Low Resource Quantitative Information Extraction via Structure Searching and Prefix-Based Text Generation

Tongliang Li, Zixiang Wang, Zhoujun Li*

State Key Lab of Software Development Environment, Beihang University, Beijing, China
{tonyliangli, wangzixiang, lizj}@buaa.edu.cn

Abstract

Quantitative information plays an important part in the financial and data analysis areas. Prior work relied on pattern-matching methods and complex hand-crafted rules to extract quantitative information due to the lack of labeled data. Such methods can be unstable and difficult to scale to the open domain. In this paper, we study quantitative information extraction in the low-resource setting. We propose a search-based approach by searching from the syntactic structures to acquire basic training data. The search process is simple yet effective. Then, a prefix-based text-to-text generation method is employed to extract the quantitative information. The prefix design can fully leverage pre-trained language models for text generation to serve the information extraction purpose. Experimental results show that our approaches achieves high performance with a limited amount of labeled data. The extraction result could further boost the performance of other tasks such as quantitative reasoning.

Introduction

Quantitative information in the text is extremely valuable in the financial and data analysis areas. For example, financial reports contain quantitative information that contributes to the understanding of companies' fundamentals which is not reflected in data tables. Quantitative information extraction helps to understand these numbers and extracts them into structured representations where standard financial tables or charts can be generated from. Despite these valuable applications, the task of Quantitative information extraction seems to be less studied and focused on. Most existing approaches in question answering and reading comprehension (Rajpurkar et al. 2016; Seo et al. 2017; Yu et al. 2018) do not explicitly address understanding quantities and might fail to answer queries that require quantitative inference and reasoning. The understanding of quantities could empower new applications like quantity search (Ho et al. 2019) to support analytic queries like "cities with more than one million inhabitants", "companies with PE small than 10". In this paper, we study quantitative information extraction, which can be considered as a prerequisite to obtaining a deep understanding of quantities.

*Corresponding Author

A major problem for quantitative information extraction is that the task is highly domain relevant, leading to a lack of labeled data for the open domain. Hence, quantitative information extraction is not well studied as a research topic. Prior work mainly rely on pure rule-based pattern-matching approaches or rules combined with statistical methods to extract quantitative information. For example, Saha, Pal, and Mausam (2017) propose BONIE that uses bootstrapping to learn the dependency patterns to extract numerical relation triples. However, not all quantities are expressed in forms of "relation". BONIE fails to extract any quantitative information from "What are the top 100 universities in the US?", and we should know that "100" refers to "universities in the US". Other similar studies like MARVE (Hundman and Mattmann 2017) design complex pattern-matching rules on the dependency structures. The rules are complex and is not generic.

We argue that only relying on pattern-matching rules is far from enough for open domain quantitative information extraction, as well as training fully-supervised models on domain-specific data. To gain higher performance on open domain information extraction, low-resource approaches should be considered first. The model should be able to fit on the smallest amount of data that is acquired from the open domain with minimum human effort. Based on these principles, we propose a low resource quantitative information extraction method based on automatically generated training data.

To obtain the training data, we propose a novel search-based approach by searching possible extraction results from the constituency parsing (CP) structures. The approach is simple yet effective, with no pattern-matching rules required. Previous studies already show the effectiveness of applying CP in information extraction (Bast and Haussmann 2013; Barkschat 2014; Evans et al. 2017; Cetto et al. 2018; Jiang and Diesner 2019). Our search-based approach significantly improves the recall compared with existing rule-matching approaches (Saha, Pal, and Mausam 2017; Hundman and Mattmann 2017). To build an information extraction model on limited amount of training data, we transform the information extraction task into a text generation task to fully leverage the pre-trained weights of a language model. Figure 1 shows the main concept of such transformation. We design a natural language prefix to guide the model on what

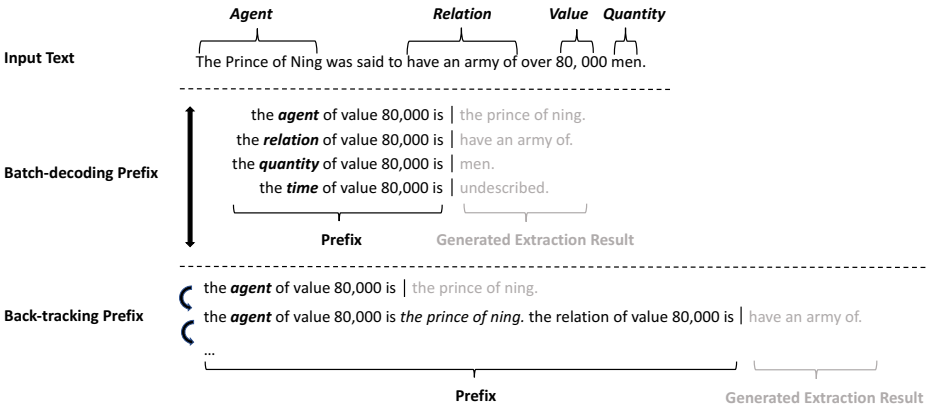

Figure 1: Generation-based quantitative information extraction overview.

to generate, and constraint the model to generate text from the input only. Therefore, the generated text can be considered as the extraction result. Our design make best use of the knowledge from pre-trained language models without further fine-tuning the model on a different objective. Experimental results show that out approach achieves high precision and recall, and the extraction results could further boost the performance of other quantity-related task such as quantitative reasoning.

Our work could further encourage endeavors on the quantitative information task and may even provide a new perspective on designing open information extraction methods by leveraging text generation models.

