Answer Extraction

Ling573
NLP Systems and Applications
May 16, 2013

Roadmap

- Deliverable 3 Discussion
  - What worked
- Deliverable 4
- Answer extraction:
  - Learning answer patterns
  - Answer extraction: classification and ranking
  - Noisy channel approaches

Reminder

- Rob Chambers
  - Speech Tech talk & networking event
  - This evening: 6:00pm
  - Johnson 203

  - Speech Technology and Mobile Applications:
    - Speech in Windows Phone

Deliverable #3

- Document & Passage Retrieval
- What was tried:
  - Query processing:

Deliverable #3

- Question Answering:
  - Focus on question processing
- What was tried:
    - Question classification

Deliverable #3

- Question Answering:
  - Focus on question processing
- What was tried:
  - Question classification
    - Data: Li & Roth, TREC – given or hand-tagged
    - Features: unigrams, POS, NER, head chunks, semantic info
    - Classifiers: MaxEnt, SVM {+ confidence}
      - Accuracies: mid-80%s

Deliverable #3

- Question Answering:
  - Focus on question processing
- What was tried:
  - Question classification
    - Data: Li & Roth, TREC – given or hand-tagged
    - Features: unigrams, POS, NER, head chunks, semantic info
    - Classifiers: MaxEnt, SVM {+ confidence}
      - Accuracies: mid-80%s
  - Application:
    - Filtering: Restrict results to have compatible class
    - Boosting: Upweight compatible answers
      - Gazetteers, heuristics, NER

Question Processing

- What was tried:
- Question Reformulation:
  - Target handling:
    - Replacement of pronouns, overlapping NPs, etc
  - Per-qtype reformulations:
    - With backoff to bag-of-words
  - Inflection generation + irregular verb handling
  - Variations of exact phrases

What was tried

- Assorted clean-ups and speedups
  - Search result caching
  - Search result cleanup, dedup-ing
  - Google vs Bing
  - Code refactoring

What worked

- Target integration: most variants helped
- Query reformulation: type specific
- Qtype boosting, in some cases
- Caching for speed/analysis

Results

- Major improvements over D2 baseline
  - Most lenient results approach or exceed 0.1 MRR
    - Current best: ~0.34
  - Strict results improve, but less than lenient

Deliverable #4

- Answer extraction/refinement
  - Fine-grained passages

Deliverable #4

- Answer extraction/refinement
  - Fine-grained passages
  - Lengths not to exceed
    - 100-char,
    - 250-char

Deliverable #4

- Answer extraction/refinement
  - Fine-grained passages
  - Lengths not to exceed
    - 100-char,
    - 250-char
  - Evaluate on 2006 Devtest
    - Final held-out evaltest from 2007
      - Released later, no tuning allowed

Deliverable #4

- Any other refinements across system
  - Question processing
  - Retrieval – Web or AQUAINT
  - Answer processing
- Whatever you like to improve final scores

Plug

- Error analysis
  - Look at training and devtest data
  - What causes failures?
    - Are the answers in any of the retrieval docs? Web/TREC
      - If not, why?
    - Are answers retrieved by not highly ranked?

Last Plugs

- Tonight: 6pm: JHN 102
  - Jay Waltmunson: Speech Tech and Mobile
    - UW Ling Ph.D.
  - Presentation and Networking

- Tomorrow: 3:30 PCAR 291
  - UW/MS Symposium
  - Hoifung Poon (MSR): Semantic Parsing
  - Chloe Kiddon (UW): Knowledge Extraction w/TML

Answer Extraction

- Pattern-based Extraction review
- Learning Answer Reranking I
- Noisy Channel Answer Extraction
- Learning Answer Reranking II

Answer Selection by Pattern

- Identify question types and terms
- Filter retrieved passages, replace qterm by tag
- Try to match patterns and answer spans
- Discard duplicates and sort by pattern precision

Pattern Sets

- WHY-FAMOUS

  1.0 <ANSWER> <NAME> called

  1.0 laureate <ANSWER> <NAME>

  1.0 by the <ANSWER> , <NAME> ,
  1.0 <NAME> - the <ANSWER> of

  1.0 <NAME> was the <ANSWER> of

- BIRTHYEAR

  1.0 <NAME> (<ANSWER> -)

