 (0.128)
Model 5	+	+	avg. of word emb.	meta-path-2vec	0.775 (0.010)	0.326 (0.057)	**1.003** (0.065)	**1.009** (0.133)
Model 6	+	+	avg. of word emb.	avg. of word emb.	0.771 (0.013)	0.338 (0.077)	1.020 (0.080)	1.047 (0.164)

The most significant improvement is in the case of cold protein and cold ligand interactions. For CI score we observe 2.2% points increase comparing protein-ligand-words graph that uses average of word embeddings to protein-ligand graph in the case where the words are not filtered (see Models 1 and 3 in Table 5.4). In cold both case, using average of word embeddings is much more effective than using metapath2vec

embeddings for cold nodes as Table 5.4 shows. The best results are achieved in the case of using the average of filtered words for cold nodes. We observe 4.5% points increase in CI score when we add word information for cold nodes compared to the case where the words are not used at all (see Models 1 and 5). When we look at the results of Models 3 and 5, we see that the CI score increases 2.3% points. If average of word embeddings is also used, we observe a little improvement on r^2 in Model 6 compared to Model 5.

Table 5.4. Effect of using word information on cold ligands and cold proteins (cold both).

	Words	**TF-IDF**	**Cold Node Emb.**	**Warm Node Emb.**	**ci**	**r2**	**rmse**	**mse**
Model 1	-	-	BERT Emb.	meta-path-2vec	0.551 (0.057)	-0.234 (0.199)	1.404 (0.439)	2.165 (1.360)
Model 2	+	-	meta-path-2vec	meta-path-2vec	0.545 (0.045)	-0.205 (0.144)	1.358 (0.283)	1.924 (0.792)
Model 3	+	-	avg. of word emb.	meta-path-2vec	0.573 (0.065)	-0.134 (0.124)	1.329 (0.337)	1.879 (0.982)
Model 4	+	+	meta-path-2vec	meta-path-2vec	0.544 (0.039)	-0.198 (0.173)	1.364 (0.357)	1.987 (1.093)
Model 5	+	+	avg. of word emb.	meta-path-2vec	**0.596** (0.032)	-0.128 (0.203)	**1.305** (0.246)	**1.763** (0.623)
Model 6	+	+	avg. of word emb.	avg. of word emb.	0.575 (0.024)	**-0.103** (0.120)	1.311 (0.324)	1.824 (0.905)

5.2. Comparison with Other Methods

Overall, filtering words according to their TF-IDF scores improves the results. So, we compare our models where the filtered words are used (Models 4, 5 and 6), with recent methods proposed for the task of drug-target affinity prediction, DeepDTA [3], BPE-DTA and LM-DTA from DebiasedDTA [21]. We also compare our results with graph-based methods GraphDTA [15] and MGraphDTA [16]. The results of the compared methods are taken from DebiasedDTA paper and the results with the guide none are used meaning that there is no debiasing [21]. Only CI and r^2 scores are noted in the paper, so we do comparison with two scores.

For warm interactions, our Model 6 outperforms other methods in CI score and DeepDTA has the best r^2 score. The of DeepDTA is only 0.5% higher than Model 4, so Model 4 or Model 6 can be used to predict interactions of warm nodes (see Table 5.5).

Table 5.5. Comparison of recent methods with proposed model for warm interactions.

	Words	**TF-IDF**	**Cold Node Emb.**	**Warm Node Emb.**	**ci**	**r2**
DeepDTA	-	-	-	-	0.888 (0.009)	**0.781** (0.028)
BPE-DTA	-	-	-	-	0.883 (0.006)	0.774 (0.013)
LM-DTA	-	-	-	-	0.876 (0.005)	0.745 (0.011)
GraphDTA	-	-	-	-	0.824 (0.010)	0.493 (0.060)
MGraphDTA	-	-	-	-	0.834 (0.013)	0.520 (0.067)
Model 4	+	+	metapath-2vec	metapath-2vec	0.889 (0.007)	0.776 (0.013)
Model 5	+	+	average of word emb.	metapath-2vec	0.884 (0.006)	0.774 (0.018)
Model 6	+	+	average of word emb.	average of word emb.	**0.891** (0.006)	0.772 (0.019)

As shown in Table 5.6 Model 5 has highest CI score. GraphDTA model is also successful method for cold ligands with highest r^2 and CI score close to our Model 5.