Related Work

Quantitative Information Extraction Quantitative Information Extraction is a sub-task of Open Information Extraction. Prior studies fall into three categories: open information extraction (OIE)-based, syntactic parsing (SP)-based, and semantic role labeling (SRL)-based approaches. For OIE-based approaches, Saha, Pal, and Mausam (2017) propose BONIE that uses bootstrapping to learn the dependency patterns to extract numerical relation triples. SP-based approaches extract numerical information from a syntax tree built by dependency parsing (DP) or constituency parsing (CP). DP-based methods (Hundman and Mattmann 2017) rely on dependency pattern matching, which is complex and might fail to handle quantitative information with long-range dependency. Alonso and Sellam (2018) extract small pieces of text which contain quantities from the constituency parsing tree. We argue that small pieces of text are not sufficient to fully understand the quantities, more details like the unit, quantity value, date, and some context are needed. In the SRL-based approach, quantitative information is considered as different types of semantic roles (Lamm et al. 2018a), and can be extracted using the SRL model trained on manually labeled data (Lamm et al. 2018b). More recently, Ho et al. (2019) propose a sequence tagging model trained on the data collected from extraction results of OpenIE using distant supervision. It is worth noting that the model trained with distant supervision only handles simple quantitative semantic roles, for fine-grained quantitative semantic roles (Lamm et al. 2018a), it will be labor-intensive to acquire the training data. (Ravichander et al. 2019) extract NUMSETs, which are grounded representations for quantity mentions, to serve the purpose of quantitative reasoning.

Text Generation Generation tasks has been widely studied recently. There are different variants of generation tasks, such as text generation, text-to-text generation, and data-to-text generation. Early generation systems can be divided into a pipeline of two sub tasks: content selection and surface realization (Reiter and Dale 1997). Recent studies tend to train neural models in an end-to-end fashion (Wiseman, Shieber, and Rush 2017; Liu et al. 2018; Wiseman, Shieber, and Rush 2018; Chen et al. 2020; Zeng et al. 2018; Zhang et al. 2020). More recently, large pre-trained models (Rothe, Narayan, and Severyn 2020; Raffel et al. 2020; Lewis et al. 2020) have also achieved new state-of-the-art results on generation tasks. To leverage a generation model for information extraction, the results can not be open-ended, which means the generation process should be more controlled. Controlled text generation is also a hot research area. It considers controlling attributes, such as identity of the speaker (Li et al. 2016), sentiment (Dou et al. 2018), tense (Hu et al. 2017), politeness (Sennrich, Haddow, and Birch 2016) and text length (Kikuchi et al. 2016). ToTTo (Parikh et al. 2020) propose content control on the topics of generated text. Lu et al. (2022) propose a unified text-to-structure generation framework for different information extraction tasks (UIE). Our work could be considered as a middle-ground between text-to-text generation and quantitative information extraction. Different from UIE, our work is based on a text-to-text generation framework to fully leverage pre-trained language models.

Task Definition and Data Construction

Task Definition

The RDF model (Lassila et al. 1998) represents each fact as a <subject, predicate, object> triple. Lamm et al. (2018a) define the quantitative semantic role labeling (QSRL) schema

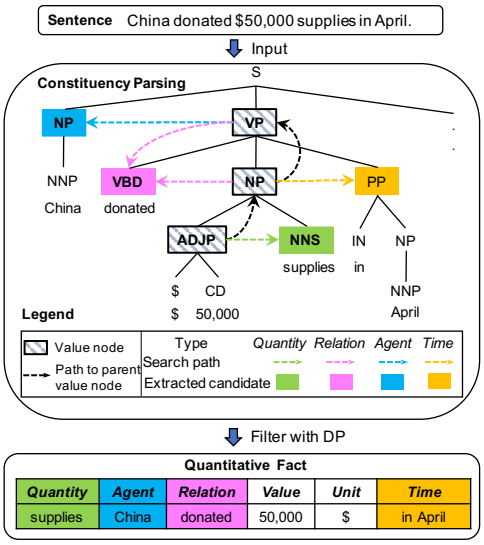

Figure 2: Search *Quantitative Facts* from the CP structures.

for quantitative fact extraction. Following QSRL, *quantity* answers the question "what does this number measure?", and we distinguish *quantity* from the *value* it takes on and the *unit* in which it is measured. As time expressions co-occur with numbers frequently, we introduce the *time* argument, which is the time when a *quantity* takes on a particular *value*. Meanwhile, we maintain the subject and predicate concept from the RDF structure to build the *agent* and *relation* argument. *agent* plays a direct role in influencing or motivating a *quantity*; *relation* here is similar to the relational phrase in OIE, which indicates the relation between *agent* and *quantity*.

In summary, for the input text, we define results of quantitative information extraction as a set of *quantitative facts*, each *quantitative fact* is a tuple of 6 arguments: <*quantity*, *agent*, *relation*, *value*, *unit*, *time*>. Our definition of *quantitative fact* is mainly syntactic-based and can be considered as an extended RDF schema or a simplified QSRL schema which preserves the most common quantitative semantic roles. Such schema allows us to perform basic *quantitative fact* extraction by leveraging syntactic features.

Building Quantitative Facts from Scratch

To build *quantitative facts* from scratch without manually labeling data, we propose a search-based approach on the constituency parsing tree, which has two advantages over the approaches that are based on DP rule matching: 1) searching on the CP tree is simpler and more robust than designing DP matching rules, 2) the extraction results are self-contained because DP rules are at the token level while quantitative facts are usually phrases. The idea of searching on the CP tree might also inspire new approaches in information extraction. In addition, we introduce several constituent-level dependency patterns to filter out the none-quantitative fact candidates. For all input text, the constituency and dependency parsing trees are obtained using (Zhou and Zhao 2019), and we use *Recognizers-Text*[1], a rule-based tool, to extract *values*, *time*, and *units*.

