

Aalto University
School of Science
Master's Programme in Computer, Communication and Information Sciences

Pin Tian

Extracting Measured Properties for Numerical Data with SciBERT model and Question Answering

Master's Thesis
Espoo, Dec 17, 2021

Supervisor: Professor Mariet Theune, University of Twente
Professor Maurice van Keulen, University of Twente
Professor Mika P. Nieminen, Aalto University
Advisor: dr. Marius Doornenbal, Elsevier B.V.

Author:	Pin Tian	
Title:	Extracting Measured Properties for Numerical Data with SciBERT model and Question Answering	
Date:	Dec 17, 2021	Pages: 50
Major:	Human-Computer Interaction and Design	Code: SCI3020
Supervisor:	Professor Mariet Theune Professor Maurice van Keulen Professor Mika P. Nieminen	
Advisor:	dr. Marius Doornenbal, Elsevier B.V.	
<p>Quantity is a measurement (e.g. 18g), which usually consists numerical data and units. Quantity is crucial and very frequently mentioned in scientific publications. At Elsevier, there are good solutions for searching quantities or their components, like numerical data or units. In these products, you can search all the papers which contain, for example, "$< 2mm$", in their full text. However, what the observed measurements represent is still unclear. For example, when you search "$< 2mm$", does "$2mm$" represent the length or diameter of a tube? The ambiguity causes many irrelevant results in their search engines. The property behind the quantity is called measured property. To solve this ambiguity and enhance the search capability, extracting what measured property a quantity represents is the next step of Elsevier. When users can search both quantity and measured property at the same time, they can definitely get more accurate results.</p> <p>In this paper, we propose a Question-Answering architecture for joint measured property and relationship extraction based on the numerical data extraction model. The Question-Answering architecture enables a named entity recognition model to extract entity and relationship jointly. We train a SciBERT model to extract quantity in the corpus and another SciBERT model to extract corresponding measured property for each quantity. Meanwhile, we annotate a dataset with the publications from the engineering domain, MeasPro, for our model training. It proves that our approach has excellent accuracy and it is better than the state-of-art models on MeasEval dataset.</p>		
Keywords:	NLP, entity extraction, relationship extraction, measurement, scientific publications	
Language:	English	

Acknowledgements

I would like to thank Dr. Mariet Theune for her supervision and guidance during this project. Thank you for guiding me to think about a problem from the research perspective, for improving my understanding of academic writing, and for handling all the operational issues to ensure the progress was on track.

I also want to thank Dr. Maurice van Keulen for his input from the middle of the project. Your input gave me brave to switch my original plan, the MRT method, to the QA method. It made my research more smooth during the rest of the study.

I am also very grateful to Dr. Marius Doornenbal, and other Elsevier colleagues, for your incredibly quick and accurate communication, for providing me with access to very helpful resources, for the annotation of the new dataset, for contributing your expertise to this project, and for the freedom of finding my way in planning and executing the work.

Espoo, Dec 17, 2021

Pin Tian

Contents

1	Introduction	6
2	Related Work	10
2.1	Entity Extraction	10
2.1.1	Rule-based Approaches	10
2.1.2	Unsupervised Learning Approaches	11
2.1.3	Feature-based Supervised Learning Approaches	12
2.1.4	Deep Learning Approaches	12
2.1.4.1	CNN and RNN Encoder	13
2.1.4.2	Tag Decoder	14
2.1.4.3	Transformer Models	14
2.2	Relationship Extraction	14
2.2.1	Ontology-based Approaches	14
2.2.2	Rule-based Approaches	17
2.2.3	Unsupervised Learning Approaches	17
2.2.4	Feature-based Supervised Learning Approaches	17
2.2.5	Deep Learning Approaches	17
2.3	The Interaction between Entity Extraction and Relationship Extraction	18
2.3.1	Pipeline Approaches	18
2.3.2	Joint minimum risk training (MRT) Approaches	18
2.3.3	Parameter Sharing Approaches	19
2.3.4	Joint Extraction Approaches	20
2.3.4.1	Special Symbol Insertion Method	20
2.3.4.2	Question Answering	20
2.3.5	Conclusion of Related Work	21
3	Method	23
3.1	Project Framework	23
3.2	Preprocessing	24
3.3	Extraction model	24

3.4	Question Answering	26
3.5	Model Training for Question-Answering	27
4	Dataset	29
4.1	MeasEval	29
4.2	MeasPro: A newly annotated dataset	30
4.2.1	Data Source	31
4.2.2	Annotation Setting	31
4.2.3	Evaluation of the Annotations	32
5	Experiments of Measured Property Extraction	33
5.1	The State-of-art Models trained on MeasEval	33
5.2	Comparison of Measured Property Extraction Models	34
5.3	How the Input Length Influences the Score	37
5.4	How the Parameters Influence the Score	38
5.5	Text Classification before Entity Extraction	41
6	Result of Full Implementation	44

Chapter 1

Introduction

There is a large number of quantities in scientific publications. The quantity usually represents a measured property of a measured entity. For example, in the following sentence,

An efficient red-emitting LEC fabricated on a glass substrate achieved a current efficiency (ηC) of 7.18 cd/A and an external quantum efficiency (η_{ext}) of 9.32% [24].

7.18 cd/A and 9.32% are numerical data, which is often called quantity. It consists of two parts, value (e.g. 7.18) and unit (cd/A). For quantity extraction, there are good solutions at Elsevier. They have two products which allow users search units and value ranges (e.g. $< 0.8 cd/A$). They are Compendex¹, an engineering literature database, and Knovel², a toolkit providing trusted engineer data and insights. The solutions are based on regular expressions and have a good accuracy. In short, the existing models can only extract one kind of entities, quantity, as below:

An efficient red-emitting LEC fabricated on a glass substrate achieved a current efficiency (ηC) of [7.18 cd/A](Quantity) and an external quantum efficiency (η_{ext}) of [9.32%](Quantity).

¹Ei Compendex, <https://www.elsevier.com/solutions/engineering-village/content/compendex>

²Knovel, <https://www.elsevier.com/solutions/knovel-engineering-information>

However, the current solutions can not solve the ambiguity of what property a quantity represents. For example, "5m", 5 meters, can represent circumference or radius in different context. In such case, when a user wants to search for a machine element within 5m circumference, the search engines may provide irrelevant results, a machine element within 5m radius.

To solve this problem, Elsevier plans to apply NLP technology to extract the context of the quantity to enhance the capability of its search engines. Measured property indicates what the quantity measures. Extracting measured property can enable users to filter the search result according to the context of a quantity. It can enhance the capability of search engines and provide more relevant results.

There are two measured properties in the case above. "current efficiency" is the measured property of the quantity "7.18 cd/A". Meanwhile, "external quantum efficiency" is the measured property of the quantity "9.32%". They are the new targeted entity that we should extract.

An efficient red-emitting LEC fabricated on a glass substrate achieved a [current efficiency] (Measured Property) (ηC) of [7.18 cd/A] (Quantity) and an [external quantum efficiency] (Measured Property) (η_{ext}) of [9.32%] (Quantity).

In the general cases, there might be multiple measured properties and quantities in a given corpus. How to link measured property and quantity is another crucial problem. A relationship model can build relationship between each property to its corresponding quantity like below:

Relationships:

[current efficiency] (Measured Property) <---> [7.18 cd/A] (Quantity)

[external quantum efficiency] (Measured Property) <---> [9.32%] (Quantity)

In summary, our research problem is how to extract what measured property a quantity represents in scientific publications. This research question consists of the following three sub-questions: (1). How to extract quantity spans in scientific texts; (2). How to extract measured property spans in scientific texts; (3). How to build a relationship between a quantity and a measured property which the quantity represents.

To extract measured property and quantity in pair, we apply the Question-Answering method [16]. Question-Answering is a method that transforms the entity-relation classification task into a QA task by appending a question before the text that indicates the relationship. For the given example, the QA method fills one of the quantities into a question template, and inserts the full question, "What is the measured property of 7.18 cd/A ?" into the beginning of the original sentence. The corresponding entity span and the new sentence will be used as the training data like below.

Sentence:

What is the measured property of 7.18 cd/A ? An efficient red-emitting LEC fabricated on a glass substrate achieved a current efficiency (η_C) of 7.18 cd/A and an external quantum efficiency (η_{ext}) of 9.32%.

Entity:

[current efficiency] (Measured Property)

In this way, a single named entity recognition model can learn the target entity and relation jointly. This architecture reduces the complexity of modeling. In our thesis, we train a SciBERT [2] model to extract quantity in the corpus and another SciBERT model to extract corresponding measured property for each quantity with QA method.

Our solution aims to extract the measured property in engineering domain scientific papers. There is no available dataset for our special requirements. Therefore, we annotate a dataset with the publications from the engineering domain, MeasPro, based on the Elsevier's literature database, for our model training and evaluation.

The contributions of our work are summarized as follows: (1) We annotate a new dataset, MeasPro, in the engineering domain for scientific measurements and properties. (2) Our model obtains 0.811 F1-score with Question-Answering and 0.770 F1-score without Question-Answering using the MeasPro dataset. (3) Our model obtains 0.617 F1-score using the SemEval-2010 Task 8 dataset, MeasEval, and outperforms existing state-of-the-art models.

The rest of this paper is organized as follows: Chapter 2 details related work. Then we describe our method, framework of the full solution, and model selection in Chapter 3. The public dataset, MeasEval, new dataset, MeasPro and MeasPro's annotation settings are described in Chapter 4. We show our experimental results on MeasEval and the comparison with the state-of-art models in Chapter 5. The final implementation on MeasPro and

its results are shown in Chapter 6. Finally, we conclude this thesis in Section 7.

