

# Baseline QA system

## Deliverable 2

C.J Hsu

Ryan Bielby

# Outline

- System Architecture
  - Frontend System (Answer generator)
  - Backend System (IR Engine)
- Evaluations
- Discussions
- References

# System Architecture

- The frontend module (n-gram answer generator) tries to mine the correct answers from the web.
- The backend module (IR Engine) tries to find the supporting documents for answers mining from frontend.

# The Architecture of Frontend System

- Generate two types of query “**baseline**” and “**inexact**” queries for each question.
- Send these queries to the Teoma (Ask.com) and fetch 100 snippets for each query.
- Generate the n-grams (unigram to tetragram) from snippets by “Voting”, “**Filtering**”, “Combining”, “Scoring” and “**Reranking**” (Lin 2007).

# The Architecture of Frontend System Filtering

- Three kinds of filters mentioned in Lin 2007, but we only implement “type-neutral” and **incomplete** “type-specific” filters.
- “type-neutral” filter removes n-grams starting and ending in stop words.
- “type-specific” filter removes the n-grams which doesn’t meet the question type. We deal with **“Person”**, **“Time/Date”**, **“Number”** and **“Location”**

# The Architecture of Frontend System

## Reranking

- According to (Lin 2007), every n-gram must be contained in at least **2** snippets, What does “contain” mean here ? A case study will be proposed in Discussion section.
- Since our filtering mechanism is not strong enough, we set this threshold to **5** to avoid including some ridiculous n-grams.

# The Architecture of Backend System

## Indexing

- We implement a plug-in to help Lucene index TREC document format.
- The AQUAINT corpus is indexed by Snowball Analyzer (stop words removal, stemming...).
- The indexing process is an independent process from our main pipeline.

# The Architecture of Backend System

## Document Retrieval/Answer Projection

- Formulate the boolean query by the stemmed terms from original question and n-gram answer candidate.
- The retrieved document **must** contains the terms from n-gram but **may not** contain the terms from original question.
- The retrieved documents are discarded if their scores are lower than a threshold.
- Eg: Q: Where is Mozart born ? Ans: Salzburg.  
The Lucene Query is  
“mozart” OR “born” AND “salzburg”

# The Architecture of Backend System

## Passage Retrieval/Answer Projection

- We Implement three different passage retrieval algorithms. MITRE V1, MITRE V2, IBM. Please refer to (Tellex et al. 2003).
- Intuitively, the term from n-gram should be more important than the term from original question, how do we integrate this concept into the scoring function ?
- The document containing the highest score passage is the supporting document.

# Evaluations

- If the quality of the n-gram generation is terrible, The whole system is doomed to fail even if you develop a brilliant document/passage retrieval algorithm. Vice versa.

# Evaluations

## Frontend System

|         | Frontend System<br>(type neutral and specific filters) | Frontend System<br>(type neutral filter) |
|---------|--------------------------------------------------------|------------------------------------------|
| strict  | 0.2361                                                 | 0.1761                                   |
| lenient | 0.2368                                                 | 0.1770                                   |

# Evaluations

## Whole System with Different Answer Projection Algorithms

|         | frontend only | whole system<br>MITRE V.1 | Whole system<br>MITRE V.2 | Whole system<br>IBM |
|---------|---------------|---------------------------|---------------------------|---------------------|
| strict  | 0.2361        | 0.1031                    | 0.0989                    | 0.0981              |
| lenient | 0.2368        | 0.2383                    | 0.2389                    | 0.2383              |

# Discussion

## Reranking Mechanism of Frontend

- Our system fail on a easy question.  
“152.1: Where is Mozart born?”.  
The correct answer is “Salzburg.\*Austria”.  
Our top one answer is “Salzburg”.
- Since only one snippet contain the bigram  
“Salzburg Austria”. So it got filtered out.  
However, a lot of snippets contain  
“Salzburg , Austria”.  
Exact match is the best solution ?

# Discussion

## Select the best Parameters

- There are a lot of parameters in our system, like the threshold to filter out documents, the weight for term when doing passage retrieval and the reranking threshold.
- For now, we decide to give Condor a break. We will not perform parameter selection until the we complete all the remaining functions.

# References

- An Exploration of the Principles Underlying Redundancy-Based Factoid Question. Lin 2007.
- Query Formulation for Answer Projection. Gilad Mishne and Marriten de Rijke 2005.
- Quantitative Evaluation of Passage Retrieval Algorithms for Question Answering. Stefaine Tellex et al. 2003.