



The University of Hong Kong

Faculty of Engineering

Department of Computer Science

COMP7704

Dissertation for Examination

Machine Comprehension of Court Cases:
Construction of Drug Trafficking Court Case Knowledge Graph and Its
Applications

Submitted in partial fulfillment of the requirements for the admission to the
degree of Master of Science in Computer Science

By
Chen Yunkun
3035561924

Supervisor's title and name: Professor Ben C.M. Kao
Date of submission: 10/04/2020

Abstract

Hong Kong uses Common Law as their legal architecture. Common Law, which is also usually called as “Case Law”, means that the judgements are normally guided by previous law cases which are like the current one. Meanwhile, the developing speed of knowledge graph is fast. As a result, it is acceptable to construct a knowledge graph which can be easily used by legal practitioners. However, the law industry currently lacks such knowledge graphs, and it is still a challenging task for people to easily make a satisfying knowledge graph about court cases. When constructing a knowledge graph, there are two important procedures: Named Entity Recognition and Relation Extraction. In this dissertation, we focus on drug trafficking cases and try to find a good approach to construct a knowledge graph. Two models are involved: Separate model and Joint model. The results show that for the legal texts, it is better to separate the procedures of Named Entity Recognition and Relation Extraction into two parts. Based on the experience above, we build a knowledge graph of drug trafficking court cases in Hong Kong. We further build a query system based on the knowledge graph for legal practitioners.

Declaration

I hereby declare that this dissertation represents my own work which is done during my study for the degree of Master of Science at the University of Hong Kong, has been composed solely by myself and that it has not been submitted, in whole or in part, in any previous application for a degree. Except where states otherwise by reference or acknowledgment, the work presented is entirely my own.

Acknowledgments

To be honest, the last 10 months is a tough period for most people in Hong Kong, especially for the people who first experienced the riots and then the Covid-19 virus.

Therefore, the difficulty of finishing my dissertation is far beyond my expectation.

Fortunately, with the help of people around me, I managed to complete this dissertation in time. My words cannot fully express how I feel grateful, but I will try to express it.

I would first like to thank my supervisor Prof. Ben Kao. Although it is hard to have much communication during the special period, he gave me valuable advice and support in the working of this project and steered me in the right direction.

Also, I would like to thank Dr. K.P. Chow, the co-examiner of this project, for reviewing my work and listening my presentation.

Besides, I would like to thank Mr. Yuan Guowen for helping me complete the project during this academic year, such as providing data, helping me decide the methodologies and teaching me the ways to get related data for this project and so on. I would also like to thank Ms. Huang Seika for helping me draw some figures in this dissertation.

In addition, I would like to thank to my project groupmate Wu Jian, Zhou Han and Wang Mingxin for providing me with unfailing support and continuous encouragement throughout this project.

Finally, I must express my very profound gratitude to my families and friends who cheered me up during my hardest time. I cannot complete this dissertation without your help. Thank you very much.

Table of Content

Abstract	2
Declaration.....	3
Acknowledgments.....	4
Table of Content.....	5
1. Introduction.....	1
1.1 Background	1
1.2 Problem statement	2
2. Objectives and outcomes	3
2.1 Objectives.....	3
2.2 Outcomes.....	4
3. Literature Review.....	6
3.1 Building Knowledge graph	6
3.1.1 Comparison: existing knowledge graph construction	6
3.1.2 The features of expected knowledge graph	6
3.1.3 Ontology construction.....	7
3.1.4 Named entity recognition.....	8
3.1.5 Knowledge fusion	9
3.1.6 Relation extraction	9
3.1.7 Knowledge inference	10
3.1.8 Storage of knowledge graph	11

3.1.9	Visualization of knowledge graph	11
3.2	Building applications based on the constructed knowledge graph.....	11
4.	Methodology	13
4.1	Data preparation	13
4.2	Data processing	14
4.3	Construction of ontology	14
4.4	Named entity recognition	14
4.5	Relation extraction	15
4.6	Knowledge fusion & knowledge inference	15
4.7	Storage & Visualization of knowledge graph.....	16
4.8	Construction of Q&A system and prediction model	16
4.9	Evaluation.....	17
5.	Implementation	18
5.1	Overview	18
5.2	Constructing a knowledge graph using labeled dataset.....	18
5.2.1	Dataset.....	18
5.2.2	Data processing	19
5.2.3	Named entity recognition.....	21
5.2.4	Relation extraction	22
5.2.5	Entity linking (disambiguation)	24
5.2.6	Ontology construction.....	25
5.2.7	Visualization and storage	26

5.3	Constructing a knowledge graph using unlabeled dataset.....	34
5.3.1	Data preparation.....	34
5.3.2	Named entity recognition on existing models	35
5.3.3	Data processing.....	36
5.3.4	Refined dataset.....	40
5.3.5	Entity types	41
5.3.6	Relation types.....	43
5.3.7	Building models of named entity recognition and relation extraction for drug trafficking court cases	44
5.3.8	Introduction of SpaCy NER model.....	44
5.3.9	Introduction of Joint Named entity recognition and Relation extraction model.....	45
5.3.10	Named entity recognition and relation extraction using SpaCy NER model	47
5.3.11	Named entity recognition and relation extraction using Joint NER-RE model	48
5.3.12	Ontology construction, knowledge fusion and knowledge inference	49
5.3.13	Visualization and storage	50
5.4	Building a Q&A system based on the knowledge graphs above.....	51
5.4.1	Knowledge graph platform and user interface of Q&A system.....	52
5.4.2	Questions in high demand and classification of the questions	52

5.4.3	Transformation from plain text to program query language.....	57
5.4.4	Designation of queries	59
5.4.5	Demo of Q&A system.....	61
6.	Evaluation	62
6.1	Setup and training of SpaCy NER model.....	62
6.2	Setup and training of Joint NER-RE model	62
6.3	Evaluation methods and performance evaluation of named entity recognition and relation extraction.....	63
6.4	Case Study of Q&A system.....	66
7.	Discussion	71
7.1	Limitations.....	71
7.1.1	Limitations on knowledge graph	71
7.1.2	Limitations on Q&A system	71
7.1.3	Limitations on NER and RE Model.....	72
7.2	Future work	72
7.2.1	Future work on knowledge graph	72
7.2.2	Future work on Q&A system	73
8.	Conclusion	74
	Reference	75
	Appendix.....	80

1.Introduction

1.1 Background

Recent years, Knowledge graph has been widely used for forming knowledge in different area, including semantic search, big data analysis and intelligent recommendation. For example, Google introduced Knowledge Graph in 2012 and then used it in Google Search. Facebook also introduced graph search in 2013. Nowadays, there are some existing knowledge graphs such as DbPedia, Freebase, YAGO and NELL. These knowledge graphs contain millions of entities and billions of relationships through different knowledge fields. Besides, knowledge graphs focusing on specific fields can be useful. In legal field, Hong Kong uses Common law to make judgements. As a result, court cases are important for lawyers and courts. Ordinary people often do not have enough law knowledge, and they will concern law cases only when they are involved. For them, court's judgment and sentence are important points. Therefore, a knowledge graph including court cases and related information will be helpful for people who are professional and involved in lawsuits. Also, some applications based on the knowledge graph, such as Q&A system and sentence prediction model, will be helpful for both professionals and ordinary people.

1.2 Problem statement

One problem is that currently it is not easy to construct a court case knowledge graph for legal practitioners. A court case knowledge graph will be useful for people who are professional and involved in lawsuits, but such knowledge graph is lacking in both categories and quantities.

The other problem is that the applications based on court case knowledge graph is not satisfying. It will be helpful for legal practitioners to use the knowledge graph better if there are applications based on the knowledge graph, such as Q&A system.

2.Objectives and outcomes

Based on the background information and problem statement, the objectives and outcomes are stated below.

2.1 Objectives

Inspired by the past work in knowledge graph, this dissertation aims to achieve three goals:

1. To figure out how should the approach be to construct a court case knowledge graph, especially on the aspect of Named Entity Recognition and Relation Extraction. Two models are involved: SpaCy model and Joint multi-head selection model.
2. To construct a drug trafficking knowledge graph of court cases in Hong Kong.

The knowledge graph will contain some useful information extracted from drug trafficking court cases. One court case may consist of many nodes, such as Plaintiff, Defendant, Charge, Drugs involved and so on. Different cases can also be connected since they may have citations. The other cases that mentioned in a graph are like pointers. When checking the knowledge graph of a specific case, users can switch to the knowledge graph of another related case through the pointers. Neo4j is will be involved to achieve this goal.
3. To make applications based on the knowledge graph mentioned above so that people can use the graph to complete some tasks and get information they want. In

this dissertation, a Q&A system will be made, which can be used for information extraction of legal practitioners.

To achieve the goals listed above, there are mainly 7 steps:

1. Finding proper datasets and doing some data processing work.
2. Constructing a knowledge graph with labelled dataset.
3. Using the SpaCy library to implement a Named Entity Recognition model.
4. Implementing the Joint multi-head selection model.
5. Training, testing and evaluating the above two models.
6. Using the two models above to construct a knowledge graph with unlabeled dataset.
7. Design a Q&A system based on the knowledge graph above.

2.2 Outcomes

There are mainly two outcomes.

1. Three knowledge graphs are constructed. One knowledge graph uses the labelled data, while the other two use unlabeled data. For the knowledge graphs using unlabeled data, one is based on the SpaCy NER model, and another one is based on Joint NER-RE model. The result shows that the SpaCy NER model is more suitable than Joint NER-RE model on original court case dataset in terms of named entity recognition. However, the performance of both named entity recognition and relation extraction of Joint NER-RE model can be much better if using refined dataset, which is a slightly labelled dataset.

2. A Q&A system is constructed based on the knowledge graph. The Q&A system can receive common queries about court case and give a proper result based on the information in the knowledge graph. It can answer various types of questions, some of which are relatively easier while others are harder.

3.Literature Review

The Literature Review consists of two parts: building knowledge graph and building applications based on knowledge graph.

3.1 Building Knowledge graph

To build a knowledge graph, it is necessary to complete these tasks: construction of ontology (That is, to set some categories so that the extracted knowledge can be classified. For example, "Tom" is plaintiff, and "Paul shoot the victim" is an action, then plaintiff and action belong to ontology.), named entity recognition, relation extraction, entity linking (ambiguation) and knowledge inference.

3.1.1 Comparison: existing knowledge graph construction

Some people have already completed different kinds of knowledge graph with different texts. Liu, Peng, Yan, Li and Hao (2018) built a knowledge graph of traditional Chinese medicine and a knowledge service system based on that. Jia, Qi, Shang, Jiang and Li (2018) constructed a knowledge graph of cybersecurity.

3.1.2 The features of expected knowledge graph

As mentioned above, the knowledge graph we want focuses more on the relationships that will influence the final sentence. The texts of court cases are different with other texts because of their special terminologies. The length of sentences also makes it

hard to extract the important information. As a result, the ontology of court case knowledge graph should be built differently based on law knowledge (if it is necessary to be built manually).

The related works of this part are on mainly several aspects under the scope of natural language processing, including ontology construction, named entity recognition, entity linking (disambiguation), relation extraction and knowledge inference. Besides, it is also necessary to choose proper approaches to store and visualize the knowledge graph.

3.1.3 Ontology construction

There are two approaches to construct knowledge graph: bottom-up and top-down.

For different approaches of knowledge graph construction, the construction of ontology will also be different. In top-down approach, ontology is defined before the relation extraction. In bottom-up approach, ontology is constructed after relation extraction. The knowledge graph of Google uses a combination of bottom-up and top-down approach. The combination can both keep the quality of entity extraction in a high position (with the help of top-down approach) and find new patterns and relations during construction of knowledge graph (with the help of bottom-up approach). Ontology construction can be done manually or automatically driven by data. Nowadays most of the ontology construction is based on the data itself.

3.1.4 Named entity recognition

There are plenty of researches on NER systems. Traditional NER systems use hidden Markov model (HMM) (Zhou & Su, 2002) or conditional random field (CRF) (Finkel, Grenager, & Manning, 2005). After that, there are also some new models using K-nearest neighbors (KNN) with CRF (X. Liu, Zhang, Wei, & Zhou, 2011), or using transfer learning to make NER model generalized (Pan, Toh, & Su, 2013). Chiu and Nichols (2016) presented bidirectional LSTM-CNN NER model. LSTM with CRF is also an available choice, which is introduced by Lample, Ballesteros, Subramanian, Kawakami and Dyer (2016). On improving the word embedding, Peters et al. (2018) introduced ELMo embedding, which can further make NER better. Recently, Shen, Yun, Lipton, Kronrod and Anandkumar (2017) combined deep learning and active learning so that the NER model needs less labeled data. Bekoulis, Deleu, Demeester and Develder (2018) showed a joint model which combines BiLSTM-CRF entity recognition and sigmoid scoring relation extraction. Pham et al. (2019) showed a multi-task learning approach with contextualized word representations to improve the performance of fine-grained NER system. In addition, an open-source library SpaCy¹ introduced an NER model with bloom embeddings and residual CNNs.

¹ <https://spacy.io/>

3.1.5 Knowledge fusion

Different entities may have the same name, such as two people with the same name in different court cases. Therefore, it is necessary to make disambiguation and link text mentions to their representations in knowledge graph. Yang and Chang (2015) proposed S-MART and used it to complete the task of entity linking of tweet. ELMo can also improve the performance of entity linking (Peters et al., 2018).

3.1.6 Relation extraction

Entities form a graph by the edges that link them together. The edges need to be built by relation extraction. In the recent years, there are several kinds of relation extraction methods: statistical method, deep learning method and open information extraction.

Statistical method needs labeled data. Kernel function and feature vector are commonly used in statistical method, and there are several researches on the kernel function (Culotta & Sorensen, 2007; Zelenko et al., 2003) and features (Miao, Zhang, Zhang, & Yu, 2012). Deep learning method is suitable when data is large enough.

Nguyen and Grishman (2015) introduced a CNN-based method and further improved it by combining it and log-linear models. Open information extraction contains binary extraction and n-ary extraction. Sahu, Anand, Oruganty and Gattu (2016) made CNN to learn features automatically and improved the performance of n-ary extraction.

Zhang, Qi and Manning (2018) used graph convolutional network (GCN) to improve relation extraction with pruned dependency trees.

Recently there are also researches about joint extraction, which means combining named entity recognition and relation extraction together. Katiyar and Cardie (2016) introduced LSTM-based joint extraction, which performs better than extracting separately. Miwa and Bansal (2016) also introduced LSTMs on Sequences and Tree Structures. Zheng et al. (2017) introduced hybrid neural network, using LSTM and CNN to reduce the manual labeling and improve the performance.

3.1.7 Knowledge inference

Knowledge inference is to find implied knowledge and to populate the knowledge graph. There are mainly four kinds of approaches: rule-based approach, graph-based approach, entity and relation embedding approach and statistical relational learning (SRL) approach. Rule-based approach is a traditional approach. As for graph-based approach, Lao and Cohen (2010) proposed path ranking algorithm (PRA). Wang, Mazaitis, Lao and Cohen (2015) showed personalized PageRank based on PRA. Entity and relation embedding approach often uses deep learning, such as Neural tensor network (Socher, Chen, Manning, & Ng, 2013). Some algorithms like TransE (Bordes, Usunier, Weston, & Yakhnenko, 2013), TransH (Z. Wang, Zhang, Feng, & Chen, 2014) and TransR (Lin, Liu, Sun, Liu, & Zhu, 2015) are also helpful. There are also researches making inferences using tensor factorization, such as RESCAL (Nickel, Tresp, & Kriegel, 2011) and TRESICAL (K.-W. Chang, Yih, Yang, & Meek, 2015). SRL approach contains two main models: Markov logic network (MLN)

(Richardson & Domingos, 2006) and constrained conditional models (CCM) (M. W. Chang, Ratinov, & Roth, 2012).