Chapter 2

Related Work

This section provides a literature review and study of entity extraction and relationship extraction. Entity extraction and relationship extraction are both well-studied topics. There are many approaches in both of them. The sections, 2.1 and 2.2, summarize the entity extraction and relationship extraction respectively. Besides, we also summarize the different interactions between entity extraction models and relationship extraction models in section 2.3.

This full panorama of the-state-of-art study provides exhaustive comparisons of different methods. It gives guidance and inspiration to our solution.

2.1 Entity Extraction

Entity Extraction is the task to extract a set of entities $\varepsilon_1, \dots, \varepsilon_m$ from a given sentence: $s = \omega_1, \dots, \omega_n$, where an entity ε is a sequence of words labelling with predefined semantic types (e.g. person, organization, etc.). [18]

2.1.1 Rule-based Approaches

Rule-based entity extraction systems rely on hand-crafted semantic and syntactic rules to recognize entities. Because the rules are usually summarized based on domain-specific knowledge, the systems cannot be transferred to other domains. Meanwhile, high precision and low recall are often observed from these systems due to the incomplete rules [15].

For example, in the earth science domain, there is a rule-based solution to extract quantities and measured properties based on the semantic structure [11]. The system is called Marve. The pattern was developed heuristically

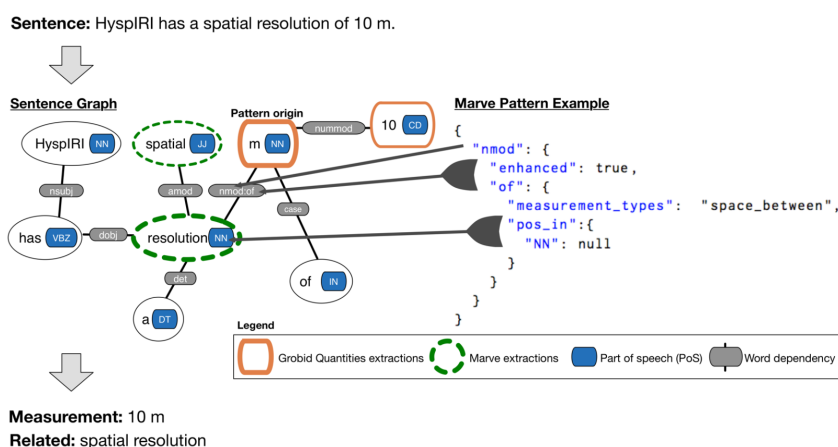


Figure 2.1: Marve context extraction based on word dependencies and PoS tags[11]

by analyzing scientific literature. It uses word dependencies and Part-of-Speech (Pos) tags to locate the entities (see Fig.2.1). High precision and low recall are observed from their experiments. The performance on earth science papers is 0.804 precision, 0.662 recall, and 0.726 F1-score. Moreover, how the system works on other domains is still unknown.

2.1.2 Unsupervised Learning Approaches

Unsupervised Learning Approaches can be used to extract entities without any given groups to learn. Jing Li et al. summarize the state-of-art unsupervised learning approaches[15]. Clustering is the typical approach to gather data based on context similarity. However, the output of clustering is many groups without any tags. If we want to apply it to entity extraction, we need more processing after the clustering.

For the unlabelled data processed by clustering, a named entity classification can be used for tagging. Named entity classification can be done by extracting domain-independent patterns to generate candidate facts. KNOWITALL [6] is a model which utilizes generic extraction patterns to extract label. For example, a generic pattern can be "We provide tours to cities such as Paris, London, and Berlin". KNOWITALL can generate an extraction rule by substituting the class name "City" into a generic rule template. The rule looks for the phrase "cities such as" and extracts the proper nouns following that phrase as instances of the class City. In this way, the labels are

based on the context. The extraction rules are generated from the semantic structure automatically rather than hand-made rules. Therefore the rules are not predefined by humans, and the tagging results are unpredictable.

Besides, in a specific domain, the entity extraction system can leverage terminologies. An external dictionary of terminologies defines the vocabularies of entities. For example, Zhang and Elhadad present an unsupervised entity extraction approach through external terminologies in biomedical domain [25]. This approach relies on exhaustive terminologies, so that it is hard to apply it in a general entity extraction task across many different domains.

Our entity extraction has predefined semantic types, measured property and quantity. Therefore, unsupervised entity extraction approaches do not apply for our case.

2.1.3 Feature-based Supervised Learning Approaches

In supervised learning, entity extraction is a multi-class classification or sequence labeling task. Models are trained on an annotated corpus. Many traditional machine learning algorithms have been applied in supervised entity extraction, including Hidden Markov Models (HMM), Decision Trees, Support Vector Machines (SVM), Conditional Random Fields (CRF) etc. [15]. Feature engineering is crucial and critical for such algorithms. Features can be extracted from word level (e.g. part-of-speech tag), list lookup features (e.g. Wikipedia gazetteer), to document and corpus features level (e.g. local syntax and multiple occurrences).

There is one open source model for extracting quantities (e.g. 2m , $7 \cdot 10^5\text{g}$, or $10 \pm 2\text{km}^2$), which is called Grobid Quantities[7]. It can also transform quantities into structured data with value (e.g. 2, $7 \cdot 10^5$, or 10 ± 2) and raw units (e.g. m, g, km^2). Besides, the raw units can be normalized into the base units (e.g grams to kg, Celsius to Kelvin, etc.) defined by the International System of Units (SI). It is trained using CRF (Conditional Random Field) algorithm.

CRFs are undirected statistical graphical models that compute a conditional probability of label sequences by a given a particular observation sequence [22]. CRFs are also used as tag decoder layer in deep learning approaches (see section 2.1.4).

2.1.4 Deep Learning Approaches

In the recent years, deep learning methods have dominated the field of name

entity recognition. Compared to feature-based supervised learning, deep learning has unique advantages. Deep learning uses multiple processing layers to learn hidden features automatically. It enables to train a model with raw data and discover representations automatically without effort on designing entity extraction features [15].

In a deep learning entity extraction system, the input can be either character-level or word-level representation. Character-level representation means using characters in words as basic input. Word-level representation means using single words as basic input. It has been proved that character-level representation is useful for exploiting explicit sub-word-level information such as prefix, suffix, and morpheme-level regularities [15]. Of course, the complexity of computation of character-level representation is much more than word-level representation. Character-level is often used to extract internal units in quantity spans [1, 8]. However, it is not our problem. In our case, word-level information should be sufficient for our model training.

A typical word-level deep learning entity extraction system consists of two parts: context encoder and tag decoder. Context encoder is to capture the context dependencies. The common models of context encoder are CNN (Convolutional Neural Networks) and RNN (Recurrent Neural Networks). Tag decoder is following the context encoder. It predicts labels of words in the given sequence based on the encoded context from context encoder. The common architectures of tag decoder are Multi-Layer Perceptron (MLP) + Softmax layer and conditional random fields (CRFs) [15].

2.1.4.1 CNN and RNN Encoder

Collobert et al. [4] proposed a sentence based CNN shown in Fig.2.3. Each word is embedded to an N-dimension vector. Then a convolution layer produces local features around each word. After that, local features are combined into global features. Finally, the tag decoder computes scores for all possible tags based on the global features.

RNN has different variants, such as gated recurrent unit (GRU) and long-short term memory (LSTM). A typical RNN based context encoder proposed by Li et al.[15] is shown in Fig.2.3. Compared to the CNN model, a bidirectional LSTM makes full use of evidence from the whole input sentence including past and future information in sequential data. However, RNNs take more time to train. Experiments show that CNNs achieve 14-20x test-time speedups compared to RNNs [15].

2.1.4.2 Tag Decoder

Tag decoder is a classifier which can classify the input tokens into predefined labels. The common architectures are Multi-Layer Perceptron (MLP) + Softmax layer and conditional random fields (CRFs) [15]. Their architectures are shown in Fig.2.5.

The Perceptron + Softmax layer shown in Fig.2.2 casts the sequence labeling task into a multi-class classification problem. The tag for each word is independently predicted without taking into account its neighbors [15].

CRFs is a feature-based machine learning approach (see section 2.1.3). CRFs are the most common choice for tag decoder of many deep learning based context encoder. However, CRFs cannot make full use of segment-level information because their properties of segments cannot be fully encoded with word-level representations [15].

2.1.4.3 Transformer Models

Besides, there are transformer models dispensing with context encoder and tag decoder architecture. The most common one is BERT (Bidirectional Encoder Representations from Transformers). Its architecture is shown in Fig.2.4. BERT is a pre-trained deep bidirectional Transformer by joining both left and right context in all layers to build basic blocks for both encoder and decoder. A separated tag decoder is unnecessary. It has been proved that transformers require significantly less time to train while being superior in quality [15].

For NLP tasks in the scientific domain, SciBERT[2] is a popular pre-trained model, which has its own vocabulary (scivocab) that's built to best match scientific texts. It allows users to train their own model using its vocabulary.

2.2 Relationship Extraction

Relationship extraction is built on entity extraction. A relationship can be described as a triple: $(\varepsilon_i, \varepsilon_j, l)$, where $\varepsilon_i, \varepsilon_j$ are any two entities, l is a relation type of the two entities (e.g, measured property-quantity). It is actually a classification problem for any combination of two entities [18].

