Pathway2Text: Dataset and Method for Biomedical Pathway Description Generation

Junwei Yang1,2, Zequn Liu2, Ming Zhang2*, Sheng Wang3*
1School of Electronics Engineering and Computer Science, Peking University, Beijing, China
2School of Computer Science, Peking University, Beijing, China
3Paul G. Allen School of Computer Science and Engineering, University of Washington, Seattle, WA

yjwtheonly, zequnliu, mzhang_cs@pku.edu.cn
swang@cs.washington.edu

Abstract

Biomedical pathways have been extensively used to characterize the mechanism of complex diseases. One essential step in biomedical pathway analysis is to curate the description of a pathway based on its graph structure and node features. Neural text generation could be a plausible technique to circumvent the tedious manual curation. In this paper, we propose a new dataset Pathway2Text, which contains 2,367 pairs of biomedical pathways and textual descriptions. All pathway graphs are experimentally derived or manually curated. All textual descriptions are written by domain experts. We form this problem as a Graph2Text task and propose a novel graph-based text generation approach kNN-Graph2Text, which explicitly exploited descriptions of similar graphs to generate new descriptions. We observed substantial improvement of our method on both Graph2Text and the reverse task of Text2Graph. We further illustrated how our dataset can be used as a novel benchmark for biomedical named entity recognition. Collectively, we envision our method will become an important benchmark for evaluating Graph2Text methods and advance biomedical research for complex diseases.1

1 Introduction

Many complex diseases, such as cancer and neurodegenerative disorders, are driven by reactions among a combination of genes and metabolites instead of one single gene (Manolio et al., 2009). These reactions, which are formally referred to as pathways (Kanehisa et al., 2017; Gillespie et al., 2022), are represented as a heterogeneous graph (Figure 1). Each node in this graph is a biomedical entity, such as gene, chemical or metabolite. Each edge is a specific biomedical reaction. Using natural language to describe this pathway graph is of great importance for scientific communication and further promotes applications in complex disease research (Whirl-Carrillo et al., 2012, 2021). To date, these descriptions are almost entirely curated manually by domain experts, thus substantially slowing down downstream biomedical applications (Naithani et al., 2019). Neural text generation has shown promising results in many applications (Bowman et al., 2016; Sutskever et al., 2014; Song et al., 2020; Brown et al., 2020; Rafail et al., 2020; Lewis et al., 2020). Among them, Graph-to-Text (Graph2Text) generation, such as AMR-to-Text (Song et al., 2018; Marcheggiani and Perez-Beltrachini, 2018; Fan and Gardent, 2020), and Knowledge-Graph-to-Text (Colas et al., 2021; Wang et al., 2021), is most similar to pathway description generation. Therefore, we hypothesize that neural text generation could also be a solution here. To fill in the gap, we first propose a novel biomedical pathway description dataset Pathway2Text, which contains 2,367 pairs of pathway and description. Each description is written by domain experts, describing the function and property of this pathway. In contrast to many other Graph2Text datasets (Banarescu et al., 2013; Colas et al., 2021) that use automatic approach to extract the graph from the text, pathways in our dataset are all experimentally measured or manually curated, presenting a high-quality structured data corresponding to the textual description. To the best of our knowledge, Pathway2Text is the first large-scale dataset studying the problem of biomedical pathway description generation.

One unique feature of our dataset is the rich textual information on each node in the graph. Specif-
Succinic acid is a dicarboxylic acid. The anion, succinate, is a component of the citric acid cycle capable of donating electrons to the electron transfer chain. Succinate dehydrogenase (SDH) plays an important role in the mitochondria, being both part of the respiratory chain and the Krebs cycle.

Hypoxia-inducible factor 1-alpha (HIF-1α) functions as a master transcriptional regulator of the adaptive response to hypoxia. Under hypoxic conditions, activates the transcription of over 40 genes, including erythropoietin, glucose transporters, glycolytic enzymes, vascular endothelial growth factor, HILPDA, and other genes whose expression is increased.