3.1.8 Storage of knowledge graph

There are mainly three kinds of storages of knowledge graph: Relational database, NoSQL database and graph database (Yan, Wang, Cheng, Gao, & Zhou, 2018). Graph database currently plays an important role. Several graph databases are popular, such as Neo4j, Titan, Apache Jena and Giraph. Some databases also use combination model, such as OrientDB and Microsoft Azure Cosmos DB.

3.1.9 Visualization of knowledge graph

It is common to visualize the knowledge graph using a browser. Some browsers on the Internet are famous, such as IsaViz, GraphViz, Fenfire and RDF Gravity. Also, some graph databases contain their own browser, which can make people easily access the graph database and view it as a knowledge graph, such as Neo4j.

3.2 Building applications based on the constructed knowledge graph

Based on a knowledge graph, there are some applications available: Q&A system, prediction model and so on. Different applications have different steps to build, but

they share similar procedures. For Q&A system, it is necessary to have the following flow:

1. Get the input from users and make the program understand what users want
2. Search from the knowledge graph to find useful information
3. Give users the answer after searching

Compared with prediction model, Q&A system is easier to be built. The question can be transformed to similar relations (in form of triples) that exist in the knowledge graph. Then the answer can be retrieved by matching the relation that the question requires. The prediction model will use similar retrieval method with Q&A system, but it will also contain a further processing to make prediction.

4. Methodology

The workflow of knowledge graph and application is shown in Figure 1.

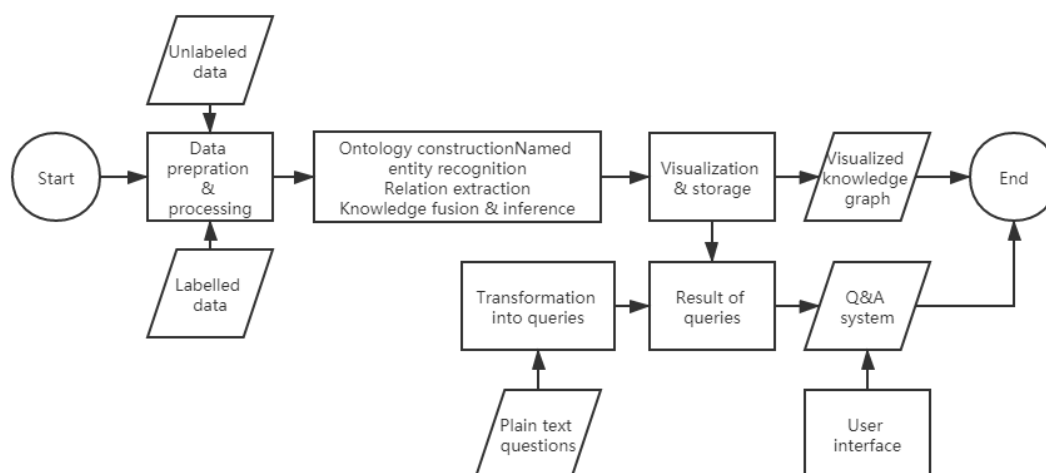


Figure 1: A brief workflow of knowledge graph and its applications.

4.1 Data preparation

HKLII² (Hong Kong Legal Information Institute) is a HK court case database which can be used to get the cases we need. We plan to download all the cases related and then start data processing. Besides, we can also use the labeled data from Department of Law of HKU. The data contains labeled data and unlabeled data, and we can use both in different ways, which will be talked in the NER section.

² <http://www.hklli.org/eng/>

4.2 Data processing

The data should be processed so that the computer program can read it. This needs some encoding works. Although court cases contain some terminologies that rarely appear in normal documents, we can still use famous word-to-vector models such as GloVe, BERT and ELMo.

4.3 Construction of ontology

The construction of ontology may contain both top-down (manually by human) and bottom-up (automatically by programming) methods. For the manual part, it needs help from people with law knowledge. If the manual part is difficult to do, then we can use the bottom-up method only since there are also many good knowledge graphs that do not require a manual ontology construction.

4.4 Named entity recognition

For this dissertation, some existing tools can be used such as Stanford named entity recognizer, OpenNLP³ and ScaPy⁴. These tools provide several pre-trained NER models. Besides, there are models that can jointly do named entity recognition and

³ <https://opennlp.apache.org/>

⁴ <https://scapy.net/>

relation extraction. Based on these technics, we can train our own model for court cases named entity recognition.

The labeled dataset, as mentioned above, is currently limited to small fields of court cases: drug trafficking. There may become an issue that the dataset is too small.

Therefore, semi-supervised learning and unsupervised learning will be considered if necessary.

4.5 Relation extraction

Similar to NER, there are also existing tools such as OpenNLP and OpenNE⁵ that provide the modern relation extraction and entity linking methods. Besides, a model combining named entity recognition and relation extraction can be used.

4.6 Knowledge fusion & knowledge inference

We plan to use OpenKE⁶ to implement TransR and RESCAL to complete the knowledge fusion and knowledge inference. If the time is not enough, we will have to simplify or skip this step since a knowledge graph can already be constructed after ontology construction, named entity recognition and relation extraction.

⁵ <https://github.com/thunlp/OpenNE>

⁶ <https://github.com/thunlp/OpenKE>

4.7 Storage & Visualization of knowledge graph

Neo4j⁷ will be used to store and visualize the knowledge graph. Neo4j is a high-performance NOSQL graph database that stores structured data on the network rather than in tables. It is a common choice for storage and visualization of knowledge graph. Besides, Neo4j contains a query language Cypher, which is similar to MySQL but with more abilities.

4.8 Construction of Q&A system and prediction model

For Q&A system, the first step is to transform user's question to a triple query. Then the answer is retrieved based on the query result. To catch the meaning of questions by the user, a simple but useful approach is to get a keyword dictionary as well as a question list. It is also a choice to use deep learning models to make machine comprehension of plain text questions. After transforming plain text questions to Cypher queries, the queries can be sent to the storage of knowledge graph to get query results as the answers and then be shown to users with a proper user interface.

⁷ <https://neo4j.com/>

4.9 Evaluation

The evaluation of named entity recognition and relation extraction can be done using precision, recall and F1-score. As for Q&A system, the performance can be measured using similar thoughts to evaluation of named entity recognition and relation extraction. A good Q&A system should be able to receive various types of questions and then give the right answers. Without manually constructing an enormous list of Q&A cases, it will be hard to get precision, recall and F1-score for to evaluate the wildness of questions that Q&A system can handle and the accuracy of the Q&A system. Therefore, we decide to use case study to complete the evaluation of Q&A system. The cases will cover various types of questions with various difficulties. The details of evaluation method will be discussed later.

5.Implementation

5.1 Overview

Currently, the implementation can be divided into three parts:

- a) Constructing a drug trafficking knowledge graph using labeled dataset
- b) Constructing a drug trafficking knowledge graph using unlabeled dataset
- c) Building a Q&A system based on the knowledge graphs above

This report will talk about these two parts one by one.

5.2 Constructing a knowledge graph using labeled dataset

The details of the construction of a court case knowledge graph using labeled dataset are stated below.

5.2.1 Dataset

With the help of Faculty of Law, the University of Hong Kong, two labeled datasets of drug trafficking cases are obtained. There are totally 6000 court cases with 64 kinds of labels. Each label represents a feature. These 64 features can be classified into several types: Offence, Case considered, Aggravating factors, Penalty, Remarks, Mitigating Factors (Guilty Plea), Mitigating Factors (Others), Defendant Background and Starting Point. The detailed feature table is shown in Appendix 1. Figure 2 shows a sample of the datasets.

[illegible]

Figure 2: A sample of the datasets.

5.2.2 Data processing

The labelled dataset needs some data processing work to make it available for the program. The labelled data needs to be fit with Neo4j API, which requires some python programming about changing the data structure and solving some encoding issues. Figure 3 shows a sample screenshot of python codes for this.

```

67 # file = open('labeled_processed.json', 'r', encoding='utf-8')
68 file = open('labeled_processed_2.json', 'r', encoding='utf-8')
69 cases = []
70 for line in file.readlines():
71     case = {}
72     dic = json.loads(line)
73     # case['content'] = dic['content']
74     # case['metadata'] = dic['metadata']
75     label = dic['annotation']
76     for i in range(len(label)):
77         for k in range(len(label[i]['label'])):
78             case[label[i]['label'][k]] = label[i]['points'][0]['text']
79     case['extras'] = dic['extras']
80     cases.append(case)
81 result = open('labeled_processed_2_neo4j_new.json', 'w', encoding='utf-8')
82 for i in range(len(cases)):
83     # result.write(str(cases[i]) + '\n')
84     json.dump(cases[i], result)
85     result.write('\n')
86 file.close()
87 result.close()

```

Figure 3: Screenshot of python codes for data processing.

Apart from the data processing above, there is another issue to be solved. The dataset we obtained is already labelled, however, it is not fully labeled and needs adding some entities. There are three useful entities (especially the last two) but not labelled: Plaintiff, Defendant and Lawyer. Entity Plaintiff contains plaintiff names, entity defendant contains defendant names, and entity Lawyer contains lawyer names. Without this, it is compulsory for us to create abstract entities to make a complete knowledge graph.

Figure 4 shows a screenshot of code of adding these entities.

```

56 loaded_json.append(json.loads(line))
57 annotation = loaded_json[-1]["annotation"]
58 loaded_json[-1]["content"] = loaded_json[-1]["content"].replace("\n", " ")
59 temp_content = loaded_json[-1]["content"].replace(", ", ", ").replace(".", ". ")
60 words = temp_content.split()
61 overlap_array = [0] * len(words)
62 entities_annotation=[0]*len(words) #letter 0, not number
63 head_array=[0]*len(words) #indexes of the last words of entities
64 relation_array = [['N'] for m in range(len(words))] #relation
65 relation_index_array = [[m] for m in range(len(words))] #index of entity w
66
67 # obtain the plaintiff, normally it will be the HKSAR
68 for w in range(len(words)):
69     if words[w] == '##':
70         entities_annotation[w + 1] = "B-PLAINTIFF"
71         overlap_array[w + 1] = 1
72         tempw = 2
73         while (w+tempw<len(words) and words[w + tempw]!='v'):
74             entities_annotation[w + tempw] = "I-PLAINTIFF"
75             overlap_array[w + tempw] = 1
76             tempw += 1
77         head_array[w+tempw-1]=1
78         break
79
80 #obtain the defendant name
81 for w in range(len(words)):
82     if words[w]=='v' and words[w+1]=='.':
83         entities_annotation[w + 2] = "B-DEFENDANT"
84         overlap_array[w + 2] = 1
85         tempw=3

```

Figure 4: Screenshot of code of adding entities of plaintiffs, defendants and lawyers.

5.2.3 Named entity recognition

For labelled dataset, the named entity recognition is unnecessary since the entities are already recognized and labelled. Still, some simple features are not labelled, such as number of defendant and name of defendant(s). For these features, a traditional approach with no deep learning technology can works well by scanning the content of court cases.

5.2.4 Relation extraction

With the labelled dataset, extraction is done by first manually defining relations and then matching the relations using labels. Triples are obtained after relation extraction.

In this task, 32 kinds of relations are defined. Figure 5 shows the matching table of relations and labels.

Entity type 1	Entity type 2	Relation
Case	Defendant	rels_defendant
Case	Offence	rels_offence
Offence	Charge	rels_charge
Charge	Defendant	rels_charge
Offence	Ordinance	rels_ordinance
Offence	Drug	rels_drug
Defendant	Drug_quantity	rels_drug_total
Offence	Drug_quantity	rels_drug_amount
Offence	Drug_value	rels_drug_value
Case	Drug_value	rels_drug_value_total
Defendant	Motive	rels_motive
Offence	Motive	rels_motive
Defendant	Role_courier	rels_courier
Offence	Role_courier	rels_courier
Defendant	Role_mastermind	rels_mastermind

Offence	Role_mastermind	rels_mastermind
Defendant	Role_operator	rels_operator
Offence	Role_operator	rels_operator
Defendant	Role_financier	rels_financier
Offence	Role_financier	rels_financier
Offence	Offence_date	rels_offence_date
Case	Offence	rels_guideline
Case	Defendant	rels_guideline
Case	Ordinance	rels_other_ordinance
Case	Case	rels_cited
Offence	Aggra_refugee	rels_refugee
Defendant	Aggra_refugee	rels_refugee
Defendant	Aggra_refugee_sent	rels_refugee_sent
Offence	Aggra_refugee_sent	rels_refugee_sent
Defendant	Aggra_total	rels_aggravating_total
Offence	Aggra_total	rels_aggravating_total
Defendant	Miti_high_court_plea	rels_plea_high_court
Offence	Miti_high_court_plea	rels_plea_high_court
Defendant	Miti_guilty_plea_sent	rels_plea_sent
Offence	Miti_guilty_plea_sent	rels_plea_sent
Defendant	Miti_total	rels_mitigating_total

Offence	Miti_total	rels_mitigating_total
Defendant	Miti_total	rels_mitigating_total_without_plea
Offence	Miti_total	rels_mitigating_total_without_plea
Defendant	Penalty	rels_penalty
Offence	Penalty	rels_penalty
Defendant	All_penalty	rels_all_penalty
Defendant	Training_center	rels_training_center
Defendant	Detention_center	rels_detention_center
Defendant	Addiction_treatment_center	rels_addiction_treatment_center
Defendant	Starting_tariff	rels_starting_tariff
Offence	Starting_tariff	rels_starting_tariff
Defendant	Starting_tariff_combined	rels_starting_tariff_combined

Figure 5: Matching table of relations and labels. The relations are directed. In an edge, the node of label 1 points to label 2.

5.2.5 Entity linking (disambiguation)

There is not too much disambiguation work to do with the labelled dataset. The noticeable work here is to make defendant name disambiguation and charge disambiguation. Both disambiguation tasks can be done during visualizing and storing the knowledge graph.

5.2.6 Ontology construction

Ontology is also manually obtained. Based on the labels, 30 different kinds of nodes are defined, which can be used for construction of the ontology. With the help of a feature table provided by Faculty of Law, the University of Hong Kong, the ontology is designed. The ontology contains several basic classes: Case, Defendant, Plaintiff, Lawyer, Offence and Penalty. Case contains ID, Guideline, Cited case and Other ordinance. Defendant contains Name and Background information. Plaintiff and Lawyer contain Name. Offence contains Charge, Ordinance, Type of drug, Quantity of drug, Value of drug, Motive, Role and Date of offence. Penalty contains Aggravating factor, Mitigating factor, Starting point and Final penalty. Figure 6 shows the ontology graph constructed.