Constituency parsing breaks text into phrases or words according to the phrase structure grammar. Non-terminal nodes of the tree are phrases while terminals are words. An observation is that for a given numerical value, the quantitative facts are usually its siblings or siblings of its ancestors in the constituency parsing tree. Candidates with close distance are more likely to be the extraction results. Thus, the generic extraction process is quite simple: **searching quantitative fact candidates on the constituency parsing tree starting from the numerical value in bottom-up order**. Figure 2 shows the overall search process starting from "50,000". Each search rule consists of "the path to the parent value node" (optional) and "the specific search path" (the colored arrow).

We consider a non-terminal node as a "value node" or "time node" if it contains a numerical value or time expression extracted by *Recognizers-Text*. The slashed squares in Figure 2 represent the value nodes. For simplicity, we define "value mention" as the least ancestor that contains the numerical value ("$50,000" is a value mention in Figure 2). Note that given the *value*, the value mention is the least value node in the constituency parsing tree.

♦ **Quantity Extraction:** *quantity* is what the numerical *value* measures, it should be close to the value mention. We find that there are two common cases: 1) *quantity* exists within the value mention or the parent node of the value mention; 2) *quantity* exists at the same level or the ancestor level of the value mention, and is positioned next to the value node. For 1), we introduce the search rule S_1 as:

Search Rule 1 (S_1)**.** Searching nouns with their modifiers right to the *value* from the value mention or the parent node of the value mention.

In Figure 2, "supplies" will be the *quantity* candidate matched by S_1 because it is the noun from the parent node of the value mention "$\$50,000$". For sentences of the second type, the *value* may measure actions or events. For example, in the sentence "it cost us $50 to ship these supplies", "to ship these supplies" is the *quantity* of "50". As such, the *quantity* will be a verb phrase (VP). Prepositional phrases (PP) could also be considered as *quantity* candidates. For example, "of supplies" in "$500 of supplies" and "for medical emergency" in "$2,000 for medical emergency" are the *quantity* candidates. Therefore, we introduce the search rule S_2 as:

Search Rule 2 (S_2)**.** Searching the right closest sibling (or cousin) NP, VP, or PP nodes of value mention and all its ancestor value nodes.

It is worth noting that the output of S_1 is one candidate and that of S_2 is a collection of candidates in bottom-up level order. Some of the candidates are not valid, because they might be value nodes of a different *value*, or they could be time nodes. For example, for "218,590" in the sentence "There were 218,590 people, 79,667 households, and 60,387 families residing in the county.", "people" will be matched by S_1, "79,667 households" and "60,387 families" will be

[1]https://github.com/microsoft/Recognizers-Text

matched by S_2, and we shall filter out "79,667 households" and "60,387 families" because they are not the *quantities* of "218,590". Therefore, we design the filter rule F_1 as:

Filter Rule 1 (F_1)**.** Filtering out candidates that are NP value nodes or time nodes.

Because constituency parsing does not express relations between constituents, we introduce some **constituent-level** dependency rules to filter out the non-*quantity* candidates.

Filter Rule 2 (F_2)**.** Filtering out the candidate if its dependency relation with the value mention is not *nsubj* or *nsubjpass* and is not "close".

A dependency relation is "close" if one constituent is the dependent (directly or indirectly) of the other or the shortest dependency path is less than two. For example, in the sentence "The oil price increased by 10% in January but fell below 20 dollars a barrel in April.", "a barrel" will be filtered out as it is not a "close" dependency relation to "10%" though they are cousins in the CP tree.

The overall process is we first search the candidate (denoted by c_1) with S_1, then we search candidates (denoted by C_2) with S_2 in bottom-up order and filter using F_1 and F_2. After that, We obtain the ordered set of candidates $\mathbb{C} = \{c_1\} \bigcup C_2$. To extract self-contained quantity, we further merge the candidates in $\mathbb{C}$ that are adjacent in the original text and share the same phrase-level parent in the constituency parsing tree. The first remained candidate is the *quantity*. For example, for the *value* "500,000" in sentence "500,000 coronavirus test kits donated by Jack Ma arrived in the US", c_1 is "coronavirus test kits" (nouns) and C_2 is {"donated by Jack Ma" (VP), "arrived in the US" (VP)}. After merging, the first remained candidate (*quantity*) will be "coronavirus test kits donated by Jack Ma".

♦ **Relation Extraction:** The *relation* here is similar to the relational phrase in OIE, which is usually a verb with and the preposition after it (if any). The verb indicates how the *value* is changed, such as "increased", or the action that is taken on the *value*, such as "donated". The preposition after the verb is also important, for example, it could help distinguish between absolute and relative change, such as "to" in "increased to" and "by" in "increased by". In the constituency parsing tree, *relation* is usually in the least verb phrase (VP) ancestor of the value mention (Jiang and Diesner 2019). Thus, the search rule for extraction is quite simple, we search verb and the preposition after it from the least VP ancestor of the value mention as *relation*.

♦ **Agent Extraction:** *agent* plays a direct role in influencing or motivating a quantity, and it is usually to the left of the *relation*. We design the search rule S_3 for agent extraction as:

Search Rule 3 (S_3)**.** The left closest sibling (or cousin) NP node of the *relation*.

