13

Graph Neural Networks in Biochemistry and Healthcare

13.1 Introduction

Graphs have been widely adopted to represent data and entities in computational biochemistry and healthcare. For example, molecules and chemical compounds can be naturally denoted as graphs with atoms as nodes and bonds connecting them as edges. The protein-protein interactions (PPIs), which record the physical contacts established between two or more proteins, can be captured as a graph. Furthermore, in the drug industry, drug-drug interactions (DDIs), which describe the adverse outcomes when using certain combinations of drugs for complex diseases, can also be represented as graphs. Given the powerful capacity in learning graph representations, graph neural network models have been adopted to facilitate many biochemistry and healthcare applications, including drug development and discovery, multi-view drug similarity integration, polypharmacy side effect prediction, medication recommendation, and disease prediction. In this chapter, we discuss GNN models for representative applications in biochemistry and healthcare.

13.2 Drug Development and Discovery

Graph neural networks have been adopted to advance many tasks that are important for drug development and discovery. Examples of these tasks include: 1) molecule representation learning, which can facilitate downstream tasks such as molecule property prediction and therefore can help narrow down search space to find more promising candidates with proper properties; 2) molecule graph generation, which aims to generate molecules with desired properties; 3) drug-target binding affinity prediction, which is to predict the drug-target interaction strength and thus can benefit the new drug development

and drug re-purposing; and 4) protein interface prediction, which targets on predicting the interaction interface of proteins and thus can allow us to understand molecular mechanisms. Next, we introduce the applications of graph neural networks in molecule representation learning, drug-target binding affinity prediction, and protein interface prediction. Note that we have introduced representative methods that are partially based on graph neural network models to generate molecular graphs in Section 9.4.2 and Section 9.5.2.

13.2.1 Molecule Representation Learning

It is important to predict the properties of novel molecules for applications in material designs and drug discovery. Deep learning methods have been adopted to perform the predictions on molecular data. Typically, deep learning methods such as feed-forward networks and convolutional neural networks cannot be directly applied to molecular data as the molecule can be of arbitrary size and shape. Hence, the prediction procedure usually consists of two stages: 1) feature extraction: extracting molecule fingerprint, a vector representation encoding the structure information of the molecule; and 2) property prediction: performing prediction with deep learning methods using the extracted fingerprint as input. Traditionally the molecular fingerprint is extracted using some non-differentiable off-the-shelf fingerprint software without guidance from the downstream prediction task. Thus, these extracted representations might not be optimal for the downstream tasks. In (Duvenaud et al., 2015), an end-to-end framework is proposed to perform the predictions, where graph neural networks are adopted to learn the molecular fingerprints in a differentiable way. Specifically, a molecule can be represented as a graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ where nodes are the atoms and edges represent the bonds between these atoms. Thus, the task of molecular property prediction can be regarded as graph classification or graph regression, which requires to learn graph-level representations. Note that in the context of molecules, these representations are called molecular fingerprints. Hence, the graph neural network model adopted to perform this task consists of both graph filtering and graph pooling layers (see general framework in Chapter 5). Specifically, in (Duvenaud et al., 2015), a global pooling method is adopted. Next, we first introduce the graph filtering layers and then introduce the global pooling layer to obtain the molecular fingerprint. For a node $v_i \in \mathcal{V}$, the graph filtering operation (in the l-th layer) can be described as:

$$\mathbf{F}_i^{(l)} = \sigma\left(\left[\mathbf{F}_i^{(l-1)} + \sum_{v_j \in \mathcal{N}(v_i)} \mathbf{F}_j^{(l-1)}\right]\mathbf{\Theta}_{|\mathcal{N}(v_i)|}^{(l-1)}\right), \tag{13.1}$$