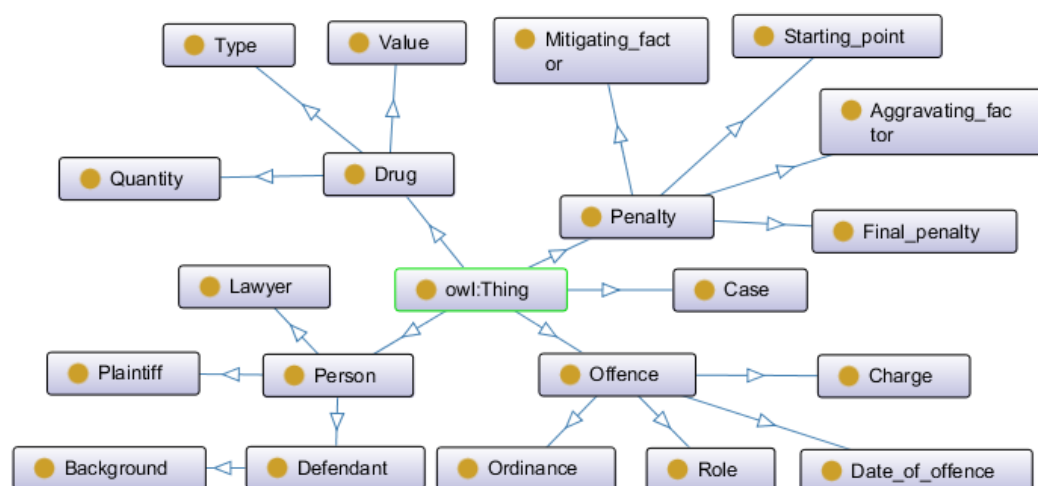


Figure 6: A rough ontology graph constructed based on feature table of court cases provided by staff of Faculty of Law, the University of Hong Kong.

5.2.7 Visualization and storage

In this dissertation, Neo4j is chosen to store and visualize the knowledge graph. To achieve this, it is necessary to set up a Neo4j database environment and use Neo4j python library with programming. To complete this, first is to set up a Java environment and initialize an empty Neo4j database. With proper configuration, the new empty Neo4j database will be able to be accessed through web browser. After that, we made some python programming to link to the database, get read the dataset and start constructing the knowledge graph. Based on the implementation above, a python class named KnowledgeGraph represents a knowledge graph in Neo4j. Each instance of KnowledgeGraph contains 30 node arrays and 32 relation arrays. Each node array contains all the nodes of a certain type. Similarly, each relation array contains all relations of a certain type (triple). The details of the structure of KnowledgeGraph are shown in Figure 7 and Figure 8. Some information, which are not chosen to be represented as nodes, are stored in the type dictionaries and then put in Neo4j as property keys. Figure 9 shows the list of property keys. Some court cases contain more than one defendant and more than one offence. As a result, two more arrays are involved to prevent the confusion within different defendants and offences. After constructing node arrays and relation arrays, it is necessary to transform the arrays to nodes and edges. A sample code of this part is shown in Figure 10. Finally, the visualized knowledge graph will be obtained after creating the nodes and edges in

Neo4j through the programming mentioned above. Figure 11 and Figure 12 reveal the sample code of Neo4j node creation and edge creation through python.

Name	Description
cases	Case
plaintiffs	Plaintiff
defendants	Defendant
lawyers	Lawyer
charges	Charge
ordinances	Ordinance
offences	Offence
drugs	Drug
motive	Motive of offence
drug_value	Value of drug
drug_quantity	Quantity of drug
role_couriers	Courier in an offence of drug trafficking
role_masterminds	Mastermind in an offence of drug trafficking
role_financier	Financier in an offence of drug trafficking

role_operators	Operator in an offence of drug trafficking
offence_date	Date of offence
penalties	Penalty
all_penalties	All the penalties
training_centers	Training center order
detention_centers	Detention center order
addiction_treatment_centers	Drug addiction treatment center order
starting_tariffs	Starting point
starting_tariffs_combined	Starting point combined
starting_aggravating_points	Starting point aggravating factors
starting_mitigating_points	Starting point mitigating factors
aggra_refugee	Refugee
aggra_refugee_sent	Sentence added for refugee
aggra_total	Total sentence added for aggravating factors
miti_guilty_plea_sent	Plead not guilty
miti_high_court_plea	High court plea
miti_high_court_plea_sent	Sentence reduced for high court plea
miti_total	Total sentence reduced for mitigating factors

Figure 7: Node arrays and the descriptions

Name	Description
rels_defendant	Has defendant
rels_offence	Has offence
rels_charge	Has charge
rels_ordinance	With ordinance
rels_drug	Carry drug
rels_drug_total	Carry drugs
rels_drug_amount	Amount of drug
rels_drug_value	Value of drug
rels_drug_value_total	Total value of drugs
rels_motive	Motive of offence
rels_courier	Role of courier
rels_mastermind	Role of mastermind
rels_operator	Role of operator
rels_financier	Role of financier
rels_offence_date	Date of offence
rels_guideline	Sentence guideline
rels_other_ordinance	Other ordinance
rels_cited	Other cited

rels_refugee	Is refugee
rels_refugee_sent	Sentence added from refugee
rels_aggravating_total	Total sentence added
rels_plea_high_court	High court guilty plea
rels_plea_sent	Sentence reduced from guilty plea
rels_mitigating_total	Total sentence reduced
rels_mitigating_total_without_plea	Total sentence reduced without plea
rels_penalty	Penalty
rels_all_penalty	All the penalties
rels_training_center	Training center order
rels_detention_center	Detention center order
rels_addiction_treatment_center	Drug addiction treatment center order
rels_starting_tariff	Sentence starting point
rels_starting_tariff_combined	Sentences combined starting point

Figure 8: Relation arrays and the descriptions

Name	Description
name	Case name
sex	Sex of defendant
nationality	Nationality of defendant

commission_age	Age at the time of commission of the offence
sentencing_age	Age at the time of sentencing
youth	Extremely young or not
relationship	Relationship/Marital status
education	Education status
family	Family background
health	Health status
previous_criminal	Previous criminal records
clear_record	Clear criminal record
drug_addict	Drug addict or not
personality	Personality e.g. stupid, single-minded, naive
occupation	Occupation
salary	Salary

Figure 9: List of property keys

```

611 elif 'STARTING_TARIFF' in key:
612     starting_tariffs.append(data_json[key])
613     for i in range(1, offence_num + 1):
614         if 'charge' + str(i) in key:
615             rels_starting_tariff.append([offence_array[i - 1], data_json[key]])
616             break
617     for i in range(1, defendant_num + 1):
618         if 'defendant' + str(i) in key:
619             rels_starting_tariff.append([defendant_array[i - 1], data_json[key]])
620             break
621
622 elif 'STARTING_TARIFF_COMBINED' in key: # SPTD1_STARTING_TARIFF_COMBINED_defendant1
623     starting_tariffs_combined.append(data_json[key])
624     for i in range(1, defendant_num + 1):
625         if 'defendant' + str(i) in key:
626             rels_starting_tariff_combined.append([defendant_array[i - 1], data_json[key]])
627             break
628

```

Figure 10: A sample code of transform the arrays to nodes and edges

```

def create_graphnodes(self):
    Cases, Defendants, Charges, Ordinances, Offences, Drugs, Motive, Drug_value, Drug_quantity, Role_couriers, Role_operators, Role_masterminds, Role_financier, \
    Offence_date, Penalties, All_penalties, Training_centers, Detention_centers, Addiction_treatment_centers, Starting_tariffs, Starting_tariffs_combined, Starting_aggravating,
    Starting_mitigating_points, Aggra_refugee, Aggra_refugee_sent, Aggra_total, Miti_high_court_plea, Miti_high_court_plea_sent, Miti_guilty_plea_sent, Miti_total, case_infos, \
    defendant2_infos, rels_defendant, rels_offence, rels_charge, rels_ordinance, rels_drug, rels_drug_total, rels_drug_amount, \
    rels_drug_value, rels_drug_value_total, rels_motive, rels_courier, rels_mastermind, rels_operator, rels_financier, rels_offence_date, rels_guideline, \
    rels_other_ordinance, rels_cited, rels_refugee, rels_refugee_sent, rels_aggravating_total, rels_plea_high_court, rels_plea_sent, rels_mitigating_total, \
    rels_mitigating_total_without_plea, rels_penalty, rels_all_penalty, \
    rels_training_center, rels_detention_center, \
    rels_addition_treatment_center, rels_starting_tariff, rels_starting_tariff_combined = self.read_nodes()
    self.create_cases_nodes(case_infos)
    # self.create_defendant_nodes(defendant1_infos)
    # self.create_defendant_nodes(defendant2_infos)

    # self.create_node('Case', Cases)
    # print(len(Cases))
    self.create_node('Defendant', Defendants)
    print(len(Defendants))
    self.create_node('Charge', Charges)
    print(len(Charges))
    self.create_node('Ordinance', Ordinances)
    print(len(Ordinances))
    self.create_node('Offence', Offences)
    print(len(Offences))
    self.create_node('Drug', Drugs)

```

Figure 11: Part of node creation code

```

def create_relationship(self, start_node, end_node, edges, rel_type, rel_name):
    count = 0

    set_edges = []
    for edge in edges:
        set_edges.append('###'.join(edge))
    all = len(set(set_edges))
    for edge in set(set_edges):
        edge = edge.split('###')
        p = edge[0].replace("'", "\'")
        q = edge[1].replace("'", "\'")
        query = "match(p:%s), (q:%s) where p.name='%s' and q.name='%s' create (p)-[rel:%s{name:'%s'}]->(q)" % (
            start_node, end_node, p, q, rel_type, rel_name)
        try:
            self.g.run(query)
            count += 1
            print(rel_type, count, all)
        except Exception as e:
            print(e)
            logging.error(e)

    return

```

Figure 12: Part of edge creation code

The knowledge graph in Neo4j can be easily checked with various options. Figure 13 shows the user interface of Neo4j database. One can view a specific type of nodes and edges by clicking the left of the user interface. Detail of a node can be obtained by clicking the node itself on the right of the user interface. Figure 14 and Figure 15 show the knowledge graph of a small dataset (5 cases) and a large dataset (1000 cases).

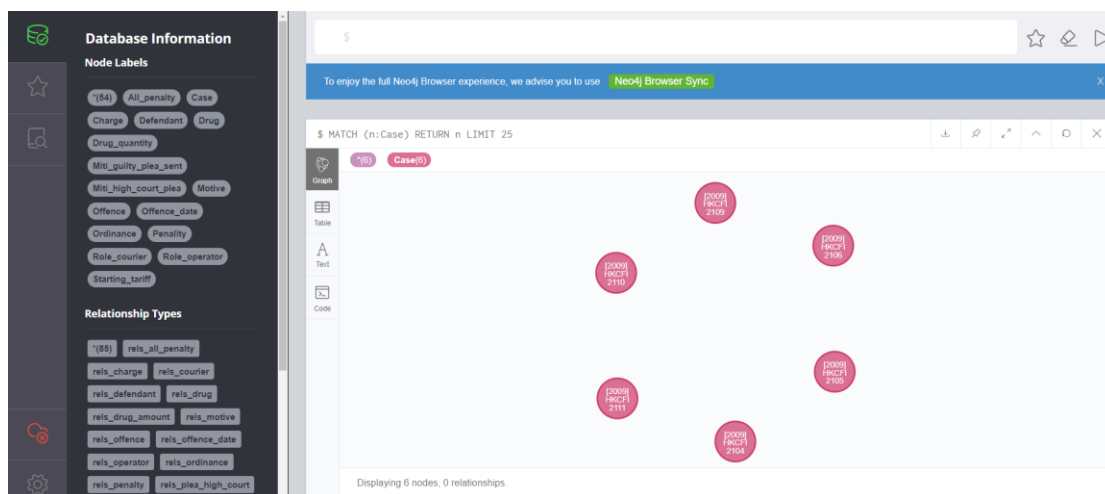


Figure 13: User interface of Neo4j

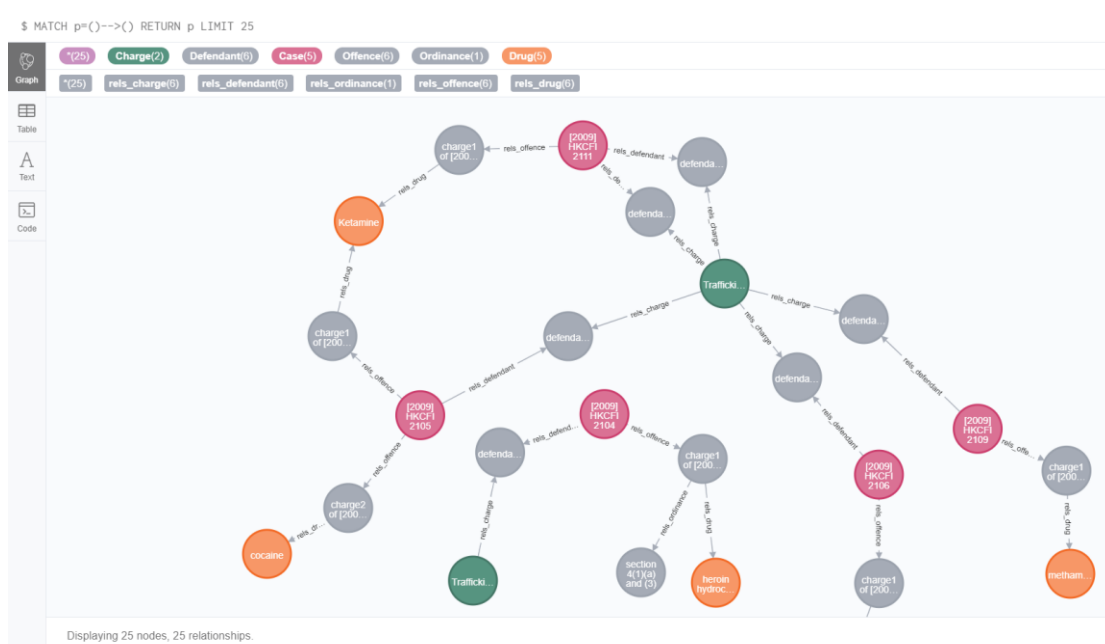


Figure 14: A knowledge graph of a small dataset, which contains 5 cases.

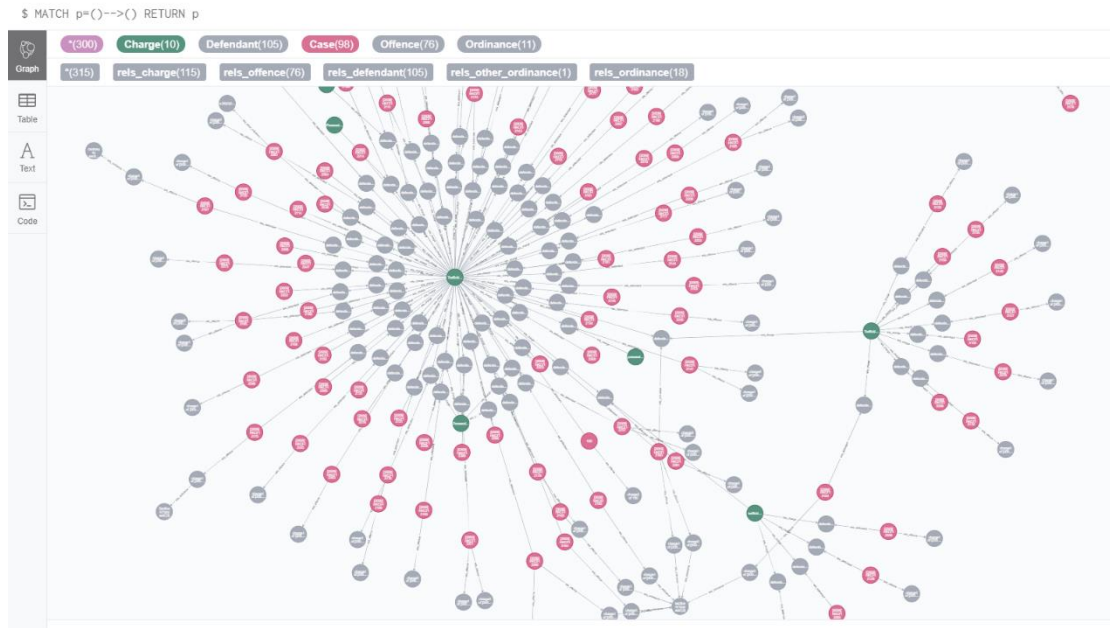


Figure 15: A knowledge graph of a large dataset (not fully expended), which contains 1000 cases.