Production of interleukin-1 beta (IL-1β) is promoted by the higher levels of HIF-1α. Succinic acid is also necessary for the succinylation of proteins, leading to changes in their structure and function.  

Figure 1: An example of a pathway and its description in our dataset. Each pathway is a heterogeneous graph containing different node types and edge types. Each node has three features: textual label, textual description and node type. For Graph2Text task, the input is the graph and the output is the graph description.
(a container for other nodes) into a tree structure, by adding additional edges between nodes and their container, (3) merged different nodes referring to the same entity into a single node for each pathway, and (4) rewrote the entire dataset into more readable JSON file (cf. Appendix B for more intuitive explanation of our data format).

**Format translation.** Reactome and Pathbank already provide SBGN files, making it straightforward to adapt pathways in both database into our data format. But KEGG only provides KGML (Kanehisa et al., 2017) file, a specific representation of KEGG pathways. So we applied additional modifications to translate KGML file into our data format: we (1) used Process node to represent a reaction instead of directly adding edges between substrates and products, (2) treated a Group of proteins acting on the same reaction as a Complex node, and (3) adjusted node types to match our definition. We refer readers to Appendix A for an illustration for these operations.

**Node description gathering.** Neither SBGN nor KGML file contains detailed node descriptions. SBGN file provides node label (a short text for display), and KGML file provides a KEGG identifier for each node. (1) For Pathbank database, each pathway is also recorded in PWML (DS et al., 2020) format, which contains textual node labels and descriptions. We therefore used node labels given by SBGN file to retrieve node descriptions from PWML file. (2) For Reactome database, each pathway is also stored in BioPAX (Demir et al., 2010) format. 49.2% nodes in BioPAX file have long descriptions while most of the others are only linked to identifiers in external biological entity databases. Among these databases, the Function attribute in UniProt and the Definition attribute in ChEBI are appropriate to be utilized as complements to node descriptions. So we aligned each node in SBGN file to node in corresponding BioPAX file using node label. And then extracted node descriptions from the union of BioPAX file, UniProt and ChEBI. (3) For KEGG database, each KEGG identifier indicates particular information (stored in a TXT file) of a specific entity. We parsed this file to pick entity name, textual Comment and external database identifiers. We used entity names as node labels, used Comments as node descriptions for entities having this attribute (3.7%), and used identifiers of UniProt and ChEBI to retrieve node descriptions for others.

### 2.2 Dataset description

After excluding duplicate pathways and pathways that do not have textual description, we finally obtained 2,367 pairs of pathway and description. An example is shown in **Figure 1**. Each textual description is a few sentences describing functions and structures of the pathway. The textual description has on average 129.5±101.4 words and 7.6±5.3 sentences. Each pathway can be viewed as a heterogeneous graph that contains different types of edges and nodes. There are 7 edge types and 7 node types in the entire dataset, where each pathway has on average 3.5±1.4 edge types and 4.5±1.4 node types. Each node type (e.g., chemical) has a large number of specific classes (e.g., succinic acid). Each class is associated with a concise textual label and a detailed textual description. The average length of the textual description is 114.8 words. We refer to the class description as the node description and the pathway description as the graph description throughout the paper. Each pathway has on average 61±52 nodes and 75±80 edges. In summary, there are four data fields for each pathway description pair: graph description, graph structure, node description and node label.

![Figure 2: Scatter plot showing the consistency between graph-based representation similarity and description-based representation similarity. Each dot is a pair of graphs.](image-url)

To examine the feasibility of conducting Graph2Text and Text2Graph tasks using our dataset, we examined the consistency between graph similarity and description similarity (**Figure 2**). We used GAT (Veličković et al., 2018) to embed each graph into a dense representation. We also obtained a dense representation for each graph description using BioBERT (Lee et al., 2020). For every two graphs, we calculated one similarity score based on their graph-based representations and an-
other similarity score based on their description-based representations. We observed a Pearson correlation 0.35 between these two similarity scores, reflecting a substantial consistency between these two similarity metrics. This indicates that graphs with similar structure tend to have similar textual descriptions, suggesting the possibility to generate textual description using the graph structure and vice versa.

