# GNN Integration to knowledge graph for Nephrology QA system

**Arvind Murari Vepa**
Department of Computer Science
University of California, Los Angeles
Los Angeles, CA 90095
amvepa@ucla.edu

**Justin Zhicheng He**
Department of Computer Science
University of California, Los Angeles
Los Angeles, CA 90095
justinhe@ucla.edu

**Sangjoon Lee**
Department of Computer Science
University of California, Los Angeles
Los Angeles, CA 90095
sangjoonlee@cs.ucla.edu

## Abstract

The goal of our project is to teach an AI system to be as proficient as a human nephrologist by "reading" popular textbooks in nephrology. We represent the AI system's knowledge using a knowledge graph (KG), which is generated using an information extraction pipeline on a novel textbook dataset. We evaluate the AI system's knowledge with 1500 multiple-choice exam questions collected by a UCLA Nephrology professor. Using state-of-the-art QA models, we are able to demonstrate significant results against our baselines.[1]

## 1 Introduction

There is significant clinical interest to create an AI application that is as as proficient as human doctors. While there have been advances in other medical fields (e.g. radiology [Patel et al., 2019]), there are no similar applications in nephrology. In our study, we create an AI application that can "read" nephrology textbooks, akin to how students would "read" textbooks to study the field. We retain knowledge using a knowledge graph (KG). Currently, there is no domain-specific knowledge base for the field of nephrology. Thus, automated KG construction has significant additional utility for nephrologists. In our study, we apply NLP information extraction algorithms to 563 chapters from relevant nephrology textbooks and 814 research articles in nephrology to construct our knowledge graph.

Similar to how medical students' knowledge would be evaluated with an exam, we evaluate the model's knowledge using 1500 multiple choice exam questions. These questions are extracted from a larger collection of exam questions from a UCLA nephrology professor using regular expression. We utilize two state-of-the-art QA models that consume the KG and question body to answer these complex medical questions and compare the results with a random choice baseline.

[1] All code is available at https://drive.google.com/drive/folders/1PSzeH5L5yJf6FkDk0PXE7KGgquU_0DBk

Preprint. Under review.

## 2 Related Work

There are several previous projects in which researchers automatically generated a KG from medical data [Kim et al., 2021, Dessì et al., 2020, Rossanez et al., 2020, Wu et al., 2021, Wang et al., 2020]. In one work, researchers were able to extract relations from unstructured and structured medical data [Wu et al., 2021]. Additionally, researchers were able to generate a knowledge graph from scientific literature and utilized a medical knowledge base to extract entities, relations, and events as well as distantly supervised models for entity recognition [Wang et al., 2020]. Rather than using a general knowledge base, in one work researchers generated a KG from a vast amount of text documents about COVID-19 [Kim et al., 2021].

There is also significant research in utilizing KGs for downstream tasks. Researchers produced a diagnosis of thyroid disease based on a learned knowledge graph from patient symptoms [Chai, 2020]. Additionally, after automatically generating a knowledge graph from scientific literature, researchers have applied this to QA and report generation [Wang et al., 2020]. While the previous model handles relatively simple QA queries, QAGNN handles complex questions with relevance scoring (where language models are used to estimate the importance of KG nodes relative to the given QA context) and joint reasoning (where one can connect the QA context and KG to form a joint graph, and mutually update their representations through graph neural networks) [Yasunaga et al., 2021]. Researchers build on this with GreaseLM which fuses encoded representations from pretrained LMs and graph neural networks over multiple layers of modality interaction operations. Information from both modalities propagates to the other, allowing language context representations to be grounded by structured world knowledge, and allowing linguistic nuances (e.g., negation, hedging) in the context to inform the graph representations of knowledge [Zhang et al., 2022].

## 3 Methodology

### 3.1 Information Extraction

The information extraction pipeline involves entity and relation extraction. The entity extraction process involves mention detection and entity linking with the UMLS knowledge base using the SciPy Biomedical Text Processing library. The relation extraction process involves applying a pretrained zero-shot relation extractor [Cetoli, 2020] to extract the relations 'associated', 'interacts', 'causes', and 'treats' from the linked entities.

