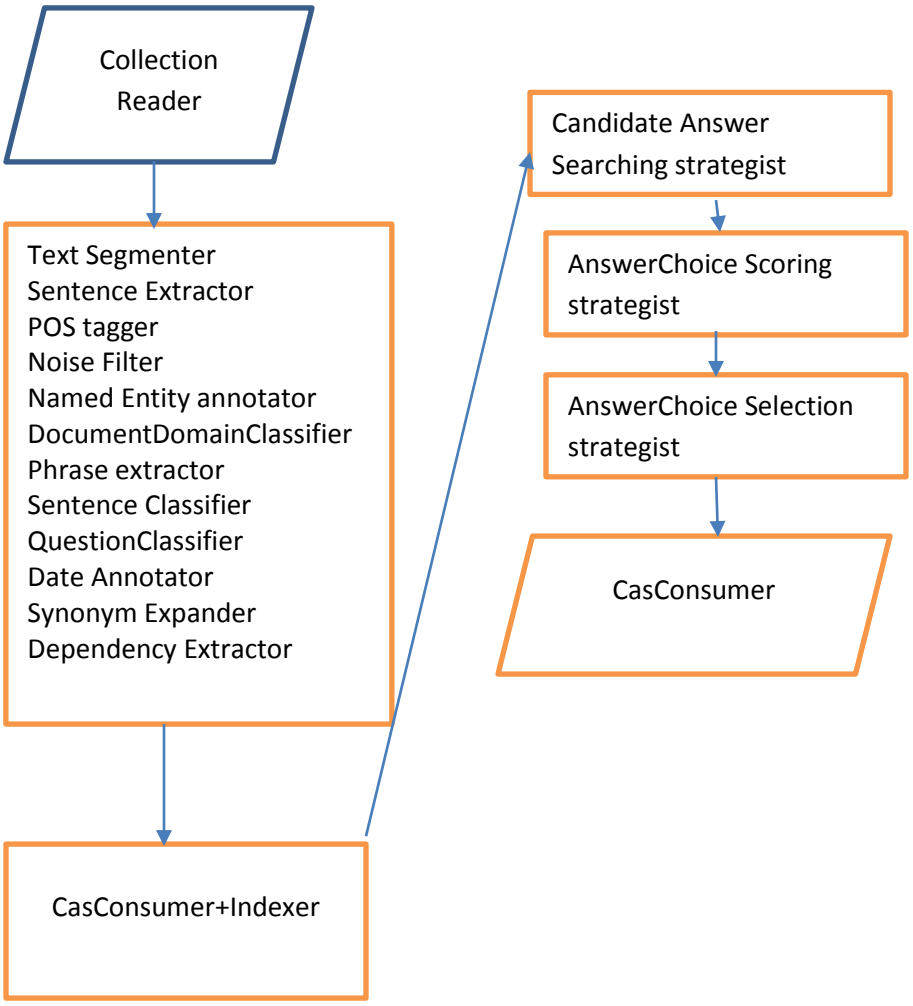
SE Project: Team 6

Ankur Gandhe, Simranjit Singh Kohli, Xiang Li, Mario Piergallini, and Wenqing Yuan

November 6, 2013

We list below our work flow design (initial design), followed by the baseline methods we plan to implement and the type system.

1. **Work Flow Design:**



2. **Baseline Methods:**

2.1 Background Corpus:

The background corpus can be used for many useful insights/information about the data, In addition to the baseline system; we plan to add the following:

a) Co-occurrence statistics: Get Word co-occurrence matrix for words in the background corpus. Using this matrix, we can identity **co-references, abbreviations, synonyms, antonyms** each with a probability score.

b) Background statistics according to domain: By making domain (medical, music, blog) statistics, we can get better estimates.

2.2 Annotations:

Most of the annotations can be used from the baseline system provided to us and using the stanford NLP toolkit. However, some annotations require building classifiers:

a) QuestionClassifier: Classifies a question into the five W's and one H . Also classifies the question as Easy, Moderate , Difficult

b) SentenceClassifier: Marks the sentences with a document segment (Title, Abstarct, related work, body , references)

c) DocumentDomainClassifier: The document domain (Medical, music, blog). This can be used to fetch the correct background statistics of the background corpus

d) Date Annotations: Since many question ask about "when", it is good to have dates annotated in the text

2.3 Scoring Algorithms:

The baseline system given to us already does a sentence matching. In addition to already existing scores, we will use the statistics from the background corpus to enhance the scoring functions. The additional scoring algorithms will use synonym information, antonym information (penalty) and abbreviation information. We plan to use the baseline system's answer choice selection algorithm since it has both the voting and aggregate functionality. For answering 'None of the above', we have 2 strategies: a) A question specific score 'threshold' (Easy, Moderate, Difficult); if none of the answers are above the threshold. Threshold determined by doing experiments over development set b) If the retrieved ranked list of sentences and the answer choices have a high distance (using the co-occurrence matrix values as word vectors), then probably the answer is not present

3. **Initial Type System**

We Propose a typeSystem for initial version. All types inherit features from UIMA type Annotation, such as start, end.

Token

- text String
- pos String part of speech tag
- ner String BIO NER tag

Answer

- text String
- id String
- questionId String
- docId String
- synonyms FSList<Synonym>
- isCorrect Boolean
- isSelected Boolean
- nounPhraseList FSList<NounPhrase>
- nerList FSList<NER>
- tokenList FSList<Token>
- dependencies FSList<Dependency>

CandidateAnswer

- qId String
- text String
- choiceIndex Integer
- PMIScore Double pointwise mutual information
- similarityScore Double cosine similarity?
- synonymScore Double

CandidateSentence

- relevanceScore Double
- sentence Sentence
- depMatchScore Double
- synonymMatchScore Double
- candAnswerList FSList<CandidateAnswer>

Dependency

- governor Token
- dependent Token
- relation String

NER

- text
- tag String BIO tag
- weight Double
- source String
- synonyms FSList<Synonym>

NounPhrase

- text
- weight Double
- synonyms FSList<Synonym>

Question

- id
- text
- dependencies
- nerList
- nounList

- tokenList
- category String factoid,causal,method,purpose,t/f

QuestionAnswerSet
- question Question
- answerList
- candidateSentenceList

Sentence
- id String
- text String
- qualityScore Double
- dependencies FSList<Dependency>
- tokenList FSList<Token>
- bFilter Boolean
- phraseList FSList<NounPhrase>
- nerList FSList<NER>
- interrogative Boolean
- section String title,abstract,intro,references,etc.

SourceDocument
- text
- id
- filteredText
- sentenceList
- authors FSList<Author>
- pubDate Date
- genre String type of document, i.e. journal blog post, etc.
- docLists FSList<DocList>

Author
- text String
- firstName String
- lastName String
- initials String
- institution String

Date
- text String
- day Integer
- month Integer
- year Integer

DocList list items can be evaluated in relation to the topic
- sentences FSList<Sentence>
- listTopic
- listItems

Synonym
- text
- source
- weight

TestDocument
- qaList FSList<QuestionAnswerSet>
- readingTestId String
- topicId String

4. **Division Of Labor:**

Ankur Gandhe: Question and Document Classifiers
Simranjit Singh Kohli: UIMA Annotations
Xiang Li: Error Analysis of baseline/subsequent systems
Mario Piergallini: Prepare background Corpus statistics
Wenqing Yuan: Scoring function Implementation