2.2.1 Ontology-based Approaches

Ontology-based approaches rely on external dictionaries or ontologies which

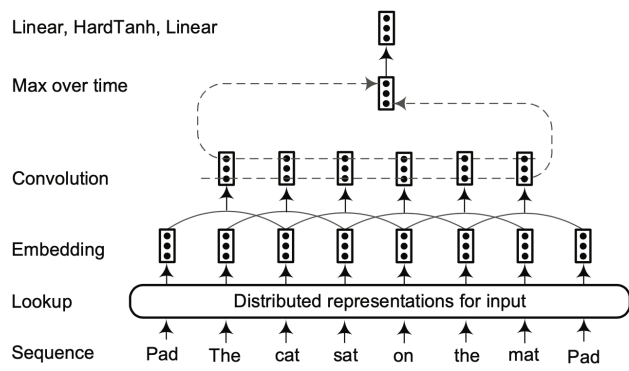


Figure 2.2: CNN Encoder [4]

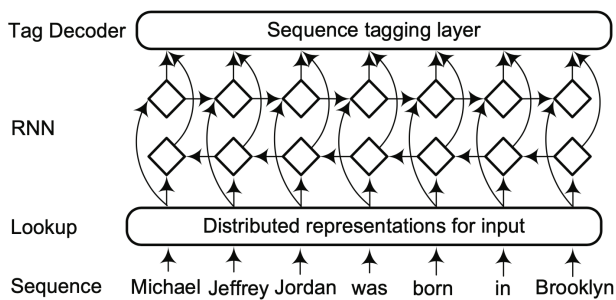


Figure 2.3: RNN Encoder [15]

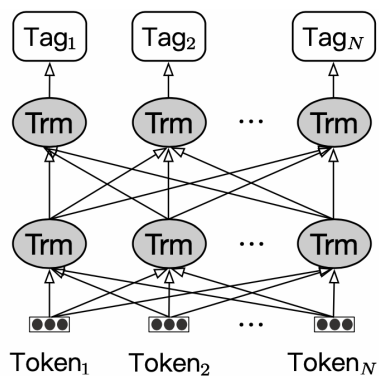


Figure 2.4: BERT [15]

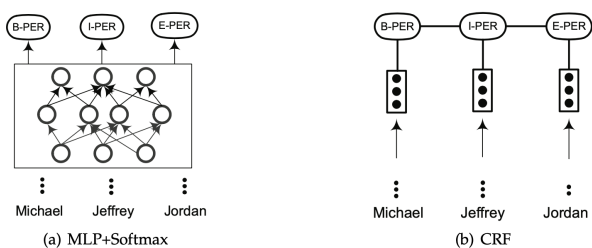


Figure 2.5: Different Tag Decoder Architectures [15]

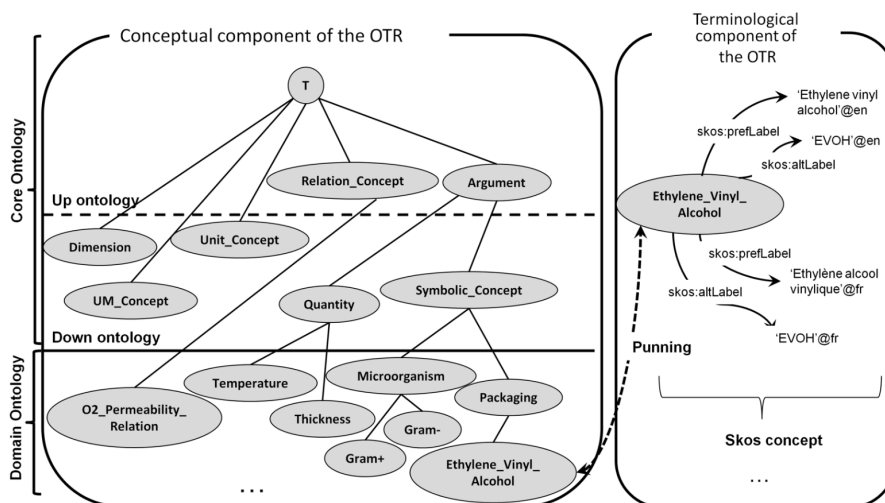


Figure 2.6: The concept hierarchy of n-Ary relationships in food packaging domain[3]

define the vocabulary and relationships between entities. [17, 27]. The ontologies must be summarized manually and a terminological resource matching to the ontologies is necessary for these approaches. An example of ontologies is shown in Fig.2.6. It is suitable to solve problems with complex hierarchy relationships (e.g. n-Ary relationships) with limited vocabulary in a specific domain [3, 14]. The complex hierarchy relationships reduce the efficiency and accuracy of the relationship model training.

However, creating a terminological resource requires much manual work for a new domain and ontologies and it is hard to make vocabularies that are exhaustive.

2.2.2 Rule-based Approaches

Rule-based relationship extraction relies on hand-crafted semantic and syntactic rules to recognize entities. Marve [11] is an example using word dependencies and Part-of-Speech (Pos) tags to extract entities and relationships (see Fig.2.1). It is also described in Section 2.1.1 as an entity extraction approach.

2.2.3 Unsupervised Learning Approaches

The unsupervised relationship extraction method can extract relationships without manually annotated training data. A typical unsupervised method is proposed by Hasegawa [10]. Hasegawa's clustering algorithm gathers words or phrases with high similarity or co-occurrence as the corresponding relationships. Then it regards the class combination with the highest frequency as the characterization of a particular relationship (e.g. PERSON-COMPANY). They are easy to apply because it doesn't require model training. However, it is impossible to define the numbers and names of clustered relationships in advance.

2.2.4 Feature-based Supervised Learning Approaches

Because relationship extraction is actually a multi-classification problem, many feature-based models apply to it, such as k-Nearest-Neighbor Classifiers, kNN, and SVM, which are discussed in section 2.1.3.

These models rely on a set of features that are obtained by textual analysis. Their performances strongly depend on the quality of the features. Therefore, how to select an efficient feature set is crucial for model training. Researchers usually suffer from manual work when constructing the structured representation into feature vectors. [13].

2.2.5 Deep Learning Approaches

Deep learning methods have the advantage of direct feature extraction over feature-based supervised learning. Deep learning reads raw data and learns hidden features automatically. Manually designing features is unnecessary [13]. The popular deep learning models for relationship extraction are CNN

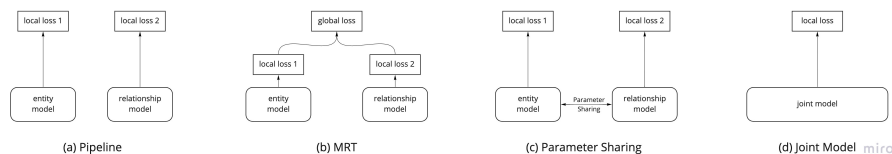


Figure 2.7: Different Interactions between Entity Extraction and Relationship Extraction [15]

and RNN. For instances, Zeng et al. [23] presented a sentence-based CNN model for relation classification. Its architecture is similar to Fig.2.2. Ebrahimi and Dou [5] proposed a chain based RNN using the shortest path between two entities in a dependency graph.

2.3 The Interaction between Entity Extraction and Relationship Extraction

According to the interaction depth between entity extraction and relationship extraction, there are four main types of approaches from no interaction to heavy interaction: pipeline, joint minimum risk training, parameter sharing, and joint extraction (see Fig.2.7).

2.3.1 Pipeline Approaches

The easiest applications are pipelined, which means the entity models and relationship models are totally separate. Entity models are trained first. Then the relationship models are trained based on the entity extraction result (see Fig.2.7(a)). The training process of pipelined training is simple and flexible to apply different algorithms and use different data sources. However, the entity model ignores the relation annotations which may be useful for identifying entities (e.g., if an A-B relation exists, the entity model can only assign A and B to its entities). Therefore, error propagation and data inefficiency are the main problems of this approach [21].

2.3.2 Joint minimum risk training (MRT) Approaches

Joint minimum risk training (MRT) [21] uses a global loss function to strengthen

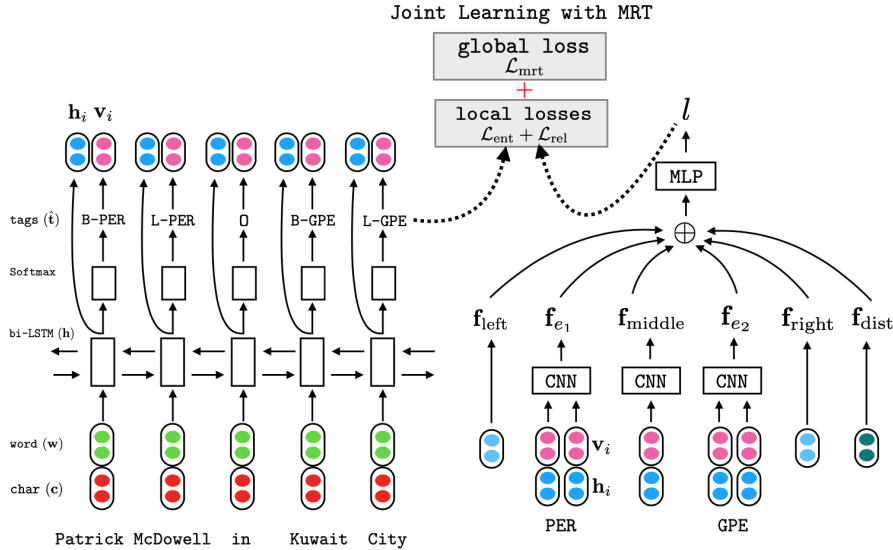


Figure 2.8: A Joint minimum risk training Architecture for the entity and relation extraction [21]

connections between the entity models and relationship models, and keeps their capacities unaffected (see Fig.2.7(b)). The entity models and relationship models are optimized simultaneously under the global loss function. One of the advantages of MRT is that the training can make use of the information from both sides: entity and relationship.

For example, Sun et al.[21] propose a MRT architecture using bi-LSTM and softmax as entity models and CNN as relationship models (see Fig.2.8).

However, the simultaneous training process highly increases the complexity of implementation. A large number of existing wrapped libraries of supervised learning, such as sklearn and keras, do not support MRT in training process. Applying MRT asks hand-crafted models and training function. It costs unaffordable time when many model combinations are tested.

2.3.3 Parameter Sharing Approaches

Parameter sharing is a strategy which shares some input features or internal hidden states between the entity models and relation models (see Fig.2.7(c)). There are no special constraints in model selection. However, naive parameter sharing can not fully utilize the inherent connection between the sub-

models due to independent models. For example, to get features from relation annotations, the entity model needs to wait for the relation model to update the shared parameters. In contrast, MRT allows the training of the entity model to directly acknowledge the loss of the relation model without waiting for shared parameters updating [21].