In our study, we apply NLP information extraction algorithms to 563 chapters from relevant nephrology textbooks and 814 research articles in nephrology to construct our knowledge graph. However, due to processing time and space constraints, we were only able to use 10% of our textbook data for the QA model. The generated graph has 47,272 entities and 473,590 relations. In order to compare the quality of our information extraction, we also evaluated our QA model on Disease Database portion of the Unified Medical Language System and DrugBank. The knowledge graph contains 9,958 nodes and 44,561 edges and was originally used for the MedQA-USMLE task for GreaseLM [Zhang et al., 2022].

### 3.2 Question Pre-processing

Pre-processing was done on the raw QA data to convert it into a compatible form for the QA models. The pre-processing pipeline is described in the following steps:

1. Separate the raw data into questions and extract the full question text and the correct answer choice
2. Create a statement from each question-answer pair by extracting the question stem and answer choices from the full question text and appending each answer choice to the stem
3. Perform entity linking on the question stem and each answer choice separately
4. Create a grounded statement from each statement by annotating its corresponding statement with extracted entities from the stem and answer choice
5. Construct the knowledge subgraph for each grounded statement by collecting the nodes of all 2-hop neighbors from the nodes of the previously extracted entities

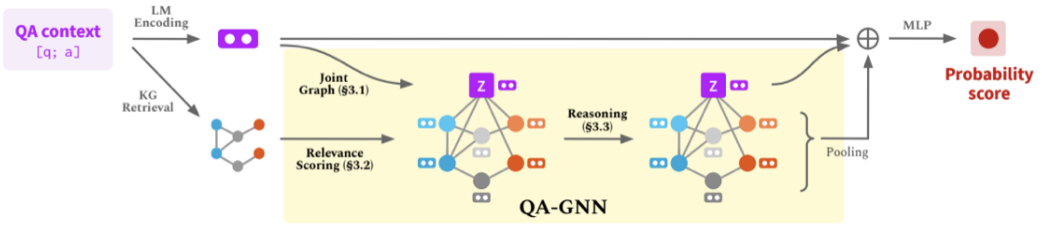


Figure 1: Flow diagram for QAGNN from original QAGNN paper [Yasunaga et al., 2021]

### 3.3 QA Model

QA models normally involve using BERT [Devlin et al., 2018] to vectorize the question body and answer. This is computationally feasible and also produces contextualized features, which are useful for the QA task. If we have prior knowledge on a topic in the form of a knowledge graph, we additionally would like to utilize use the the entities and relationships between different entities that are encoded in the knowledge graph.

With the Graph Neural Network's increased popularity in the machine learning community, we believe generating a GNN from the knowledge graph can further generalize its knowledge and even interpolate unseen knowledge and relationships between entities. This was the motivation for using QAGNN and the improved version, GreaseLM. Both of these models involve addressing QA by combining BERT and GNN in meaningful ways.

#### 3.3.1 QAGNN

QAGNN is one of the first models that incorporates a knowledge graph as a GNN in a QA system. To answer a question, the QAGNN first extracts a feature vector from the question body and answer choices using a pretrained BERT. Because it would be infeasible to use the entire knowledge graph to answer a question, the feature vector is used to extract a relevant subgraph of the knowledge graph. This involves selecting subgraphs by relevance scoring, which involves counting entities up to 3 hops away from the entities seen in the question body and answer choices. Additionally, the CLS feature vector is added as a node to the subgraph and this is used to create a GNN. This GNN undergoes several Graph Attention Network updates, which modifies the representation of these nodes. Finally, the model predicts the answer choice using the updated answer nodes, the updated CLS embedding node, and the original CLS embedding with a multilayer perceptron and softmax layer.

#### 3.3.2 GreaseLM

GreaseLM is an improved version of QAGNN. Instead of using the original CLS embedding node, updated answer and CLS nodes to make the prediction, these vectors are fed into the GreaseLM's signature GreaseLM layers. A GreaseLM layer consists of two components: the pretrained BERT layer (i.e Language Model ("LM") layer) that uses the original CLS embedding to generate the next layer CLS embedding and, additionally, a Graph Attention Network update layer (i.e. GM layer) that uses the updated answer and CLS nodes and generates a modified graph representation. These LM and GM layers run in parallel. Using a modality interaction unit (Mint), information is exchanged between the LM and GM layers via a multiplayer perceptron. The researchers believed that the mixture of information between LM layer and GNN layer can facilitate communication between layers to better predict answer choices. At the end of the architecture, the GreaseLM uses essentially the same three component like QAGNN to make the prediction: final CLS embedding, updated answer nodes, and the updated CLS node, all from the final GreaseLM layer.