Table 5.6. Comparison of recent methods with proposed model for cold ligand interactions.

	Words	**TF-IDF**	**Cold Node Emb.**	**Warm Node Emb.**	**ci**	**r2**
DeepDTA	-	-	-	-	0.687 (0.096)	0.039 (0.243)
BPE-DTA	-	-	-	-	0.657 (0.083)	-0.143 (0.202)
LM-DTA	-	-	-	-	0.688 (0.046)	-0.027 (0.175)
GraphDTA	-	-	-	-	0.701 (0.024)	**0.143** (0.138)
MGraphDTA	-	-	-	-	0.684 (0.030)	-0.125 (0.134)
Model 4	+	+	metapath-2vec	metapath-2vec	0.648 (0.103)	-0.168 (0.330)
Model 5	+	+	average of word emb.	metapath-2vec	**0.705** (0.041)	-0.025 (0.164)
Model 6	+	+	average of word emb.	average of word emb.	0.694 (0.069)	0.068 (0.154)

LM-DTA performs the best for the case of cold proteins. We achieve close results to LM-DTA with Models 4, 5 and 6. Our proposed models also outperform graph-based methods, GraphDTA and MGraphDTA.

Table 5.7. Comparison of recent methods with proposed model for cold protein interactions.

	Words	**TF-IDF**	**Cold Node Emb.**	**Warm Node Emb.**	**ci**	**r2**
DeepDTA	-	-	-	-	0.759 (0.006)	0.315 (0.049)
BPE-DTA	-	-	-	-	0.653 (0.060)	-0.256 (0.411)
LM-DTA	-	-	-	-	**0.780** (0.016)	**0.384** (0.083)
GraphDTA	-	-	-	-	0.685 (0.039)	0.040 (0.114)
MGraphDTA	-	-	-	-	0.754 (0.013)	0.289 (0.070)
Model 4	+	+	metapath-2vec	metapath-2vec	0.772 (0.012)	0.342 (0.063)
Model 5	+	+	average of word emb.	metapath-2vec	0.775 (0.010)	0.326 (0.057)
Model 6	+	+	average of word emb.	average of word emb.	0.771 (0.013)	0.338 (0.077)

In cold both case, our Models 5 and 6 outperform other methods in terms of CI score with the effect of cold nodes' word embeddings as seen in Table 5.8. GraphDTA outperforms other models with r^2 equals to -0.047, followed by Model 6 and Model 5.

Table 5.8. Comparison of recent methods with proposed model for interactions between cold protein and cold ligand nodes (cold both).

	Words	**TF-IDF**	**Cold Node Emb.**	**Warm Node Emb.**	**ci**	**r2**
DeepDTA	-	-	-	-	0.554 (0.047)	-0.154 (0.164)
BPE-DTA	-	-	-	-	0.522 (0.054)	-0.442 (0.349)
LM-DTA	-	-	-	-	0.572 (0.028)	-0.226 (0.205)
GraphDTA	-	-	-	-	0.558 (0.077)	**-0.047** (0.162)
MGraphDTA	-	-	-	-	0.555 (0.059)	-0.448 (0.497)
Model 4	+	+	metapath-2vec	metapath-2vec	0.544 (0.039)	-0.198 (0.173)
Model 5	+	+	average of word emb.	metapath-2vec	**0.596** (0.032)	-0.128 (0.203)
Model 6	+	+	average of word emb.	average of word emb.	0.575 (0.024)	-0.103 (0.120)

6. DISCUSSION

6.1. Evaluation of the Results

Comparing Model 1 with Models 4 and 6, adding words to the graph improved the results for warm interactions by a small amount. The increase is expected to be small since the warm nodes are already in the graph even if the words are not used. In Model 1 we have information on the connection of warm nodes, so we can learn the embeddings of the warm nodes nearly as good as Models 4 and 6, using that information.