2.3.4 Joint Extraction Approaches

Joint extraction algorithms can simultaneously encode relations in the sequential labelling tag set (see Fig.2.7(d)). The solutions are usually complex and heavyweight. Adding constraints on the relation model is necessary (e.g. it cannot handle entities which appear in multiple relations) [26].

2.3.4.1 Special Symbol Insertion Method

For one-on-one relationships, there are some convenient solutions. Gangwar et al.[8] propose a method with special symbol insertion to extract measured property based on known quantity using the SemEval-2010 Task 8 dataset, MeasEval. They insert the special symbol $\hat{\$}$ at the beginning and end of the quantity span. Then, the span of the measured entity related to quantity enclosed in the $\hat{\$}$ symbol is used as the true-label for training the model. The modified sentences are used as input for training the model. The extracted measured property is linked to the quantity span directly which is surrounded by "\$".

2.3.4.2 Question Answering

There is another way to merge the original two tasks entity extraction and relation extraction into a single Question Answering (QA) task. Li et al.[16]. propose a method of treating the entity-relation extraction task as a multi-turn Question Answering task. Question Answering consists of sequential questions made by templates and answers to extract structural data. A relationship is built based on the corresponding question and returned answer. QA provides an elegant way to extract entities and relations simultaneously.

Avram et al.[1] apply the QA architecture to extract measured property based on known quantity using the SemEval-2010 Task 8 dataset, MeasEval. Avram et al.[1] insert a question, "What is the measured property of the quantity XXX" before the original sentence. XXX is the single quantity span. Then, the span of the measured entity related to quantity in the question is used as the true-label for training the model. The modified sentences are

used as input for training the model. The extracted measured property is linked to the quantity span indicated in the question directly.

The most important advantage of QA is that it enables the entity extraction model to extract both entity and relationship in one shot. Hence, a separate relationship model is not necessary when QA is applied. It can definitely simplify the architecture of entity and relationship extraction solution and save the implementation time. Therefore, we use a similar solution as Avram et al.'s [24].

2.3.5 Conclusion of Related Work

We summarize the four types of entity extraction approaches, five kinds of relationship extraction approaches, and three different interactions between entity extraction and relationship extraction.

In entity extraction, rule-based approaches are based on domain-specific knowledge and hand-crafted rules. They do not generalize well across domains. Unsupervised learning approaches can not control the numbers and names of tags. Feature-based supervised learning can extract entities and label them using predefined tags but researchers suffer from how to construct a good feature set. Deep learning approaches can process raw data and discover entities without effort on designing features. Therefore, deep learning approaches are the best choice for our problem.

Relationship extraction is similar to entity extraction. Ontology-based approaches require much manual work to summarize the domain vocabularies and relationships. Moreover, the domain vocabularies sometimes are hard to be exhaustively summarized by humans. Rule-based approaches have a generalization problem when applying a solution to another domain. Unsupervised learning can not label a relationship as predefined relationship types. Feature-based supervised learning suffers from feature construction. Deep learning is easy to be implemented because it doesn't require feature design.

There are four kinds of interaction between entity extraction and relationship extraction. In pipeline approaches, entity extraction and relationship extraction are separated models. It is easy to implement but they can not utilize the information from another side. MRT approaches use a global loss function to optimize entity extraction and relationship extraction models together. But it is not supported by major machine learning libraries, like sklearn and keras. It costs much time on coding for your training function. Parameter sharing is lighter than MRT. It only shares features among models. However, naive parameter sharing can not fully utilize the inherent connection between sub-models. The last type is joint extraction approaches.

It simultaneously encodes relations in entity extraction. It enables an entity extraction model to extract relationships at the same time. It is the lightest approach but utilizes the coherent information between relationship and entity. Therefore, we decide to apply one of the Joint Extraction Approaches, Question-Answering in our thesis. Meanwhile, Question-Answering doesn't require a separate relationship model. We don't have to choose a relationship model for our solution.

Chapter 3

Method

3.1 Project Framework

Our research question is to extract two types of entities, the quantities and corresponding measured properties, and link them in pairs. Therefore, there are two main tasks, entity extraction and relationship extraction.

Because Question-Answering (QA) architecture can enable the entity extraction model to extract both entity and relationship at the same time, it merges our two tasks into one. We decide to apply QA in our system to make the framework simpler and save much time for a separate relationship model training and testing.

QA asks sequential entity models to extract multiple entities because the former extraction results will be filled in the question template as the input question for the latter extraction model.

Therefore, we have two entity extraction models to extract quantity and measured property respectively. After preprocessing the text and splitting it into sentences, the Quantity extraction model goes first to extract as many as quantity entities in the given text. Then, all the quantity spans are marked and the raw text and quantity span are sent into the Question Answering processing. Question Answering processing constructs the questions with the question template and each quantity span. Every quantity span can be used to construct one question. After that, the raw text and one question are sent to the Measured Property extraction model. The Measured Property extraction model extracts the measured property span based on the raw text and the question. The extracted measured property is linked to the quantity indicated in the question naturally (see Fig.3.1).

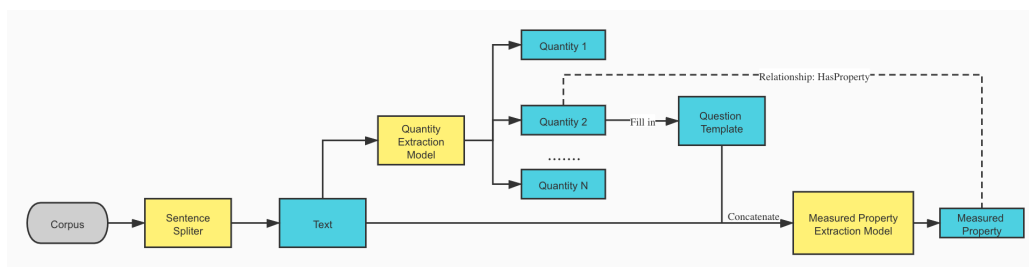


Figure 3.1: The framework of the project

3.2 Preprocessing

Our corpus is drawn from scientific full texts. The input for the quantity model and QA system can be a full document, a paragraph, or a sentence. To normalize the input length, we assume that a quantity and its responding measured property should occur in one sentence, so we split all the corpus into sentence-level input.

Then we use the spacy English tokenizer to tokenize the sentence-level text into a list of tokens. Finally, we remove all the punctuation in the token list. The punctuation removal must be done after the tokenization because there may be meaningful punctuation in the quantities, such as "1.8g" or "3m/s". These punctuations should not be removed from the text.

3.3 Extraction model

The extraction model can be any models which are described in section 2.1. According to our study, deep learning approaches are the best choice for our problem. More specifically, unlike CNN, a bidirectional LSTM (bi-LSTM) can utilize the evidence from the whole input sentence including past and future information in sequential data. Bi-LSTM can be a candidate of our entity extraction model. Moreover, SciBERT[2] is quite popular when solving nlp tasks in the scientific domain. It can be another candidate of our entity extraction model because it matches our domain so well.

For bi-LSTM and SciBERT models, the input must be word representation. A token list works for both models. Their output is in BILUO scheme. It uses B-, I-, L- prefix to indicate the beginning, inside, last of a multi-token entity. The U- prefix is used to label a single-token entity. O is for tokens outside any entity classes (see Fig.3.2).

If	0	
we	0	
assume	0	
that	0	
the	0	
average		0
seawater		0
1870s/1880s		U-MP
prior	0	
to	0	
the	0	
LIP	0	
onset	0	
was	0	
-0.8	0	
and	0	
use	0	
an	0	
average		0
Os	B-MP	
abundance		L-MP
in	0	
seawater		0
of	0	
10	0	
ppq	0	
based	0	
on	0	
the	0	
present		0
day	0	
average		0

Figure 3.2: BILOU Scheme

3.4 Question Answering

Question Answering enables the entity extraction model to extract both entity and relationship at the same time. It simplifies our architecture and replaces the architecture of separate entity and relationship models.

The core of Question Answering is to construct a question to indicate a known entity span and its potential relationship. The question is appended before the original text. For example, we have such a sentence as below. The quantity spans have been labelled by the quantity extraction model in advance.

An efficient red-emitting LEC fabricated on a glass substrate achieved a current efficiency (ηC) of [7.18 *cd/A*](Quantity) and an external quantum efficiency (η_{ext}) of [9.32%](Quantity)[24].

Then, we construct a question based on question template and quantity span. Our question template is like below:

What is the measured property of _____?

Then, we fill one of the quantity spans into our question template. Then we have a specific question:

What is the measured property of 7.18 *cd/A*?

After that, we concatenate the question with the raw text. The modified sentence is the input of the measured property extraction model.

What is the measured property of 7.18 *cd/A*? An efficient red-emitting LEC fabricated on a glass substrate achieved a current efficiency (ηC) of 7.18 *cd/A* and an external quantum efficiency (η_{ext}) of 9.32%.

Finally, the measured property extraction model extracts the measured property and it is linked to the quantity span given by the question directly.

What is the measured property of 7.18 cd/A ? An efficient red-emitting LEC fabricated on a glass substrate achieved a [current efficiency] (Measured Property) (ηC) of [7.18 cd/A] (Quantity) and an external quantum efficiency (η_{ext}) of 9.32%.

[current efficiency] (Measured Property) <---> [7.18 cd/A] (Quantity)

This is how our system works. To make Question-Answering work as our expectation, we have to train the entity extraction model with question template questions too. We will discuss it in the next section.

3.5 Model Training for Question-Answering

In our training process, the quantity extraction model is trained by raw texts and quantity spans without any modification. But we have to modify the training data for the measured property extraction model because it must understand questions and give predictions like our design in the framework.