5.3 Constructing a knowledge graph using unlabeled dataset

The construction of drug trafficking knowledge graph using unlabeled dataset involves two models for named entity recognition and relation extraction: SpaCy NER model and Joint NER-RE model. The details of construction procedures are shown below.

5.3.1 Data preparation

To train a model receiving unlabeled dataset, it is necessary to prepare labelled dataset first. So, the dataset used here is the same as previous. The unlabeled dataset is obtained by deleting the label information from labeled dataset.

5.3.2 Named entity recognition on existing models

Several existing NER tools are tested to handle unlabeled data, such as Stanford named entity recognizer and AllenNLP. The results are not satisfying. There are two reasons. First, compared with common texts, the court cases focus on different entities (such as drug and penalty). However, the existing pretrained tools are not trained for these kinds of entities. Figure 16 shows the result from AllenNLP. Also, Court cases contains some proper nouns, which are low-frequency words and makes the court case text different with common texts. As a result, training a proper named entity recognition model (Figure 17), which we will discussed more later, becomes necessary.

The particulars of the offence are that the defendant, on the 18th day of September 2014, in Hong Kong, together with Yeung Hoi Ting, unlawfully trafficked in a dangerous drug, namely 4.96 kilogrammes of a crystalline solid containing 4.86 kilogrammes of methamphetamine hydrochloride

Figure 16: Result from AllenNLP, which cannot recognize the entities of drugs.

```

train.py x
52
53 def MakeSummary(name, value):
54     """Creates a tf.Summary proto with the given name and value."""
55     summary = tf.Summary()
56     val = summary.value.add()
57     val.tag = str(name)
58     val.simple_value = float(value)
59     return summary
60
61 def make_shape(array, last_dim):
62     output = []
63     for i in array:
64         for j in i:
65             output.append(j)
66     output = np.array(output)
67     if np.shape(output)[-1] == last_dim:
68         return output
69     else:
70         print('Make Shape Error!')
71
72 def main():
73
74     print('reading word embedding')
75     word_vec = np.load(export_path + 'vec.npy')
76     print('reading training data')
77
78     instance_triple = np.load(export_path + 'train_instance_triple.npy')
79     instance_scope = np.load(export_path + 'train_instance_scope.npy')
80     train_len = np.load(export_path + 'train_len.npy')
81     train_label = np.load(export_path + 'train_label.npy')
82     train_word = np.load(export_path + 'train_word.npy')
83     train_pos1 = np.load(export_path + 'train_pos1.npy')
84     train_pos2 = np.load(export_path + 'train_pos2.npy')
85     train_mask = np.load(export_path + 'train_mask.npy')
86     train_head = np.load(export_path + 'train_head.npy')
87
88     main()

```

Figure 17: Part of the named entity recognition model.

5.3.3 Data processing

Different models receive different structures of data. For each model, there is a processing program so that the data can be read by the model. SpaCy NER model receives dictionary of entity words together with indices of the words, while Joint NER-RE receives a csv file, each row contains the index, word, entity type using BIO

labelling scheme, list of relations and list of indices of relations. Figure 18 and Figure 19 shows two screenshots about data processing of SpaCy NER model and Joint NER-RE model. Figure 20 and Figure 21 shows examples of processed data of SpaCy NER model and Joint NER-RE model.

```

        label="BACKGROUND"
    elif "STARTING" in label:
        label="STARTING"

    start=int(annotation[i]["points"][0]["start"])
    if line["content"][start]==" ":#to make sure the entity doesn'
        continue
    end = int(annotation[i]["points"][0]["end"]) + 1
    overlap=False
    for ch in range(start,end):
        if overlap_array[ch]==1:
            overlap=True
            break
    if overlap:
        continue
    for ch in range(start,end):
        overlap_array[ch]=1
    entity=(start,end,label)
    entities.append(entity)
    case=(line["content"], {"entities": entities})
    result.append(case)
# f=open("./processed_data_spacy_20200329_all", 'wb')
# f = open("./processed_data_gform_spacy_20200331_all", 'wb')
f = open("./processed_data_spacy_20200408_brief_500", 'wb')
# f = open("./processed_data_gform_spacy_20200408_brief_500", 'wb')
pickle.dump(result,f,protocol=2)
# print(result)
- - -

```

Figure 18: A screenshot showing part of work about data processing of SpaCy NER model.

```

236
237 drop_index_offset=[0 for r in range(len(words))]
238 for i in range(1,len(words)):
239     if (words[i-1]==',' or words[i-1]==',') and entities_annotation[i]=='O':
240         drop=True
241         drop_len=0
242         for k in range(len(words)-i):
243             if (words[i+k]==',' or words[i+k]==',') and entities_annotation[i+k]=='O':
244                 drop_len=k+1
245                 break
246             if entities_annotation[i+k]!='O':
247                 drop=False
248                 break
249         if drop:
250             # print(words[i:i+drop_len])
251             # print(entities_annotation[i:i+drop_len])
252             for k2 in range(drop_len):
253                 entities_annotation[i+k2]='DROP'
254             for k3 in range(len(words)-i):
255                 drop_index_offset[i+k3]+=drop_len
256
257 for arr in relation_index_array:
258     for i in range(len(arr)):
259         arr[i]=drop_index_offset[arr[i]]
260 # print(len(words))
261 drop_counter=0
262 for i in range(len(words)):
263     if entities_annotation[i]!='DROP':
264         csv_output.write(str(i-drop_index_offset[i])+"\t"+words[i]+"\t"+entities_annotation[i]
265     else:

```

Figure 19: A screenshot showing part of work about data processing of Joint NER-RE model.

TRAIN_DATA = ('' # HK SAR v. MWANIKI, CHERLINE WAITHIRA [2009] HKCFI 2104; HCCC 241/2008 (30/03/2009) \n\nHCCC241/2008\nHONG KONG SPECIAL ADMINISTRATIVE REGION\nHONG KONG COURT OF FIRST INSTANCE\nHONG KONG JUDGE Ms. 241 of 2008\n\n // '' \n\n# HK SAR v. \n\n# MWANIKI, Catherine Waithira \n\n // \n\n // \n\nBefore Deputy High Court Judge Geiser \n\nDate: 30 March 2009 at 11.52 am\nPresent: \n\nMiss Grace Chan, SPP, of the Department of Justice, for \n\nHK SAR \n\nMr Frederic C Whitehouse, instructed by Chin & Associates, for the last Accused \n\n\nOffence: Trafficking in dangerous drugs (販運危險藥物) \n\n\nTranscript of the \n\nAudio Recording \n\nThe Sentence in the above case\n\nCOURT: You, madam, pleaded guilty to an offence of trafficking in dangerous drugs contrary to section 4(1)(a) and (3) of the Dangerous Drugs Ordinance, \n\nCap. 134, Laws of Hong Kong, the particulars being that, on 8 June of last year, at Chek Lap Kok International Airport, you had in your possession 881.38 grammes of a mixture containing 613.74 grammes of heroin hydrochloride. \n\n\nThe facts which you have agreed indicate that upon your arrival at Hong Kong International Airport, you were searched and a quantity of heroin in the form of pellets were found inside the pants that you were wearing and, subsequently, when you were taken to hospital, a further 65 pellets of heroin were discharged from your body. \n\n\nYou told the police, in a record of interview, that you had been recruited by a person called 'Mike' in Mumbai to collect the drugs from Delhi and bring them to Hong Kong. In Delhi, you met another male called 'Joe' who gave you the pellets to swallow, which you did, and you then brought them to Hong Kong. For this, you were to be paid US\$5,000 for successful delivery to Hong Kong. \n\n\nI have read the letter that you have written to me and which has been handed to me by your counsel Mr Whitehouse. It makes for tragic reading. You are 48 years of age now, of clear record, a single mother, having three children of your own. You also, I understand, have the responsibility of looking after seven other children, four nephews and three nieces. These are children of your two sisters, both of whom, together with their respective husbands, have now died of Aids. \n\n\nIn addition to this, to add to your overwhelming difficulties, your shop and your home were burnt down during the post-election riots that took place in Kenya at the end of 2007. As a result of your desperate situation, you say in your letter, you were easy prey to the marauding drug bosses who brought you into their cartel. \n\n\nI am asked by your counsel to take all of these matters into account in sentencing you. I will do so and I will also give you credit for your plea of guilty to this offence, which is an indication of your genuine remorse. \n\n\nI must, however, also bear in mind that there is an aggravating feature here in that it is, quite obviously, an international element to this offence in the sense that this large quantity of drugs was brought into Hong Kong by yourself from overseas. \n\n\nThe sentencing guidelines contained in Regina v Lau Tak Ming & Others (1990) 2 HKLR 370 puts you just outside the 15 to 20-year starting point bracket, as you had in your possession over 600 grammes of heroin. I am prepared, however, to invoke a starting point that keeps you within that sentencing bracket but, undue to the aggravating feature that I have referred to, I find that I am unable to adopt a starting point of less than 20 years' imprisonment. \n\n\nI do adopt that as my starting point. I will discount that starting point by one-third to take account of your plea of guilty, coming to 13 years and 4 months' imprisonment. \n\n\nIn addition, and as an act of mercy, I will give you a further discount of 10 months' imprisonment to take account of the tragic circumstances that you found yourself in, leading you to commit this offence, arriving at a sentence of 10 years' imprisonment.' (entitles : (19945, 3967, 'P1CID1_INDIVIDUAL_PENALTY'), (3833, 3917, 'MC0CID_MITIG_OTHER'), (3787, 3810, 'MC0CID_MITIG_OTHER_SENTENCE'), (3652, 3666, 'PH0CID1_PLEA_H_UNKNOWN'), (3618, 3627, 'P5CID1_PLEA_SENTENCE'), (3508, 3530, 'P5CID1_STARTING_TARIFF'), (3078, 3287, 'OC0CID_GUIDELINE'), (2935, 2959, 'A0CID1_INTERNATIONAL'), (2804, 2811, 'M0RCID1_REMOSE'), (2449, 2473, 'OM0CID_MOTIVE'), (1968, 1974, 'BD01_RELATIONSHIP'), (1952, 1964, 'BD1D_CLEAR_RECORD'), (1928, 1943, 'B5D1_SENTENCING_AGE'), (1709, 1761, 'OR0CID1_ROLE_COURIER'), (1105, 1388, 'A0CID1_INTERNATIONAL'), (1036, 'ED06', 'OT0CI_TYPE_DRUGS'), (1018, 1032, 'OC0CI_QUANTITY_DRUGS'), (892, 898, 'OD0CI_DATE_OF_OFFENCE'), (774, 840, 'OC0CI_ORIGINE'), (699, 713, 'PH0CID1_PLEA_H_UNKNOWN'), (568, 598, 'OC0CID_CHARGE'), (268, 295, 'BFD1_FEMALE'), (41, 58, 'NC0TRUAL_CITATION'))). ('' # HK SAR v. LAI CHI WAI [2009] HKCFI 2105; HCCC 18/2009 (17 March 2009) \n\nHCCC18/2009\nHONG KONG COURT OF FIRST INSTANCE\nHONG KONG JUDGE Mr. 18 of 2009\n\n // ''

```

1 #doc 0
2 0 ## 0 ['N'] [0]
3 1 HKSAR B-PLAINTIFF ['N'] [1]
4 2 v 0 ['N'] [2]
5 3 . 0 ['N'] [3]
6 4 MWANIKI B-DEFENDANT ['N'] [4]
7 5 , I-DEFENDANT ['N'] [5]
8 6 CATHERINE I-DEFENDANT ['N'] [6]
9 7 WAITHIRA I-DEFENDANT ['motive', 'role', 'aggravating_factor', 'aggravating_factor', 'mitigating_factor', 'mitigating_factor', 'mitigating_factor'
'mitigating_factor', 'mitigating_factor', 'background', 'background', 'background', 'background', 'background', 'background', 'background', 'starting_
375, 106, 359, 371, 465, 485, 504, 44, 242, 247, 251, 259, 305, 334, 456]
10 8 [ B-NEUTRAL_CITATION ['N'] [8]
11 9 2009 I-NEUTRAL_CITATION ['N'] [9]
12 10 ] I-NEUTRAL_CITATION ['N'] [10]
13 11 HKCFI I-NEUTRAL_CITATION ['N'] [11]
14 12 2104 I-NEUTRAL_CITATION ['lawyer_of_case', 'lawyer_of_case', 'plaintiff_of_case', 'defendant_of_case', 'charge_of_case', 'case_considered'] [67]
15 13 : 0 ['N'] [13]
16 14 HCCC 0 ['N'] [14]
17 15 241/2008 0 ['N'] [15]
18 16 ( 0 ['N'] [16]
19 17 30 0 ['N'] [17]
20 18 March 0 ['N'] [18]
21 19 2009 0 ['N'] [19]
22 20 ) 0 ['N'] [20]
23 21 HCCC241/2008 0 ['N'] [21]
24 22 IN 0 ['N'] [22]
25 23 THE 0 ['N'] [23]
26 24 HIGH 0 ['N'] [24]
27 25 COURT 0 ['N'] [25]
28 26 OF 0 ['N'] [26]
29 27 THE 0 ['N'] [27]
30 28 HONG 0 ['N'] [28]
31 29 KONG 0 ['N'] [29]
32 30 SPECIAL 0 ['N'] [30]
33 31 ADMINISTRATIVE 0 ['N'] [31]
34 32 REGION 0 ['N'] [32]

```

5.3.4 Refined dataset

Some NER models, such as Joint NER-RE model, performs well when doing normal NER tasks. However, court case is different with normal text, not only because the terminologies in the text, but also the length of the text. The Joint NER-RE model is originally designed for named entity recognition and relation extraction within a sentence, not among different sentences. If we regard the court case text as a long sentence consists of sentences, then court case is longer than normal input, which weakens the performance of memory functions (BiLSTM layer) of the model. As a result, we decide to come up with a “slightly labelled” version of dataset, named “refined dataset”, to improve the performance of NER models with LSTM.

The concept of refined dataset is simple: It is a new dataset that only contains sentences that has at least one named entity from the original dataset. Assuming there is a lazy person who knows the entities but do not want to mark the named entities precisely and do not want to tell the entity category, and only roughly mark a big area that contains the named entity. Such processing can change the original dataset to refined dataset. It is not totally unlabeled, but it is also not like normal labelled dataset. If using the refined dataset to train models, then the test dataset also needs to be refined. The length of refined dataset is about 40% of the original dataset. As a result, the frequency of named entities is indirectly improved, which makes models easier to find these entities.