where $\mathbf{\Theta}^{(l-1)}_{|\mathcal{N}(v_i)|}$ is a transformation matrix depending on the size of the neighborhood $|\mathcal{N}(v_i)|$ of node v_i. Specifically, in organic molecules, an atom can have up to 5 neighbors, and hence there are 5 different transformation matrices to be learned. The molecular fingerprint $\mathbf{f}_{\mathcal{G}}$ for a molecule $\mathcal{G}$ can be obtained by the following global pooling operation:

$$\mathbf{F}_{\mathcal{G}} = \sum_{l=1}^{L} \sum_{v_i \in \mathcal{V}} \text{softmax}\left(\mathbf{F}_i^{(l)} \mathbf{\Theta}^{(l)}_{pool}\right), \tag{13.2}$$

where L denotes the number of graph filtering layers, and $\mathbf{\Theta}^{(l)}_{pool}$ is utilized to transform the node representations learned in the l-th layer. The global pooling method in Eq. (13.2) aggregates the information from node representations learned in all the graph filtering layers. The obtained molecule fingerprint $\mathbf{f}_{\mathcal{G}}$ can then be adopted for downstream tasks such as property prediction. Both the graph filtering process in Eq. (13.1) and the pooling process in Eq. (13.2) are guided by the given downstream task such as molecule property prediction (Liu et al., 2018a). In fact, besides the method we introduced above, any graph neural networks designed for learning graph level-representations can be utilized to learn the molecular representations. Specifically, we can compose a graph neural network model with graph filtering layers and graph pooling layers, as introduced in Chapter 5. In particular, the MPNN-Filter discussed in Section 5.3.2 was introduced under the context of the molecular representation learning (Gilmer et al., 2017).

13.2.2 Protein Interface Prediction

Proteins are chains of amino acids with biochemical functions (Fout et al., 2017) as shown in Figure 13.2. As shown in Figure 13.1, an amino acid is an organic compound. It contains amine (-NH_2), carboxyl (-COOH) functional groups and a side chain (R group), which is specific to each amino acid. To perform their functions, proteins need to interact with other proteins. Predicting the interface where these interactions occur is a challenging problem with important applications in drug discovery and design (Fout et al., 2017). The protein interaction interface consists of interacting amino acid residues and nearby amino acid residues in the interacting proteins. Specifically, in (Afsar Minhas et al., 2014), two amino acid residues from different proteins are considered to be a part of the interface if any non-hydrogen atom in one amino acid residue is within 6$\mathring{A}$ of any non-hydrogen atom in the other amino acid residue. Therefore, the protein interface prediction problem can be modeled as a binary classification problem where a pair of amino acid residues from dif-

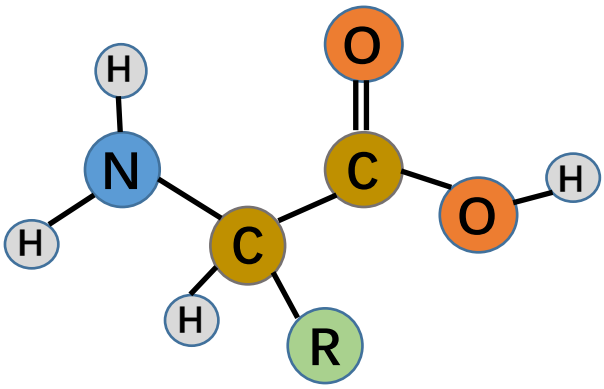

Figure 13.1 An illustrative example of amino acids.