See Fig.3.3, the training data consists of three parts: raw text, entities, and relationships. There are multiple pairs of quantity and measured property. For each pair, we create a line of training data. We use the quantity span to fill the question template "What is the measured property of XXX?" and concatenate the question with the raw text as the X input for measured property extraction model. Meanwhile, the corresponding measured property entity is sent as the Y input for the model.

In this way, the measured property extraction model learns relationship in pairs. When it extract a measured property with a question, it links to the span filled in the question just like how we train it.

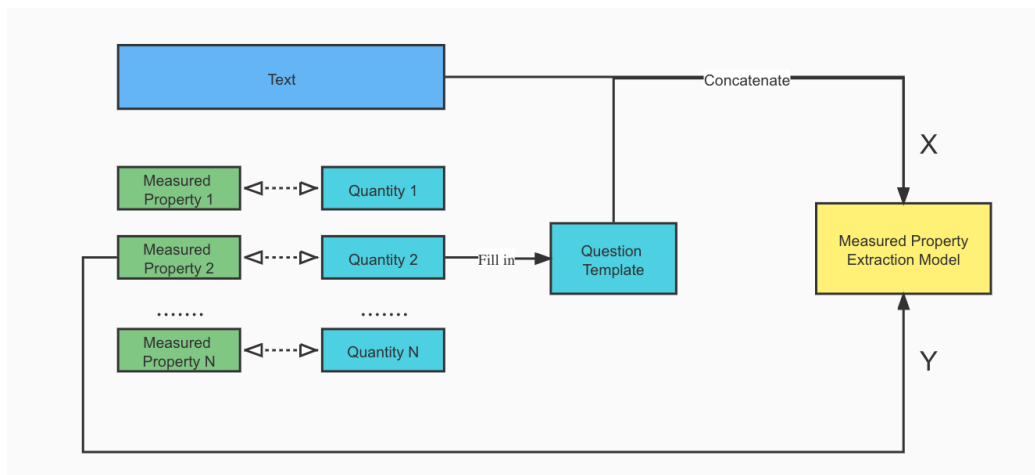


Figure 3.3: Model Training for Question-Answering

Chapter 4

Dataset

Datasets that include quantities and measured properties are rare on the internet. We only find one, which is MeasEval. But it covers 10 different scientific domains, which doesn't match our requirements. Our target is to build a system to extract measured properties only in the engineering domain. Therefore, we annotate a new dataset, MeasPro, to train our model.

4.1 MeasEval

MeasEval[9] is the dataset of the SemEval-2010 Task 8: extracting counts, measurements, and related context from scientific documents. The task focused on finding quantities, units, measured properties, measured entities and measurement contexts. The task consists of five sub-tasks:

1. Identify quantities (e.g. 12 kg)
2. For each identified Quantity, identify the Unit of Measurement (e.g. kg)
3. For each identified Quantity, identify the Measured Entity (e.g. bed inventory) and Measured Property (e.g. concentration)
4. Identify and mark the span of any Qualifier (e.g. after incubation)
5. Identify relationships between Quantity, Measured Entity, Measured Property, and Qualifier spans using the HasQuantity, HasProperty, and Qualifies relation types (see in Fig.4.1).

MeasEval is drawn from scientific papers from 10 different domains including agriculture, astronomy, biology, chemistry, computer science, earth

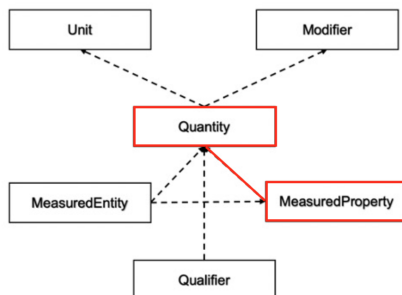


Figure 4.1: The relationship of the entities in MeasEval[9]

science, engineering, materials science, mathematics and medicine. It has 1164 quantities, 911 measured entities, 651 measured properties and 278 qualifiers.

Our problem only includes two entities, quantity and measured property, and one relationship from quantity to measured property. Our task is a subset of the MeasEval task (see the red frames in Fig.4.1). However, our solution will be used to extract measured properties and quantities in engineering papers. The engineering part of MeasEval only has less than 100 measured properties. It is not enough for our model training.

MeasEval is still useful for us because there are over 75 submissions for its NLP task. We can train our model based on MeasEval and compare our results with the state-of-art models. Then we implement the best practice on MeasPro for the final solution. In this thesis, we train measured property extraction model for the comparison.

4.2 MeasPro: A newly annotated dataset

Because MeasEval can not match our needs, we construct a new dataset called MeasPro (MeasuredProperty). We annotated 3300 sentences sampled from engineering domain papers with two entities: Quantity and Measured-Property and one relationship: HasProperty (Quantity - MeasuredProperty). Moreover, MeasPro has more useful entities than MeasEval. MeasEval has only 1164 quantities and 651 measured properties in 10 domains. But MeasPro has 7316 quantities and 4606 measured properties in engineering domain.

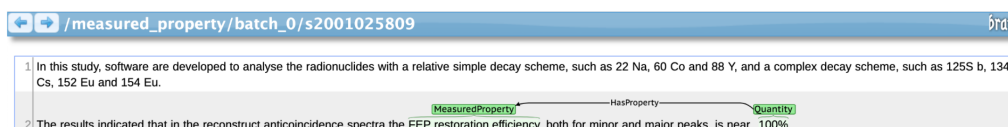


Figure 4.2: The Screenshot of an Example Annotation

Dataset	Quantity	MeasuredProperty
MeasPro	0.938	0.72
MeasEval	0.943	0.641

Table 4.1: Krippendorff's Alpha scores in Inter-Annotator Agreement batch

4.2.1 Data Source

The sentences are drawn from the unitsOfMeasure database, Scopus 19-20, of Elsevier. The database is sampled from engineering publications and it has over 700k sentences with quantities. The data structure includes id, concept url and sample sentence. An example is shown below:

```
2001568977 https://data.elsevier.com/e/unitsOfMeasure/Concept-520274148
The introduction of the posttransition iridium(III) ion not only
significantly enhances the absorption intensity at the end of visible region,
but also displays efficient singlet oxygen quantum yield (76%), which is
applicable for photodynamic therapy in living cells.
```

The data has been processed and the units in the sentences have been extracted and normalized according to Elsevier's standard unit library. The normalized units can be searched by the concept url like below:

```
https://data.elsevier.com/e/unitsOfMeasure/Concept-520407122
microwatt per cubic metre
```

4.2.2 Annotation Setting

To have a balanced dataset, we group sentences by their normalized unit given by Elsevier unitsOfMeasure and sample an equal number of sentences, 100, from each unit group. In result, we get 33 unique units and 3300 sampled

sentences. Then we randomly batch all the samples into 10 batches for task assignment.

Three colleagues from the demand department of Elsevier take charge of the annotation task. We have 10 even batches in total. Batch 0 is annotated by all three annotators. Batch 0 is used to evaluate the consistency of annotation by different annotators. Then each annotator is assigned 3 non-overlapping batches respectively to reduce the workload.

We used the brat annotation tool[20] to carry out our task. The screenshot of an example annotation is in Fig.4.2.

4.2.3 Evaluation of the Annotations

There are variations when humans annotate data. The inter-annotator agreement (IAA) is used to evaluate the consistency. Krippendorff's alpha is a reliability coefficient developed to measure the IAA. Krippendorff suggests it is customary to require $\alpha \geq 0.800$. Where tentative conclusions are still acceptable, $\alpha \geq 0.667$ is the lowest conceivable limit [12].

The summary paper of MeasEval provides the Krippendorff's alpha of each MeasEval's entity [9]. The comparison of Krippendorff's alpha between MeasPro and MeasEval's key entities is shown in Table.4.1. It shows that the consistency of Quantity in both datasets is good enough. However, the consistency of MeasuredProperty of MeasEval is unacceptable. The one of MeasPro is acceptable.

Chapter 5

Experiments of Measured Property Extraction

5.1 The State-of-art Models trained on MeasEval

As we discuss in section 4.1, MeasEval is a dataset of the SemEval-2010 Task 8. There are five sub-tasks in it:

1. Identify quantities (e.g. 12 kg)
2. For each identified Quantity, identify the Unit of Measurement (e.g. kg)
3. For each identified Quantity, identify the Measured Entity (e.g. bed inventory) and Measured Property (e.g. concentration)
4. Identify and mark the span of any Qualifier (e.g. after incubation)
5. Identify relationships between Quantity, Measured Entity, Measured Property, and Qualifier spans using the HasQuantity, HasProperty, and Qualifies relation types.

We have the same goal to extract quantities and measured property, which matches sub-task 1 and half of sub-task 3. Fortunately, Harper et al.'s summary paper [9] makes a conclusion about the scores of quantities and measured property extraction from different submissions. It enables us to compare our extraction results with the state-of-art models.

However, their definitions of the relationships is different from ours. In their summary paper [9], there are only the scores of two relationships,

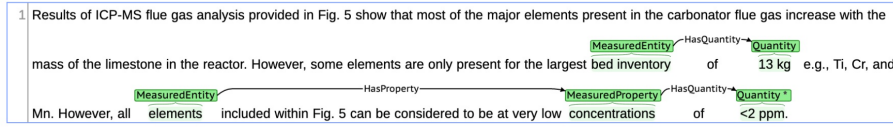


Figure 5.1: Example of a Quantity with related annotations in MeasEval[9]

HasQuantity and HasProperty. From Fig.5.1, we can see HasProperty is the relationship from MeasuredEntity to MeasuredProperty. HasQuantity is a relationship from either MeasuredEntity or MeasuredProperty to Quantity. We can't access the scores about the relationship only from MeasuredProperty to Quantity. Hence, we can't compare our relationship extraction results with the state-of-art models.

