Final Project Write Up: Software Engineer Final Project Team 15

Team members: Shen Wu, Mengda Yang , Yi Song , Qiwen Zhu, Ying Sheng

Git repository URL:
https://github.com/Mengda/hw5-team15

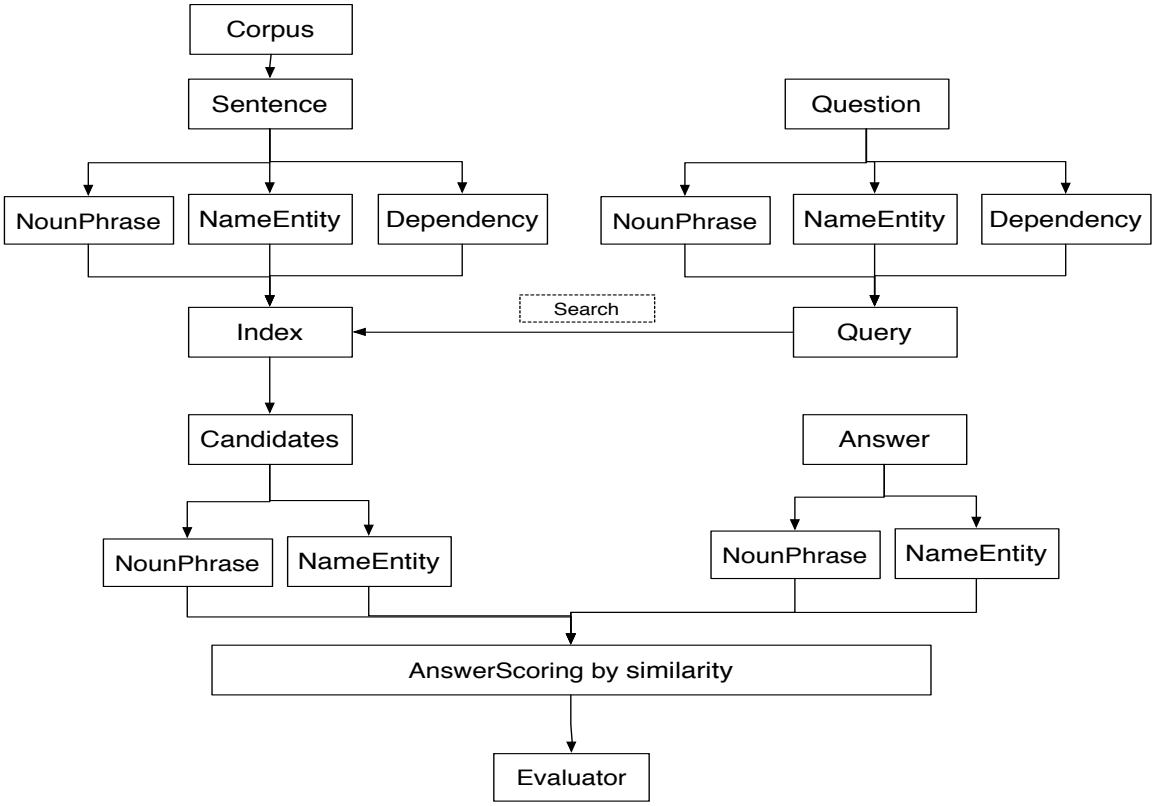
1) Initial pipeline/workflow design

1. The pipeline first process the articles and parse into sentences and then tokens. In parallel, the question and answer are annotated and parse into tokens.

2. NameEntity annotation: this annotation will be applied to both QAsentences and article sentences. Multiple technics can be added here in the future to help the following match process.

3. Match the question in the article to find multiple candidate sentences. Score the answers corresponding to the question by the matchness of the answer and the candidate sentences in the article.

4. Using AnswerScore for ranking and compare to goldenAnswer to get the rank and evaluation.



2) Initial type-system design

- SourceDocument: sentenceList(Sentence)
- TestDocument: qaList(QuestionAnswerList)
- Sentence: dependencyList(Dependency)
- Question: tokenlist(Token) nounList(NounPhrase) nerList(NER) dependencies(Dependency)
- Answer: synonyms(Synonym) nounPhraseList(NounPhrase)
- nerlist (NER) tokenList(Token) dependencyList(Dependency)
- QuestionAnswerSet: answerList(Answer) candidateSentenceList(CandidateSentence)
- CandidateSentence: candAnswerList(CandidateAnswer)
- CandidateAnswer
- Token
- Synonym:
- NounPhrase: synonyms(Synonym)
- NER: synonyms(Synonym)

- Dependency:

3) Ideas to improve the baseline system

1. Noise Filter:

Threshold was used in the given baseline system for noise filtering. We believe this task can be better conquered by applying machine learning algorithms to train a model for noise filtering. We can use current model and train these parameters, or we can add them with bag of word arrays as features. This may require manual labelling.

StanfordNLPAnnotator

Use StanfordNLPAnnotator to annotate tokens inside a sentence.
Annotates POS, NamedEntityTag for each token.
Annotates dependency graph of each sentence.

No improvements can be applied.

StanfordQuestionNLPAnnotator
Same as StanfordNLPAnnotator, but works on Question strings.

2. Candidate Sentence Searching:

In the given baseline system, question keywords are used to form search query. We believe adding relationship between keywords would improve the system performance, so we will rich the search query by adding **dependency information**

3. Answer Choice Scoring:

Point wise Mutual Information (PMI) between noun phrases/named entities in candidate sentence and answer is used in baseline. We believe more feature can be extracted and further used to match sentences. For example: Subject-Verb Comparison, Textual Entailment and Chunk Boundary. We will select models as many as we can, and then train the relative weights among those models to select the best combination of features.

4. Question preprocessing

We believe it's easy however important to classify questions before searching. Because various kinds of question can be linked to different types of answer. For instance, if it's a "What" question, more weight should be put on NE or N phrase, but when it's a "How" question, verb should be considered to be the most valuable element.

5) Initial division of work among team-members

Shen Wu: Question preprocessing

Mengda Yang: : Question preprocessing
Yi Song: Answer Choice Scoring improvement
Qiwen Zhu: Question preprocessing & wiki, issue, team meeting record
Ying Sheng: Answer Choice Scoring improvement