  0.85 <NAME> was born on <ANSWER> ,

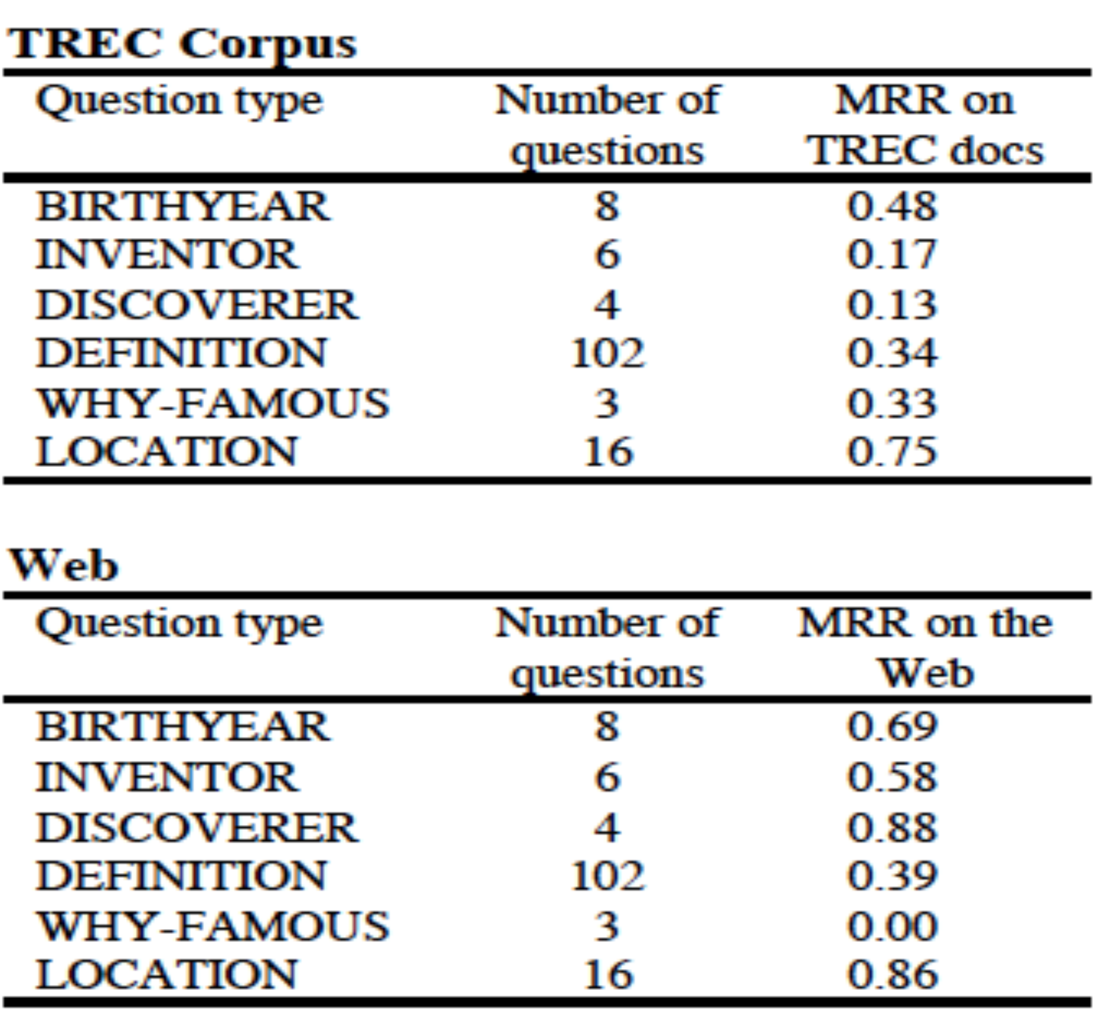
  0.6 <NAME> was born in <ANSWER>

  0.59 <NAME> was born <ANSWER>

  0.53 <ANSWER> <NAME> was born

Results

- Improves, though better with web data



TREC Corpus

Question type	Number of questions	MRR on TREC docs
BIRTHYEAR	8	0.48
INVENTOR	6	0.17
DISCOVERER	4	0.13
DEFINITION	102	0.34
WHY-FAMOUS	3	0.33
LOCATION	16	0.75

Web

Question type	Number of questions	MRR on the Web
BIRTHYEAR	8	0.69
INVENTOR	6	0.58
DISCOVERER	4	0.88
DEFINITION	102	0.39
WHY-FAMOUS	3	0.00
LOCATION	16	0.86