3 Task Description

We aim to generate the textual description for a given biomedical pathway graph and generate the biomedical pathway graph from a given textual description. Let $D = \{D_G, D_S\} = \{(G_i, S_i)\}_{i=1}^N$ be a dataset of paired pathway and its textual description. Each pathway is a directed graph $G = (V, E, F)$, where $V$ represents the set of nodes, $E \subseteq V \times V$ represents the set of edges, and $F$ represents node features. Since each pathway is a heterogeneous graph, we refer to pathway as graph in this paper.

One unique property of the graphs in our dataset is the rich node features $F = \{g, t, d\}$. In particular, each node $v$ is associated with three features $g_v, t_v$, and $d_v$. $g_v \in \{0, 1\}^{n_v}$ is a one-hot vector representing the node type of $v$. $t_v = 1$ if node $v$ is type $i$. $t_v = \langle t_{v1}, t_{v2}, \ldots, t_{vl_v} \rangle$ is the textual label of node $v$. $d_v = \langle d_{v1}, d_{v2}, \ldots, d_{vl_v} \rangle$ is the textual description of node $v$. $t_{vi} \in C$ and $d_{vi} \in C$, where $C$ is the vocabulary. In practice, the textual label is often a phrase and the textual definition is a few sentences. As a result, $|d_v|$ is often much larger than $|t_v|$. Each edge is associated with an edge type $r \in R$, where $R$ is the set of edge types in the dataset. Each graph description is a token sequence defined as $S = \langle S_1, S_2, \ldots, S_l(S) \rangle$, where $S_i \in C$.

We use an inductive learning framework in our experiment. The whole dataset $D$ is randomly divided into $D_{\text{train}} = \{(G_i, S_i)\}_{i=1}^{|D_{\text{train}}|}$ and $D_{\text{test}} = \{(G_i, S_i)\}_{i=|D_{\text{train}}|+1}^N$. For each task, we train our model on $D_{\text{train}}$ and evaluate its performance on $D_{\text{test}}$. Graph $G$ and textual description $S$ are always observed for the training data. We define three tasks based on the unobserved information in the test data as follows:

**Graph2Text.** The input of this task is a graph $G$. All node features are observed on this graph. The output is the description text $S$ for this graph.

**Text2Graph link prediction.** This task aims to predict missing links in a test graph. The inputs are graph description $S$, all node features $F$ and a subset of edges $\{e\}$ in the graph $G$. For a test edge $e_{uv} \in V \times V - \{e\}$, our goal is to classify $e_{uv}$ into a specific edge type $r \in R$.

**Text2Graph node classification.** This task aims to classify each test node into a specific node type in graph $G$. We split nodes in $G$ into training nodes and test nodes. For training nodes, we observed all node features $F$, including textual label, textual description and node type, whereas none of these features is observed for the test node. We also observed the graph description $S$ for $G$. Instead of predicting the node type, we aim at predicting the specific textual label, which is a more challenging task. We form this problem as a node classification task instead of textual generation.

4 Methods

4.1 Graph2Text

The overall framework of our method is shown in Figure 3. We propose a two-step approach. In the first step, we embed each graph into a dense representation through jointly considering its graph structure and node features. In the second step, we use the learnt graph embeddings to find similar graphs for each test graph and then leverage the description of these similar graphs to help the generation.

4.1.1 Description guided graph embedding

One unique property of our dataset is the rich textual features on each node. We hypothesize that unsupervised graph embedding methods might be unable to fully exploit these textual features. Therefore, we first use a supervised approach to obtain
graph embeddings. Since we don’t have any class label for each graph, we treat the graph description as the pseudo label in the supervised learning framework to embed graphs.