We further apply the filter rule F_2 to the output of S_3 to obtain the extracted *agent*.

♦ **Time Extraction:** We extract *time* from the value mention or the first sibling or cousin time node of value nodes in the bottom-up search order.

Summary

For each *value*, we extract the quantitative semantic roles via searching on the constituency parsing tree starting from its value mention in bottom-up order. The search rules utilized are quite simple. As a benefit of constituency parsing, our approach could partially resolve the ellipsis for text with multiple quantities. For example, in the sentence "revenues for private commercial stations increased from $1.4 billion to $1.5 billion", the correct *agent* "revenues for private commercial stations" for both "$1.4 billion" and "$1.5 billion" could be extracted. While for DP rule-matching, it requires complex dependency patterns, which will inevitably introduce errors. The extraction results from the search-based approach can be considered as distant supervision for the generation model.

Infomation Extraction via Text Generation

Due to the lack of manually labeled data and existing open datasets, fully training a classic (i.e. sequence labeling) information extraction model can be difficult. Recently, large-scale pre-trained language models have achieved state-of-the-art results in various tasks by fine-tuning on task-specific data. However, the fine-tune and pre-train objectives are not strictly aligned. Language models are mainly pre-trained with the Masked Language Modeling (MLM) objective, while the downstream tasks use task-specific heads such as classification for fine-tuning.

Under the low-resource setting, how to better leverage the knowledge from the pre-trained model becomes particularly important. Models with task-specific heads enjoy the low-level knowledge of the pre-trained language model to achieve better performance. We argue that only low-level knowledge is insufficient for low-resource tasks lack of training data. Prior studies on prompt-tuning suggests that the cloze prompt (Schick and Schütze 2021) and the prefix prompt (Li and Liang 2021; Lester, Al-Rfou, and Constant 2021) are forms that can fully leverage the language models, as these forms strictly match the pre-train objectives.

Inspired by TWT (Li et al. 2021), a data-to-text task conditioned on both the input data and a given prefix, we transform the information extraction task into a text generation task. The decoder of the text generation model is fed with a manually constructed prefix as a hint or control factor, which allows the model to generate the following text conditioned on the prefix. The generated text are considered as the extraction result. For the example in Figure 1, to extract the *quantitative facts* from text "The prince of Ning was said to have an army of over 80,000 men.". We first input the text into the encoder of the model. Then we construct a prefix "The agent of 80,000 is" with one numerical *value* included. Finally, the decoder of the model generates the following text "the prince of Ning" as the *agent* of 80,000. By changing the prefix, we can extract all the arguments of one *quantitative fact* and all the *quantitative facts*. Low resources quantitative information extraction benefits from such transformation, due to:

- **Open-ended Target**: Different from the relation extraction task, the information extraction task is not a classifi-

cation problem. The arguments are open-ended, making the task a close match to text generation.

- **Fixed Prefix Template**: Unlike prompt tuning, which requires prompt construction based on different inputs and answer construction based on different labels. The prefix of our method is rather fixed and the answer can be considered as the result directly without mapping to true labels, which decreases the loss from these stages.
- **Pre-acquired Numerical Values**: The numerical *value* can be pre-acquired easily with heuristic rules or other tools, making the *value* an known argument for the prefix. The *value* may serve as the knowledge to guide the model "generate" results of other unknown arguments.

Model Overview

We adopt a transformer-based model (Rothe, Narayan, and Severyn 2020) with the complete encoder-decoder structure. Both the encoder and decoder are initialized with pre-trained parameters. The text to be extracted is the input of the encoder and the decoder generates the extraction results. We design different forms of prefixes to allow a generation model serve the information extraction purpose. For the decoder, we employ different decoding strategies to accelerate decoding speed or gain higher performance. We also introduce the copy mechanism and a source tracing approach to ensure the generated extraction result can be strictly aligned and back-traced to the input.

Prefix Design

Given the input data x, we design a natural language prefix template "the arg of $value$ is", where $value$ is one of *VALUES* $= \{value_1, value_2, ..., value_n\}$ extracted by *Recognizers-Text* from x, and arg is one of *ARGS* $= \{quantity, agent, relation, time\}$. The prefix construction process to build a set of prefixes $\mathbb{P} = \{p_1, p_2, ..., p_n\}$ can be described as Algorithm 1.

Decoding Strategy

To leverage the prefix for extraction purposes, the decoding strategy should be designed to generate extraction results instead of next words. Typically, the extraction results are labeled as tagged sequences(Panchendrarajan and Amaresan 2018) or span-based segments(Li et al. 2019). Such labeling schema allow the model to extract all results at the same time. For generation-based methods, the results are decoded in an auto-aggressive style. In order to complete a full extraction, the model is required to complete n different generations, where n equals to the amount of arguments to extract. For example, to extract the *quantity*, *agent*, *relation* and *time* for each *value*, the model is required to generate 4 times, where each time the result for a different argument is extracted. To address this problem, as shown in Figure 1, we design two different decoding strategies to complete a full extraction.

