Automatic Entity-based Distractor Generation

Shunyao Li
Advisor: Prof. Teruko Mitamura
Language Technologies Institute
Carnegie Mellon University
Pittsburgh, PA 15213
shunyaol@andrew.cmu.edu

Abstract

SmartReader System requires a time-efficient, high quality distractor generator to test readers' understanding about English passages. This project proposed a novel way to generate entity-based distractors based on fine-grained question classification and BERT embeddings. The automatic entity-based distractor generation system can be initialized in 10 seconds and the averaging time consumed to generate three distractors for a question-answer pair is 2.3s. The system also beats baseline model on overall, grammar and semantics score in human evaluation.

1 Introduction

The SmartReader System is a tool to help English as a Second Language (ESL) learners develop their reading comprehension skills in English. It consists of three modules including question generation, answer generation and distractor generation. In reading comprehension tests, distractors play a significant role in testing ESL learners' understanding of the passage as well as the ability to locate the correct information.

Distractors are misleading wrong answer options, and sometimes true in common sense but not in the case of a specific passage. Specifically, a "good" distractor[1] should be at least semantically related to the key, grammatically correct given the stem, and consistent with the semantic context of the stem. As discussed in the previous meeting, distractor generation can be categorized into entity-based distractor generation and event-based distractor generation. For entity-based distractor generation, a good distractor should also appear in the passage. In the scenario of event-based distractor generation, questions are looking for events that are not captured by named entities. A good distractor should be semantically related to the context and the correct answer.

Two LTI students have worked on distractor generation task for SmartReader System. Goutam Nair implemented entity-based and event-based distractor generation modules[5]. In practice the initialization process requires several hours since it have to query WikiData by all possible distractor candidates. Furthermore, Goutam used Word2Vec embedding similarities when ranking candidates, so there are plenty of room to improve the ranking process by contextual embeddings including BERT.

Chentao Ye worked on automatic distractor generation based on event relation distance and automatic distractor evaluation[6, 7]. Event-based distractor generation and automatic distractor evaluation are promising future work directions.

2 Methods

The pipeline to implement entity-based distractor generation module involves the following steps.

Syntactic Map Extraction	*Question Rewrite*	Rewrites questions that are in non-standard form.
	Parse Tree Analysis	Extract structure information from the question using Constituency-based parse trees
Word, Phrase and Entity Extraction	*Headword Extraction*	Extract headwords from noun phrases in the question using **a)** Possessive Unrolling **b)** Preposition Rolling **c)** Entity Identification
	Verb, Wh-word and Adjective Extraction	Extract the Auxiliary and Major Verbs, the Wh-word and all adjectives from the question.
Rule-based Classification	*Match Rules based on the Question Syntax and Word Type*	Using a hierarchy of syntactic positions in a question, iteratively check to see if there exists a rule for mapping the word at that position to a QC.

Figure 1: Madabushi and Lee's system pipeline.

1. Determine question type. A high accuracy rule-based question classification algorithm is applied.
2. Determine golden answer type. This step is only possible when there exists a true answer. For questions without a true answer, or the true answer is "none of the above", the step can be skipped.
3. Named entity recognition by spaCy.
4. Combine question type and answer type as target class. Select entities from the text that are of the same class as distractors candidates.
5. Feature extraction and ranking. Features including BERT sentence embedding similarities are extracted and used for ranking.

2.1 Fine-grained Question Classification

Li and Roth[2] introduced the task of question type classification in 2002 and defined a question taxonomy in both coarse and fine grains as shown in table 1[1]. The taxonomy has been a standard metrics in question classification and followed by successor works.

Madabushi and Lee proposed a high accuracy rule-based question classification model using question syntax and semantics[4]. Their system (figure 1) consists of three parts: **a)** extracting a Question's Syntactic Map, **b)** identifying the headword, of the noun phrase in the question, while handling Entity Identification and phrase detection, and **c)** using rules to map words at different positions in the Syntactic Map to identify the Question Classification. These are further broken down into the following steps as shown in the middle and right part of the figure. The rule-based model proposed by Madabushi and Lee can achieve a fine-grained question classification accuracy of 97.2% the dataset TREC 10 dataset, significantly outperformed other classifier based models.