5.3.5 Entity types

The knowledge graph with labelled dataset has all the entities with their original entity type. However, the knowledge graph constructed with the unlabeled dataset will not have such enormous ontology. This is because there are some unpopular entities with few instances, which make the model hard to find it during named entity recognition. Therefore, some entity types will be dropped or combined. After selection, the entity types contained in the knowledge graph are: Case, Plaintiff, Defendant, Lawyer, Drug, Role, Considered, Aggravation, Mitigation, Penalty, Remark, Background and Starting. The descriptions of entity types are shown in Figure 22. Figure 23 is a screenshot showing a part of work on entity types selection and combination.

Name	Description
Case	Case ID
Plaintiff	Plaintiff of a case
Defendant	Defendant of a case
Lawyer	Lawyer of a case
Drug	Drug information
Drug_quantity	Quantity of drug
Drug_value	Value of drug
Role	Role of defendant in an offence
Offence_date	Date of offence
Considered	Cases and guidelines that related to the

	current case
Aggravation	Aggravation factors
Mitigation	Mitigation factors
Penalty	Penalty of an offence
Remark	Other features of a case
Background	Background information of a defendant
Starting	Starting tariff

Figure 22: Entity types and their descriptions for SpaCy NER model and Joint NER-RE model.

```

186         #change_label
187         #unchanged: NEUTRAL_CITATION, CHARGE, ORDINANCE MOTIVE
188         if "DRUG" in label:
189             label="DRUG"
190         elif "ROLE" in label:
191             label="ROLE"
192         elif "GUIDELINE" in label or "OTHER_ORDINANCE" in label or "CASE_CITED" in label:
193             label="CONSIDERED"
194         elif "REFUGEE" in label or "BAIL" in label or "STREET" in label or "PERSISTENT" in label or "INTERNATIONAL" in label or "AGGREV" in label:
195             label="AGGREV"
196         elif "PENALTY" in label or "CENTER" in label:
197             label="PENALTY"
198         elif "MORE" in label or "NOT_SENTENCE" in label:
199             label="REMARK"
200         elif "PLEA" in label or "HC_" in label or "DC_" in label or "REMORSE" in label or "CONSUME" in label or "CONTROLLED" in label or "TESTIMONY"
201         in label or "GOOD" in label or "MITIG" in label:
202             label="MITIG"
203         elif "MALE" in label or "HONG_KONG" in label or "FOREIGNER" in label or "COMMISSION_AGE" in label or "SENTENCING_AGE" in label or "YOUTH"
204         in label or "RELATIONSHIP" in label or "EDUCATION" in label or "FAMILY" in label or "HEALTH" in label or "PREVIOUS_CRIMINAL" in label or
205         "CLEAR_RECORD" in label or "DRUG_ADDICT" in label or "PERSONALITY" in label or "OCCUPATION" in label or "SALARY" in label:
206             label="BACKGROUND"
207         elif "STARTING" in label:
208             label="STARTING"
209
210         overlap_array[start] = 1
211         entities_annotation[start] = "B-"+label
212         for ch in range(start+1, end):
213             overlap_array[ch] = 1
214             entities_annotation[ch] = "I-"+label

```

Figure 23: A part of work on entity types selection and combination of Joint NER-RE model.

5.3.6 Relation types

Since the entity types becomes fewer compared to knowledge graph with labelled data, the relations also become fewer. Figure 24 is a list of relations among different entity types.

Entity type 1	Entity type 2	Relation
Case	Lawyer	Lawyer_of_case
Case	Plaintiff	Plaintiff_of_case
Case	Defendant	Defendant_of_case
Case	Charge	Charge_of_case
Charge	Defendant	Defendant_involved_in
Charge	Ordinance	Ordinance
Charge	Drug	Drug_of_charge
Drug	Drug_quantity	Quantity_of_drug
Drug	Drug_value	Value_of_drug
Defendant	Motive	Motive_of_defendant
Defendant	Role	Role_of_defendant
Charge	Date_of_offence	Date_of_offence
Case	Remark	Remark_of_case
Defendant	Aggravation	Aggravation_of_defendant
Defendant	Mitigation	Mitigation_of_defendant
Defendant	Background	Background_of_defendant

Defendant	Starting	Starting_point
-----------	----------	----------------

Figure 24: A list of relations among different entity types.

5.3.7 Building models of named entity recognition and relation extraction for drug trafficking court cases

To improve the performance of Named entity recognition on court cases, two models are involved: SpaCy NER model and Joint NER-RE model. As we mentioned above, SpaCy has a build-in model, which is easy for users to train and test. As for Joint NER-RE model, it aims to combine NER and RE into a single model to make the procedure simpler and improve the performance.

5.3.8 Introduction of SpaCy NER model

The SpaCy NER model is easy to use. Users can train NER models without knowing how the model is designed. However, it is still not bad to briefly know the design of the model. In short, the biggest feature of SpaCy NER Model is that it uses incremental parsing with Bloom embeddings and residual CNNs. The residual CNN layer can catch morphology features of nearby words, which improves the performance of recognition.

5.3.9 Introduction of Joint Named entity recognition and Relation extraction model

For this model, there is no existing program to use. But it is kind of convenient to make proper changes based on some codes on GitHub⁸ to implement the model.

Although the model is named “Joint”, the steps of NER and RE are completed consecutively instead of simultaneously. The model consists of 4 layers: embedding layer, BiLSTM layer, CRF layer and sigmoid scoring layer. Embedding layer is for word embedding, BiLSTM layer and CRF layer are for named entity recognition, and sigmoid scoring layer is for relation extraction.

The embedding layer of this model consists of both character-level embedding and word-level embedding. For each word, each character of this word is transformed to a vector and then be fed to a character-level BiLSTM neural network. Then the final states of two directions on BiLSTM are combined as character-level embedded vector. The word itself is also embedded using pretrained word embedding model. Finally, the embedded vectors of character-level and word-level are concatenated as the result of word embedding. Figure 25 shows the embedding layer in detail.

⁸ https://github.com/bekou/multihead_joint_entity_relation_extraction

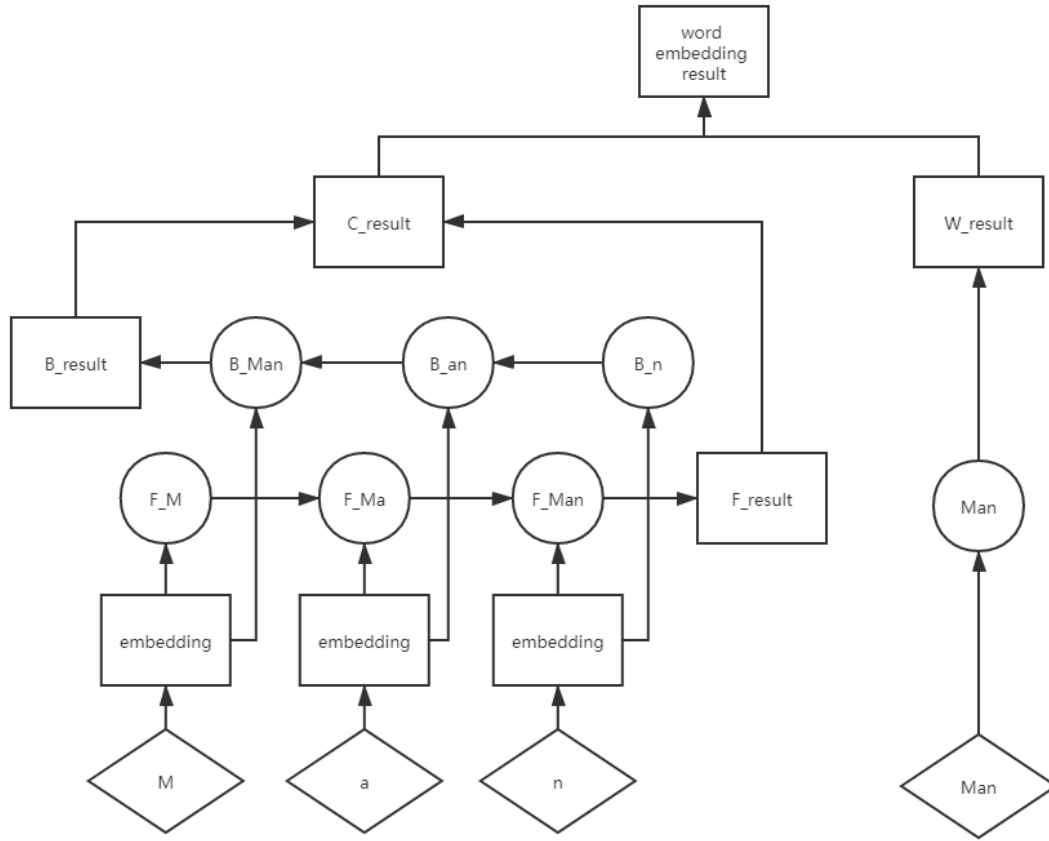


Figure 25: Embedding layer in detail with an example of word “Man”. The left part is character-level BiLSTM neural network, while the right part is pretrained word embedding model. The word embedding result is the combination of C_result (character-level embedding result) and W_result (word-level embedding result).

After word embedding, BiLSTM encoding layer is to encode the court case texts. Similar to character-level word embedding, the encoding result is a combination of vectors of two directions of BiLSTM.

After BiLSTM encoding, CRF layer is to recognize named entity in court case texts. The output of CRF is an entity type (label) using BIO labelling schema.

Next is label embedding, which make the labels obtained from CRF layer embedded to

vectors. Then a sigmoid layer is used to determine whether there is a relation between two named entities. Since a named entity may consists of multiple words, only the last word of the named entity (called “head”) is fed to the sigmoid layer. The sigmoid layer therefore is making selection among these “heads” and is a multi-head selection model. Finally, the outputs for each word consists of a named entity label and a set of tuples of the relations and entity head of relations. Figure 26 is a diagram from the essay of Joint NER-RE model, which gives an overview of the whole structure.

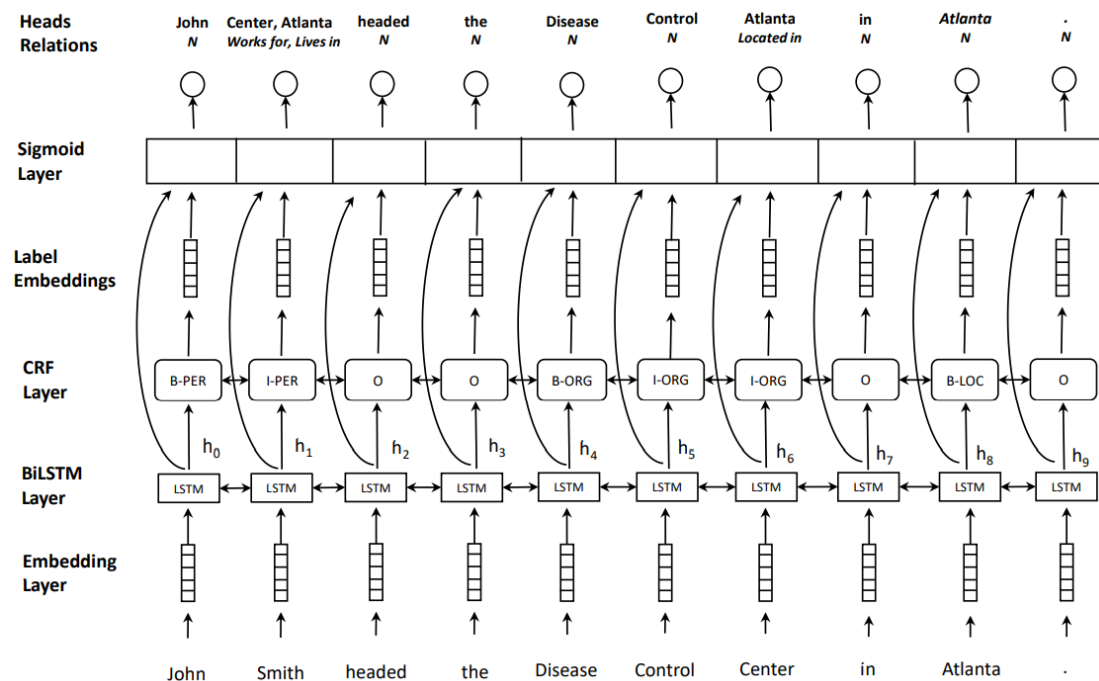


Figure 26: Overview of the Joint NER-RE model.

5.3.10 Named entity recognition and relation extraction using SpaCy NER model

Using the SpaCy library, an NER model can be trained. Figure 27 shows a part of work

of training the model. After named entity recognition, the relation extraction is completed similar to constructing knowledge graph using labelled data, which is discussed above.

```

116     # add labels
117     for _, annotations in TRAIN_DATA:
118         for ent in annotations.get("entities"):
119             ner.add_label(ent[2])
120
121     # get names of other pipes to disable them during training
122     pipe_exceptions = ["ner", "trf_wordpiecer", "trf_tok2vec"]
123     other_pipes = [pipe for pipe in nlp.pipe_names if pipe not in pipe_exceptions]
124
125     dropout = decaying(0.6, 0.2, 1e-4)
126     with nlp.disable_pipes(*other_pipes): # only train NER
127         # reset and initialize the weights randomly - but only if we're
128         # training a new model
129         prev_loss=0
130         stop_counter=0
131         if model is None:
132             optimizer=nlp.begin_training() # optimizer=nlp.begin_training()
133         else:
134             optimizer=nlp.resume_training()
135         for itn in range(n_iter):
136             currentTime = time.strftime("%Y%m%d_%H%M%S", time.localtime())
137             print(currentTime)
138             random.shuffle(TRAIN_DATA)
139             losses = 0
140             # batch up the examples using spaCy's minibatch
141             batches = minibatch(TRAIN_DATA, size=compounding(20.0, 120.0, 1.025)) #compounding originally
142             counter=0
143             for batch in batches:
144                 texts, annotations = zip(*batch)
145                 # print(texts)
146                 nlp.update(
147                     texts, # batch of texts
148                     annotations, # batch of annotations
149                     drop=next(dropout), # dropout - make it harder to memorise data, originally 0.5
150                     losses=losses

```

Figure 27: A part of work of training SpaCy NER model.

5.3.11 Named entity recognition and relation extraction using Joint NER-RE model

The model is implemented base on open source code with some amendments. Using the Joint NER-RE model, the named entity recognition and relation extraction can be

done. Figure 28 shows a part of work of training the model.

```
80
81     print("*****start training*****")
82
83     continue_training=False
84     if continue_training:
85         saver = tf.train.Saver(write_version=tf.train.SaverDef.V2)
86         module_file = tf.train.latest_checkpoint("saved_models/20200403_171540/")
87         saver.restore(sess,module_file)
88
89     for iter in range(config.nepochs+1):
90
91         model.train(train_data, operations, iter)
92         print("model.train done")
93
94         dev_score=model.evaluate(dev_data, operations, 'dev')
95         print("model.evaluate dev done")
96
97         model.evaluate(test_data, operations, 'test')
98         print("model.evaluate test done")
99
100        print("epoch "+str(iter))
101        print(time.strftime("%Y%m%d_%H%M%S", time.localtime()))
102
103        if dev_score>=best_score:
104            nepoch_no_imprv = 0
105            best_score = dev_score
106            saver = tf.train.Saver(write_version=tf.train.SaverDef.V2) # added on 20200326
107            saver.save(sess, save_path=save_path + currentTime + "/model.ckpt", global_step=iter,
108                        write_meta_graph=True) #added on 20200326
109            print("- Best dev score {} so far in {} epoch".format(dev_score, iter))
110
111        else:
112            nepoch_no_imprv += 1
113
```

Figure 28: A part of work of training Joint NER-RE model.