Algorithm 1: Prefix Construction

Input: T
Parameter: *VALUES*, *ARGS*
Output: $\mathbb{P} = \{p_1, p_2, ..., p_n\}$

```
Initialize ℙ ← {}
for each value ∈ VALUES do
    for each arg ∈ ARGS do
        p_i = the arg of value is
        Update ℙ ← ℙ + p_i
    end for
end for
return ℙ
```

To maintain a matching inference speed with the traditional extraction models, we adopt a batch-decoding strategy for prefixes with different lengths. We left pad all the prefixes from $\mathbb{P}$ for each *value* to the same length with the token representing the end of sequence (EOS token). The batched prefixes are input into the model and the results are generated in parallel:

$$\mathbb{P}_{vocab}(w) = softmax(f([p_1; p_2; ...; p_n], c_t))$$

$\mathbb{P}_{vocab}(w)$ is a set of next word probabilities conditioned on different prefixes from $\{p_1, p_2, ..., p_n\}$. c_t is the context vector. To gain higher extraction performance, we design a back-tracking strategy for each argument to be extracted. We build a prefix including all history extraction results as prior knowledge for the next generation to back-track:

$$P_{vocab}(w_n) = softmax(f([p_1, o_2, p_2, ..., p_m, o_m], c_t))$$

P_{vocab} is the probability of the next word w_n conditioned on the prefix. o_m is the history generated tokens conditioned on prefix p_m. The more accurate the prior extractions are, the higher performance the following extractions acquire.

Extraction Confidence and Source Tracing

Generation models tend to generate open-ended targets with hallucinated content which is not faithful to the input (Xiao and Wang 2021). For information extraction, we argue that the generated text should be more than faithful to the input. The concept of faithful refers to the generations is relevant to the input, but the expressions (how to say it) can be different. Text with the same semantics as the ground truth but realized differently cannot be considered as a true positive, as it is not a hard alignment with the input text.

To maintain a high extraction confidence, where the generated results can be aligned to the input text, we adopt a copy strategy. Li and Wan (2018) introduced the copy mechanism (Oriol, Meire, and Navdeep 2015; Gu et al. 2016) to improve faithfulness of the generations. The copy mechanism is mainly proposed to address the out-of-vocabulary (OOV) problem. For the extraction-purposed generation, we expect the text to be generated in an out-of-vocabulary way. Specifically, the generated tokens after the prefix should originate from the vocabulary of the input text, while maintaining the generation abilities of pre-trained language models.

To achieve such goal, we adopt the copy mechanism. For an encoder-decoder model, the context vector is $h_t^* = \sum_i a_i^t h_i$, where a_i^t denote the encoder-decoder attention weight. Let $P_{vocab}(w)$ denote the probability of generating

a token w, the final probability distribution over both the vocabulary and the input text will be:

$$P(w) = (1 - p_{copy})P_{vocab}(w) + p_{copy} \sum_{i:w_i=w} a_i^t$$

When $p_{copy} = 1$, the tokens are hard copied from the input text over a probability distribution represented by the encoder-decoder attention weight. p_{copy} is calculated using a binary classifier based on the decoder input word embedding x_t, the output of last decoder layer s_t, and the context vector weighted on the encoder-decoder cross attentions h_t^*. Additionally, we explicitly "teach" the model when to copy from the input text. We consider tokens of the extraction results in the golden target as copied tokens, denoted by V_a. Following Chen et al. (2020), we maximize the copy probability p_{copy} with an extra loss term at the copied tokens:

$$L = L_c + \lambda \sum_{w_j \in V_a} (1 - p_{copy}^j)$$

where L_c is the loss between the model's output and the target, w_j is the target token at position j. λ is a hyperparameter representing the weight for the copy loss. V_a is a set of vocabularies corresponding to the aligned words.

One remaining problem for generation-based extraction models is source tracing. As the extraction results are generated as pure text, tracing the source of the extraction result, i.e., the start and end positions of the extraction results from the input can be difficult. To trace the source of the results, we find the highest probabilities from the encoder-decoder weight between the generated result and the input tokens. Tokens from the input text with the highest probabilities are considered as the source of the extraction result.

Experiments

We first perform human evaluation on the search-based approach to examine it's performance on building high quality training data. Then we evaluate quantitative information extraction on two tasks: 1) *quantitative fact* extraction and 2) quantitative reasoning.

Search-based Approach Human Evaluation

We sample and label 200 sentences from the DROP dataset (Dua et al. 2019) and 200 sentences from the test data of BONIE (Saha, Pal, and Mausam 2017) with the schema of *quantitative fact*. Each sentence contains at least one numerical value. In total, we labeled 453 *agents*, 475 *relations*, 496 *quantities*, and 186 *time*, which is a total of 613 *quantitative facts*. We use partial match and exact match as the evaluation metrics, denoted by PM and EM, respectively. We compare our approach with a simplified search-based baseline Search$_s$ and two DP-rule-matching baselines: MARVE (Hundman and Mattmann 2017) and BONIE (Saha, Pal, and Mausam 2017). Search$_s$ extracts *quantity* from the value mention, *time* from the time node closest to the value mention, *relation* from the first VP value node in top-bottom order, and *agent* from the left sibling node of the *relation*.

For a fair comparison between BONIE (Saha, Pal, and Mausam 2017), MARVE (Hundman and Mattmann 2017), and our approach, we make some adjustments to the output schemas. BONIE outputs numerical triple <*arg1*, *relation*, *arg2*> with an extra *additional*. One of the *arg1* and *arg2* contains numerical value while the other is the subject or object, *relation* is similar to the relational phrase in OIE, and *additional* is the additional information of the sentence. MARVE outputs <*value*, *quantified*, *related*> where *quantified* is the object or concept measured by the *value*, and *related* is a collection of words or entities related to a measurement. For BONIE, we consider a correct extraction of *agent* or *quantity* if either *arg1* or *arg2* matches the labeled text. Some *relation* phrases from BONIE are normalized, we consider all these types of *relations* correct. For MARVE, we only evaluate *quantity*. Either *quantified* or *related* matches the labeled *quantity* will be considered as correct.