In particular, we learn an encoder Enc that projects the graph \( G \) into a dense representation \( h_G \), and then a decoder Dec that maps this representation into the textual description \( S \). The decoder will be discarded in the second step, while the encoder will be used to obtain the representation of an input graph.

Our encoder could be any existing graph neural network architectures (Kipf and Welling, 2017; Veličković et al., 2018; Xu et al., 2019). We first use a pretrained language model BioBERT to encode the textual label \( t_v \) and the description \( d_v \) of each node \( v \) into a dense vector \( t_v \) and a dense vector \( d_v \), and fuse them to get the initial node embedding for node \( v \):

\[
h^0_v = \text{RELU}([t_v] || d_v)W,
\]

where \( W \) represents a trainable parameter matrix and \( || \) is the concatenation operation.

We then propagate this embedding on the graph using a chosen graph neural network architecture, which learns representation of node \( v \) through iteratively updating it with neighbors’ information \( h^l_N(v) \) as:

\[
\begin{align*}
h^l_N(v) &= \text{AGG}((\{h^{-1}_u, e_{uv}\})_{u \in N(v)}), \\
h^l_v &= \text{UPDATE}(h^{-1}_v, h^l_N(v)),
\end{align*}
\]

where \( N(v) \) denotes the set of neighbors for \( v \). AGG and UPDATE are the aggregation and the update function of the specific graph neural network architecture. We studied the performance of using GIN, GCN and GAT as the neural network architecture in our experiments.

After \( L \) iterations, the final embedding \( h^L_v \) can be used to represent the local subgraph comprising node \( v \)’s \( L \)-hop neighbors. Next, for each node, we concatenate its node embeddings from all layers to fuse the information from different ranges of neighbors. We then calculate the graph-level representation by applying a READOUT function to the concatenated node embedding:

\[
\begin{align*}
h_v &= [h^1_v || h^2_v || \cdots || h^L_v]W, \\
h_G &= \text{READOUT}(\{h_v\}_{v \in V}).
\end{align*}
\]

Our decoder is a Transformer based on the pretrained BioBERT. It generates textual description conditioned on \( h_G \):

\[
P(\hat{S}^i|h_G) = \text{Dec}(h_G, S^1^{i-1}).
\]

Finally, the decoder Dec and the encoder Enc are trained jointly using the following loss function:

\[
L_1 = -\frac{1}{|D_{\text{train}}|} \sum_{(G,S) \in D_{\text{train}}} \sum_{S' \in S} \log P(S'|h_G). 
\]

4.1.2 Exploiting descriptions of similar graphs in generation

The above encoder-decoder framework could already be used to generate the description for a given test graph. However, we observed that such generations were not of great quality in our experiment, partially due to the poor utilization of the node textual features. We thus propose to train a new decoder by leveraging the descriptions of similar graphs.

We first use \( h_G \), to find \( k \) similar graphs in the training data:

\[
\begin{align*}
\text{dis}_{ij} &= ||h_G_i - h_G_j||^2, \\
\bar{S}_i &= \{ G_j \in k\text{NN}(G_i) \},
\end{align*}
\]

where \( S_j \) is the description for \( k \) nearest graphs measured by \( \text{dis}_{ij} \). We then embed neighbor’s description \( \bar{S}_i \) into a dense representation \( \bar{s}_i \) using BioBERT:

\[
\begin{align*}
\langle \bar{s}^i_k \rangle &= \text{BioBERT}(\bar{S}_i)W, \\
\bar{s}_i &= \text{Maxpooling}(\langle \bar{s}^i_k \rangle).
\end{align*}
\]