In summary, we decide to only train two entity extraction models to extract quantity and measured property respectively for comparison with the state-of-art models, regardless of relationship extraction in this experiment phase.

In the summary paper [9], they use a single SQuAD-style [19] F1 (Overlap) score as the evaluation of entity extraction. The overlap F1 score is a metric which measures the average overlap between the prediction and ground truth answer. If A is the prediction, B is the ground truth answer, and A and B are both bags of tokens, the overlap area is where the tokens from A match the tokens from B. Then, the metric of overlap F1 can be performed as the following calculations:

$$F1 = (2 * length(Area of Overlap)) / (length(A) + length(B))$$

The summary image (see Fig.5.2) and Harper et al.'s paper [9] show almost all participants have a high F1-score for Quantity extraction. Their median F1 score is over 0.800. The highest score for Quantity extraction is 0.861 from team LIORI and Counts@IITK. However, the scores for MeasuredProperty extraction are unsatisfactory. The median score is less than 0.400. The two highest scores for Measured Property extraction are 0.467 and 0.437 from team LIORI and jarvis@tencent.

5.2 Comparison of Measured Property Extraction Models

From Fig.5.2, we can easily see the scores for Quantity extraction are high enough. On the other hand, there is a large space to improve in the extraction of measured property where we should put more efforts on. Therefore, we

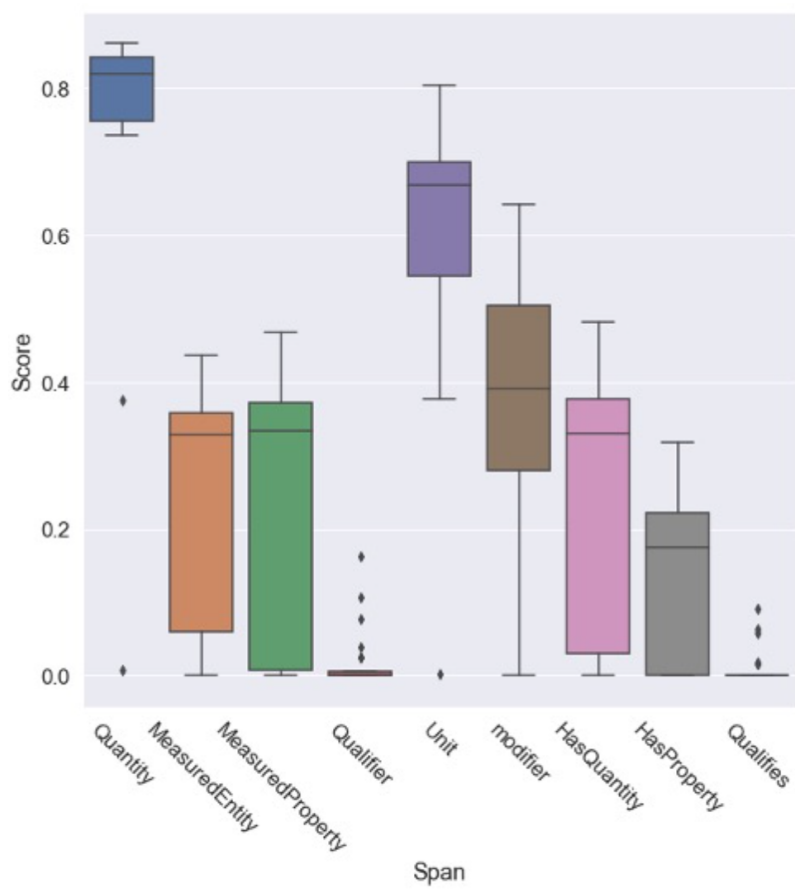


Figure 5.2: Visualization of average scores for each scoring component across top score for all participants[9]

Parameters	Value
Input_dim	125
Output_dim	125
Input_length	125
LSTM units	6
recurrent_dropout	1e-4
Tag decoder	softmax

Table 5.1: Parameters of bi-LSTM

Parameters	Value
bert_model	scibert-scivocab-uncased
max_seq_length	125
epochs	10
gradient_accumulation_steps	20
learning_rate	3e-05
train_batch_size	32

Table 5.2: Parameters of SciBERT

mainly test our measured property extraction models by overlap F1-score regardless of Quantity extraction.

As we discuss in section 3.3, we select bi-LSTM and SciBERT as our candidate models for entity extraction. We train and test these two models and compare them with the best two state-of-art models in Measure Property extraction (from team LIORI and jarvis@tencent).

MeasEval’s data is paragraph level. There are multiple sentences in each row of data. According to our system design, we split them into sentences, then use the sentence level data as new training data. Following the baseline notebook provided by MeasEval dataset, we randomly split 70% of the dataset as the training data and 30% as the testing data.

The parameter values that we used for training our bi-LSTM model is shown in Table 5.1. For SciBERT model, we use a scikit-wrapper SciBERT library¹ which simplifies the parameters and makes finetuning the SciBERT model more easy. We provide the parameter values we used for training our SciBERT model (Table 5.2).

We summarize the results of bi-LSTM and SciBERT models and the results from the best state-of-art models in the Table.5.3. We can see our SciBERT model has the highest performance (F1: 0.617) on measured property extraction, which is much more than LIORI and jarvis@tencent’s models

¹scikit-learn wrapper to finetune BERT, <https://github.com/charles9n/bert-sklearn>

Models	MeasuredProperty
bi-LSTM	0.134
SciBERT	0.617
LIORI	0.467
jarvis@tencent	0.437

Table 5.3: F1-score of Measured Property Extraction from different models

(0.467 and 0.437 respectively). In contrast, bi-LSTM is the worst one. The F1 score is only 0.134.

We think the pretrained scientific vocabulary of SciBERT make it outstanding among these models. SciBERT is a domain-related pre-trained model. It leverages unsupervised pretraining on a large multi-domain corpus of scientific publications (1.14M papers, 3.1B tokens), to construct a new scientific vocabulary (size: 30K) called SCIVOCAB. There is only 42% overlap between BERT’s pretrained vocabulary and SCIVOCAB. Beltagy et al. prove that SciBERT’s performance on NLP problems in the scientific domain is better than the one of the BERT with base vocabulary [2]. It indicates that scientific language has many domain-related words and this vocabulary influences the performance on NLP task of the scientific publication. In contrast, bi-LSTM doesn’t have a pretrained vocabulary. Its vocabulary is constructed only from the limited training data. For example, we train bi-LSTM with 70% MeasEval of the dataset. There are only 4334 unique words to construct a vocabulary. It limits the generalization of bi-LSTM on unknown tokens from scientific publications. We think it is the main reason that SciBERT has a high score but the performance of bi-LSTM is poor.

5.3 How the Input Length Influences the Score

MeasEval is a paragraph-level dataset. But in our system design, we split the paragraphs into sentence-level for training. To figure out whether the sentence split is the best choice for training, we train two SciBERT models based on two different conditions: one is trained on paragraph level input, while for the other, we split the corpus into sentences.

In MeasEval, there are 742 relationships between a quantity and a measured property. Its maximum paragraph has 1780 tokens. After sentence splitting, the maximum text length becomes 125 tokens. Besides, according to statistics, 95.7% quantities and its responding measured properties occur in one sentence in MeasEval. It proves that our assumption in section 3.2 is valid.

Input level	Max Input Length	MeasuredProperty
Paragraph level	1780	0.062
Sentence level	125	0.617

Table 5.4: F1-score of Measured Property Extraction based on different input length

The model testing results in paragraph level and sentence level are shown in Table.5.4. It shows that the scikit-wrapper SciBERT works badly on long input length. The F1 score is only 0.062. However, if we use sentence-level input as training data, the SciBERT model can reach 0.617 F1 score. Therefore, our best choice is to train the model based on sentence-level input. That’s why we designed our system based on sentence-level input in section 3. We also constructed the new dataset, MeasPro, at sentence level from the beginning in section 4.2.

5.4 How the Parameters Influence the Score

There are several key parameters for finetuning the SciBERT model training, "gradient_accumulation_steps", "learning_rate", and "epoch". "Gradient_accumulation_steps" is the number of update steps to accumulate before performing a backward/update pass. "Learning_rate" indicates how much the Bert Optimizer remembers from each iteration. "Epochs" is the number of training iterations. We test multiple values to find the best combination.

We find that as the learning rate grows, the F1 score increases first before $learning\ rate < 1e-5$, then keeps stable when $learning\ rate \in [1e-5, 3e-5]$, and decreases to 0 from $3e-5$ to $7e-5$ (see Fig.5.3). In summary, the valid window of learning rate is quite narrow. F1 score is higher than 0.500 only when the $learning\ rate \in [1e-5, 3e-5]$. This phenomenon also occurs in other conditions, for example, different epoches and gradient accumulation steps. Learning rate is about the remember rate of the optimizer. If it is too low, the optimizer almost remembers nothing in each iteration. It causes underfit. On the contrary, if it is too high, the optimizer remembers too much in the first few iterations. Then the model gets overfit due to it. This is the reason why the valid window of learning rate is quite narrow.