Madabushi and Lee also provided an API to access the question classification system[2] We accessed the question classification system by API for fine-grained question classification.

2.2 Named Entity Recognition

To collect distractor candidates appeared in passage, it is necessary to preprocess the passage and do Named Entity Recognition (NER). We applied spaCy, an industrial-strength natural language processing open-source library. spaCy NER labels and descriptions are listed in table 2. The accuracy of spaCy NER is reported as 92.6% in its official website[3].

Furthermore, we check if the golden answer is a named entity. If so, the named entity type is treated as the answer type. Otherwise, the answer type is "None".

[1] A detailed definition of question classes. `https://cogcomp.seas.upenn.edu/Data/QA/QC/definition.html`

[2] Question classification API documentation. `http://www.harishmadabushi.com/research/questionclassification/question-classification-api-documentation/`

[3] `https://spacy.io/usage/facts-figures`

Coarse	Fine
ABBR	abbreviation; expansion
DESC	definition; description; manner; reason
ENTY	animal; body; color; creation; currency; disease; event; food; instrument; language; letter; other; plant; product; religion; sport; substance; symbol; technique; term; vehicle; word; extraterrestrial
HUM	description; group; individual; title; indgr
LOC	city; country; mountain; other; state
NUM	code; count; date; distance; money; order; other; percent; period; speed; temperature; size; weight

Table 1: Question Taxonomy introduced by Li and Roth.

Label	Description
PERSON	People, including fictional.
NORP	Nationalities or religious or political groups.
FAC	Buildings, airports, highways, bridges, etc.
ORG	Companies, agencies, institutions, etc.
GPE	Countries, cities, states.
LOC	Non-GPE locations, mountain ranges, bodies of water.
PRODUCT	Objects, vehicles, foods, etc. (Not services.)
EVENT	Named hurricanes, battles, wars, sports events, etc.
WORK_OF_ART	Titles of books, songs, etc.
LAW	Named documents made into laws.
LANGUAGE	Any named language.
DATE	Absolute or relative dates or periods.
TIME	Times smaller than a day.
PERCENT	Percentage, including ”%“.
MONEY	Monetary values, including unit.
QUANTITY	Measurements, as of weight or distance.
ORDINAL	“first”, “second”, etc.
CARDINAL	Numerals that do not fall under another type.

Table 2: spaCy Named Entity Recognition labels and descriptions.

2.3 Combine Question Type and Answer Type

Obviously, question taxonomy in table 1 and NER labels in table 2 differs. To combine these two classification standards for distractor extraction, we proposed an algorithm shown in Algorithm 1. *fine_tag* refers to question types and *spacy_tag* refers to NER tags. The algorithm's motivation is to union two sets of tags when both tags are available. This approach can potentially correct mistakes made by one of two systems and hence improve quality of distractor generation, as discussed in section 4.

We manually created a conversion table between spaCy tag and fine-grained question type, as shown in table 3. Both match and convert functions in the algorithm follow rules in the conversion table.

2.4 Feature extraction and Ranking

Liang et al.[3] investigated into distractor generation for multiple choice questions using learning to rank. Their proposed models can learn to select distractors that resemble those in actual exam questions, which is different from most existing unsupervised ontology-based and similarity-based methods. They empirically studied feature-based and neural net (NN) based ranking models with experiments on SciQ dataset and MCQL dataset. Experimental results showed that feature-based ensemble learning methods (random forest and LambdaMART) outperformed both the NN-based method and unsupervised baselines. Specifically, they proposed 10 features: *Emb Sim, POS Sim, Edit Dist, Token Sim, Length, Suffix, Freq, Single, Num* and *Wiki Sim*.