Next, we use multi-head attention framework to calculate a new dense representation \( v^a \) based on description embedding \( \bar{s}_i \) and \( \langle \bar{s}^i_k \rangle \), and a new dense representation \( v^g \) based on graph embedding \( h_G \) and \( \{h_v\} \) as:

\[
s^a(u, v_i, V) = \frac{\exp(Q^a(u)^T K^a(v_i))}{\sum_{v_j \in V} \exp(Q^a(u)^T K^a(v_j))},
\]

Attention\(^a(u, V) = \text{LeakyReLU}(\sum_{v_j \in V} s^a(u, v_i, V)v_j),
\]

\[
v^a = \text{Attention}^a(h_G, \{h_v\}),
\]

\[
v^g = \text{Attention}^a(\bar{s}_i, \langle \bar{s}^i_k \rangle),
\]

where \( a \in \{1, \ldots, A\} \) indicates the attention head number. \( Q^a \) is a projection function mapping a vector to the query space, which is defined as \( Q^a(v) = \tanh(vQ^a) \), where \( Q^a \) represents a trainable parameter matrix. Similarly, we use \( K^a \) to map a vector to the key space.

Finally, we concatenate the new graph embedding \( v^g \) and new description embedding \( v^a \), and use a pretrained Transformer as the decoder to gen-
We then define the predicted label distribution and the prediction of the graph description feature.

To classify a test node \( v \), we applied a similar attention mechanism on its node embedding \( h_v \) and graph description feature \( ⟨s⟩ \) as:

\[
\langle s_i \rangle = \text{BioBERT}(S_i)W, \\
h_v = \text{Attention}(h_{u,v}, \langle s_i \rangle).
\] (14)

Then we define the predicted label distribution and loss function accordingly as:

\[
P(\hat{\ell}_v|S) = \text{softmax}(\text{MLP}([|h_v|]|h_i)),
\]

\[
L_4 = \frac{1}{|D_{train}|} \sum_{(G, S) \in D_{train}} \sum_{\{v\} \in S} P(\hat{\ell}_v|S).
\] (15)

5 Results

5.1 Experimental setup

We exclude any pathway that is a subgraph of another pathway in all experiments to avoid data leakage. For Graph2Text, we randomly split the graph description pairs into 75% training pairs and 25% test pairs. We used a fixed Transformer encoder in BioBERT and initialized the GNN with xavier initialization. We used a learning rate 5e-5. We found that this method performed better than using a fixed Transformer and warming GNN before the training. We used GAT (Veličković et al., 2018), GCN (Kipf and Welling, 2017) and GIN (Xu et al., 2019) as different graph encoders. The hidden state embedding dimension was set to 128 for GAT and 512 for others. The number of heads of GAT was set as 4. AGG and UPDATE functions were implemented according to the original papers. Global mean pooling was used as the READOUT function. Since Transformer can hardly generate more than 512 tokens, we calculated the loss functions and evaluated the generation only on the first 3 sentences, which have an average token length 69±23 (maximum token length is 471). However, the entire text was used as the input in all tasks through the attention mechanism, and we set the attention head number \( A = 128 \). We set \( k \) to 1 in the \( k \)NN framework. We focused on the 1,173 pathway from Pathbank (DS et al., 2020) in our experiments.

For Text2Graph node classification, we randomly split the graph and description pairs into 75% training pairs and 75% test pairs. We used 10% nodes as the test node in each graph. In Text2Graph link prediction task, we varied the proportion of the test set (10%, 30%, 50%, 70%, 90%). We sampled 40% edges for each graph and the same number of edges from the complementary graph as the test edge. In link prediction and node classification, we only used GAT since it obtained the best performance in Graph2Text. We set the learning rate to 5e-4. We used Adam optimizer for all optimizations.