About the epochs, the F1-score increases when the epochs increase. However, when the number of epochs is over 10, F1-score increases hardly (see Fig.5.4). It is easy to understand why it happens because epochs is the number of training iterations. More iterations make the model learn the same things more times. After a certain number of iterations, the model has

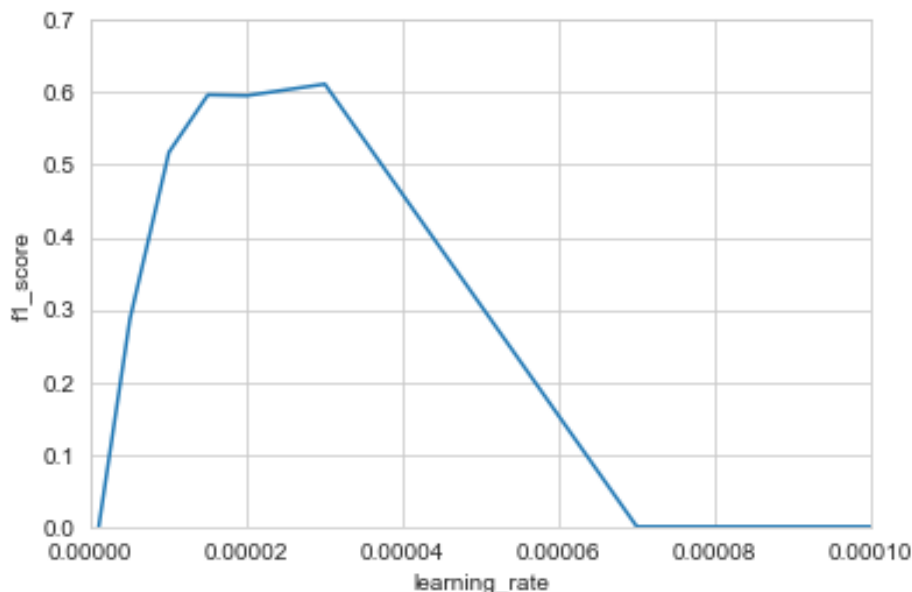


Figure 5.3: F1-score when $epochs = 10$ and $gradient\ accumulation\ steps = 16$

learned almost everything in the training data. It can not learn much in the next iteration. This phenomenon also occur in other conditions, for example, different learning rate and gradient accumulation steps, even though the curve may move to the left or right to a certain extent in the coordinates.

For the gradient accumulation steps aspect, it hardly influences the F1-score (see Fig.5.5). Gradient accumulation is the number of update steps to accumulate before performing a backward/update pass. Gradient accumulation is associated with the batch size. The batch size is the number of samples which are sent to a model in every single training step. Larger batch sizes need more GPU memory. Gradient accumulation is used to split the batch of samples into a few mini-batches to reduce the storage in the training process. It is a technique which aims to reduce the memory use but it doesn't influence the training result.

In summary, learning rate must be in the zone $\in [1e - 5, 3e - 5]$. 10-20 epochs is enough for training. Gradient accumulation steps don't influence the results. We provide our best practice in Table.5.10. It is epochs: 10, gradient accumulation steps: 20, learning rate: 3e-05.

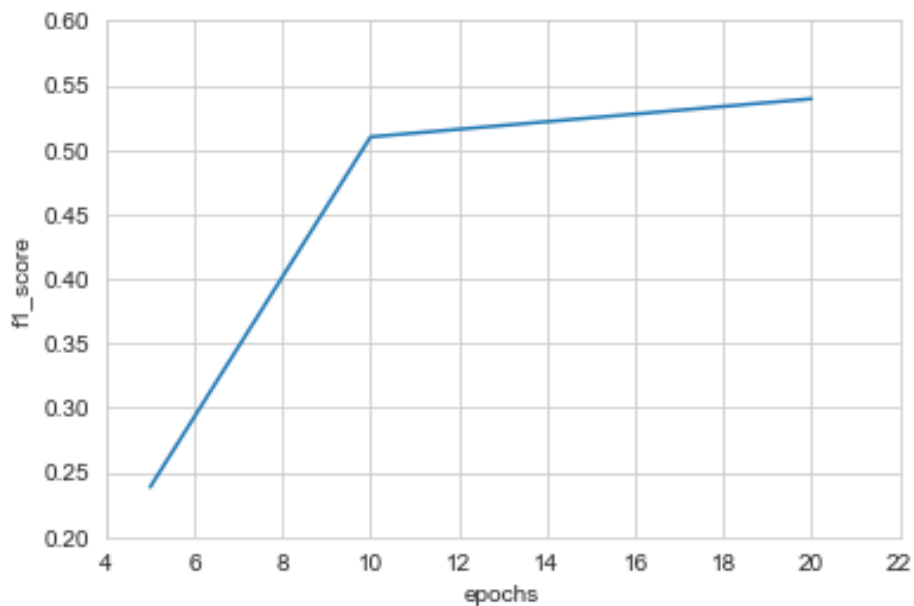


Figure 5.4: F1-score when $learning\ rate = 1e - 5$ and $gradient\ accumulation\ steps = 4$

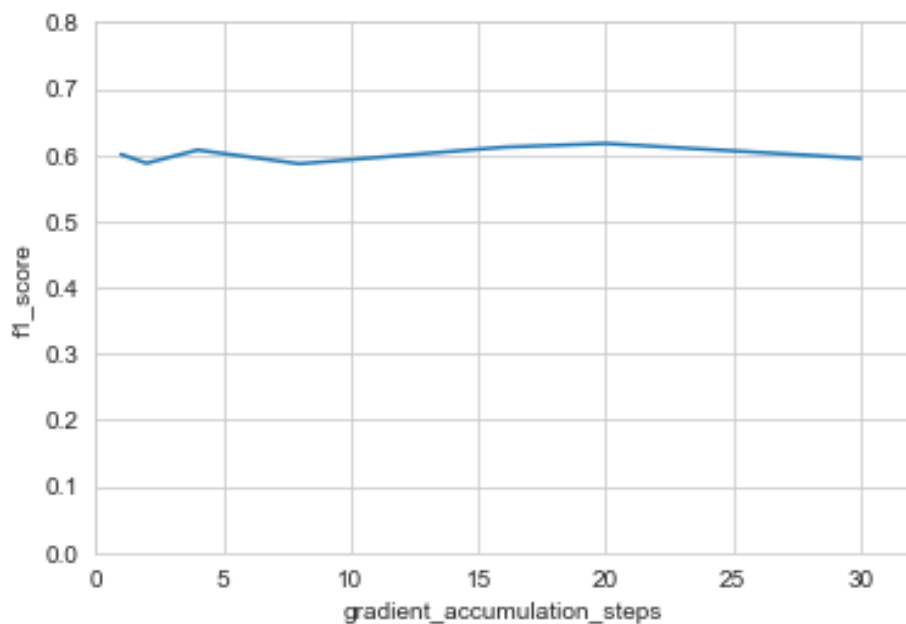


Figure 5.5: F1-score when $learning\ rate = 3e - 5$ and $epochs = 10$

Parameters	Value
class_prior	[0.001, 0.999]

Table 5.5: Parameters of BernoulliNB

Parameters	Value
loss	hinge
penalty	l2
alpha	$1e - 3$
max_iter	5
class_weight	0 : 0.25, 1 : 0.75

Table 5.6: Parameters of SVM

5.5 Text Classification before Entity Extraction

When we observe the scores of the bi-LSTM model, we find that the precision of bi-LSTM is extremely low. As we show in section 4.1, there are 1164 quantities and 651 measured properties. After sentence splitting, there are only one fourth of the sentences has at least one measured property. As the consequence, there are many false positives in the predictions of bi-LSTM.

To reduce the false positives, we implement a text classification model which detects if a sentence has measured property and gives a true or false tag. Only sentences with a measured property tag are sent to the extraction model.

Firstly, we train two text classification models, Bernoulli Naive Bayes (BernoulliNB) and SVM. We choose them because it is fast to train these two models. We can quickly validate our idea of adding a text classifier before entity extraction. The parameters of BernoulliNB and SVM are shown in Table 5.5 and 5.6.

The test results of the two models are shown in Table.5.7. From the F1-score, BernoulliNB has a higher score. But from the confusion matrix of the two models (see Table.5.8, we can see BernoulliNB model has less true positives which means BernoulliNB drops more sentences with a measured property tag. We hope to reduce the sentences without a measured property tag not the opposite. Therefore, we choose SVM model to join with the bi-LSTM model.

SciBERT supports text classification task too. We also train a SciBERT classification model to join it with SciBERT extraction model to compare with SVM + biLSTM combination. The parameters of the SciBERT text classifier

tag	BernoulliNB	SVM	SciBERT
0	0.82	0.76	0.94
1	0.59	0.59	0.86
overall	0.75	0.69	0.91

Table 5.7: F1-scores of the Text Classifiers

	BernoulliNB		SVM	
	Negative	Positive	Negative	Positive
0	286	75	239	122
1	54	92	33	113

Table 5.8: Confusion Matrix of BernoulliNB and SVM Text Classification Models

are shown in Table 5.10. The evaluation of the SciBERT text classifier is shown in Table 5.7.

The result is shown in Table.5.9. The result shows that the classification model increases the performance of bi-LSTM, but it doesn't work on the SciBERT model. The most likely reason why they are different is that the score of SciBERT is high enough. The text classification model can not help to increase the precision.

Models	MeasuredProperty
bi-LSTM	0.134
CLS(SVM) + bi-LSTM	0.221
SciBERT	0.617
CLS(SciBERT) + SciBERT	0.609

Table 5.9: F1-score of two-step models

Parameters	Value
bert_model	scibert-scivocab-uncased
max_seq_length	125
epochs	10
gradient_accumulation_steps	20
learning_rate	3e-05
train_batch_size	32

Table 5.10: Parameters of SciBERT

Chapter 6

Result of Full Implementation

As we discuss in section 4, MeasEval doesn't match the target domain in our project. We created a new dataset, MeasPro, instead. The sampled data of MeasPro is all cut from engineering papers. The annotation consistency of MeasPro is better than MeasEval. MeasPro also contains more quantities and measured properties of MeasPro than MeasEval. Therefore, we train our models based on MeasPro to implement our final solution.

In our system design (see section 3), there are two entity extraction models, including a Quantity extraction model and a MeasuredProperty extraction model. We have proved that the SciBERT is better than the state-of-art extraction models in a scientific context based on the MeasEval dataset. Hence, we directly apply SciBERT to construct our implementation on MeasPro.