Algorithm 1 Combine Question Type and Answer Type

```
if fine_tag and spacy_tag:
    if match(fine_tag, spacy_tag):
        return extract(spacy_tag)
    else:
        return extract(spacy_tag) union extract(convert(fine_tag))
else if fine_tag:
    return extract(convert(fine_tag))
else if spacy_tag:
    return extract(spacy_tag)
else:
    do nothing
```

spaCy Tag	Fine-grained Question Type
PERSON	HUM.all
NORP	ENTY.religion, ENTY.lang
FAC	ENTY.all
ORG	ENTY.other
GPE	LOC.all
LOC	LOC.all
PRODUCT	ENTY.all
EVENT	ENTY.event, ENTY.sport
WORK_OF_ART	ENTY.cremat
LAW	ENTY.all
LANGUAGE	ENTY.lang
DATE	NUM.date, NUM.period
TIME	NUM.period
PERCENT	NUM.percent
MONEY	NUM.money
QUANTITY	NUM.distance, NUM.speed, NUM.temp, NUM.size, NUM.weight
ORDINAL	NUM.order
CARDINAL	NUM.count

Table 3: Conversion table between spaCy tag and fine-grained question type.

Considering the difference between actual exam questions (SciQ and MCQL datasets) and SQuAD dataset, only part of these features are applied in our system.

- BERT Embedding
  Embedding cosine similarity between question(q) and distractor(d)
  Embedding cosine similarity between answer(a) and distractor(d)
- POS Tagging
  Jaccard similarity between a and d's simple POS tags
  Jaccard similarity between a and d's detailed POS tags
- Edit Distance
  Edit distance between a and d
- Token Similarity
  Jaccard similarities between a and d's tokens
- Character Level Length
  the difference of a and d's character lengths

We replaced Word2Vec embwdding similarity with BERT embedding similarity using 768 dimension sentence vector returned by bert-as-service[4].

[4]bert-as-service. `https://github.com/hanxiao/bert-as-service`

Instructions
1. Please rate baseline distractors and generated distractors on a scale of 1 (worst) - 5 (best) in three aspects: grammar, semantic and overall. 5 - Strong distractors, very misleading. 4 - Plausible distractors, can have minor errors. 3 - Possible distractors, can be easily identified as incorrect. 2 - Weak distractors, somewhat unrelated to the question/answer. 1 - Empty distractors or poor distractors completely unrelated to the question. 2. The golden answer can be "None" when the question is unanswerable. 3. Grammar Score: Grammatically correct given the question. 4. Semantics Score: Semantically related to the question and the golden answer (if there is an answer). 5. Overall Score: Overall evaluation.

Table 4: Instructions for human evaluators.

Model	Grammar	Semantics	Overall
Baseline	2.75	2.76	2.79
Proposed Model	**4.29**	**4.16**	**4.18**

Table 5: Human evaluation result.

3 Experiment and Result

Experiments were conducted on SQuAD 2.0 dateset. It combines the 100,000 questions in SQuAD1.1 with over 50,000 unanswerable questions written adversarially by crowdworkers to look similar to answerable ones. Distractors were generated for first 60 articles in the SQuAD 2.0 training set (442 articles in total), including 21k questions, both answerable and unanswerable. Top 3 candidates with highest scores were used as generated distractors. We also used 'Q_Coarse', 'Q_Fine', 'Gold_spaCy', 'Candidate_Type' to track how distractors were generated for error analysis. As for speed, initialization finished in 10s. Generating distractors for a question took 2.3s on average, while the question classification consumed more than 95% of the time. For baseline, we applied a model without fine-grained question classification and ranking by Word2Vec embedding similarity.

60 question-answer pairs and corresponding generated distractors were randomly selected for human evaluation. Each pair was evaluated by exactly 3 evaluators. SQuAD dataset has extremely long articles, so it is impossible to acquire evaluators to read the related article. Evaluators are given question, answer, and three generated distractors. All distractors existed in the article. Instructions provided with human evaluators are shown in table 4.

The human evaluation results are listed in table 5. There is a huge gap between baseline and our proposed model. The main reason is that baseline model cannot generate distractors for about half question-answer pairs because either the question is unanswerable or the answer does not have a valid NER tag. For these pairs, baseline model only gets 1 out of 5, so the average score for baseline is relatively low. Meanwhile, grammar, semantics and overall scores of proposed model falls in to the interval between 4 and 5. According to evaluation instructions, 4 points stands for plausible distractors and 5 for strong candidates. Therefore, distractors generated by out model are of high quality and very misleading.