5.3.12 Ontology construction, knowledge fusion and knowledge inference

Compared with ontology of knowledge graph using labelled data, the ontology remains unchanged. Works about knowledge fusion and knowledge inference are skipped due to time limitation.

5.3.13 Visualization and storage

Visualization and storage here are similar to the ones of knowledge graph using labelled data. Figure 29 shows a screenshot of knowledge graph using SpaCy NER model.

Figure 30 shows a screenshot of knowledge graph using Joint NER-RE model.



Figure 29: A screenshot of knowledge graph using SpaCy NER model.

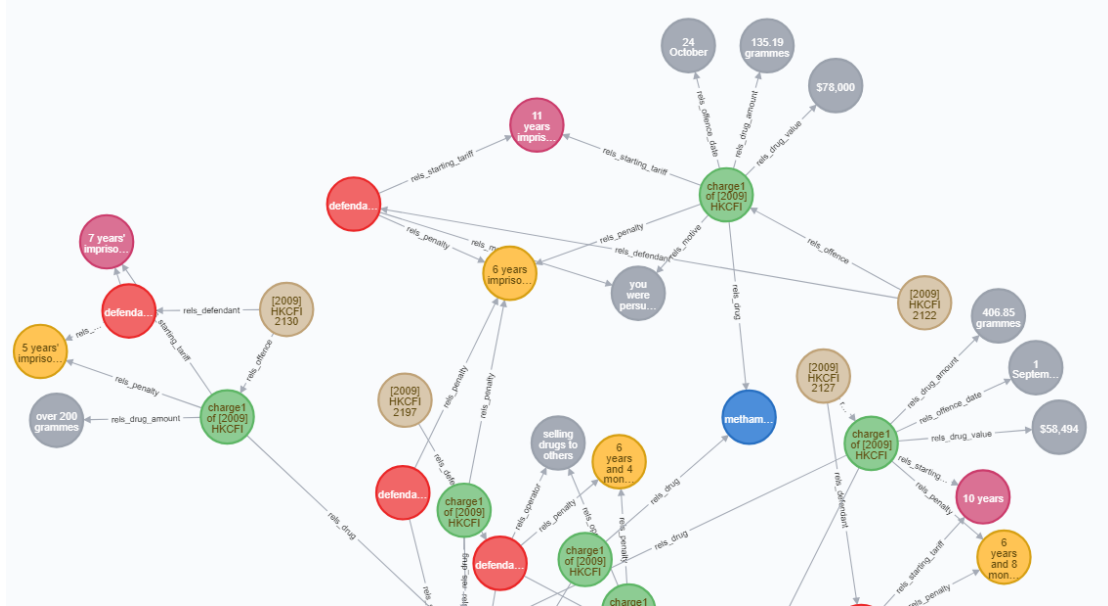


Figure 30: A screenshot of knowledge graph using Joint NER-RE model.

5.4 Building a Q&A system based on the knowledge graphs above

A Q&A system is a question-answering program which receives questions and give answers. In this dissertation, a Q&A system about court cases based on the knowledge graph is made. The design of Q&A system is inspired by an open-source project⁹.

There are several steps to build a Q&A system based on the knowledge graphs above:

1. Finding a good platform to store the knowledge graph and process the query that users make, and then making a simple user interface for users to input the questions and receive answers.

⁹ <https://github.com/liuhuanyong/QASystemOnMedicalKG>

2. Figuring out the questions that in high demand among users.
3. Using proper methods to transform users' plain texts of questions into specific program query language so that the program can know the users' demands.
4. Extracting the information that users' want to get and showing the information to them as answers.

The details of building this Q&A system is shown below.

5.4.1 Knowledge graph platform and user interface of Q&A system

When constructing the knowledge graph, we have used Neo4j for storage and visualization. In this case, we still use Neo4j. Apart from being good at storing and visualizing knowledge graph, Neo4j also contains a useful query language named Cypher. In the Q&A system, Cypher can play an important role in seeking answers given questions.

User interface can be simple if it can receive questions and show answers. In this dissertation, there is not much design of it.

5.4.2 Questions in high demand and classification of the questions

Although users can input anything they want, the questions can be classified into several categories:

1. Asking for basic information of the knowledge graph
2. Asking for information of a node given certain constraints
3. Asking for processed information of nodes given certain constraints
4. Other complicate questions

When asking for basic information of the whole knowledge graph, the answer will be related to a query which is related to the whole knowledge graph. For example: “How many labels are included in the knowledge graph?”. The question above is asking for the information of the knowledge graph itself. Figure 31 shows an example list of possible questions of querying basic information of the knowledge graph.

Example question
How many labels are included in the knowledge graph?
Help / How to use the knowledge graph?
How many cases are included in the knowledge graph?
What kind of cases is this knowledge graph about?

Figure 31: An example list of possible questions of querying basic information of the knowledge graph.

Asking for information of a node given certain condition contains different kinds of nodes. For example: “What is the name of the defendant of case [2009] HKCFI 2159?”. The question above is asking for the identity (which is the expected

information) of defendant in a specific case (which is the constraint). Similarly, other entities can also be queried often, including but not limited to Plaintiff, Lawyer, Drug type, Drug value, Date of offence, Motive and so on. Such questions can be called as “what is” questions. However, not all the “what is” questions can be handled using this approach, and the rest are discussed in the next part. In addition, some “yes or no” questions can also be answered in similar way. For example: “Is CHAN HON CHUNG a healthy person?” can be answered by showing the health entity linked to the defendant. If there is no health entity linked, then answer “I have no idea of it.”. Figure 32 shows an example list of possible questions of querying information of a node given certain constraints.

Example question
What is the drug of [2009] HKCFI 2159?
Who is the defendant of [2009] HKCFI 2159? / What is the name of the defendant of case [2009] HKCFI 2159?
What is the charge of case [2009] HKCFI 2159?
Who (What) is the plaintiff of [2009] HKCFI 2159?
Who is the lawyer of [2009] HKCFI 2159?
What is the motive of defendant in [2009] HKCFI 2159?
What is the motive of CHAN HON CHUNG?
What is the offence date of case [2009] HKCFI 2159?
What is the offence date of CHAN HON CHUNG?

What is the starting tariff of case [2009] HKCFI 2159?
What is the penalty of defendant in case [2009] HKCFI 2159?
What is the starting tariff of CHAN HON CHUNG?
What is the penalty of CHAN HON CHUNG?
Is CHAN HON CHUNG a healthy person?

Figure 32: An example list of possible questions of querying information of a node given certain constraints.

When asking for processed information of nodes given certain constraints, it is not a simple task of finding a node, but finding a processed answer made from the nodes. For example: “Is CHAN HON CHUNG a drug courier?”. To answer this question, it is necessary to figure out whether a drug courier entity is linked to CHAN HON CHUNG. Apart from “yes or no” questions, finding the most or the least is also a common pattern of question. For example: “What is the maximum quantity of cocaine involved among the cases in this knowledge graph?”. Such questions are “what is” questions but with different focus, which results in different approaches to be handled. To answer this question, it is necessary to find all the cases which are related to cocaine and then find the one with maximum drug quantity. In addition, there are also some “how many” questions such as “How many cases are related to ketamine?”, which can be solved similarly with the previous question. Figure 33 shows an

example list of possible questions of querying processed information of nodes given certain constraints.

Example question
Is CHAN HON CHUNG a drug courier?
What role does CHAN HON CHUNG play in the offence?
What is the maximum quantity of cocaine involved among the cases in this knowledge graph?
What is the minimum of penalty in the knowledge graph?
How many cases are related to ketamine?

Figure 33: An example list of possible questions of querying processed information of nodes given certain constraints.

The difficulty of answering questions is related to the complexity of program query.

Compared with the questions mentioned above, some common questions are more complicated and more difficult to answer. For example: “Is Mr Paul Leung attends more cases than Mr Caesar Lo?”. To answer this question, it is necessary to first figure out how many cases they are involved respectively and then make a comparison. Figure 34 shows an example list of possible complicate questions.

Example question
Is Mr Paul Leung attends more cases than Mr Caesar Lo?
What is the most popular drug in the knowledge graph?

What is the average of penalty for carrying 100 grammes of cocaine?

Figure 34: An example list of possible complicate questions.

5.4.3 Transformation from plain text to program query language

The plain text of questions cannot be directly understood by the program, so it is necessary to process the text and transform the text into query language that program can read. There are mainly two approaches to implement the transformation. One is designing and training a deep learning model, and the other is setting some patterns using keyword dictionaries.

So, which one is better? As for this dissertation, there are plenty of words possibly contained in question texts. However, the diversity of meanings of query words is limited. Also, keyword matching can make users easily input the question without worrying about grammar of the sentence. Users can get expected answers as long as they input the correct keywords without typos, instead of writing a whole sentence. As a result, keyword matching with dictionaries is better. To implement the Q&A system in this dissertation, there are mainly two kinds of dictionary: content word dictionary and question word dictionary.

Content word dictionary is a set generated from dataset. For each entity, there is a set that store all the words of the entity. Figure 35 shows sample sets of entity Motive and Drug_quantity of a small dataset consisting of 100 cases.

1	to earn some money	1	209.01 grammes
2	your desperate situation	2	385.68 grammes
3	unable to provide the support for his family	3	415.62 grammes
4	you were persuaded to carry the drugs on this occasion in order	4	192.56 grammes
	to satisfy all the debts you owed to the person who induced you	5	188.91 grammes
	to carry out this exercise	6	824.42 grammes
5	financial reasons	7	817 grammes
6	to earn money to support his young family in Mainland China	8	53.46 grammes
7	The facts reveal that you were selling it to feed your own habit	9	156 grammes
8	family problems, especially financial problems	10	47.73 grammes
9	in debt	11	1,236.59 grammes
10	I am prepared to accept that you had found yourself with the	12	over 200 grammes
	burden of looking after a very large number of dependants, your	13	slightly under 40 grammes
	siblings, your own children, and the children of one of your	14	168.66 grammes
	sisters who had been murdered	15	212.67 grammes
11	monetary gain	16	16.06 grammes
12	tempted to traffic in dangerous drugs in return for the promise	17	12.57 grammes
	of the payment of US\$5,000 as a result of the lorry which you	18	183.63 grammes
	drive and thereby earn your living having been burnt in a fire	19	662 grammes
	in 2008	20	30 grammes
13	troubles with your business, that you had seven children to	21	5.84 grammes

Figure 35: Two sample sets of entity Motive and Drug_quantity of a small dataset consisting of 100 cases.

Question word dictionary is a set representing a keyword that often related to a symbol of a specific kind of question. Adding more question word dictionaries can improve the accuracy of transformation. Figure 36 shows a screenshot of question word dictionaries of different keywords.

```

76 self.Plaintiff_qwds = ['plaintiff', 'plaintiffs']
77 self.Drugs_qwds = ['drugs', 'drug'] #drug involved
78 self.Drug_value_qwds = ['value']
79 self.Lawyer_qwds = ['lawyer', 'lawyers']
80 self.Motive_qwds = ['motive', 'motivation']
81 self.Offence_date_qwds = ['offence date', 'date of of
82 self.Starting_tariffs_qwds = ['starting tariffs', '
83 self.Penalties_qwds = ['penalties', 'penalty']
84
85 #role qwds
86 self.Role_select_qwds = ['carrier', 'financier', 'ma
87 self.Role_qwds = ['role']
88
89 #help words
90 self.Help_qwds = ['Help', 'help']
91 self.What_qwds = ['What', 'what']
92 self.Max_qwds = ['max', 'maximum', 'largest', 'bigges
93 self.Min_qwds = ['min', 'minimum', 'smallest']
94 self.Many_qwds = ['How many']

```

Figure 36: A screenshot of question word dictionaries of different keywords.

Every question text can be processed to determine the core of meaning using content word dictionaries and question word dictionaries. For example, “What is the maximum quantity of cocaine involved among the cases in this knowledge graph?” contains keywords “maximum”, “quantity”, “cocaine” and “knowledge graph”. “maximum”, “quantity” and “knowledge graph” be extracted by question word dictionaries, while “cocaine” can be extracted by content word dictionaries. Then the question text can be transformed to a structured query based on the keywords it contains:

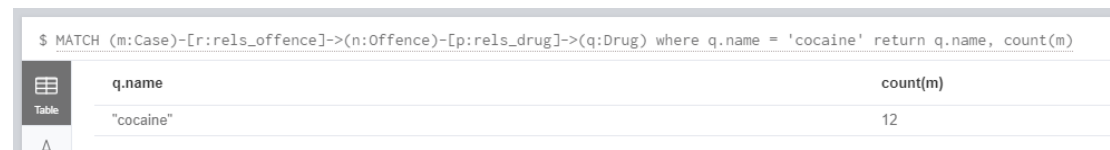
```
{'args': {'cocaine': ['Drug']}, 'question_types': ['max_drug_quantity']}
```

The ‘question_types’ represents the type of question, while ‘args’ represents the parameters of a node and its entity type. Then the structured query can be transformed into Cypher query, which will be discussed in the next part.

5.4.4 Designation of queries

Cypher, the query language of Neo4j, is similar to SQL. Most query commands and functions, such as SELECT, GROUP BY, count(), sum() and so on, are also available in Cypher. It also contains some features that are designed for matching relations. A representative query command in Cypher is MATCH RETURN. The MATCH RETURN can extract not only the nodes of knowledge graph, but also edges, triples and even chains. Figure 37 shows a Cypher query example in Neo4j.

Using the question type that generated in the previous step, the query patterns are designed respectively. Figure 38 shows a screenshot of Cypher query patterns made with respect of different question types.



The screenshot shows the Neo4j Cypher query interface. At the top, a query is entered: `$ MATCH (m:Case)-[r:rels_offence]->(n:Offence)-[p:rels_drug]->(q:Drug) where q.name = 'cocaine' return q.name, count(m)`. Below the query, a table displays the results. The table has two columns: 'q.name' and 'count(m)'. The first row shows 'cocaine' and 12.

q.name	count(m)
"cocaine"	12

Figure 37: A Cypher query example in Neo4j. The query equals to the question “How many cases are related to cocaine?”.

```

147         elif question_type == 'starting_tariff_case':
148             sql = ["MATCH (m:Case)-[r:rels_offence]->("
n:Offence)-[p:rels_starting_tariff]->(q:Starting_tariff)
where m.name = '{0}' return m.name, r.name, n.name, p.name,
q.name".format(i) for i in entities]

149
150         elif question_type == 'penalty_case':
151             sql = ["MATCH (m:Case)-[r:rels_offence]->("
n:Offence)-[p:rels_penalty]->(q:Penalty) where m.name =
'{0}' return m.name, r.name, n.name, p.name, q.name".format(i)
for i in entities]

169         elif question_type == 'role_defendant': #what role
does [defendant name] play?
170             sql = ["MATCH (m:Defendant)-[r:rels_courier
]:rels_operator]->(n) where m.name = '{0}' return
m.name, type(r), n.name".format(i) for i in entities]

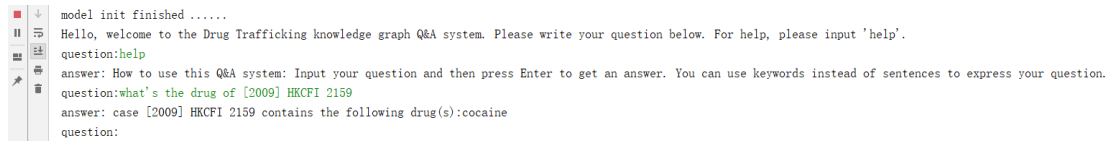
171
172         elif question_type == 'max_drug':
173             sql = ["MATCH (m:Offence)-[r:rels_drug]->("
n:Drug) where n.name = '{0}' with m MATCH (m)-[
p:rels_drug_amount]->(q:Drug_quantity) with q,m order by
q.name DESC LIMIT 1 RETURN m.name, q.name".format(i) for i in
entities]

```

Figure 38: A screenshot of Cypher query patterns made with respect of different question types.