There are 10 even batches in MeasPro. Because batch 0 is annotated by three annotators and we don't have extra human resources to solve the disagreement inside, we use the batch 1-9 as our training dataset, which are annotated by only one annotator respectively. We merge all batches and randomly split 80% data as training data and the other 20% as the testing data. In the testing part, we continue using the overlap F1-score as our main metric.

To evaluate every component in our system separately, firstly we train a Quantity extraction model using SciBERT and MeasPro's quantity entities. We simply use the same as in Table 5.10. Using these parameters, SciBERT obtains 0.944 F1-score (Table 6.1) on quantity extraction in MeasPro, which is significantly high. The Quantity extraction model is good enough as the model before the QA processing.

Secondly, we train a Measured Property extraction model directly without QA processing to validate the extraction efficiency. Still, we simply apply the same parameters as in Table 5.10. The model obtains 0.770 F1-

Models	MeasPro	MeasEval
Quantity Extraction	0.944	–
Measured Property Extraction	0.770	0.617
Measured Property Extraction with QA	0.811	–

Table 6.1: The Scores of Different SciBERT models

score (Table 6.1) on measured property extraction in MeasPro. Considering the experiments we did in the section 5, the SciBERT Measured Property extraction model obtains 0.617 on MeasEval (Table 6.1) and is beyond the state-of-art models. 0.770 is good enough for our project. In our dataset, SciBERT can obtain a higher score, which means MeasPro has better quality than the training dataset for this given task.

Moreover, with the same parameters, the transfer of SciBERT between these two datasets is very successful, which means SciBERT has high generalizability in the transfer between the scientific datasets.

The next step is to apply QA processing before the Measured Property extraction model. QA processing modifies the training data by appending a question about the relationship before the raw text. Naturally, we have to retrain the Measured Property extraction model to process the data after QA processing.

For example, *text1* is the original text from MeasPro like below:

An efficient red-emitting LEC fabricated on a glass substrate achieved a current efficiency (ηC) of 7.18 *cd/A* and an external quantum efficiency (η_{ext}) of 9.32% [24].

We know the 7.18 *cd/A* is the quantity so that we can construct *question1* like below:

What is the measured property of 7.18 *cd/A*?

Then, the differences between the original data and processed data can be represented like the Table 6.2.

We train a new SciBERT Measured Property extraction model based on the training data processed by QA. Still, we keep the same parameters while training the SciBERT model. The model obtains 0.811 F1-score (Table 6.1). Comparing to the former model (F1-score 0.770), it shows that if the information of the relationship is indicated in the training data as a question format,

Model Type	X	Y
original model	<i>text1</i>	concentration
model with QA	<i>question1 + text1</i>	concentration

Table 6.2: Different training data after QA processing

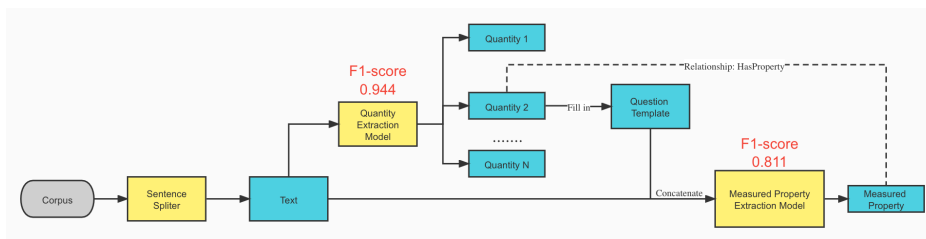


Figure 6.1: Results of Our Final Implementation

the prediction performance increases. The entity extraction model utilizes the coherent information hiding between the entities and relationships. It is one of the advantages of joint extraction.

In summary, we apply SciBERT model with the parameters which have been proved in our former experiments with MeasEval. The quantity extraction model obtains 0.944 F1-score and the measured property extraction model with QA data obtains 0.811 F1-score. The final system with the evaluation from each component is shown in the Fig.6.1. The scores show our system has high performance to solve our research question.

Bibliography

- [1] AVRAM, A.-M., ZAHARIA, G.-E., CERCEL, D.-C., AND DASCALU, M. Upb at semeval-2021 task 8: Extracting semantic information on measurements as multi-turn question answering. *arXiv preprint arXiv:2104.04549* (2021).
- [2] BELTAGY, I., LO, K., AND COHAN, A. Scibert: Pretrained language model for scientific text. In *EMNLP* (2019).
- [3] BERRAHOU, S. L., BUCHE, P., DIBIE, J., AND ROCHE, M. Xart system: discovering and extracting correlated arguments of n-ary relations from text. In *Proceedings of the 6th International Conference on Web Intelligence, Mining and Semantics* (2016), pp. 1–12.
- [4] COLLOBERT, R., WESTON, J., BOTTOU, L., KARLEN, M., KAVUKCUOGLU, K., AND KUKSA, P. Natural language processing (almost) from scratch. *Journal of machine learning research 12*, ARTICLE (2011), 2493–2537.
- [5] EBRAHIMI, J., AND DOU, D. Chain based rnn for relation classification. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (2015), pp. 1244–1249.
- [6] ETZIONI, O., CAFARELLA, M., DOWNEY, D., POPESCU, A.-M., SHAKED, T., SODERLAND, S., WELD, D. S., AND YATES, A. Unsupervised named-entity extraction from the web: An experimental study. *Artificial intelligence 165*, 1 (2005), 91–134.
- [7] FOPPIANO, L., ROMARY, L., ISHII, M., AND TANIFUJI, M. Automatic identification and normalisation of physical measurements in scientific literature. In *Proceedings of the ACM Symposium on Document Engineering 2019* (2019), pp. 1–4.

- [8] GANGWAR, A., JAIN, S., SOURAV, S., AND MODI, A. Counts@iitk at semeval-2021 task 8: Scibert based entity and semantic relation extraction for scientific data. *arXiv preprint arXiv:2104.01364* (2021).
- [9] HARPER, C., COX, J., KOHLER, C., SCERRI, A., DANIEL JR, R., AND GROTH, P. Semeval 2021 task 8: Measeval—extracting counts and measurements and their related contexts. In *Proceedings of the Fifteenth Workshop on Semantic Evaluation (SemEval-2021), Bangkok, Thailand (online)*. Association for Computational Linguistics (2021).
- [10] HASEGAWA, T., SEKINE, S., AND GRISHMAN, R. Discovering relations among named entities from large corpora. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)* (2004), pp. 415–422.
- [11] HUNDMAN, K., AND MATTMAN, C. A. Measurement context extraction from text: discovering opportunities and gaps in earth science.
- [12] KRIPPENDORFF, K. Estimating the reliability, systematic error and random error of interval data. *Educational and Psychological Measurement* 30, 1 (1970), 61–70.
- [13] LENG, J., AND JIANG, P. A deep learning approach for relationship extraction from interaction context in social manufacturing paradigm. *Knowledge-Based Systems* 100 (2016), 188–199.
- [14] LENTSCHAT, M., BUCHE, P., DIBIE-BARTHELEMY, J., AND ROCHE, M. Scipure: a new representation of textual data for entity identification from scientific publications. In *Proceedings of the 10th International Conference on Web Intelligence, Mining and Semantics* (2020), pp. 220–226.
- [15] LI, J., SUN, A., HAN, J., AND LI, C. A survey on deep learning for named entity recognition. *IEEE Transactions on Knowledge and Data Engineering* (2020).
- [16] LI, X., YIN, F., SUN, Z., LI, X., YUAN, A., CHAI, D., ZHOU, M., AND LI, J. Entity-relation extraction as multi-turn question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics* (2019), pp. 1340–1350.
- [17] MCDOWELL, L. K., AND CAFARELLA, M. Ontology-driven information extraction with ontosyphon. In *International Semantic Web Conference* (2006), Springer, pp. 428–444.

- [18] MIWA, M., AND BANSAL, M. 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)* (2016), pp. 1105–1116.
- [19] RAJPURKAR, P., ZHANG, J., LOPYREV, K., AND LIANG, P. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing* (2016), pp. 2383–2392.
- [20] STENETORP, P., PYYSALO, S., TOPIĆ, G., OHTA, T., ANANIADOU, S., AND TSUJII, J. Brat: a web-based tool for nlp-assisted text annotation. In *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics* (2012), pp. 102–107.
- [21] SUN, C., WU, Y., LAN, M., SUN, S., WANG, W., LEE, K.-C., AND WU, K. Extracting entities and relations with joint minimum risk training. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing* (2018), pp. 2256–2265.
- [22] WALLACH, H. M. Conditional random fields: An introduction. *Technical Reports (CIS)* (2004), 22.
- [23] ZENG, D., LIU, K., LAI, S., ZHOU, G., AND ZHAO, J. Relation classification via convolutional deep neural network. In *Proceedings of COLING 2014, the 25th international conference on computational linguistics: technical papers* (2014), pp. 2335–2344.
- [24] ZENG, Q., LI, F., CHEN, Z., YANG, K., LIU, Y., GUO, T., SHAN, G.-G., AND SU, Z. Rational design of efficient organometallic ir (iii) complexes for high-performance, flexible, monochromatic, and white light-emitting electrochemical cells. *ACS applied materials & interfaces* 12, 4 (2020), 4649–4658.
- [25] ZHANG, S., AND ELHADAD, N. Unsupervised biomedical named entity recognition: Experiments with clinical and biological texts. *Journal of biomedical informatics* 46, 6 (2013), 1088–1098.
- [26] ZHENG, S., WANG, F., BAO, H., HAO, Y., ZHOU, P., AND XU, B. Joint extraction of entities and relations based on a novel tagging scheme. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (2017), pp. 1227–1236.

- [27] ZHOU, X., ZHANG, X., AND HU, X. Maxmatcher: Biological concept extraction using approximate dictionary lookup. In *Pacific rim international conference on artificial intelligence* (2006), Springer, pp. 1145–1149.