The left part of Table 1 shows that the search-based approach achieves the best overall performance, the recall is improved significantly. BONIE extracts empty results for 319 out of 400 sentences on our data, which is consistent with the original paper results (1512 out of 2000 sentences have empty results from the ClueWeb12)[2]. BONIE and MARVE perform poorly on sentences with multiple numerical values. The search-based approach has significantly better results under the EM metric, the reason is that DP-based pattern matching works at token-level while the quantitative facts are mostly phrases. It should be noted that the searching rules are quite simple, which also indicates the effectiveness of search-based approaches in information extraction as simple methods often work better than complex ones. The significantly improved recall and performance on extractions makes the search-based approach a strong baseline for building training data.

Quantitative Information Extraction

As the search-based approach produces high quality quantitative facts, we consider the quantitative facts labeled data. Therefore, we are able to train the generation-based quantitative information extraction model without manually labeled data. Theoretically unlimited amount of labeled data can be produced with such approach. To further examine the performance of our model under the low resource setting, we only search the quantitative facts from 1,000 sentences. Altogether 4000 quantitative facts are produced. We roughly split the sentences and quantitative facts with the ratio of 8:1:1 for training, validation and testing.

We adopt T5 (Raffel et al. 2020), a pre-trained text-to-text using the transformer framework as the backbone of our model. T5 achieved state-of-the-art results on many text generation benchmarks, including ToTTo (Parikh et al. 2020). For our model, we adopt the back-tracking prefix, as the batch-decoding prefix is mainly designed to improve decoding speed, which is not a priority in this work. We compare our model with a vanilla T5-based generation model by introducing the prefix only at inference time. We train both the

[2]https://github.com/dair-iitd/OpenIE-standalone/tree/master/data

		Search		Search$_s$		BONIE		Marve		Baseline Gen		Our Gen	
		P	R	P	R	P	R	P	R	P	R	P	R
quantity	PM	0.84	0.91	0.66	0.82	0.57	0.15	0.82	0.28	0.81	0.75	0.86	0.88
	EM	0.64	0.89	0.33	0.70	0.34	0.10	0.31	0.11	0.73	0.46	0.80	0.60
agent	PM	0.85	0.94	0.67	0.98	0.80	0.20	-	-	0.86	0.82	0.90	0.90
	EM	0.58	0.91	0.57	0.97	0.72	0.18	-	-	0.74	0.39	0.84	0.55
relation	PM	0.89	0.97	0.44	0.82	0.85	0.21	-	-	0.71	0.42	0.83	0.71
	EM	0.74	0.97	0.26	0.73	0.69	0.17	-	-	0.62	0.28	0.80	0.56
time	PM	0.71	0.92	0.73	0.80	-	-	-	-	0.56	0.96	0.79	0.81
	EM	0.61	0.91	0.66	0.78	-	-	-	-	0.56	0.94	0.73	0.57

Table 1: Extraction results. The left part evaluates the search-based approach and the right part evaluates the gen-based models.

	RTE-Q	Δ	NewsNLI	Δ	RedditNLI	Δ	NR ST	Δ	AWPNLI	Δ	Nat. Avg. Δ	Synth. Avg. Δ	All Avg. Δ
MAJ	57.8	0.0	50.7	0.0	58.4	0.0	33.3	0.0	50.0	0.0	0.0	0.0	0.0
GPT	68.1	+10.3	72.2	+21.5	52.4	-6.0	36.4	+3.1	50.0	+0.0	+8.6	+1.6	+5.8
BERT	57.2	-0.6	72.8	+22.1	49.6	-8.8	36.9	+3.6	42.2	-7.8	+4.2	-2.1	+1.7
Q-REAS	56.6	-1.2	61.1	+10.4	50.8	-7.6	63.3	+30	71.5	+21.5	+0.5	+25.8	+10.6
+ Our method	59.6	+1.8	62.8	+12.1	55.2	-3.2	62.7	+29.4	68.6	+18.6	+3.6	+24.0	+11.7

Table 2: Accuracies(%) of quantitative reasoning. Δ captures the improvement over the baseline MAJ.

baseline and our approach with 8 NVIDIA Tesla V100 32G GPUs. Following (Raffel et al. 2020), we employ a constant learning rate of $1e-3$ with AdaFactor optimizer (Shazeer and Stern 2018). Decoding is conducted via greedy search. For other settings (including the baselines), the batch size is 128, and the maximum number of input and output tokens are 256 and 128, respectively. Tokens shorter or longer than the maximum length will be padded or truncated. For simplicity, the best checkpoint is chosen based on the BLEU metric on the validation set. We evaluate the precision and recall on the test set with the best checkpoint.

The right part of Table 1 shows the experiment results of the generation-based models. Our model achieves high performance on the data built with the search-based approach. The comparison with the baseline generation model suggests that the copy mechanism and the back-tracking decoding strategy are both effective designs. By examining specific cases, we find that the generation model may even remove some of the false positive results from the search-based approach. The results also demonstrate that with only a few amount of labeled data, the generation-based model is capable of completing extraction tasks. By adopting more data produced by the search-based approach, the generation model may achieve even better results. But we tend to prove the effectiveness of our model designs under the low resource setting, making our model also possible to work with a limited amount of manually labeled data.

Quantitative Reasoning

We further examine the effectiveness of the extracted quantitative facts on the downstream task: quantitative reasoning in natural language inference (NLI) on the EQUATE (Ravichander et al. 2019) dataset. EQUATE consists of 3 natural (RTE-Q, NewsNLI and RedditNLI) and 2 synthetic (StressTest and AwpNLI) test sets. Ravichander et al. (2019) design a strong heuristic baseline Q-REAS. We replace the quantity parser in Q-REAS with our search-based approach and compare the best results with the majority class (MAJ) baseline and the original Q-REAS.