5.4.5 Demo of Q&A system

After completing the previous steps, the Q&A system is completed. Figure 39 shows a demo of the Q&A system. More Q&A cases are discussed in the Chapter of Case Study.



```
model init finished .....
Hello, welcome to the Drug Trafficking knowledge graph Q&A system. Please write your question below. For help, please input 'help'.
question:help
answer: How to use this Q&A system: Input your question and then press Enter to get an answer. You can use keywords instead of sentences to express your question.
question:what's the drug of [2009] HKCFI 2159
answer: case [2009] HKCFI 2159 contains the following drug(s):cocaine
question:
```

Figure 39: A demo of the Q&A system. More Q&A cases are discussed in the Chapter of Case Study.

6.Evaluation

As mentioned above, there are two models involved: SpaCy NER model and Joint NER-RE model. Also, two datasets are involved: normal dataset and refined dataset. Although it is not reasonable to compare two results with different model and different dataset, the result itself is important because we can have an intuitive thinking on the result without comparing with others.

6.1 Setup and training of SpaCy NER model

In this dissertation, SpaCy 2.2.4 is used with Python 3.6. The operating system for this model is Windows 10 1909.

As for hyperparameters, the model uses a decaying dropout rate from 0.6 to 0.2 with decay coefficient of 1×10^{-4} . The batch size is also dynamic. It increases from 25 to 150 with a compounding coefficient of 1.025. The training epoch is set to 100 with early stop of 5.

6.2 Setup and training of Joint NER-RE model

Based on the existing code from Internet, the model is implemented using Python 3.5 with libraries of TensorFlow 1.12.0, Numpy 1.14.1 and Gensim 3.4.0. The operating system for this model is Windows 10 1909.

After running the model multiple times with different configurations, the Joint NER-RE model should be trained with the following hyperparameters: The character embedding size is 25, and the word embedding is initialized using 200-dimension GloVe embedding. The optimizer for training is Adam optimizer with a learning rate of $1*10^{-3}$. Also, dropout is used to prevent the training from overfitting. The hidden size of LSTM layer is 64, and for each layer of neural network, the width is also 64. The activation function among the hidden layers is tanh. The training epoch should be around 100 with early stop of 5.

6.3 Evaluation methods and performance evaluation of named entity recognition and relation extraction

Both SpaCy NER model and Joint NER-RE model are classifying data with certain labels, so they are classification models. A simple but effective way to evaluate the performance of classification models is using precision, recall and F1-score. In evaluation of NER, precision measures the performance of successfully finding a certain entity, while recall measures the performance of the accuracy of finding the right entity. For example, if there are two entities A and B, precision of A measures how well does the model marks the words of entity A currently. Recall of A measures the proportion of words with true entity A in the set of words with marked entity A. In evaluation of RE, precision measures the performance of successfully finding a certain relation, while recall measures the performance of the accuracy of finding the right

relation. F1-score is the harmonic mean of precision and recall, which can also be a standard of evaluation.

Using precision, recall and F1-score as evaluation standard, the results of different models are shown in Figure 40 and Figure 41 below.

	SpaCy NER	Joint NER-RE	Joint NER-RE with refined data
Avg. NER Precision	0.572	<0.350	0.712
Avg. NER Recall	0.611	<0.350	0.564
Avg. NER F1-score	0.591	<0.350	0.629

Figure 40: The average precision, recall and F1-score of different NER models and different datasets.

	Joint NER-RE	Joint NER-RE with refined data
Avg. RE Precision	<0.100	0.633
Avg. RE Recall	<0.100	0.575
Avg. RE F1-score	<0.100	0.603

Figure 41: The average precision, recall and F1-score of RE model with different datasets.

It is hard to determine which one is better between SpaCy NER model and Joint NER-RE model since they are using different dataset. The SpaCy uses unlabeled data, while Joint NER-RE model uses refined data. According to the results, SpaCy NER model can handle the original data better than Joint NER-RE model, but the Joint NER-RE model is also acceptable with refined data. Generally, SpaCy is still a better choice compared with Joint NER-RE. The reason of such difference may be mainly related to the difference of model design, especially the CNNs used by SpaCy. The CNN, although often used in computer vision instead of natural language processing, can also store the sequence feature of words so that to replace LSTM.

As for relation extraction, Joint NER-RE model with refined data also performs much better than it with a normal dataset, which is not beyond expectation.

For details of the results of NER and RE (shown in Appendix 2 and Appendix 3), the results show diversity among different entity types and different relations. In general, if words of an entity type have similar pattern or in similar position of the content text, then it is easier to be recognized. For example, words of entity Defendant often appears in the beginning of content text, while words of entity Penalty often contains word “year” and “punishment”. On the other hand, for those entity types which contain various words in various positions, the recognition results are not as satisfying as previous entity types. For instance, entities of Motive have different length, different index and different words among different content texts. Similarly, relations that contain similar patterns and positions are easier to be extracted (e.g. plaintiff of case), while relations that contain various patterns and positions are harder to be extracted (e.g. role of

defendant).

6.4 Case Study of Q&A system

Several questions are selected to show the performance of the system. To make the case easy to see, the questions and answers are arranged and stated in Figure 42. The answers of knowledge graph using labelled dataset are shown in the column Answer. Because of different knowledge graphs constructed using different approaches, not all questions can be answered by the knowledge graphs using unlabeled dataset. The column Unlabeled means whether the knowledge graphs using unlabeled dataset can answer the question. “Yes” means the question can be answered, “Partial” means the question can be answered with lower accuracy, and “No” means the question cannot be answered.

No.	Question	Answer	Unlabeled
1	help	How to use this Q&A system: Input your question and then press Enter to get an answer. You can use keywords instead of sentences to express your question.	Yes
2	what's the drug of [2009] HKCFI 2159	case [2009] HKCFI 2159 contains the following drug(s): cocaine	Yes
3	How many labels are included in the knowledge	Total Entity amount: 33	Yes

	graph?		
4	How many cases are included in the knowledge graph?	the amount of Case: 100	Yes
5	What kind of cases is this knowledge graph about?	This is a Q&A system of the knowledge graph of Hong Kong drug trafficking court cases.	Yes
6	defendant [2009] HKCFI 2159	case [2009] HKCFI 2159 contains the following defendant(s): SUM KA WA	Yes
7	What is the charge of case [2009] HKCFI 2159?	case [2009] HKCFI 2159 contains the following charge(s): Trafficking in dangerous drugs	Yes
8	Who (What) is the plaintiff of [2009] HKCFI 2159?	case [2009] HKCFI 2159 contains the following plaintiff(s): HKSAR	Yes
9	Who is the lawyer of [2009] HKCFI 2159?	case [2009] HKCFI 2159 contains the following lawyer(s): Mr Vincent Wong Ms Lee Mo-kit, Rebecca	Yes
10	What is the motive of defendant in [2009] HKCFI 2160?	case [2009] HKCFI 2160 contains the following motive(s): unable to provide the support for his family	Partial

11	What is the motive of defendant in [2009] HKCFI 2159?	answer not found	Partial
12	What is the motive of CHAN HON CHUNG?	answer not found	Partial
13	What is the offence date of case [2009] HKCFI 2160?	case [2009] HKCFI 2160 contains the following offence date: 21 July 2008	Yes
14	What is the offence date of CHAN HON CHUNG?	answer not found	Yes
15	What is the starting tariff of case [2009] HKCFI 2159?	case [2009] HKCFI 2159 contains the following starting tariff: 8 years and 3 months' imprisonment	Yes
16	What is the penalty of defendant in case [2009] HKCFI 2159?	case [2009] HKCFI 2159 contains the following penalty: 5½ years	Yes
17	What is the starting tariff of CHAN HON CHUNG?	defendant CHAN HON CHUNG contains the following starting tariff: 7 years' imprisonment	Yes
18	What is the penalty of CHAN HON CHUNG?	defendant CHAN HON CHUNG contains the following penalty: 56	Yes

		months.	
19	Is CHAN HON CHUNG a healthy person?	answer not found	No
20	Is CHAN HON CHUNG a drug courier?	defendant CHAN HON CHUNG is drug_courier: you had been asked by somebody you knew called Anthony to deliver the cigarette packet to a person called Ho Ma	No
21	What role does CHAN HON CHUNG play in the offence?	defendant CHAN HON CHUNG is drug_courier: you had been asked by somebody you knew called Anthony to deliver the cigarette packet to a person called Ho Ma	No
22	What is the maximum quantity of cocaine involved among the cases in this knowledge graph?	maximum of Drug cocaine: 742.13 grammes	Yes
23	What is the minimum of penalty in the knowledge graph?	minimum of Penalty: 3 years	Yes
24	How many cases are	Case of Drug ketamine is related: 48	Yes

	related to ketamine?		
25	Is Mr Paul Leung attends more cases than Mr Caesar Lo?	no	Yes
26	What is the most popular drug in the knowledge graph?	most in Drug: ketamine	Yes
27	What is the average of penalty for carrying 100 grammes of cocaine?	average of Drug cocaine: 477.9525	Partial (Wrong answer)

Figure 42: Some questions and answers of the Q&A system. Most of the questions get the right answer or receive no answer, except the last one which gives a wrong answer.

According to the questions and answers shown above, most of the questions can get right answers or “answer not found” which means the question cannot be answered. Due to time limitation, some complicate questions such as the last question cannot be correctly understood by the system and result in wrong answers.

7. Discussion

On closing of this dissertation, there still exist some limitations and some things that can be done in the future.

7.1 Limitations

There are mainly three aspects of limitations: knowledge graph, Q&A system and model design.

7.1.1 Limitations on knowledge graph

This dissertation mainly focuses on named entity recognition and relation extraction but does not have enough work on Knowledge fusion and knowledge inference. For example, some drug words have the same meaning, such as “ice” and “methamphetamine”. These two words cannot be regarded as the same in the current knowledge graph without entity linking. Besides, the dataset is limited to the labelled court cases, which is not large enough to train the models well.

7.1.2 Limitations on Q&A system

The performance of Q&A system depends on both the quality of knowledge graph and the design of transformation from plain text questions to Cypher queries. According to previous case study, there are still some questions cannot be answered correctly. This means the design and implementation of keyword matching still needs to be improved. Besides, if the keyword matching work becomes enormous, it will be also acceptable

to use deep learning models to extract the Cypher query by making classification on texts of questions.

7.1.3 Limitations on NER and RE Model

The Joint NER-RE model aims to two tasks: named entity recognition and relation extraction. The model is a multi-task model, and the final loss function contains two functions representing two tasks. For a multi-task model, it is hard to determine proper configurations to train the model well. The training effect depends on the coefficients of each loss function and how it is trained. In this project, all the coefficients of loss functions are 1, and all the variables are trained together. During training, it is hard to get all the loss functions simultaneously low because some of the variables are related to more than one loss function. For certain variables, one loss function may be near the minimum, but other functions may be far from the minimum. As a result, to train all the functions well is kind of contradictory.

7.2 Future work

Based on the limitations discussed above, some plans of future work are shown below.

7.2.1 Future work on knowledge graph

As mentioned above, we plan to implement knowledge fusion and knowledge inference to make the knowledge graph better. As for dataset, we also plan to use more data in the future, including unlabeled drug trafficking court cases from Hong Kong Legal

Information Institute (HKLII) and drug trafficking court cases of other countries using common law. Besides, some transfer learning technics can be considered so that to expand the knowledge graph to other aspect of court cases.

7.2.2 Future work on Q&A system

We plan to improve the performance of keyword matching and will also try to use deep learning approaches if necessary.

8. Conclusion

In this dissertation, two kinds knowledge graphs are constructed: knowledge graph using labeled data and knowledge graph using unlabeled data. When constructing knowledge graph using unlabeled data, two models are used and compared: SpaCy NER model and Joint NER-RE model. The result shows that normally SpaCy NER model is a better choice for named entity recognition, while the performance of Joint NER-RE model can also be improved if using refined dataset, which is a slightly labelled version of unlabeled dataset. With refined dataset, the relation extraction can also be handled using Joint NER-RE model. We also design a Q&A system, which uses the constructed knowledge graphs to receive questions and give answers. The Q&A system can receive plain texts and finding answers by transforming the texts into Cypher queries.

Due to time limitation, there are also some weaknesses yet to be improved. The construction procedure lacks knowledge fusion and knowledge inference, which makes the knowledge graphs unable to merge different entity nodes of the same meaning and unable to inference knowledge based on the existing information. The Q&A system still has some problems and the NER models are not trained well.

To sum up, we managed to constructed knowledge graphs and make a Q&A system related to them. There are still some weaknesses yet to be improved, but we believe it will become better after amelioration in the future.