For cold ligands, including words in the input graph and especially using average of the word embeddings to represent cold nodes improved the results significantly. We can do the similar comment also for the cold both case. The improvement in cold protein is less than the improvement in cold ligands and cold both, however, we can say that word information can also be used for cold proteins. For cold cases, the results are consistent with expectations. Cold ligands are not connected to proteins, we only have the sequence and the word information of the cold ligands. The information on the cold ligands' interaction with proteins are learned via ligands' word embeddings through the graph. Cold protein and cold both interactions are also learned in a similar way using molecule words.

In the experiment with no words, cold ligand and cold proteins are represented with pretrained embeddings [27,28]. In our models where we calculated the embeddings of cold nodes taking the average of their word embeddings learned using metapath2vec, we outperformed the model with no words. Thus, we can say that using the average of word embeddings performs better than pretrained models. Moreover, the complexity of metapath2vec algorithm that we have used to learn the word embeddings is much less than the complexity to form prot_bert and PubChem10M_SMILES_BPE_450k models.

6.2. Analysis of Identified Words

We compared the predictions done using protein-ligand graph as input (our Model 1) and the predictions obtained using the average of warm and cold compounds' word embeddings from protein-ligand-words graph (our Model 6). After identifying the protein-ligand pairs in the test set that performed well in the case of using average word embeddings and not that well in the case of the graph without words, we investigated if some of these words have a chemical meaning.

Looking at the molecules that bind to Beta-2 adrenergic receptor (ADRB2), our algorithm extracted a naphthalene backbone-bearing fragment. We identified 1-naphthyloxymethyl fragment corresponding to chemical word "COc1cccc2ccccc12", shown in green in Figure 6.1, learned using BPE algorithm. ADRB2 is a cell membrane-spanning receptor that is targeted to treat bronchial spasm in patients with bronchial asthma and chronic obstructive pulmonary disease (COPD) [29]. The first generation of beta blocker drugs, particularly propranolol in Figure 6.1, having naphthalene moiety inhibit beta-adrenergic receptors (beta-1 and beta-2) in a non-selective manner and new generations of selective beta-2 blockers developed to reduce side effects [30]. However, we can say that our algorithm yields to promising results extracting 1-naphthyloxymethyl fragment, because the oxygen atom in the fragment binds the receptor through an H-bond and also the extracted fragment is an important portion of the chemical structure of first generation beta blockers and Class 2 Antiarrhythmic Drugs [31].

Carbonic anhydrases (CAs) take part in ion transport and the overall acid-base balance in various body organs. Increase in their activity causes many diseases such as glaucoma, cancer etc. A common approach to treat these disorders is usage of CA inhibitors. BPE algorithm found benzimidazole-bearing fragments in their parent molecules. The algorithm was able to identify the word "CCCc1nc2ccccc2n1CC" in 2-chloro-4-(2-(2-propyl-1H-benzo[d]imidazol-1-yl)acetyl)benzenesulfonamide as shown in Figure 6.2a. Another benzimidazole-bearing fragment extracted by BPE algorithm

is “CCCCc1nc2ccccc2n1CC” from compound 4-(2-(2-butyl-1H-benzo[d]imidazol-1-yl) acetyl)-2-chlorobenzenesulfonamide represented in Figure 6.2b. Finally, BPE algorithm successfully found the word “CSc1nc2ccccc2n1CC” in 4-(2-(2-(methylthio)-1H-benzo[d]imidazol-1-yl)acetyl) benzenesulfonamide (see Figure 6.2c). Zubrienė et al. stated a group of benzimidazole-containing CA inhibitors which has augmented binding affinity to CA enzymes [32]. Obtaining 1-substituted-1H-benzimidazole fragment from BPE algorithm is meaningful since in the literature, it is noted that the benzimidazole core forms an H-bond complex with some CA isoforms [33].