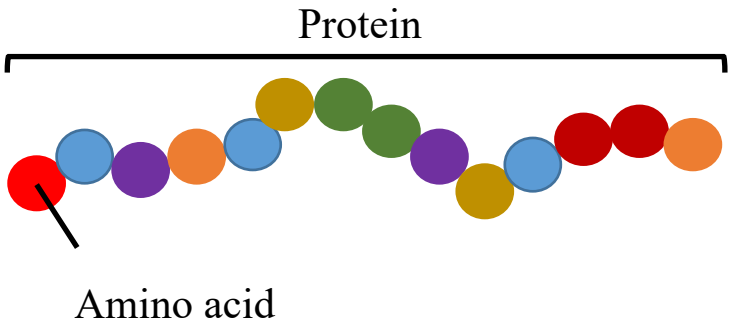

Figure 13.2 A protein consists of a chain of amino acids

ferent proteins are served as input. In (Fout et al., 2017), a protein is modeled as a graph. In the graph, amino acid residues in the protein are treated as nodes, and relations between them are defined as edges. Then, graph neural network models are employed to learn node representations, which are then utilized for classification. Next, we describe how proteins are treated as graphs and then introduce the approach to perform the protein interface prediction.

Representing Proteins as Graphs

A protein can be represented as a graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$. In detail, each amino acid residue in the protein is treated as a node. The spatial relations between the amino acid residues are utilized to build the edges between them. Specifically,

each amino acid residue node is connected to k closest amino acid residues determined by the mean distance between their atoms. Each node and edge in the graph are associated with some features. Specifically, features for node $v_i \in \mathcal{V}$ are denoted as $\mathbf{x}_i$ and features for an edge (v_i, v_j) are represented as $\mathbf{e}_{ij}$.

Protein Interface Prediction

Given a pair of amino acid residues, one from a ligand protein $\mathcal{G}_l = \{\mathcal{V}_l, \mathcal{E}_l\}$ and the other one from a receptor protein $\mathcal{G}_r = \{\mathcal{V}_r, \mathcal{E}_r\}$, the task of protein interface prediction is to tell whether these two residues are in the protein interface. It is treated as a binary classification problem where each sample is a pair of amino acid residues (v_l, v_r) with $v_l \in \mathcal{V}_l$ and $v_r \in \mathcal{V}_r$. Graph filtering operations are applied to $\mathcal{G}_l$ and $\mathcal{G}_r$ to learn the node representations and then the node representations for v_l and v_r are combined to obtain a unified representation for this pair, which is then utilized for the classification by fully-connected layers. A graph filter similar to GCN-Filter (see details of GCN-Filter in Section 5.3.2) is adopted to learn the node representations as (for l-th layer):

$$\mathbf{F}_i^{(l)} = \sigma\left(\mathbf{F}_i^{(l-1)}\mathbf{\Theta}_{\mathrm{c}}^{(l-1)} + \frac{1}{|\mathcal{N}(v_i)|}\sum_{v_j \in \mathcal{N}(v_i)} \mathbf{F}_j^{(l-1)}\mathbf{\Theta}_{\mathrm{N}}^{(l-1)} + \mathbf{b}\right),$$

where $\mathbf{\Theta}_{\mathrm{c}}^{(l-1)}$ and $\mathbf{\Theta}_{\mathrm{N}}^{(l-1)}$ are learnable matrices specific to the centering node and the neighboring node respectively, and $\mathbf{b}$ is the bias term. Furthermore, to incorporate the edge features, the following graph filtering operation is proposed in (Fout et al., 2017):

$$\mathbf{F}_i^{(l)} = \sigma\left(\mathbf{F}_i^{(l-1)}\mathbf{\Theta}_{\mathrm{c}}^{(l-1)} + \frac{1}{|\mathcal{N}(v_i)|}\sum_{v_j \in \mathcal{N}(v_i)} \mathbf{F}_j^{(l-1)}\mathbf{\Theta}_{\mathrm{N}}^{(l-1)} + \frac{1}{|\mathcal{N}(v_i)|}\sum_{v_j \in \mathcal{N}(v_i)} \mathbf{e}_{ij}\mathbf{\Theta}_{\mathrm{E}}^{(l-1)} + \mathbf{b}\right),$$

where $\mathbf{e}_{ij}$ denotes the edge features for the edge (v_i, v_j) and $\mathbf{\Theta}_E^{(l-1)}$ is the learnable transformation matrix for edges. Note that the edge features are fixed during the training process.