Limitations & Extensions

- Where are the Rockies?
- ..with the Rockies in **the background**

Limitations & Extensions

  - Where are the Rockies?
  - ..with the Rockies in **the background**

- Should restrict to semantic / NE type

Limitations & Extensions

  - Where are the Rockies?
  - ..with the Rockies in **the background**
- Should restrict to semantic / NE type
  - London, which...., lies on the River Thames
  - <QTERM> word* lies on <ANSWER>
    - Wildcards impractical

Limitations & Extensions

- Where are the Rockies?
- ..with the Rockies in **the background**

- Should restrict to semantic / NE type
  - London, which...., lies on the River Thames
  - <QTERM> word* lies on <ANSWER>
    - Wildcards impractical
- Long-distance dependencies not practical

Limitations & Extensions

  - Where are the Rockies?
  - ..with the Rockies in **the background**
- Should restrict to semantic / NE type
  - London, which...., lies on the River Thames
  - <QTERM> word* lies on <ANSWER>
    - Wildcards impractical
- Long-distance dependencies not practical
  - Less of an issue in Web search
    - Web highly redundant, many local dependencies
    - Many systems (LCC) use web to **validate** answers

Limitations & Extensions

- When was LBJ born?
- Tower lost to Sen. LBJ, *who ran for both the*…

Limitations & Extensions

  - When was LBJ born?
  - Tower lost to Sen. LBJ, *who ran for both the*...

- Requires information about:
  - Answer length, type; logical distance (1-2 chunks)

Limitations & Extensions

  - When was LBJ born?
  - Tower lost to Sen. LBJ, *who ran for both the…*
- Requires information about:
  - Answer length, type; logical distance (1-2 chunks)
- Also,
  - Can only handle single continuous qterms
  - Ignores case
  - Needs handle canonicalization, e.g of names/dates

Integrating Patterns II

- Fundamental problem:

Integrating Patterns II

- Fundamental problem:
  - What if there's no pattern??

Integrating Patterns II

- Fundamental problem:
  - What if there's no pattern??
    - No pattern -> No answer!!!
- More robust solution:
  - Not JUST patterns

Integrating Patterns II

- Fundamental problem:
  - What if there's no pattern??
    - No pattern -> No answer!!!
- More robust solution:
  - Not JUST patterns
  - Integrate with machine learning
    - MAXENT!!!
    - Re-ranking approach

Answering w/Maxent

$$P(a \mid \{a_1, a_2, \ldots a_A\}, q) = \frac{\exp[\sum_{m=1}^{M} \lambda_m f_m(a, \{a_1, a_2, \ldots a_A\}, q)]}{\sum_{a'} \exp[\sum_{m=1}^{M} \lambda_m f_m(a', \{a_1, a_2, \ldots a_A\}, q)]}$$

$$\widehat{a} = \underset{a}{\operatorname{argmax}}[\sum_{m=1}^{M} \lambda_m f_m(a, \{a_1, a_2, \ldots a_A\}, q)]$$

Feature Functions

- Pattern fired:
  - Binary feature

Feature Functions

- Pattern fired:
  - Binary feature
- Answer frequency/Redundancy factor:
  - # times answer appears in retrieval results

Feature Functions

- Pattern fired:
  - Binary feature
- Answer frequency/Redundancy factor:
  - # times answer appears in retrieval results
- Answer type match (binary)

Feature Functions

- Pattern fired:
  - Binary feature
- Answer frequency/Redundancy factor:
  - # times answer appears in retrieval results
- Answer type match (binary)
- Question word absent (binary):
  - No question words in answer span

Feature Functions

- Pattern fired:
  - Binary feature
- Answer frequency/Redundancy factor:
  - # times answer appears in retrieval results
- Answer type match (binary)
- Question word absent (binary):
  - No question words in answer span
- Word match:
  - Sum of ITF of words matching b/t questions & sent

Training & Testing

- Trained on NIST QA questions
  - Train: TREC 8,9;
  - Cross-validation: TREC-10
- 5000 candidate answers/question
- Positive examples:
  - NIST pattern matches
- Negative examples:
  - NIST pattern doesn't match
- Test: TREC-2003: MRR: 28.6%; 35.6% exact top 5

Noisy Channel QA

- Employed for speech, POS tagging, MT, summ, etc
- Intuition:
  - Question is a noisy representation of the answer

Noisy Channel QA

- Employed for speech, POS tagging, MT, summ, etc
- Intuition:
  - Question is a noisy representation of the answer
- Basic approach:
  - Given a corpus of (Q,S_A) pairs
  - Train $P(Q|S_A)$
  - Find sentence with answer as
    - $S_{i,Aij}$ that maximize $P(Q|S_{i,Aij})$

QA Noisy Channel

- A: Presley died of heart disease at Graceland in 1977, and..
- Q: When did Elvis Presley die?