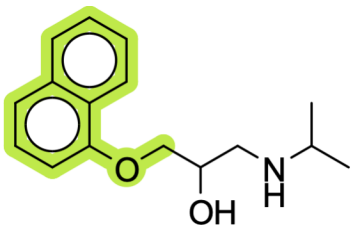

Figure 6.1. 2D structure of propranolol, highlighted part in green representing the 1-naphthyloxymethyl fragment corresponding to the chemical word “COc1cccc2ccccc12” identified by BPE algorithm.

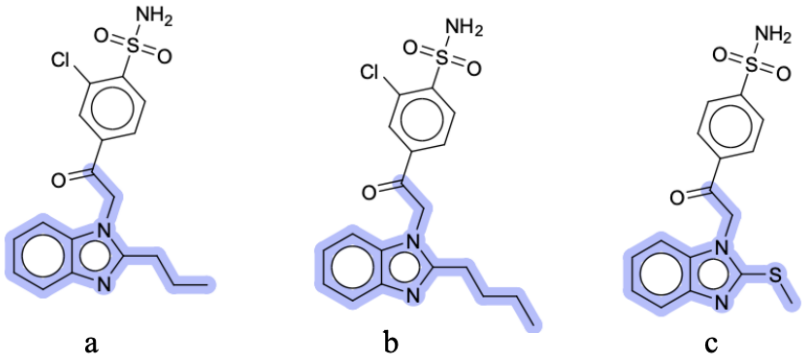

Figure 6.2. 2D structure of the compound with (a) PubchemCID 78319193, (b) PubchemCID 78319194 and (c) PubchemCID 129627000 where the chemical word identified by BPE is highlighted in purple on the molecule.

7. CONCLUSION

In this work we examined the effect of protein and ligand words in drug-target affinity prediction. We represented the data in the form of heterogeneous graph and learned the representations of each node in the graph using metapath2vec algorithm. We compared results of graphs with and without molecule words. We have seen that including word information improves the results. Next, to identify the words that have the most effect on drug-target affinities, we used TF-IDF method and filtered the words of each compound according to their importance score. We proposed using the average of their word embeddings to represent ligands and proteins and we showed the compounds can be represented using their words. Compared to the case where the words are not used at all, we observed a significant improvement especially for cold nodes. To investigate the affinity of unknown protein and drug, we propose first tokenizing them into their words and then averaging the embeddings of the words that we already know. Since we do not re-train the metapath2vec algorithm the proposed model has low complexity and is very effective in terms of time constraints which is the main problem in wet-lab experiments. In addition, we obtained comparable results with the state-of-the-art graph-based methods [15, 16].

8. FUTURE WORK

For the part of word identification, different tokenization methods such as counting k-mers, N-gram model or tokenizing according to punctuations for ligands can be used as a simple baseline to see the effect of BPE algorithm. To filter words learned using BPE, a better threshold than the average of the words' TF-IDF values can be found. One approach can be analysing the words of proteins and ligands in the training set of the prediction part. The words that are not in the binding site can be eliminated.

In Section 6.2, we look at the words in the cold both case where the predictions done using word information (Model 6) is better than the predictions done without words (Model 1). There are 178 different proteins in the cold both dataset that satisfy these conditions. There are both single domain and multi domain proteins among these 178 proteins. Thus, in the future bigger test datasets can be formed separately for single domain and multi domain proteins to see how much the word information is effective on different protein types for cold both case.

As future work, different graph learning methods other than metapath2vec can be used to learn node embeddings. Additional information can be added to the protein-ligand-words input graph such as disease, side-effect information. We can add also attention layers to the prediction part to be able to predict affinities considering important features.

In the future, the proposed method can be tested for different tasks, such as classifying similar proteins and ligands. Word information of proteins and ligands can be useful also to identify similarity between proteins and ligands.

REFERENCES

1. Pahikkala, T., A. Airola, S. Pietilä, S. Shakyawar, A. Szwajda, J. Tang and T. Aittokallio, "Toward More Realistic Drug-Target Interaction Predictions", *Briefings in Bioinformatics*, Vol. 16, No. 2, pp. 325–337, 2015.