13.2.3 Drug-Target Binding Affinity Prediction

The development of a new drug is usually time-consuming and costly. The identification of drug-target interactions (DTI) is vital in the early stage of the drug development to narrow down the search space of candidate medications. It can also be used for drug re-purposing, which aims to identify new targets for existing or abandoned drugs. The task of drug-target binding affinity prediction is to infer the strength of the binding between a given drug-target pair. It can

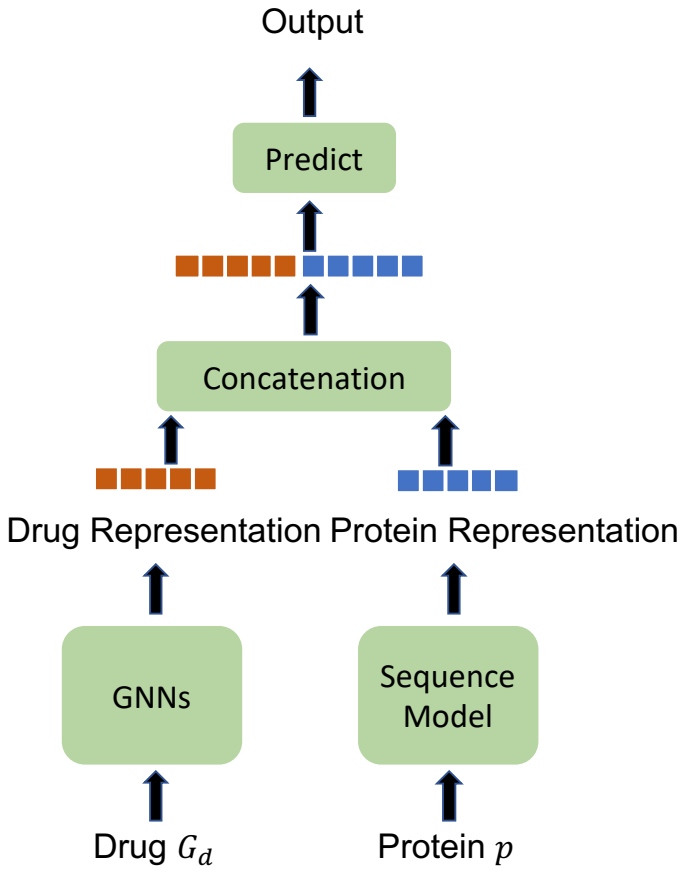

Figure 13.3 A general framework for drug-target binding affinity prediction

be considered as a regression task. There are mainly four types of targets, i.e., protein, disease, gene, and side effect, frequently involved in the task of drug-target interaction prediction. In this section, we use protein as the target to illustrate how graph neural network models can be employed to facilitate this task.

A drug-protein pair is denoted as $(\mathcal{G}_d, p)$, with $\mathcal{G}_d$, p denoting the drug and protein, respectively. The drug $\mathcal{G}_d$ is represented as a molecular graph with atoms as nodes and the chemical bonds as edges. The protein can be denoted as either a sequence or a graph, as shown in Figure 13.2. In (Nguyen et al., 2019), the proteins are represented as sequences of amino acids, which we

adopt to illustrate the framework for drug-target binding affinity prediction. An overview of a general framework for drug-target binding affinity prediction is shown in Figure 13.3. In this framework, the drug $\mathcal{G}_d$ is fed into a graph neural network model to learn a graph-level drug representation, and the protein is fed into a sequence model to learn a protein representation. These two representations are concatenated to generate a combined representation for the pair, and it is then leveraged to predict the drug-target binding affinity. The graph neural network model consists of graph filtering and graph pooling layers, as introduced in Chapter 5. The GNN models introduced in Section 13.2.1 for molecule representation learning can also be used to learn the drug representation. Sequence models such as 1-D CNNs, LSTM, and GRU, can learn the protein representation. Furthermore, if we model the protein as a graph, we can also use the GNNs to replace the sequence model in Figure 13.3 to learn its representation.