In Graph2Text task, we compared our methods to supervised graph neural network which jointly trains a graph neural network and a transformer. We denote them as GNN (des.), GNN (label), GNN (des. + label) and GNN (structure only) based on the node features used. In particular, GNN (des.) uses textual description as node feature. GNN (label) uses textual label as the node feature. GNN (des. + label) uses both textual label
We sought to evaluate the performance of our method on the task of Graph2Text (Figure 4a, Table 1). Overall, we found that our method achieves the best performance on all metrics (0.296 BLEU-1 score, 0.230 METEOR, 2.2 NIST, and 0.244 ROUGE-L), demonstrating the effectiveness of jointly modeling graph structure, node description and node label. We first compared our method to a GNN model which trained a transformer using descriptions of similar graphs to the final description. Different GNN architectures are used to identify nearest neighbors in kNN based on the graph information.

<table>
<thead>
<tr>
<th>Method</th>
<th>BLEU1</th>
<th>BLEU2</th>
<th>BLEU3</th>
<th>METEOR</th>
<th>NIST</th>
<th>ROUGE-L</th>
</tr>
</thead>
<tbody>
<tr>
<td>GIN (structure only)</td>
<td>14.5</td>
<td>2.2</td>
<td>0.9</td>
<td>12.1</td>
<td>0.8</td>
<td>19.4</td>
</tr>
<tr>
<td>GNN (des.)</td>
<td>18.7</td>
<td>2.5</td>
<td>0.9</td>
<td>11.9</td>
<td>1.1</td>
<td>16.6</td>
</tr>
<tr>
<td>GNN (label)</td>
<td>21.4</td>
<td>4.2</td>
<td>1.3</td>
<td>15.2</td>
<td>1.2</td>
<td>17.1</td>
</tr>
<tr>
<td>GNN (des. + label)</td>
<td>27.1</td>
<td>11.9</td>
<td>10.8</td>
<td>20.5</td>
<td>1.9</td>
<td>23.9</td>
</tr>
<tr>
<td>kNN-Transformer</td>
<td>26.8</td>
<td>12.3</td>
<td>10.6</td>
<td>20.4</td>
<td>1.9</td>
<td>24.3</td>
</tr>
<tr>
<td>kNN-Graph2Text (Ours)</td>
<td>29.6</td>
<td>13.8</td>
<td>11.4</td>
<td>23.0</td>
<td>2.2</td>
<td>24.4</td>
</tr>
</tbody>
</table>

5.2 Graph2Text

We next investigated the performance of node classification on the task of Text2Graph. Here, we studied two classic graph classification tasks: link prediction and node classification. We summarized the performance of link prediction in Figure 5a. We obtained an average of 0.781 accuracy score across 8 different edge types, demonstrating an accurate prediction of the graph structure using the graph description. We further examined the effect of using the graph description in Figure 4b and observed that all 8 edge types had better F1 score when the graph description was used. We observed the same improvement of using the graph description when evaluated using the accuracy. We also performed the ablation study for the graph structure and observed similar improvement Figure 4c. These results collectively confirm that our method can generate the graph structure based on the graph description, offering biologists novel insights in pathway analysis.

We then studied the performance of node classification. We considered 3 most frequent node types in our dataset: macromolecule, multimer
and chemical. For each node type, we formed the node classification task as a multi-class classification problem, where each test node is classified into a specific class defined by the textual label. We noticed that each node type has a large number of classes. Therefore, we first evaluated two naive baselines: random guess and majority vote. Random guess obtained 0.0009 average accuracy, while majority vote obtained 0.046 average accuracy, suggesting a challenging classification task.

Our method obtained a desirable classification performance, which was substantially higher than the performance of the variant that does not consider the graph description (Figure 5b). The improvement of using graph description on both node classification and link prediction further confirm that our dataset could be a promising benchmark for Text2Graph task.

6 Application to Named Entity Recognition

Named entity recognition (NER) is essential in detecting chemicals, genes, and diseases from biomedical text (Leaman et al., 2016; Luo et al., 2018; Kim et al., 2019; Yoon et al., 2019), and further facilitating downstream bioNLP applications, such as relation extraction (Xing et al., 2020). A major bottleneck in NER is the lack of curated benchmarks since such curation often requires substantial domain expertise. Our dataset Path2wayText can be used as a novel curated benchmark for NER.