2. He, T., M. Heidemeyer, F. Ban, A. Cherkasov and M. Ester, "SimBoost: A Read-Across Approach for Predicting Drug–Target Binding Affinities Using Gradient Boosting Machines", *Journal of Cheminformatics*, Vol. 9, No. 24, pp. 1–14, 2017.

3. Öztürk, H., E. Ozkirimli and A. Ozgur, "DeepDTA: Deep Drug-Target Binding Affinity Prediction", *Bioinformatics*, Vol. 34, No. 17, pp. i821–i829, 2018.

4. Özçelik, R., H. Öztürk, A. Ozgur and E. Ozkirimli, "ChemBoost: A Chemical Language Based Approach for Protein–Ligand Binding Affinity Prediction", *Molecular Informatics*, Vol. 40, No. 5, pp. 1–13, 2020.

5. Chen, T. and C. Guestrin, "XGBoost", *22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco California, USA, pp. 785–794, 2016.

6. Gilson, M. K., T. Liu, M. Baitaluk, G. Nicola, L. Hwang and J. Chong, "BindingDB in 2015: A Public Database for Medicinal Chemistry, Computational Chemistry and Systems Pharmacology", *Nucleic Acids Research*, Vol. 44, No. 1, pp. 1045–1053, 2016.

7. Öztürk, H., E. Ozkirimli and A. Ozgur, "A Novel Methodology on Distributed Representations of Proteins Using Their Interacting Ligands", *Bioinformatics*, Vol. 34, No. 13, pp. i295–i303, 2018.

8. Gage, P., "A New Algorithm for Data Compression", *The C Users Journal*, Vol. 12, No. 2, pp. 23–38, 1994.

9. Shim, J., Z.-Y. Hong, I. Sohn and C. Hwang, "Prediction of Drug–Target Binding Affinity Using Similarity-Based Convolutional Neural Network", *Scientific Reports*, Vol. 11, No. 4416, pp. 1–9, 2021.

10. Dong, Y., N. V. Chawla and A. Swami, "metapath2vec: Scalable Representation Learning for Heterogeneous Networks", *23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, New York, NY, USA, pp. 135–144, 2017.

11. Mikolov, T., K. Chen, G. Corrado and J. Dean, "Efficient Estimation of Word Representations in Vector Space", *ICLR Workshop*, edited by Bengio Y. and Y. LeCun, Scottsdale, Arizona, pp.1–12, 2013.

12. Zhang, C., D. Song, C. Huang, A. Swami and N. V. Chawla, "Heterogeneous Graph Neural Network", *25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Anchorage AK, USA, pp. 793–803, 2019.

13. Wang, X., H. Ji, C. Shi, B. Wang, Y. Ye, P. Cui and P. Yu, "Heterogeneous Graph Attention Network", *WWW '19: The World Wide Web Conference*, San Francisco CA, USA, pp. 2022–2032, 2019.

14. Li, M., X. Cai, S. Xu and H. Ji, "Metapath-Aggregated Heterogeneous Graph Neural Network for Drug–Target Interaction Prediction", *Briefings in Bioinformatics*, Vol. 24, No. 1, pp. 1–17, 2023.

15. Nguyen, T., H. Le, T. P. Quinn, T. Nguyen, T. D. Le and S. Venkatesh, "GraphDTA: Predicting Drug–Target Binding Affinity with Graph Neural Networks", *Bioinformatics*, Vol. 37, No. 8, pp. 1140–1147, 2021.

16. Yang, Z., W. Zhong, L. Zhao and C. Yu-Chian Chen, "MGraphDTA: Deep Multiscale Graph Neural Network for Explainable Drug-Target Binding Affinity Prediction", *Chemical Science*, Vol. 13, No. 3, pp. 816–833, 2022.