13.3 Drug Similarity Integration

With the rapid development of technology, drug data from multiple sources are collected for computational drug discovery research, and drug safety studies. For example, the structural information of drugs can be extracted by chemical fingerprints software and drug indication information is extracted from the packages of drugs (Kuhn et al., 2016). To better facilitate the downstream task such as drug-drug interaction (DDI) prediction, it is necessary to fuse the drug data from multiple sources, as they contain information of drugs from various perspectives. These multiple data sources of drugs may encode different similarities between drugs and thus have different levels of association with targeting outcomes. For example, drugs' structural similarity could have more impact on their interaction profiles than drugs' indication similarity. In (Ma et al., 2018c), an attentive algorithm based on graph neural networks is proposed to fuse the drug similarity information from various sources with the guidance of the down-stream tasks. Specifically, each source of drug features is regarded as a view in (Ma et al., 2018c). For view $t \in \{1, \ldots, T\}$, the features for all nodes in this view are denoted as a matrix $\mathbf{X}_t \in \mathbb{R}^{N \times d_t}$, where N is the number of drugs and d_t is the dimension of the features in this view. Furthermore, the similarity information of the drugs in this view is encoded into a similarity matrix $\mathbf{A}_t \in \mathbb{R}^{N \times N}$. The goal of multi-view drug similarity integration is to fuse the features and similarity matrices from different views to generate integrated features $\mathbf{Z} \in \mathbb{R}^{N \times d}$ and similarity matrix $\mathbf{A}$ across all views.

The similarity matrices from different views are combined as follows:

$$\mathbf{A} = \sum_{t=1}^{T} \text{diag}(\mathbf{g}_t)\mathbf{A}_t, \tag{13.3}$$

where $\mathbf{g}_t \in \mathbb{R}^N$ is the attention scores learned as:

$$\mathbf{g}'_t = \mathbf{\Theta}_t \mathbf{A}_t + \mathbf{b}_t, \forall \quad t = 1, \dots T,$$
$$[\mathbf{g}_1, \dots, \mathbf{g}_T] = \text{softmax}([\mathbf{g}'_1, \dots, \mathbf{g}'_T]),$$

where $\mathbf{\Theta}_t, \mathbf{b}_t \in \mathbb{R}^N$ are parameters to be learned and the softmax function is applied to each row. With the fused similarity matrix $\mathbf{A}$, the fused features are obtained by applying a GNN-Filter on the multi-view features as:

$$\mathbf{Z} = \alpha(\text{GNN-Filter}(\mathbf{A}, \mathbf{X})), \tag{13.4}$$

where $\mathbf{X} = [\mathbf{X}_1, \dots, \mathbf{X}_T]$ is the concatenation of the features from different views. Specifically, the GCN-Filter (see details about GCN-Filter in Section 5.3.2) is adopted in (Ma et al., 2018c) and the softmax function (applied row-wisely) is adopted as the non-linear activation function. A decoder is then used to reconstruct $\mathbf{X}$ from $\mathbf{Z}$ expecting that the fused representations contain as much information from $\mathbf{X}$ as possible. The decoder is also modeled by a GNN-Filter as:

$$\mathbf{X}' = \alpha(\text{GNN-Filter}(\mathbf{A}, \mathbf{Z})), \tag{13.5}$$

where the GCN-Filter is again adopted as the graph filter and sigmoid function is adopted as the non-linear activation function $\alpha()$ in (Ma et al., 2018c). The reconstruction loss is as:

$$\mathcal{L}_{ed} = \|\mathbf{X} - \mathbf{X}'\|^2.$$

The parameters in Eq. (13.3), Eq. (13.4) and Eq. (13.5) can be learned by minimizing the reconstruction loss. Furthermore, the fused representations $\mathbf{Z}$ can be used for downstream tasks and the gradient from the downstream tasks can also be leveraged to update the parameters in Eq. (13.3), Eq. (13.4) and Eq. (13.5).