Specifically, we used the graph description as the sentences that one wants to perform NER. We then obtained the ground truth entity type of phrases in these sentences according to their curated node types in the graph. Since the graphs, including all node types, are curated by domain experts, such node types can be used as the ground truth entity types for NER. Here, we focused on two most frequent entity types in our dataset: protein and chemical. We noticed that some phrases in the graph description sentences might also be a protein or chemical, even though they were not curated in the graph. We excluded such phrases in the evaluation in order to maintain the quality of our NER benchmark.

To this end, we obtained the graph-based curation of 8,779 protein entities and 1,621 chemical entities, offering a good complementary to existing biomedical NER datasets (Kim et al., 2003; Smith et al., 2008; Doğan et al., 2014; Krallinger et al., 2015; Li et al., 2016; Wei et al., 2018). To further investigate the performance of our novel NER datasets, we tested a few state-of-the-art biomedical NER methods, including BERN (Kim et al., 2019), CollaboNet (Yoon et al., 2019), Multi-BioNER (Wang et al., 2019), and NeuroNER (Dernoncourt et al., 2017). We observed that NeuroNER obtained the best performance on protein and Multi-BioNER achieved the best performance on Chemical (Figure 5c). Moreover, existing approaches only consider the graph description sentences when labelling entity types. In addition to graph description, our dataset also contains the corresponding graph structure, which has been shown to be critical in graph description generation in our experiments. Therefore, we hypothesize that graph structure might be also helpful in NER, and envision our dataset to be an important resource for benchmarking graph-based NER methods (Radford et al., 2015; Rijhwani et al., 2020; He et al., 2020; Nie et al., 2021).
7 Related Work

Graph2Text, which aims at generating a textual description for a structured graph, has attracted attentions in different applications. Existing Graph2Text datasets aims to generate text from RDF data (Gardent et al., 2017), knowledge graph (Koncel-Kedzior et al., 2019; Jin et al., 2020; Cheng et al., 2020; Colas et al., 2021; Wang et al., 2021), street view map (Schumann and Riezler, 2021), Abstract Meaning Representation (AMR) (Banarescu et al., 2013; Marcheggiani and Perez-Beltrachini, 2018; Song et al., 2018; Ribeiro et al., 2019; Zhu et al., 2019; Hajdič et al., 2019; Damonte and Cohen, 2019; Mager et al., 2020; Zhang et al., 2020; Zhao et al., 2020; Fan and Gardent, 2020; Wang et al., 2020), terminology ontology (Liu et al., 2021) and graph-transduction grammars (Belz et al., 2011; Mille et al., 2019, 2020). Our dataset is the first Graph2Text dataset that focuses on biomedical pathway generation. In addition, our dataset has more complicated node features than many existing Graph2Text datasets, where each node in our dataset has a node type, a concise textual label and a detailed textual description.

Text2Graph can be viewed as an information extraction task, which aims at mining structured knowledge from free text. The datasets that are more relevant to our task could be generating a knowledge graph from long document (Kertkeidkachorn and Ichise, 2017; Bosselut et al., 2019; Kannan et al., 2020; Wu et al., 2020). Many of these existing datasets use automatic annotation to extract the graph information from corpus (Kertkeidkachorn and Ichise, 2017; Bosselut et al., 2019), which might introduce bias from the extraction method. In contrast, graphs in our dataset are either experimentally derived or manually curated, presenting a high-quality complementary to existing Text2Graph datasets.