17. Davis, M. I., J. P. Hunt, S. Herrgard, P. Ciceri, L. M. Wodicka, G. Pallares, M. Hocker, D. K. Treiber and P. P. Zarrinkar, "Comprehensive Analysis of Kinase Inhibitor Selectivity", *Nature Biotechnology*, Vol. 29, No. 11, pp. 1046–1051, 2011.

18. Metz, J. T., E. F. Johnson, N. B. Soni, P. J. Merta, L. Kifle and P. J. Hajduk, "Navigating the Kinome", *Nature Chemical Biology*, Vol. 7, No. 4, pp. 200–202, 2011.

19. Tang, J., A. Szwajda, S. Shakyawar, T. Xu, P. Hintsanen, K. Wennerberg and T. Aittokallio, "Making Sense of Large-Scale Kinase Inhibitor Bioactivity Data Sets: A Comparative and Integrative Analysis", *Journal of Chemical Information and Modeling*, Vol. 54, No. 3, pp. 735–743, 2014.

20. Gilson, M. K., T. Liu, M. Baitaluk, G. Nicola, L. Hwang and J. Chong, "BindingDB", 2015, `https://www.bindingdb.org`, accessed on February 27, 2022.

21. Özçelik, R., A. Bağ, B. Atıl, M. Barsbey, A. Özgür and E. Özkırımlı, "Debiased-DTA: A Framework for Improving the Generalizability of Drug-Target Affinity Prediction Models", ArXiv:2107.05556, 2023.

22. Imming, P., C. Sinning and A. Meyer, "Drugs, Their Targets and The Nature and Number of Drug Targets", *Nature Reviews Drug Discovery*, Vol. 5, No. 10, pp. 821–834, 2006.

23. Weininger, D., "SMILES, A Chemical Language and Information System. 1. Introduction to Methodology and Encoding Rules", *Journal of Chemical Information and Computer Sciences*, Vol. 28, No. 1, pp. 31–36, 1988.

24. The UniProt Consortium, "UniProt: The Universal Protein Knowledgebase", *Nucleic Acids Research*, Vol. 45, No. 1, pp. 158–169, 2017.

25. Kim, S., P. A. Thiessen, E. E. Bolton, J. Chen, G. Fu, A. Gindulyte, L. Han, J. He, S. He, B. A. Shoemaker, J. Wang, B. Yu, J. Zhang and S. H. Bryant, "PubChem

Substance and Compound Databases", *Nucleic Acids Research*, Vol. 44, No. 1, pp. 202–213, 2016.

26. Patry, N., "huggingface/tokenizers", 2019, `https://github.com/huggingface/tokenizers`, accessed on January 15, 2023.

27. Chithrananda, S., G. Grand and B. Ramsundar, "ChemBERTa: Large-Scale Self-Supervised Pretraining for Molecular Property Prediction", ArXiv:2010.09885, 2020.

28. Elnaggar, A., M. Heinzinger, C. Dallago, G. Rehawi, Y. Wang, L. Jones, T. Gibbs, T. Feher, C. Angerer, M. Steinegger, D. Bhowmik and B. Rost, "ProtTrans: Toward Understanding the Language of Life Through Self-Supervised Learning", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 44, No. 10, pp. 7112–7127, 2022.

29. Stiles, G. L., M. G. Caron and R. J. Lefkowitz, "Beta-Adrenergic Receptors: Biochemical Mechanisms of Physiological Regulation", *Physiological Reviews*, Vol. 64, No. 2, pp. 661–743, 1984.

30. Srinivasan, A. V., "Propranolol: A 50-Year Historical Perspective", *Annals of Indian Academy of Neurology*, Vol. 22, No. 1, p. 21, 2019.

31. Vaughan Williams, E. M., *Classification of Antiarrhythmic Actions*, Springer, Berlin, Heidelberg, 1989.

32. Zubrienė, A., E. Čapkauskaitė, J. Gylytė, M. Kišonaitė, S. Tumkevičius and D. Matulis, "Benzenesulfonamides with Benzimidazole Moieties as Inhibitors of Carbonic Anhydrases I, II, VII, XII and XIII", *Journal of Enzyme Inhibition and Medicinal Chemistry*, Vol. 29, No. 1, pp. 124–131, 2014.