13.4 Polypharmacy Side Effect Prediction

A lot of complex diseases can not be treated by a single drug. A promising strategy to combat these diseases is polypharmacy. It means using a combination of several drugs to treat patients. However, a major adverse consequence of polypharmacy is that it is highly risky to introduce side effects due

to drug-drug interactions. Hence, it is important to predict the side effects of polypharmacy when adopting novel drug combinations to treat diseases. The polypharmacy side effect prediction task is not only to predict whether a side effect exists between a pair of drugs but also to tell what type of side effect it is. Exploratory analysis in (Zitnik et al., 2018) shows that co-prescribed drugs tend to have more target proteins in common than random drug pairs, which indicates that the interactions between drug and target proteins are important for polypharmacy modeling. Hence, the interactions between drug and target proteins and the interactions between the target proteins, are incorporated for polypharmacy side effect prediction in (Zitnik et al., 2018). In detail, a multi-modal graph is built on the drug-drug interactions (polypharmacy side effects), the drug-protein interactions and the protein-protein interactions. The task of polypharmacy prediction is thus modeled as a multi-relational link prediction task over the multi-modal graph. The goal is to predict whether a link exists between a pair of drugs and then what type of the link it is if existing. Graph neural network models have been adopted to learn the node representations, which are then used to perform the prediction. Next, we first describe how to construct the multi-modal graph and then introduce the framework to perform the polypharmacy side effect prediction.

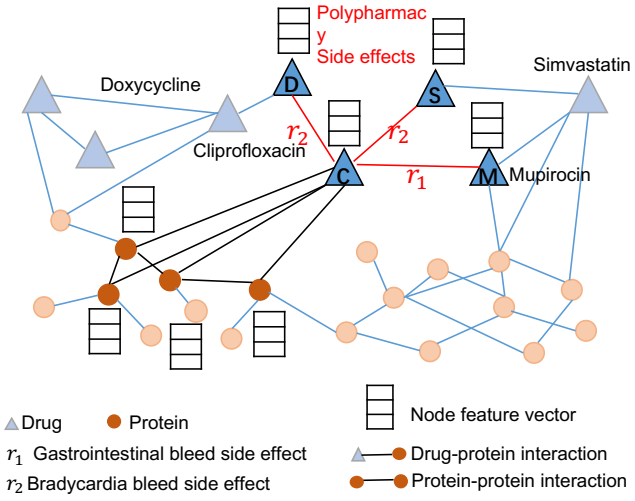

Figure 13.4 An illustrative example of the two-layer multi-modal graph with drug-drug interactions, drug-protein interactions, and protein-protein interactions

Multi-modal Graph Construction

A two-layer multi-modal graph with two types of nodes (drugs and proteins) are built upon three different interactions. As shown in the illustrative example in Figure 13.4, there are drug-drug interactions, drug-protein interactions, and protein-protein interactions. Drug-drug interactions encode the observed polypharmacy side effects. For example, in Figure 13.4, the drug Doxycycline (node D) and drug Ciprofloxacin (node C) are connected by the relation r_2 (bradycardia side effect), which indicates that taking a combination of these two drugs likely leads to the bradycardia side effect. The drug-protein interactions describe the proteins that a drug targets on. For example, in Figure 13.4, the drug Ciprofloxacin (node C) targets on 4 proteins. The protein-protein interactions encode the physical binding relations between proteins in humans. In particular, this two-layer multi-modal graph can be denoted as $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathcal{R}\}$. In $\mathcal{G}$, $\mathcal{V}$ is the set of nodes consisting of drugs and proteins, $\mathcal{E}$ denotes the edges. Each $e \in \mathcal{E}$ is in the form of $e = (v_i, r, v_j)$ with $r \in \mathcal{R}$ and $\mathcal{R}$ is the set of relations, which includes: 1) protein-protein interactions, 2) a target relationship between a drug and a protein; and 3) various types of side effects between drugs.