8 Conclusion and Future work

We have presented a novel dataset Pathway2Text for biomedical pathway description generation. Our dataset contains 2,367 pairs of curated pathway and its associated description. To generated description for biomedical pathways, we have proposed a kNN-Graph2Text approach, which utilizes neighbor’s description to enhance the text generation. We have extensively evaluated our method and observed substantial improvement in comparison to conventional graph neural network architectures. Furthermore, we have investigated the reverse task of Text2Graph and illustrated how our dataset can serve as a novel benchmark for biomedical NER.

In addition to Graph2Text, Text2Graph and NER, our dataset can also be used to investigate other important applications. For example, our dataset can be used as a relation extraction benchmark by regarding graph descriptions as sentences and graph edge types as the ground truth relation type. We can also use our dataset to study other graph-based tasks, such as generating node description given the graph structure and the graph description. Another interesting application is to identify the importance of each node in the graph, which has important applications in recommender system and social media. The order of mentions of each node in the graph description can be used to evaluate the node importance since the graph description often starts from the most important node.

From a methodological perspective, we plan to develop semi-supervised approaches to leverage many other biomedical pathways that currently do not have curated description. For example, we can train a Graph Transformer (Cai and Lam, 2020) on these unlabelled pathways and then fine-tune the model on pathways with graph description. We also want to explore other geometric embedding methods, such as hyperbolic embedding (Cvetkovski and Crovella, 2009) and spherical embedding (Meng et al., 2019, 2020), since biomedical pathways often form a hierarchical structure.

More importantly, our dataset could also open up new venues in biomedical research. Any computational biology tools that utilize biomedical pathways as features in their pipeline can exploit the graph description as additional features. For biomedical pathways that do not have the corresponding description, one can use the description generated by our kNN-Graph2Text as the feature. We envision this will substantially advance a wide range of biomedical research that involves pathway analysis, and our dataset will introduce other new text generation tools developed in the NLP community to broader audience in biomedicine.

Acknowledgement

This paper is partially supported by National Key Research and Development Program of China with Grant No. 2018AAA0101902 as well as the National Natural Science Foundation of China (NSFC Grant No. 62106008 and No. 62006004).


Nathawut Kertkeidkachorn and Ryutaro Ichise. 2017. T2KG: an end-to-end system for creating knowledge graph from unstructured text. In *AAAI*, volume WS-17 of *AAAI Workshops*.


Yu Meng, Yunyi Zhang, Jiaxin Huang, Yu Zhang, Chao Zhang, and Jiawei Han. 2020. Hierarchical topic mining via joint spherical text and tree embedding. In KDD, pages 1908–1917.


Xuan Wang, Yu Zhang, Xiang Ren, Yuhao Zhang, Marinka Zitnik, Jingbo Shang, Curtis Langlotz, and Jiawei Han. 2019. Cross-type biomedical named entity recognition with deep multi-task learning. *Bioinformatics*, 35(10):1745–1752.


### A KGML Translation

![Figure 6: An illustration of KGML translation mentioned in Section 2.1. The first and second operations aim to unify the expression of reaction. The third operation aims to eliminate inconsistencies between node types. The Orthology nodes in KGML file are omitted during this translation.](image)

### B Data Format

Our dataset is stored in a JSON file. And the hierarchy structure is organized as follows:

```json
{
    "Graph_identifier": {
        "Name": "",
        "Graph_description": "",
        "Node_dict": {
            "Node_identifier": {
                "type": "",
                "label": "",
                "description": ""}
        },
        ...
    },
    "Arc_list": {
        "arc_source": "",
        "arc_target": "",
        "arc_type": "",
        ...
    }
}
```
The node types include Submap, Macromolecule, Process, Complex, Multimer, Simple Chemical and Others. The Others is the union of several types occurring only in a single database (e.g., Unspecified Entity, Association in Reactome and Transport in Pathbank). Nodes in this type account for 7% over the whole dataset. The edge types include Catalysis, Consumption, Stimulation, Inhibition, Production, Logic Arc and Belong To, where the Belong To represents edges that were added for Complex node reconstruction mentioned in Section 2.1.