33. Čapkauskaitė, E., A. Zakšauskas, V. Ruibys, V. Linkuvienė, V. Paketurytė, M. Gedgaudas, V. Kairys and D. Matulis, "Benzimidazole Design, Synthesis, and

Docking to Build Selective Carbonic Anhydrase VA Inhibitors", *Bioorganic & Medicinal Chemistry*, Vol. 26, No. 3, pp. 675–687, 2018.

APPENDIX A: BPE EXAMPLE

This section explains how the BPE algorithm works. For simplicity, we use only one protein sequence as document. The amino acid sequence of protein with Uniprot ID P04566 is "HADGVFTSDYSRLLGQLSARKYLESLIHSDALFTDTYTRLRKQMAMKKYLNSVLN". The application of BPE on the example document is shown below:

- Step 1: Counting the occurrence of each pair: HA:1, AD:1, DG:1, GV:1, VF:1, FT:2, TS:1, SD:2, DY:1, YS:1, SR:1, RL:2, LL:1, LG:1, GQ:1, QL:1, LS:1, SA:1, AR:1, RK:2, KY:2, YL:2, LE:1, ES:1, SL:1, LI:1, IH:1, HS:1, DA:1, AL:1, LF:1, TD:1, DT:1, TY:1, YT:1, TR:1, LR:1, KQ:1, QM:1, MA:1, AM:1, MK:1, KK:1, LN:2, NS:1, SV:1, VL:1
- Step 2: Replacing first pair having the highest occurrence with a character that never appears in the sequence (replace FT with X): HADGV**X**SDYSRLLGQLSARKYLESLIHSDAL**X**DTYTRLRKQMAMKKYLNSVLN
- Step 3: Counting the occurrence of each pair: HA:1, AD:1, DG:1, GV:1, VX: 1, XS:1, SD:2, DY:1, YS:1, SR:1, RL:2, LL:1, LG:1, GQ:1, QL:1, LS:1, SA:1, AR:1, RK:2, KY:2, YL:2, LE:1, ES:1, SL:1, LI:1, IH:1, HS:1, DA:1, AL:1, LX:1, XD:1, DT:1, TY:1, YT:1, TR:1, LR:1, KQ:1, QM:1, MA:1, AM:1, MK:1, KK:1, LN:2, NS:1, SV:1, VL:1
- Step 4: Replacing first pair having the highest occurrence with a character that never appears in the sequence (replace SD with Z): HADGVX**Z**YSRLLGQLSARKYLESLIH**Z**ALXDTYTRLRKQMAMKKYLNSVLN
- Step 5: Counting the occurrence of each pair: HA:1, AD:1, DG:1, GV:1, VX:1, XZ:1, ZY:1, YS:1, SR:1, RL:2, LL:1, LG:1, GQ:1, QL:1, LS:1, SA:1, AR:1, RK:2, KY:2, YL:2, LE:1, ES:1, SL:1, LI:1, IH:1, HZ:1, ZA:1, AL:1, LX:1, XD:1, DT:1, TY:1, YT:1, TR:1, LR:1, KQ:1, QM:1, MA:1, AM:1, MK:1, KK:1, LN:2, NS:1, SV:1, VL:1
- Step 6: Replacing first pair having the highest occurrence with a character that never appears in the sequence (replace RL with W): HADGVXZYS**W**LGQLSAR-