QA Noisy Channel

  - A: Presley died of heart disease at Graceland in 1977, and..
  - Q: When did Elvis Presley die?
- Goal:
  - Align parts of Ans parse tree to question
    - Mark candidate answers
    - Find highest probability answer

Approach

- Alignment issue:

Approach

- Alignment issue:
  - Answer sentences longer than questions
  - Minimize length gap
    - Represent answer as mix of words/syn/sem/NE units

Approach

- Alignment issue:
  - Answer sentences longer than questions
  - Minimize length gap
    - Represent answer as mix of words/syn/sem/NE units
  - Create ‘cut’ through parse tree
    - Every word –or an ancestor – in cut
    - Only one element on path from root to word

Approach

- Alignment issue:
  - Answer sentences longer than questions
  - Minimize length gap
    - Represent answer as mix of words/syn/sem/NE units
  - Create ‘cut’ through parse tree
    - Every word –or an ancestor – in cut
    - Only one element on path from root to word

Presley died of heart disease at Graceland in 1977, and..
Presley died PP PP in DATE, and..
When did Elvis Presley die?

Approach (Cont'd)

- Assign one element in cut to be 'Answer'
- Issue: Cut STILL may not be same length as Q

Approach (Cont’d)

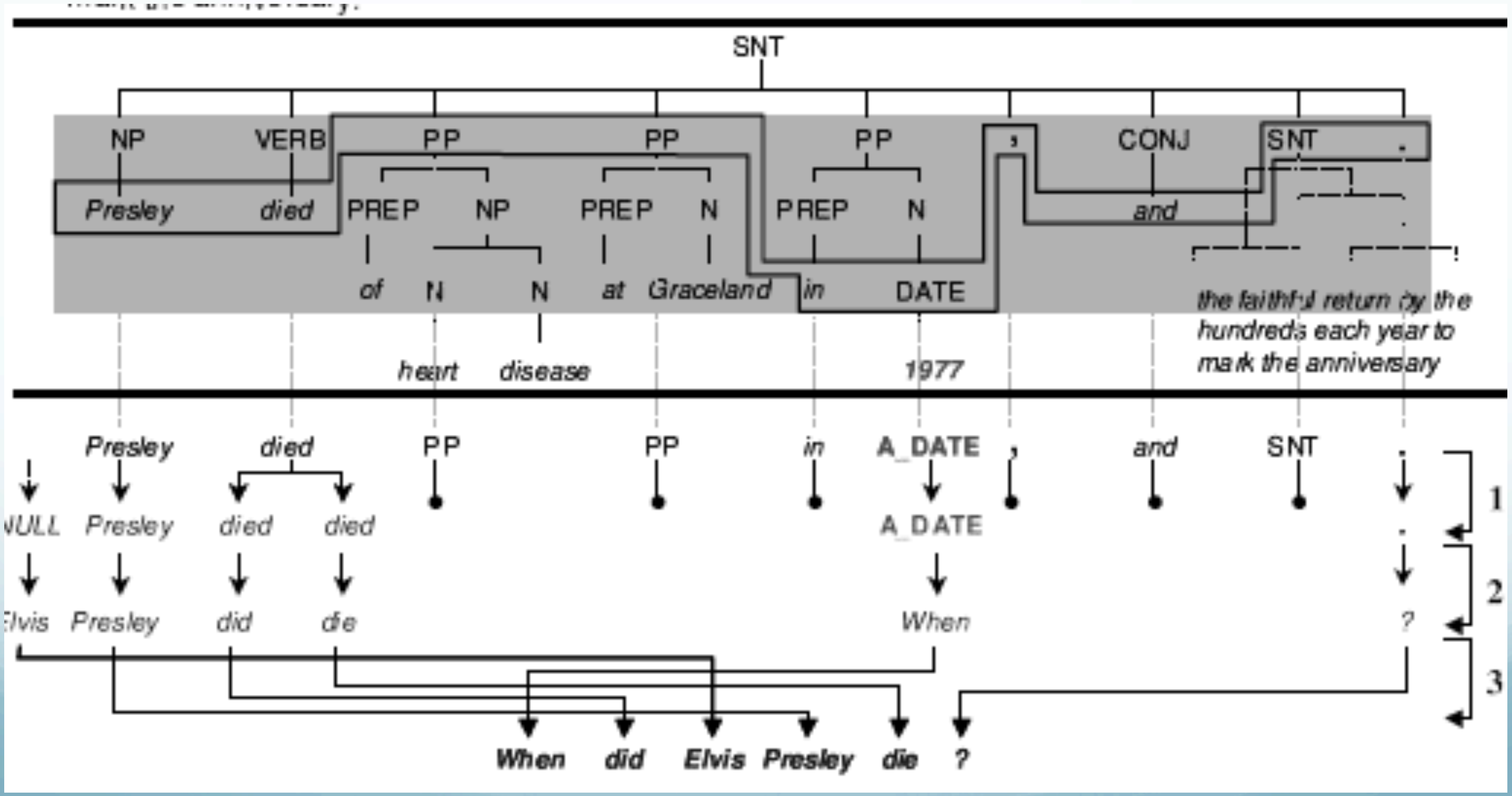
- Assign one element in cut to be ‘Answer’
- Issue: Cut STILL may not be same length as Q
- Solution: (typical MT)
  - Assign each element a fertility
    - 0 – delete the word; > 1: repeat word that many times