Polypharmacy Side Effect Prediction

The task of polypharmacy side effect prediction is modeled as a relational link prediction task on $\mathcal{G}$. In particular, given a pair of drugs $\{v_i, v_j\}$, we want to predict how likely an edge $e_{ij} = (v_i, r, v_j)$ of type $r \in \mathcal{R}$ exists between them. In (Zitnik et al., 2018), graph filtering operations are adopted to learn the node representations, which are them utilized to perform the relational link prediction. Specifically, the graph filtering operation designed for knowledge graph (Schlichtkrull et al., 2018) is adapted to update the node representations as:

$$\mathbf{F}_i^{(l)} = \sigma\left(\sum_{r\in\mathcal{R}}\sum_{v_j\in\mathcal{N}_r(v_i)} c_r^{ij}\mathbf{F}_j^{(l-1)}\mathbf{\Theta}_r^{(l-1)} + c_r^i\mathbf{F}_i^{(l-1)}\right), \tag{13.6}$$

where $\mathbf{F}_i^{(l)}$ is the hidden representation of node v_i after the l-th layer, $r \in \mathcal{R}$ is a relation type and $\mathcal{N}_r(v_i)$ denotes the set of neighbors of node v_i under the relation of r. The matrix $\mathbf{\Theta}_r^{(l-1)}$ is a transform matrix specific to the relation type r. c_r^{ij} and c_r^i are normalization constants defined as follows:

$$c_r^{ij} = \frac{1}{\sqrt{|\mathcal{N}_r(v_i)||\mathcal{N}_r(v_j)|}},$$
$$c_r^i = \frac{1}{|\mathcal{N}_r(v_i)|}.$$

The input of the first layer is the node features $\mathbf{F}_i^{(0)} = \mathbf{x}_i$. The final node representations are the output of the L-th layer, i.e., $\mathbf{z}_i = \mathbf{F}_i^{(L)}$, where $\mathbf{z}_i$ denotes the final representation of node v_i. With the learned node representations, given a pair of drugs v_i, v_j, the probability of a relational edge with relation r existing between them is modeled as:

$$p(v_i, r, v_j) = \sigma\left(\mathbf{z}_i^T \mathbf{D}_r \mathbf{R} \mathbf{D}_r \mathbf{z}_j\right), \tag{13.7}$$

where $\sigma()$ is the sigmoid function, $\mathbf{R}$ is a learnable matrix shared by all relations and $\mathbf{D}_r$ is a learnable diagonal matrix specific to relation r. The reason to use a shared matrix $\mathbf{R}$ is that many relations (side effects) between drugs are rarely observed, and learning specific matrices for them may cause overfitting. Introducing the shared parameter matrix $\mathbf{R}$ largely reduces the number of parameters of the model; hence it can help prevent overfitting. During the training stage, the parameters in both the graph filters in Eq. (13.6) and the prediction model in Eq. (13.7) are optimized by maximizing the probability in Eq. (13.7) for those observed side effects between drugs. Note that in (Zitnik et al., 2018), other types of relations, i.e., protein-protein interactions and drug-protein interactions, are also reconstructed during the training with their probabilities formulated similar to Eq. (13.7).

