11-791 Initial Project Writeup

Team 3: Zhengzhong Liu, Da Teng, Guoqing Zheng, Xiawen Chu, Ryan Carlson

Github repository:

https://github.com/hunterhector/hw5-team03

Section 1: Initial Pipeline / Workflow Design

Our changes to the pipeline/workflow will have the following stages:

- **Pre-process training data**. This will consist of cleaning up the document. The input text are likely to comes from parsed PDF documents. Sections are not clear in between boundaries, for example, abstract is closely connected to introduction. Some segmentation should be done to extract useful contents, for example, a language model could be build to detect core sentences. It's kind of a mess of contentful information, citations, authors, and a whole bunch of other stuff. We need to write some regular expressions and probably do some POS tagging to identify the relevant sections that we want to feed into our algorithms. The output of this will be a set of tags similar to named entities, where the tags will be something like "relevant info", "author", "citation", etc.. We would like to propose a less "ad-hoc" way like the given code is currently doing.
- **Linking to external resources:** Understanding Bio-domain information will be difficult even for human, where we lack understanding to the knowledge involved. This should be the same case for the machine. We propose to annotate terms with external resources. Currently we propose to use DBPedia resources, while progressing we are likely to introduce other resources that are more closely related to the Bio-domain knowledge.
- **Indexing and searching**: We use information retrieval techniques to store the document corpora and to search with queries constructed from question and answer candidates. Specifically, each sentence in the document corpora will be treated as a single document and thus indexed. Annotations from previous

phases can serve to construct better queries. Beyond the current implementation, we may also try to use higher order interactions from terms and alternative retrieval models to help retrieve better results from the index.

- **Semantic and dependency parsing**: In the searching approach we tackle question answering problem using the bag-of-word assumption. We are also interested in investigating using dependency or event semantic parsing. If we can identify the dependency or semantic role of the query term, we have more evidence in selecting an answer. The use of semantic labelling can possibly reduce the problem caused by variation in dependency links. Specifically, we propose to use the following tools for this problem:
    - **Stanford Corenlp:** Stanford CoreNLP provides a set of natural language analysis tools which can take raw English language text input and give the base forms of words, their parts of speech, normalize dates, times, and numeric quantities, and mark up the structure of sentences in terms of phrases and word dependencies, and indicate which noun phrases refer to the same entities. In our project, we are using Stanford CoreNLP to annotate tokens, sentences, time, name entities, part of speech and entity coreference in the corpus.
    - **Fanse Parser**: fanse parser is based on the tree conversion of the Penn Treebank(Marcus, et al., 1993) which could support non-projective trees and can provide informative dependency labels and shallow semantic annotation for prepositions sense disambiguation, possessives interpretation, propbank SRL, and noun compound relations. In the initial pipeline, we will use the fanse parser to identify the semantic roles for the query term.
- **Identify Question Type**. We know there are 5 different types of questions. It seems reasonable that we could create specialized classifiers to solve each type of question.
- **Develop Specialized Answer Classifiers**. For each question type, develop a specialized classifier to answer questions of that type. These might all use the

same basic algorithm, or might be fairly different from each other. This is a question for future exploration.

    - These should involve good metrics of confidence so we can choose whether or not to actually select an answer.
    - There will also probably be a generic machine to answer the questions for which we have very little confidence that we can correctly identify the type.

- **Combining** There will be different methods for answering a question. We need to merge the results in certain way to provide a robust prediction.

Section 2: Initial Type-System Design

- We first retain some basic types given by the baseline system, such as question, answer and sentences.
- In the pre-processing step described above, we need a new set of annotations (TextGenre) describing all of the spans in that document as “relevant info”, “author”, “citation”, etc.
- Each Question will now have a QuestionType
- FanseDependencyRelation: In this annotation, we will try to annotate the token relationship. The annotation will include three elements.First is head which annotate the head node of sentence and second is child node. We will also include the dependency for the sentence.
- FanseSemanticRelation: This annotation will include the basic semantic annotation. The first part will record the semantic relationship. We also need to include the head node and the child node.
- FanseTokenAnnotation: This annotation will annotate each token in the original document by using fanse resources and models. Currently the annotation will include these elements: tokenid, part-of-speech tagging, coarse Pos, lexical sense, head dependency relation, child dependency relation, head semantic relations and child semantic relations.
- DbpediaAnnotation: This annotation annotate text span in the document that can be identified as DBpedia resources. For example, the term Alzheimer will

be link to “Alzheimer's Disease”. This annotation will provide additional knowledge about the terms, thus bringing in new knowledge

    - Uri: The DBpedia URI
    - similarityScore: The confidence that this text is really linked to the resource
    - resourceType: The type of the resource, for example “Person”.
    - abstract: A summary string for this resource, it is actually the first paragraph from Wikipedia
- StanfordCorenlpSentence: This annotation annotates sentences in the documents, questions and answers based on Stanford CoreNLP. It uses one attribute:
    - sentenceId: The Id to distinguish different sentences.
- StanfordCorenlpToken: This annotation annotates tokens in the documents, questions and answers based on Stanford CoreNLP. It uses one attribute:
    - tokenId: The Id to distinguish different sentences.
- StanfordDependencyNode: This annotation is the component of the tree-like type StanfordDepenRelation. It is annotated using Stanford CoreNLP. As a node of a dependency tree, this type has three attributes:
    - headRelations: The head relations of current node
    - childRelations: The child relations of current node
    - token: The token of current node
- StanfordDependencyRelation: This annotation stores the grammatical relation between words in a sentence. It is annotated using Stanford CoreNLP. It has 4 attributes:
    - head: Head node of the sentence
    - child: Child node
    - relationType: The type of dependency relation
    - rootNode: The root node of the dependency relation tree
- StanfordEntityMention:This annotation stores a collection of words in that are the names of things, such as person name and company names. It is

annotated using Stanford CoreNLP. As a collection of name entities, it has 3 attributes:

- entityMentionId: An Id to distinguish different types of name entity collections
- entityMentions: A collection of name entities with the same type
- entiryType: The type of name entities in the collection

Section 3: Baseline Methods to Implement

We propose to conduct experiments from two lines:

1. The “Information Retrieval” way

   We will first adopt a bag of word assumption, and use a similar method like that TA is currently using, which calculates PMI for occurrences. We propose to improve this method by incorporating a larger background corpus and use additional fields in the index and query expansion with external knowledge (such as descriptions from DBpedia). The basic PMI method will be our baseline.

2. The semantic way

   We are interested in whether we can parse the questions and figure out what is the question exactly, and then searching for the answer with a more specified target. If the given data is not enough, we propose to use an unsupervised method on a larger dataset.

We will divide our work as follows:

- **Zhengzhong Liu**: connects the pipeline and merge results from different methods; Explore and annotate external knowledge; incorporate dependency and semantic parsing to answer questions.

- **Da Teng**: add additional annotations from Stanford CoreNLP and OpenNLP to help provide more information about the document, answers and questions; implement annotators for these additional annotator.
- **Guoqing Zheng**: build local Solr index to search on the corpus; construct better queries from annotations on the question and answer candidates to extract information from the index; explore alternative ways of retrieval process from the index if time permits.
- **Xiawen Chu**: figure out the type system for fanse annotation and implement its annotator and find userful semantic information.
- **Ryan Carlson**: pre-process the data, identify question types