UIMA Components

Input

- OpenTrivaQA parser
  - Parses files in OpenTriviaQA format
  - Each question is a document
  - For each question the following annotations are added
    - Question → the question text
    - Answer → the correct answer

Question Processing

- Keyword extraction
  - The question should be analyzed by other components (e.g. POS tagger) before
  - Extract all key words from question
  - Beyond the baseline: the component may add an importance score to each keyword

Tag Cloud Enrichment

- Category and hypernym detection
  - This is just a single component which is internally multi-threaded
  - For each resource (e.g. Wikipedia, WordNet) a separate thread is started
  - Each resource creates a list of categories/hypernyms
  - After the research for each resource is finished, the list is set as the corresponding feature in the answer annotation (thereby building the "large tag cloud")
  - TODO: decide how parallelization will be implemented
  - **Parameters**
    - Annotation type: over which annotation to iterate; this will be used to run the analysis engine on the correct answer first and later on the answer candidates

Candidate Extraction

- Category ranking
  - Iterates over all categories and hypernyms (possibly in parallel) of the correct answer
  - Uses a measurement to derive a score for each category which reflect its relevance
    - For instance, page rank could be used for Wikipedia categories
    - The measurement may change to allow different system configurations
  - The scores are assigned to the categories
  - **Parameters**
    - Relevance measure
- Candidate extraction
  - Extract categories/hypernyms from the correct answer with the highest relevance score

    - For each of the extracted categories, retrieve candidates
    - Create an “answer candidate” annotation for each retrieved candidate
    - **Parameters**
        - How many categories/hypernyms to use
- Synonym resolution
    - Iterate over all “answer candidate” annotations
    - Perform synonym resolution with the correct answer
        - E.g. by using WordNet
        - Baseline: perform a simple string comparison between the correct answer and the candidate answer’s title
    - Remove all synonymous candidate answers by removing the corresponding “answer candidate” from the index

Similarity Detection

- Similarity detection
    - Iterate over all “answer candidate” annotations
    - Perform the similarity measure between the candidate’s tag cloud and the correct answer’s tag cloud
    - Assign the resulting scores to the answer candidates
    - **Parameters**
        - The similarity measure

CAS Consumer

- Candidate selection
    - Iterate over all “answer candidate” annotations
    - Select the candidates with the highest scores and output them
    - **Parameters**
        - Number of candidates to select

UIMA Types

Question

- Standard UIMA annotation
- No special features

Answer

- Annotation for an answer (correct or candidate)

- Features
    - Tag cloud
        - Contains keywords, categories, and hypernyms
        - Elements in tag cloud are linked to each other
    - Answer type (optional since it requires an additional analysis engine)

Correct Answer

- Inherits everything from the answer type
- Is used to distinguish the correct answer annotation from the candidate annotation

Candidate Answer

- Inherits everything from the answer type
- Features
    - Wikipedia page (of the candidate entity found by the candidate extraction analysis engine)
    - Similarity score
- Is used for iterating over answer candidates

Final Pipeline