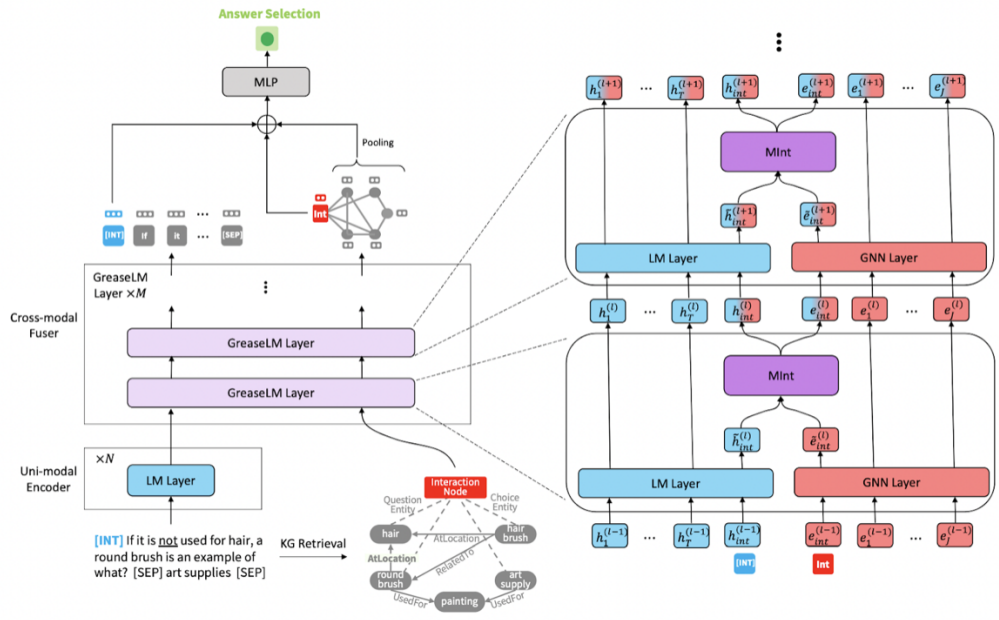


Figure 2: Flow diagram for GreaseLM from original GreaseLM paper [Zhang et al., 2022]

| Model | DB | Test Accuracy (%) |
|---|---|---|
| GreaseLM | NephDB | 35.81 |
| GreaseLM | DiseaseDB | **37.21** |
| QAGNN | NephDB | 34.42 |
| QAGNN | DiseaseDB | 34.88 |
| Random | N/A | 21.80 |

Table 1: Results from our experiments using the two QA models and the two different DBs. We also computed the expected accuracy for a random chance baseline

# 4 Experiments

## 4.1 Approach

In our experiments, we trained and evaluated the QA models on the db generated on the nephrology textbooks (NephDB) and the db corresponding to both the DiseaseDB and Drugbank data (DiseaseDB). We partitioned the dataset into 85% train and 15% test. We additionally partitioned the train set into 90% train and 10% validation. We used early stopping to pick the checkpoint with the best validation accuracy over the last ten epochs with no improvement and used that checkpoint for test set evaluation. We also calculated a random chance baseline: because the questions varied from 2–5 answer choices, we calculated the expected number of correct answers and provided this for comparison. While we provided some results in Table 1, the full results can be seen in the Appendix.

## 4.2 Results

You can see the results from our experiments in Table 1. From this, we can see that all our model results improve on the random chance baseline significantly: approximately 13–16% improvement. However, we did not see an improvement with using the NephDB versus the DiseaseDB. The GreaseLM model trained on the DiseaseDB did have more hyperparameter experiments (see Appendix); however, looking at both the QAGNN and GreaseLM models, there does not seem to improvement using the NephDB. We have not had the opportunity to do a more rigorous error analysis: we do know that the NephDB is a significantly larger DB which could have played a part. Additionally the performance does not suffer significantly (1–2%) and clearly the models are better than a random chance baseline.

More analysis needs to be done to generate further improvements to our information extraction pipeline.