Reference

- Bekoulis, G., Deleu, J., Demeester, T., & Develder, C. (2018). Joint entity recognition and relation extraction as a multi-head selection problem. *Expert Systems with Applications*, 114, 34–45. <https://doi.org/10.1016/j.eswa.2018.07.032>
- Bordes, A., Usunier, N., Weston, J., & Yakhnenko, O. (2013). TransE: Translating Embeddings for Modeling Multi-Relational Data. In *Advances in NIPS* (Vol. 26). <https://doi.org/10.1007/s13398-014-0173-7.2>
- Chang, K.-W., Yih, W., Yang, B., & Meek, C. (2015). *Typed Tensor Decomposition of Knowledge Bases for Relation Extraction*. <https://doi.org/10.3115/v1/d14-1165>
- Chang, M. W., Ratnoff, L., & Roth, D. (2012). Structured learning with constrained conditional models. *Machine Learning*, 88(3), 399–431. <https://doi.org/10.1007/s10994-012-5296-5>
- Chiu, J. P. C., & Nichols, E. (2016). Named Entity Recognition with Bidirectional LSTM-CNNs. *Transactions of the Association for Computational Linguistics*, 4, 357–370. https://doi.org/10.1162/tacl_a_00104
- Culotta, A., & Sorensen, J. (2007). *Dependency tree kernels for relation extraction*. <https://doi.org/10.3115/1218955.1219009>
- Finkel, J. R., Grenager, T., & Manning, C. (2005). *Incorporating non-local information into information extraction systems by Gibbs sampling*. <https://doi.org/10.3115/1219840.1219885>

- Jia, Y., Qi, Y., Shang, H., Jiang, R., & Li, A. (2018). A Practical Approach to Constructing a Knowledge Graph for Cybersecurity. *Engineering*, 4(1), 53–60. <https://doi.org/10.1016/j.eng.2018.01.004>
- Katiyar, A., & Cardie, C. (2016). *Investigating LSTMs for Joint Extraction of Opinion Entities and Relations*. 919–929. <https://doi.org/10.18653/v1/p16-1087>
- Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., & Dyer, C. (2016). Neural Architectures for Named Entity Recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. <https://doi.org/10.18653/v1/N16-1030>
- Lao, N., & Cohen, W. W. (2010). *Fast query execution for retrieval models based on path-constrained random walks*. <https://doi.org/10.1145/1835804.1835916>
- Lin, Y., Liu, Z., Sun, M., Liu, Y., & Zhu, X. (2015). *Learning Entity and Relation Embeddings for Knowledge Graph Completion*. Retrieved from www.aaai.org
- Liu, X., Zhang, S., Wei, F., & Zhou, M. (2011). *Recognizing Named Entities in Tweets*. Retrieved from Association for Computational Linguistics website: <http://sourceforge.net/projects/opennlp/>
- Liu, Z., Peng, E., Yan, S., Li, G., & Hao, T. (2018). T-Know: a Knowledge Graph-based Question Answering and Information Retrieval System for Traditional {C}hinese Medicine. In *Proceedings of the 27th International Conference on Computational Linguistics: System Demonstrations*. Retrieved from <https://www.aclweb.org/anthology/C18-2004>

- Miao, Q., Zhang, S., Zhang, B., & Yu, H. (2012). Extracting and Visualizing Semantic Relationships from Chinese Biomedical Text. In *Proceedings of the 26th Pacific Asia Conference on Language, Information, and Computation*. Retrieved from <http://aclweb.org/anthology/Y12-1010>
- Miwa, M., & Bansal, M. (2016). End-to-End Relation Extraction using LSTMs on Sequences and Tree Structures. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. <https://doi.org/10.18653/v1/P16-1105>
- Nguyen, T. H., & Grishman, R. (2015). *Relation Extraction: Perspective from Convolutional Neural Networks*. <https://doi.org/10.3115/v1/w15-1506>
- Nickel, M., Tresp, V., & Kriegel, H. (2011). A Three-Way Model for Collective Learning on Multi-Relational Data 28th International Conference on Machine Learning. In *Cip.IfI.Lmu.De*. Retrieved from <http://www.cip.ifi.lmu.de/~nickel/data/slides-icml2011.pdf>
- Pan, S. J., Toh, Z., & Su, J. (2013). Transfer joint embedding for cross-domain named entity recognition. *ACM Transactions on Information Systems*, 31(2), 1–27. <https://doi.org/10.1145/2457465.2457467>
- Peters, M., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., & Zettlemoyer, L. (2018). Deep Contextualized Word Representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. <https://doi.org/10.18653/v1/N18-1202>

- Pham, T.-H., Mai, K., Trung, N. M., Duc, N. T., Bolegala, D., Sasano, R., & Sekine, S. (2019). *Multi-Task Learning with Contextualized Word Representations for Extended Named Entity Recognition*. Retrieved from <http://arxiv.org/abs/1902.10118>
- Richardson, M., & Domingos, P. (2006). Markov logic networks. *Machine Learning*, 62(1-2 SPEC. ISS.), 107–136. <https://doi.org/10.1007/s10994-006-5833-1>
- Sahu, S., Anand, A., Oruganty, K., & Gattu, M. (2016). Relation extraction from clinical texts using domain invariant convolutional neural network. *Proceedings of the 15th Workshop on Biomedical Natural Language Processing*, 206–215. <https://doi.org/10.18653/v1/W16-2928>
- Shen, Y., Yun, H., Lipton, Z., Kronrod, Y., & Anandkumar, A. (2017). Deep Active Learning for Named Entity Recognition. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*. <https://doi.org/10.18653/v1/W17-2630>
- Socher, R., Chen, D., Manning, C. D., & Ng, A. Y. (2013). *Reasoning With Neural Tensor Networks for Knowledge Base Completion*. Retrieved from www.socher.org.
- Wang, W. Y., Mazaitis, K., Lao, N., & Cohen, W. W. (2015). Efficient inference and learning in a large knowledge base: Reasoning with extracted information using a locally groundable first-order probabilistic logic. *Machine Learning*, 100(1), 101–126. <https://doi.org/10.1007/s10994-015-5488-x>
- Wang, Z., Zhang, J., Feng, J., & Chen, Z. (2014). *Knowledge Graph Embedding by Translating on Hyperplanes*. Retrieved from www.aaai.org

- Yan, J., Wang, C., Cheng, W., Gao, M., & Zhou, A. (2018). A retrospective of knowledge graphs. *Frontiers of Computer Science*, Vol. 12, pp. 55–74.
<https://doi.org/10.1007/s11704-016-5228-9>
- Yang, Y., & Chang, M.-W. (2015). *S-MART: Novel Tree-based Structured Learning Algorithms Applied to Tweet Entity Linking*. 504–513.
<https://doi.org/10.3115/v1/p15-1049>
- Zelenko, D., Richardella, A., Kandola, J., Hofmann, T., Poggio, T., & Shawe-Taylor, J. (2003). Kernel Methods for Relation Extraction Chinatsu Aone CHINATSU AONE@SRA.COM. In *Journal of Machine Learning Research* (Vol. 3).
Retrieved from <http://www.jmlr.org/papers/volume3/zelenko03a/zelenko03a.pdf>
- Zhang, Y., Qi, P., & Manning, C. D. (2018). *Graph Convolution over Pruned Dependency Trees Improves Relation Extraction*. 2205–2215.
<https://doi.org/10.18653/v1/d18-1244>
- Zheng, S., Hao, Y., Lu, D., Bao, H., Xu, J., Hao, H., & Xu, B. (2017). Joint entity and relation extraction based on a hybrid neural network. *Neurocomputing*, 257, 59–66. <https://doi.org/10.1016/j.neucom.2016.12.075>
- Zhou, G., & Su, J. (2002). *Named entity recognition using an HMM-based chunk tagger*. <https://doi.org/10.3115/1073083.1073163>

Appendix

Code	Name	Description
NC	NEUTRAL_CITATION	Neutral/HKLII citation
Offence		
OCCnDn	CHARGE	Charge i.e. trafficking/trafficking in a dangerous drug
OOCn	ORDINANCE	Ordinance i.e. Dangerous Drugs Ordinance
OTCn	TYPE_DRUGS	Type of drugs e.g. heroin, cocaine, ketamine
OQCn	QUANTITY_DRUGS	Quantity/Amount of drugs [Must come in “pairs” with OTCn. If the judgment mentions the drug multiple times, label only once.]
OQTDn	QUANTITY_DRUGS_TOTAL	Quantity of drugs in total (if the judge combined all the drugs w.r.t. one defendant in sentencing)
OVCn	VALUE_DRUGS	Value of drugs

OVT	VALUE_DRUGS_TOTAL	Value of drugs in total [If the total amount of drugs of all charges is reported]
OMCnDn	MOTIVE	Motive/reason for commission of offence
ORCCnDn	ROLE_COURIER	The defendant played the role of courier in the commission of offence
ORMCnDn	ROLE_MASTERMIND	The defendant played the role of mastermind in the commission of offence
OROCnDn	ROLE_OPERATOR	The defendant played the role of operator in the commission of offence
ORFCnDn	ROLE_FINANCIER	The defendant played the role of financier in the commission of offence
ODCn	DATE_OF_OFFENCE	Date of offence [Label day, month and year. For example, if “12 July, this year” is used in the judgement,

		label “12 July” and the year that appears in date of judgement.]
Case Considered		
CGCnDn	GUIDELINE	Sentencing guidelines
CO	OTHER_ORDINANCE	Other ordinances cited
CC	CASE_CITED	Other cases cited
Aggravating Factors		
ARCnDn	REFUGEE	Commission of offences by Refugees or torture claimants
ARSCnDn	REFUGEE_SENT	Sentence added for commission of offences by refugees or torture claimants
ABCnDn	BAIL	Commission of offences on Bail
ABSCnDn	BAIL_SENT	Sentence added for ...
ASCnDn	STREET	Trafficking in drugs on the Streets
ASSCnDn	STREET_SENT	Sentence added for ...
APCnDn	PERSISTENT	Persistent offender /Re-offender
APSCnDn	PERSISTENT_SENT	Sentence added for ...
AICnDn	INTERNATIONAL	International element e.g. mainland to HK
AISCnDn	INTERNATIONAL_SENT	Sentence added for ...

AXCnDn	AGGREV_OTHER	Other aggravating factors
AXSCnDn	AGGREV_OTHER_SENT	Sentence added for ...
ATCnDn	AGGREV_TOTAL	Total sentence added for aggravating factors
Penalty		
PICnDn	INDIVIDUAL_PENALTY	Penalty for individual charges
PADn	ALL_PENALTY	Penalty for all charges [Use only when there are multiple charges]
PTDn	TRAINING_CENTER	Training Centre Order
PDDn	DETENTION_CENTER	Detention Centre Order
PMDn	ADDICTION_TREATMENT_CENTRE	Drug Addiction Treatment Centre Order
Remarks		
XC	MORE_CHARGES	More than 3 charges
XD	MORE_DEFENDANT	More than 2 defendants
XS	NOT_SENTENCE	Not related to sentencing
Mitigating Factors – Guilty Plea		
PNCnDn	PLEA_NOT_GUILTY	Plead not guilty

High Court		
(Highlight keywords that shows the stage of pleading guilty)		
PH0CnDn	HC_UNKNOWN	Stage unknown
PH1CnDn	HC_COMMITTAL	Up to committal in Magistrates' Courts
PH2CnDn	HC_BEFORE_FIX_TRIAL	After committal and up to and until trial dates are fixed
PH3CnDn	HC_AFTER_FIX_TRIAL	After trial dates are fixed but before the first date of trial
PH4CnDn	HC_FIRST_DAY	First day of trial
PH5CnDn	HC_DURING_TRIAL	During the trial
District Court		
(Highlight keywords that shows the stage of pleading guilty)		
PD0CnDn	DC_UNKNOWN	Stage unknown
PD1CnDn	DC_PLEA_DAY	At plea day
PD3CnDn	DC_AFTER_FIX_TRIAL	After trial dates are fixed but before the first date of trial
PD4CnDn	DC_FIRST_DAY	First day of trial
PD5CnDn	DC_DURING_TRIAL	During the trial
Sentence Reduced		

PSCnDn	PLEA_SENTENCE	Sentence reduced for pleading guilty (Highlight the amount of discount given)
Mitigating Factors – Others		
MRCnDn	REMORSE	Remorse
MRSCnDn	REMORSE_SENT	Sentence reduced for remorse
MSCnDn	SELF_CONSUME	Substantial amount for self-consumption
MSSCnDn	SELF_CONSUME_SENT	Sentence reduced for ...
MCCnDn	CONTROLLED	Assistance to the authorities: Controlled delivery
MCSCnDn	CONTROLLED_SENT	Sentence reduced for ...
MYCnDn	TESTIMONY	Assistance to the authorities: Give testimony in court
MYSCnDn	TESTIMONY_SENT	Sentence reduced for ...
MGCnDn	GOOD_CHARACTER	Good character

MGSCnD n	GOOD_CHARACTER_SE NT	Sentence reduced for ...
MXCnDn	MITIG_OTHER	Other mitigating factors
MXSCnD n	MITIG_OTHER_SENT	Sentence reduced for ...
MOcDnDn	MITIG_TOTAL_WO_PLE A	Total sentence reduced for mitigating factors, excluding guilty plea
MTcDnDn	MITIG_TOTAL_W_PLEA	Total sentence reduced for mitigating factors, including guilty plea
Background of the Defendant		
BMDn	MALE	Male
BFDn	FEMALE	Female
BKDn	HONG_KONG	Hong Kong Resident
BGDn	FOREIGNER	Foreigner
BCDn	COMMISSION_AGE	Age at the time of commission of the offence
BSDn	SENTENCING_AGE	Age at the time of sentencing
BYDn	YOUTH	Extreme youth
BRDn	RELATIONSHIP	Relationship/Marital status
BEDn	EDUCATION	Education
BIDn	FAMILY	Family background

BHDn	HEALTH	Health status
BPDn	PREVIOUS_CRIMINAL	Previous criminal records
BLDn	CLEAR_RECORD	Clear criminal record
BDDn	DRUG_ADDICT	Drug addict
BNDn	PERSONALITY	Personality e.g. stupid, single-minded, naive
BODn	OCCUPATION	Occupation
BADn	SALARY	Salary
Starting Point		
SPCnDn	STARTING_TARIFF	Starting point
SPTDn	STARTING_TARIFF_COMBINED	Starting point if drugs in multiple charges are combined
SPA	STARTING_AGGRAVATING	The starting point labeled includes aggravating factors (use together with SPCnDn/SPTCnDn)
SPM	STARTING_MITIGATING	The starting point labeled includes mitigating factors excluding guilty plea (use together with SPCnDn/SPTCnDn)

Appendix 1: Feature table of the labelled court case datasets.

	SpaCy NER	Joint NER-RE with refined dataset
DATE_OF_OFFENCE Precision	0.491	0.942
DATE_OF_OFFENCE Recall	0.780	0.841
DATE_OF_OFFENCE F1-score	0.603	0.889
CHARGE Precision	0.902	0.900
CHARGE Recall	0.895	0.897
CHARGE F1-score	0.899	0.898
ORDINANCE Precision	0.556	0.941
ORDINANCE Recall	0.625	0.901
ORDINANCE F1-score	0.707	0.921
DRUG Precision	0.624	0.931
DRUG Recall	0.816	0.907
DRUG F1-score	0.707	0.919
STARTING Precision	0.547	0.929
STARTING Recall	0.769	0.917

STARTING F1-score	0.639	0.923
CONSIDERED Precision	0.383	0.620
CONSIDERED Recall	0.416	0.553
CONSIDERED F1-score	0.400	0.585
CASE Precision	0.987	0.906
CASE Recall	1.000	0.969
CASE F1-score	0.993	0.936
MOTIVE Precision	0.294	0.583
MOTIVE Recall	0.172	0.556
MOTIVE F1-score	0.214	0.569
ROLE Precision	0.223	0.485
ROLE Recall	0.044	0.386
ROLE F1-score	0.074	0.430
AGGREV Precision	0.558	
AGGREV Recall	0.341	
AGGREV F1-score	0.423	
MITIG Precision	0.448	
MITIG Recall	0.422	
MITIG F1-score	0.435	
BACKGROUND Precision	0.489	

BACKGROUND Recall	0.445	
BACKGROUND F1-score	0.466	

Appendix 2: Details of evaluation results of named entity recognition models.

	Joint NER-RE model with refined dataset
case_considered Precision	0.618
case_considered Recall	0.370
case_considered F1-score	0.463
charge_involved_drug Precision	0.414
charge_involved_drug Recall	0.358
charge_involved_drug F1-score	0.384
charge_of_case Precision	0.434
charge_of_case Recall	0.429
charge_of_case F1-score	0.431
date_of_offence Precision	0.349
date_of_offence Recall	0.262
date_of_offence F1-score	0.300
defendant_involved_in Precision	0.452
defendant_involved_in Recall	0.421
defendant_involved_in F1-score	0.436
defendant_of_case Precision	0.906

defendant_of_case Recall	0.967
defendant_of_case F1-score	0.935
lawyer_of_case Precision	0.914
lawyer_of_case Recall	0.958
lawyer_of_case F1-score	0.936
Motive Precision	0.580
Motive Recall	0.519
Motive F1-score	0.548
Ordinance Precision	0.285
Ordinance Recall	0.208
Ordinance F1-score	0.240
plaintiff_of_case Precision	0.906
plaintiff_of_case Recall	0.969
plaintiff_of_case F1-score	0.937
Role Precision	0.475
Role Recall	0.363
Role F1-score	0.412
starting_point Precision	0.946
starting_point Recall	0.852
starting_point F1-score	0.897

Appendix 3: Details of evaluation results of relation extraction model.