Table 2 shows the comparison results. By only replacing the quantity parser with our method, Q-REAS achieves better accuracy on all the natural-sourced data with an improvement of 3.1%, while slightly underperforms on the synthetic-sourced data. The overall accuracy is improved by 1.1% compared to the original Q-REAS and is the best among all the baselines in Ravichander et al. (2019), including GPT (Radford et al. 2018) and BERT (Devlin et al. 2019) fine-tuned on NLI. The synthetic data is specifically designed for arithmetic problems with many ellipses and coreferences in *quantity* or *unit*, e.g., "Joan had 8 kittens, she gave 2 to Tom.", this leads to slightly decreased accuracy. Handling such synthetic data is not the focus of this paper. It should be noted that there are several cascade components in Q-REAS. The quantity parser of Q-REAS is the only component we changed, and it can be difficult to utilize all arguments in a *quantitative fact* for reasoning. The improved performance by only replacing the quantity parser proves the effectiveness of our approach.

Conclusion and Future Work

We propose a search-based approach to build quantitative facts from scratch and a prefix-based generation model that fully leverages pre-trained weights for quantitative information extraction. Our method could significantly improve the recall and the extraction could further boost the performance of quantity related downstream tasks. The idea of searching might inspire new approaches on how to build training data from a cold start and the generation schema may also encourage further endeavors on low-resource information extraction. For future work, we will focus on paragraph-/document-level extraction, and integrate the extraction for more downstream tasks such as reading comprehension.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (Grant Nos. 62276017, U1636211, 61672081), the 2022 Tencent Big Travel Rhino-Bird Special Research Program, and the Fund of the State Key Laboratory of Software Development Environment (Grant No. SKLSDE-2021ZX-18).

References

Alonso, O.; and Sellam, T. 2018. Quantitative Information Extraction From Social Data. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, 1005–1008.

Barkschat, K. 2014. Semantic Information Extraction on Domain Specific Data Sheets. In *11th Extended Semantic Web Conference (ESWC)*, 864–873.

Bast, H.; and Haussmann, E. 2013. Open Information Extraction via Contextual Sentence Decomposition. In *2013 IEEE Seventh International Conference on Semantic Computing*, 154–159.

Cetto, M.; Niklaus, C.; Freitas, A.; and Handschuh, S. 2018. Graphene: a Context-Preserving Open Information Extraction System. In *Proceedings of the 27th International Conference on Computational Linguistics: System Demonstrations*, 94–98.

Chen, Z.; Eavani, H.; Chen, W.; Liu, Y.; and Wang, W. Y. 2020. Few-Shot NLG with Pre-Trained Language Model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 183–190.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 4171–4186.

Dou, L.; Qin, G.; Wang, J.; Yao, J.-G.; and Lin, C.-Y. 2018. Data2Text Studio: Automated Text Generation from Structured Data. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 13–18.

Dua, D.; Wang, Y.; Dasigi, P.; Stanovsky, G.; Singh, S.; and Gardner, M. 2019. DROP: A Reading Comprehension Benchmark Requiring Discrete Reasoning Over Paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 2368–2378.

Evans, M. C.; Bhatia, J.; Wadkar, S.; and Breaux, T. D. 2017. An Evaluation of Constituency-Based Hyponymy Extraction from Privacy Policies. In *2017 IEEE 25th International Requirements Engineering Conference (RE)*, 312–321.

Gu, J.; Lu, Z.; Li, H.; and Li, V. O. 2016. Incorporating Copying Mechanism in Sequence-to-Sequence Learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 1631–1640.

Ho, V. T.; Ibrahim, Y.; Pal, K.; Berberich, K.; and Weikum, G. 2019. Qsearch: Answering Quantity Queries from Text. In *The 18th International Semantic Web Conference (ISWC)*, 237–257.

Hu, Z.; Yang, Z.; Liang, X.; Salakhutdinov, R.; and Xing, E. P. 2017. Toward Controlled Generation of Text. In *Proceedings of the 34th International Conference on Machine Learning*, 1587–1596.

Hundman, K.; and Mattmann, C. A. 2017. Measurement Context Extraction from Text: Discovering Opportunities and Gaps in Earth Science. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Data-Driven Discovery Workshop*.

Jiang, M.; and Diesner, J. 2019. A Constituency Parsing Tree based Method for Relation Extraction from Abstracts of Scholarly Publications. In *Proceedings of the Thirteenth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-13)*, 186–191.

Kikuchi, Y.; Neubig, G.; Sasano, R.; Takamura, H.; and Okumura, M. 2016. Controlling Output Length in Neural Encoder-Decoders. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 1328–1338.

Lamm, M.; Chaganty, A.; Jurafsky, D.; Manning, C. D.; and Liang, P. 2018a. QSRL: A Semantic Role-Labeling Schema for Quantitative Facts. In *First Financial Narrative Processing Workshop at LREC 2018*, 44–51.

Lamm, M.; Chaganty, A.; Manning, C. D.; Jurafsky, D.; and Liang, P. 2018b. Textual Analogy Parsing: What's Shared and What's Compared among Analogous Facts. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 82–92.

Lassila, O.; R. Swick, R.; Wide, W.; and Consortium, W. 1998. Resource Description Framework (RDF) Model and Syntax Specification.