Additionally, we see that GreaseLM slightly outperforms the QAGNN models. The GreaseLM is considered the more advanced architecture and, thus, it is reasonable that it slightly outperforms the QAGNN model. While the improvement for the models trained on the NephDB is relatively small (1.39%), the improvement for the models trained on the DiseaseDB is higher (2.33%).

## 5 Conclusion

In our project, we successfully demonstrated our ability to teach an AI system about the field of nephrology by "reading" popular nephrology textbooks. We accomplished this by generating a knowledge graph using an information extraction pipeline on the nephrology texts. We successfully evaluated our knowledge extraction on complex nephrology exam questions with state-of-the-art QA models and achieved performance surpassing baselines. In future work, we hope to improve the performance of our QA model with rigorous error analysis. We hope that with further error analysis we can isolate why the QA model performance suffered slightly with the nephrology KG and make the necessary improvements to our information extraction pipeline.

## References


Alberto Cetoli. Exploring the zero-shot limit of FewRel. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1447–1451, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics. doi: 10.18653/v1/2020.coling-main.124. URL https://aclanthology.org/2020.coling-main.124.

Xuqing Chai. Diagnosis method of thyroid disease combining knowledge graph and deep learning. *IEEE Access*, 8:149787–149795, 2020. doi: 10.1109/ACCESS.2020.3016676.

Danilo Dessì, Francesco Osborne, Diego Reforgiato Recupero, Davide Buscaldi, Enrico Motta, and Harald Sack. Ai-kg: An automatically generated knowledge graph of artificial intelligence. In Jeff Z. Pan, Valentina Tamma, Claudia d'Amato, Krzysztof Janowicz, Bo Fu, Axel Polleres, Oshani Seneviratne, and Lalana Kagal, editors, *The Semantic Web – ISWC 2020*, pages 127–143, Cham, 2020. Springer International Publishing. ISBN 978-3-030-62466-8.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2018. URL https://arxiv.org/abs/1810.04805.

Taejin Kim, Yeoil Yun, and Namgyu Kim. Deep learning-based knowledge graph generation for covid-19. *Sustainability*, 13(4):2276, 2021. ISSN 2071-1050. doi: 10.3390/su13042276. URL https://www.mdpi.com/2071-1050/13/4/2276.

Bhavik N. Patel, Louis Rosenberg, Gregg Willcox, David Baltaxe, Mimi Lyons, Jeremy Irvin, Pranav Rajpurkar, Timothy Amrhein, Rajan Gupta, Safwan Halabi, Curtis Langlotz, Edward Lo, Joseph Mammarappallil, A. J. Mariano, Geoffrey Riley, Jayne Seekins, Luyao Shen, Evan Zucker, and Matthew P. Lungren. Human–machine partnership with artificial intelligence for chest radiograph diagnosis. *npj Digital Medicine*, 2(1):1–10, 2019. doi: 10.1038/s41746-019-0189-7. URL https://doi.org/10.1038/s41746-019-0189-7.

Anderson Rossanez, Julio Cesar dos Reis, Ricardo da Silva Torres, and Hélène de Ribaupierre. Kgen: a knowledge graph generator from biomedical scientific literature. *BMC Medical Informatics and Decision Making*, 20(4):1–24, 2020. doi: 10.1186/s12911-020-01341-5. URL https://doi.org/10.1186/s12911-020-01341-5.

Qingyun Wang, Manling Li, Xuan Wang, Nikolaus Parulian, Guangxing Han, Jiawei Ma, Jingxuan Tu, Ying Lin, Haoran Zhang, Weili Liu, Aabhas Chauhan, Yingjun Guan, Bangzheng Li, Ruisong Li, Xiangchen Song, Yi R. Fung, Heng Ji, Jiawei Han, Shih-Fu Chang, James Pustejovsky, Jasmine Rah, David Liem, Ahmed Elsayed, Martha Palmer, Clare Voss, Cynthia Schneider, and Boyan Onyshkevych. Covid-19 literature knowledge graph construction and drug repurposing report generation, 2020. URL https://arxiv.org/abs/2007.00576.

Jialun Wu, Yang Liu, Zeyu Gao, Tieliang Gong, Chunbao Wang, and Chen Li. Bioie: Biomedical information extraction with multi-head attention enhanced graph convolutional network, 2021. URL https://arxiv.org/abs/2110.13683.

Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. Qa-gnn: Reasoning with language models and knowledge graphs for question answering, 2021. URL https://arxiv.org/abs/2104.06378.

Xikun Zhang, Antoine Bosselut, Michihiro Yasunaga, Hongyu Ren, Percy Liang, Christopher D. Manning, and Jure Leskovec. Greaselm: Graph reasoning enhanced language models for question answering, 2022. URL https://arxiv.org/abs/2201.08860.

# A Appendix

| Model | Batch | GNN Layers | Decoder LR | Encoder LR | GNN Dim | Val Acc | Test Acc |
|---|---|---|---|---|---|---|---|
| Default | 128 | 5 | $1e-5$ | $1e-3$ | 200 | 36.89 | 34.88 |
| K=3 | 128 | 3 | $1e-5$ | $1e-3$ | 200 | 34.43 | 30.23 |
| K=7 | 128 | 7 | $1e-5$ | $1e-3$ | 200 | 29.51 | 29.77 |

Table 2: Results from QAGNN trained on DiseaseDB

| Model | Batch | GNN Layers | Decoder LR | Encoder LR | GNN Dim | Val Acc | Test Acc |
|---|---|---|---|---|---|---|---|
| Default | 128 | 5 | $1e-5$ | $1e-3$ | 200 | 36.89 | 31.16 |
| K=3 | 128 | 3 | $1e-5$ | $1e-3$ | 200 | 32.79 | 29.30 |
| K=2 | 128 | 2 | $1e-5$ | $1e-3$ | 200 | 31.97 | 31.16 |
| elr = $5e-5$ | 128 | 5 | $5e-5$ | $1e-3$ | 200 | 40.16 | 34.42 |
| K=7 | 128 | 7 | $1e-5$ | $1e-3$ | 200 | 32.79 | 29.77 |
| K=9 | 128 | 9 | $1e-5$ | $1e-3$ | 200 | 22.13 | 25.12 |

Table 3: Results from QAGNN trained on NephDB

| Model | Batch | GNN Layers | Decoder LR | Encoder LR | Seq len | Val Acc | Test Acc |
|---|---|---|---|---|---|---|---|
| Default | 128 | 3 | $1e-3$ | $5e-5$ | 256 | 34.43 | 35.35 |
| K=5 | 128 | 5 | $1e-3$ | $5e-5$ | 256 | 33.62 | 29.30 |
| K=2 | 128 | 2 | $1e-3$ | $5e-5$ | 256 | 28.69 | 23.72 |
| elr = $1e-5$ | 128 | 3 | $1e-3$ | $1e-5$ | 256 | 30.33 | 30.23 |
| Seq len=512 | 128 | 3 | $1e-3$ | $5e-5$ | 512 | 27.05 | 23.26 |
| K=7 | 128 | 7 | $1e-3$ | $5e-5$ | 256 | 28.69 | 37.21 |
| dlr = $1e-4$ | 128 | 5 | $1e-4$ | $5e-5$ | 256 | 31.97 | 37.21 |
| Seq len=512 | 128 | 9 | $1e-3$ | $5e-5$ | 512 | 30.33 | 34.42 |
| K=7 | 128 | 1 | $1e-3$ | $5e-5$ | 256 | 31.97 | 35.81 |

Table 4: Results from GreaseLM trained on DiseaseDB

| Model | Batch | GNN Layers | Decoder LR | Encoder LR | Seq len | Val Acc | Test Acc |
|---|---|---|---|---|---|---|---|
| Default | 128 | 3 | $1e-3$ | $5e-5$ | 256 | 34.43 | 35.81 |
| K=4 | 128 | 4 | $1e-3$ | $5e-5$ | 256 | 32.79 | 33.95 |
| K=2 | 128 | 2 | $1e-3$ | $5e-5$ | 256 | 31.97 | 34.42 |
| elr = $1e-5$ | 128 | 3 | $1e-3$ | $1e-5$ | 256 | 33.61 | 31.16 |
| elr = $1e-4$ | 128 | 3 | $1e-3$ | $1e-4$ | 256 | 32.79 | 33.02 |

Table 5: Results from GreaseLM trained on NephDB