KYLESLIHZALXDTYT**W**RKQMAMKKYLNSVLN

- Step 7: Counting the occurrence of each pair: HA:1, AD:1, DG:1, GV:1, VX:1, XZ:1, ZY:1, YS:1, SW:1, WL:1, LG:1, GQ:1, QL:1, LS:1, SA:1, AR:1, RK:2, KY:2, YL:2, LE:1, ES:1, SL:1, LI:1, IH:1, HZ:1, ZA:1, AL:1, LX:1, XD:1, DT:1, TY:1, YT:1, TW:1, WR:1, KQ:1, QM:1, MA:1, AM:1, MK:1, KK:1, LN:2, NS:1, SV:1, VL:1
- Step 8: Replacing first pair having the highest occurrence with a character that never appears in the sequence (replace RK with B): HADGVXZYSWLGQLSA**B**YLESLIHZALXDTYTW**B**QMAMKKYLNSVLN
- Step 9: Counting the occurrence of each pair: HA:1, AD:1, DG:1, GV:1, VX:1, XZ:1, ZY:1, YS:1, SW:1, WL:1, LG:1, GQ:1, QL:1, LS:1, SA:1, AB:1, BY:1, YL:2, LE:1, ES:1, SL:1, LI:1, IH:1, HZ:1, ZA:1, AL:1, LX:1, XD:1, DT:1, TY:1, YT:1, TW:1, WB:1, BQ:1, QM:1, MA:1, AM:1, MK:1, KK:1, LN:2, NS:1, SV:1, VL:1
- Step 10: Replacing first pair having the highest occurrence with a character that never appears in the sequence (replace YL with C): HADGVXZYSWLGQLSAB**C**ESLIHZALXDTYTWBQMAMKK**C**NSVLN

- Step 11: Counting the occurrence of each pair: HA:1, AD:1, DG:1, GV:1, VX:1, XZ:1, ZY:1, YS:1, SW:1, WL:1, LG:1, GQ:1, QL:1, LS:1, SA:1, AB:1, BC:1, CE:1, ES:1, SL:1, LI:1, IH:1, HZ:1, ZA:1, AL:1, LX:1, XD:1, DT:1, TY:1, YT:1, TW:1, WB:1, BQ:1, QM:1, MA:1, AM:1, MK:1, KK:1, KC:1, CN:1, NS:1, SV:1, VL:1

  Since there is no pair left, the words are FT, SD, RL, RK, YL, H, A, D, G, V, Y, S, L, Q, E, I, T, M, K and N.

APPENDIX B: LOSSES OF METAPATH2VEC FOR DIFFERENT GRAPHS

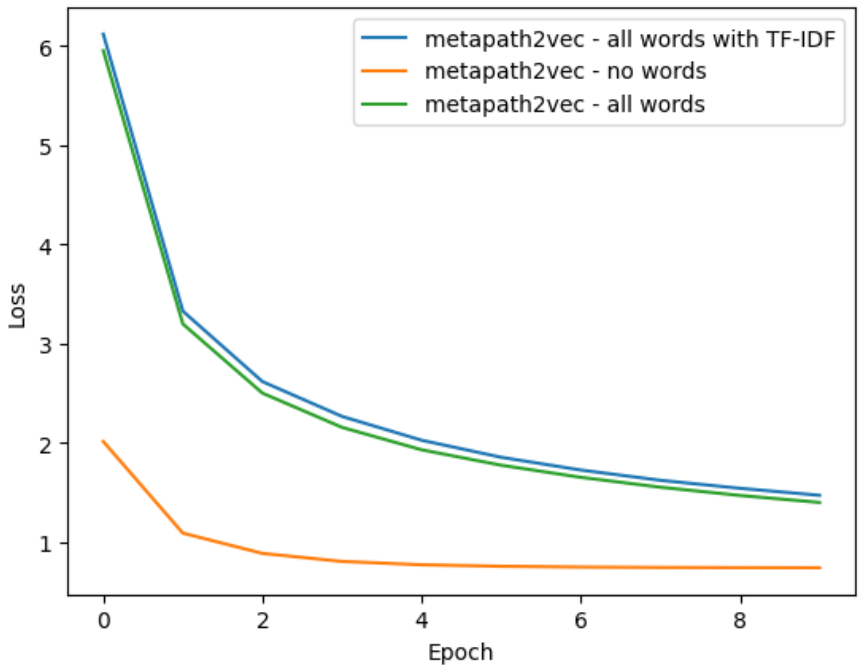

Figure B.1. Plot of average loss for each epoch.