4 Error Analysis

Three distractor generation examples are shown in figure 2. In the first example, expected distractor type is an amount of money but spaCy NER made a mistake and recognized "US$234 million" as an organization. However, question classification made a correct decision as "NUM". As a result, both organization and money are candidate distractor types according to the union operation in section 2.3. This is an example of correcting mistake made by one of two classification systems.

Question	What dollar amount of chocolate does New York export annually?	What part of China did the earthquake occur in?	What are two radio stations that broadcast from KU?
Gold	US$234 million	rural part	KJHK, 90.7 FM, and KANU, 91.5 FM
Baseline	['The New School', 'The Port of New York', 'the State of New York']	[]	['Kansas City', 'Overland Park', 'America']
Model	['US$100 million', 'US$510 million', 'US$3.2 billion']	['SilkAir', 'CCTV-1', 'the Tibet Hotel']	['the Beach Center on Disability, Lied Center of Kansas', 'The KU Memorial Unions Corporation', 'the Lawrence Campus']
Q_Coarse	NUM	ENTY	ENTY
Q_Fine	other	product	product
Gold_spaCy	ORG	None	GPE
Candidate_Type	{'ORG', 'CARDINAL', 'MONEY'}	{'PRODUCT'}	{'GPE', 'PRODUCT'}
Baseline_Grammar	1.0	1.0	2.3
Baseline_Semantic	1.0	1.0	1.3
Baseline_Overall	1.0	1.0	1.3
Model_Grammar	5.0	3.3	2.0
Model_Semantic	5.0	2.67	1.3
Model_Overall	5.0	2.67	1.3

Figure 2: Examples for error analysis.

In the second example, no spaCy tag is provided because the answer "rural part" is not recognized as a named entity. Question classification model made a mistake. Correct question type should be geographical location but it produced "product" instead, resulting low model scores.

In the third example, both spaCy NER and question classification made mistakes. Therefore, the performance of our model is limited by accuracy of spaCy NER and question classification model.

5 Conclusion and Future Work

This project proposed a novel way to generate entity-based distractors based on fine-grained question classification and BERT embeddings. The automatic entity-based distractor generation system can be initialized in 10 seconds and the average time to generate three distractors for a question-answer pair is 2.3s. The system significantly outperforms baseline model on overall, grammar and semantics scores in human evaluation.

For future work, there are two possible directions. One is to investigate in event-based distractor generation and automatic distractor evaluation. Another is to research Neural Network based approach inspired by state-of-art distractor generation models, especially hierarchical encoder-decoder framework with static and dynamic attention mechanisms.

Acknowledgments

We gratefully acknowledge Ziqian Luo, Yanwen Lin, Zanpeng Ma, Xiangyu Lin, Jianfeng Xia for their time and efforts in human evaluation.

References

[1] GOODRICH, H. C. Distractor efficiency in foreign language testing. *Tesol Quarterly* (1977), 69–78.

[2] LI, X., AND ROTH, D. Learning question classifiers. In *Proceedings of the 19th international conference on Computational linguistics-Volume 1* (2002), Association for Computational Linguistics, pp. 1–7.

[3] LIANG, C., YANG, X., DAVE, N., WHAM, D., PURSEL, B., AND GILES, C. L. Distractor generation for multiple choice questions using learning to rank. In *Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications* (2018), pp. 284–290.

[4] MADABUSHI, H. T., AND LEE, M. High accuracy rule-based question classification using question syntax and semantics. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers* (2016), pp. 1220–1230.

[5] NAIR, G. Automatic entity and event-based distractor generation for the smartreader system. *Goutam Nair's MIIS directed study report*.

[6] YE, C. Automatic distractor evaluation. *Chentao Ye's MIIS directed study report*.

[7] YE, C. Automatic distractor generation based on event relation distance and semantic similarity. *Chentao Ye's MIIS directed study report*.