Lester, B.; Al-Rfou, R.; and Constant, N. 2021. The Power of Scale for Parameter-Efficient Prompt Tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 3045–3059.

Lewis, M.; Liu, Y.; Goyal, N.; Ghazvininejad, M.; Mohamed, A.; Levy, O.; Stoyanov, V.; and Zettlemoyer, L. 2020. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 7871–7880.

Li, J.; Galley, M.; Brockett, C.; Spithourakis, G.; Gao, J.; and Dolan, B. 2016. A Persona-Based Neural Conversation Model. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 994–1003.

Li, L.; and Wan, X. 2018. Point Precisely: Towards Ensuring the Precision of Data in Generated Texts Using Delayed Copy Mechanism. In *Proceedings of the 27th International Conference on Computational Linguistics*, 1044–1055.

Li, T.; Fang, L.; Lou, J.-G.; and Li, Z. 2021. TWT: Table with Written Text for Controlled Data-to-Text Generation.

In *Findings of the Association for Computational Linguistics: EMNLP 2021*, 1244–1254.

Li, X.; Yin, F.; Sun, Z.; Li, X.; Yuan, A.; Chai, D.; Zhou, M.; and Li, J. 2019. Entity-Relation Extraction as Multi-Turn Question Answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 1340–1350.

Li, X. L.; and Liang, P. 2021. Prefix-Tuning: Optimizing Continuous Prompts for Generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 4582–4597.

Liu, T.; Wang, K.; Sha, L.; Chang, B.; and Sui, Z. 2018. Table-to-Text Generation by Structure-Aware Seq2seq Learning. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, 4881–4888.

Lu, Y.; Liu, Q.; Dai, D.; Xiao, X.; Lin, H.; Han, X.; Sun, L.; and Wu, H. 2022. Unified Structure Generation for Universal Information Extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 5755–5772.

Oriol, V.; Meire, F.; and Navdeep, J. 2015. Pointer Networks. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, 2692—-2700.

Panchendrarajan, R.; and Amaresan, A. 2018. Bidirectional LSTM-CRF for Named Entity Recognition. In *Proceedings of the 32nd Pacific Asia Conference on Language, Information and Computation*.

Parikh, A.; Wang, X.; Gehrmann, S.; Faruqui, M.; Dhingra, B.; Yang, D.; and Das, D. 2020. ToTTo: A Controlled Table-To-Text Generation Dataset. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 1173–1186.

Radford, A.; Narasimhan, K.; Salimans, T.; and Sutskever, I. 2018. Improving language understanding by generative pre-training. *OpenAI Blog*.

Raffel, C.; Shazeer, N.; Roberts., A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; and Liu, P. J. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research*, 1–67.

Rajpurkar, P.; Zhang, J.; Lopyrev, K.; and Liang, P. 2016. SQuAD: 100,000+ Questions for Machine Comprehension of Text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 2383–2392.

Ravichander, A.; Naik, A.; Rose, C.; and Hovy, E. 2019. EQUATE: A Benchmark Evaluation Framework for Quantitative Reasoning in Natural Language Inference. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, 349–361.

Reiter, E.; and Dale, R. 1997. Building applied natural language generation systems. *Natural Language Engineering*, 57–87.

Rothe, S.; Narayan, S.; and Severyn, A. 2020. Leveraging Pre-trained Checkpoints for Sequence Generation Tasks. *Transactions of the Association for Computational Linguistics*, 264–280.

Saha, S.; Pal, H.; and Mausam. 2017. Bootstrapping for Numerical Open IE. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 317–323.

Schick, T.; and Schütze, H. 2021. Exploiting Cloze-Questions for Few-Shot Text Classification and Natural Language Inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, 255–269.

Sennrich, R.; Haddow, B.; and Birch, A. 2016. Controlling Politeness in Neural Machine Translation via Side Constraints. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 35–40.

Seo, M.; Kembhavi, A.; Farhadi, A.; and Hajishirzi, H. 2017. Bidirectional Attention Flow for Machine Comprehension. In *5th International Conference on Learning Representations*.

Shazeer, N.; and Stern, M. 2018. Adafactor: Adaptive Learning Rates with Sublinear Memory Cost. In *Proceedings of the 35th International Conference on Machine Learning*, 4596–4604.

Wiseman, S.; Shieber, S.; and Rush, A. 2017. Challenges in Data-to-Document Generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2253–2263.

Wiseman, S.; Shieber, S.; and Rush, A. 2018. Learning Neural Templates for Text Generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 3174–3187.

Xiao, Y.; and Wang, W. Y. 2021. On Hallucination and Predictive Uncertainty in Conditional Language Generation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, 2734–2744.

Yu, A. W.; Dohan, D.; Luong, M.-T.; Zhao, R.; Chen, K.; Norouzi, M.; and Le, Q. V. 2018. QANet: Combining Local Convolution with Global Self-Attention for Reading Comprehension. In *6th International Conference on Learning Representations*.

Zeng, X.; Zeng, D.; He, S.; Liu, K.; and Zhao, J. 2018. Extracting Relational Facts by an End-to-End Neural Model with Copy Mechanism. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 506–514.

Zhang, R. H.; Liu, Q.; Fan, A. X.; Ji, H.; Zeng, D.; Cheng, F.; Kawahara, D.; and Kurohashi, S. 2020. Minimize Exposure Bias of Seq2Seq Models in Joint Entity and Relation Extraction. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, 236–246.

Zhou, J.; and Zhao, H. 2019. Head-Driven Phrase Structure Grammar Parsing on Penn Treebank. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2396–2408.