13.5 Disease Prediction

Increasing volume of medical data, which usually contains imaging, genetic and behavioral data, is collected and shared to facilitate the understanding of disease mechanisms. The task of disease prediction is to tell whether a subject is diseased or not given its corresponding medical image and non-image data. The medical images are often the MRI images of the subjects, and the non-image data usually includes phenotypic data such as age, gender, and acquisition site. These two types of information are complementary to each other. Hence, facilitating both information effectively is necessary to enhance the performance of disease prediction. The image data directly provides the features corresponding to some diseases of the subject, and the phenotypic information provides some association between the subjects. For example, subjects with similar ages tend to have more similar outcomes than those with very different ages when all other conditions are the same. Graphs provide an intuitive way of modeling these two types of information, where we treat subjects as nodes with their image as node features and the associations between them as edges. With the built graph, the disease prediction task can be treated as a binary semi-supervised node classification task. It can be tackled by the graph neural

network models, as introduced in Section 5.5.1. Next, we briefly introduce the process of building the graph using the ABIDE database (Di Martino et al., 2014) as an illustrative example (Parisot et al., 2018).

The ABIDE database contains neuroimaging (functional MRI) and phenotypic data of subjects from different international acquisition sites. With this database, we aim to predict whether a subject is healthy or suffering from the autism spectrum disorder (ASD). Each subject is modeled as a node v_i in the graph $\mathcal{G}$, and $\mathbf{x}_i$ denotes the features extracted from its corresponding fMRI image. To build the edges between the nodes (i.e. the adjacency matrix for the graph), we consider both the image data and the non-image phenotypic measures $\mathcal{M} = \{M_h\}_{h=1}^{H}$ as:

$$\mathbf{A}_{i,j} = \text{sim}(\mathbf{x}_i, \mathbf{x}_j) \sum_{h=1}^{H} \gamma_h(M_h(v_i), M_h(v_j)),$$

where $\mathbf{A}$ denotes the adjacency matrix, $\text{sim}(\mathbf{x}_i, \mathbf{x}_j)$ computes the similarity between the features of two nodes v_i and v_j, $M_h(v_i)$ is the h-th phenotypic measure of node v_i and $\gamma_h()$ calculates their similarity. The similarity function $\text{sim}(\mathbf{x}_i, \mathbf{x}_j)$ can be modeled with Gaussian kernels, where nodes with smaller distance have a higher similarity score. Three phenotypic measures, acquisition site, gender and age, are utilized, i.e. we have $H = 3$ for the ABIDE database. The acquisition site is considered since the database is acquired from very different sites with diverse acquisition protocol, which results in less comparable images across different acquisition sites. Gender and age are considered since gender-related and age-related group differences have been observed (Werling and Geschwind, 2013; Kana et al., 2014). For gender and acquisition site, the function $r_h()$ is defined as the Kronecker delta function, which takes value 1 if and only if the two inputs are the same (i.e. they are from the same acquisition site or they have the same gender), otherwise 0. For age, the function $r_h()$ is defined as:

$$\gamma_h\left(M_h(v_i), M_h(v_j)\right) = \begin{cases} 1 & \text{if } \left|M_h(v_i) - M_h(v_j)\right| < \theta \\ 0 & \text{otherwise} \end{cases},$$

where θ is a predefined threshold. This means that subjects with age difference smaller than θ are considered to be similar with each other.

13.6 Conclusion

In this chapter, we introduced various representative applications of graph neural network models in biochemistry and healthcare. We discuss the applications

of graph neural network models in molecule representation learning, drug-target binding affinity prediction, and protein interface prediction. These tasks can facilitate the development and discovery of novel drugs. Next, we introduced an autoencoder based on graph filers to integrate multi-view drug similarities. We also described how graph neural network models can be used for polypharmacy side effect prediction. Besides, we discussed how graph neural network models can be leveraged for disease prediction.

13.7 Further Reading

In addition to the applications we have detailed in this chapter, graph neural networks have been adopted or served as building blocks to benefit many other biochemistry tasks. They are employed as building blocks in models designed for medication recommendation (Shang et al., 2019b,c). In (You et al., 2018a), the graph neural network models have been leveraged as policy networks for molecule graph generation with reinforcement learning. Besides, graph neural network models have also been employed for computational phenotyping (Zhu et al., 2019b; Choi et al., 2020) and disease association prediction (Xuan et al., 2019; Li et al., 2019a).