Approach (Cont’d)

- Assign one element in cut to be ‘Answer’
- Issue: Cut STILL may not be same length as Q
- Solution: (typical MT)
  - Assign each element a fertility
    - 0 – delete the word; > 1: repeat word that many times
- Replace A words with Q words based on alignment
- Permute result to match original Question
- Everything except cut computed with OTS MT code

Schematic

- Assume cut, answer guess all equally likely



Training Sample Generation

- Given question and answer sentences
- Parse answer sentence
- Create cut s.t.:
  - Words in both Q & A are preserved
  - Answer reduced to 'A_' syn/sem class label
  - Nodes with no surface children reduced to syn class
  - Keep surface form of all other nodes
- 20K TREC QA pairs; 6.5K web question pairs

Selecting Answers

- For any candidate answer sentence:
  - Do same cut process

Selecting Answers

- For any candidate answer sentence:
  - Do same cut process
  - Generate all candidate answer nodes:
    - Syntactic/Semantic nodes in tree

Selecting Answers

- For any candidate answer sentence:
  - Do same cut process
  - Generate all candidate answer nodes:
    - Syntactic/Semantic nodes in tree
  - What's a bad candidate answer?

Selecting Answers

- For any candidate answer sentence:
  - Do same cut process
  - Generate all candidate answer nodes:
    - Syntactic/Semantic nodes in tree
  - What’s a bad candidate answer?
    - Stopwords
    - Question words!
  - Create cuts with each answer candidate annotated
  - Select one with highest probability by model

Example Answer Cuts

- Q: When did Elvis Presley die?
- S_{A1}: Presley died A_PP PP PP, and …
- S_{A2}: Presley died PP A_PP PP, and ….
- S_{A3}: Presley died PP PP in A_DATE, and …
- Results: MRR: 24.8%; 31.2% in top 5

Error Analysis

- Component specific errors:
  - Patterns:
    - Some question types work better with patterns
    - Typically specific NE categories (NAM, LOC, ORG..)
    - Bad if ‘vague’

Error Analysis

- Component specific errors:
  - Patterns:
    - Some question types work better with patterns
    - Typically specific NE categories (NAM, LOC, ORG..)
    - Bad if ‘vague’
  - Stats based:
    - No restrictions on answer type – frequently ‘it’

Error Analysis

- Component specific errors:
  - Patterns:
    - Some question types work better with patterns
    - Typically specific NE categories (NAM, LOC, ORG..)
    - Bad if 'vague'
  - Stats based:
    - No restrictions on answer type – frequently 'it'
  - Patterns and stats:
    - 'Blatant' errors:
      - Select 'bad' strings (esp. pronouns) if fit position/pattern

Combining Units

- Linear sum of weights?

Combining Units

- Linear sum of weights?
  - Problematic:
    - Misses different strengths/weaknesses

Combining Units

- Linear sum of weights?
  - Problematic:
    - Misses different strengths/weaknesses
- Learning! (of course)
  - Maxent re-ranking
    - Linear

Feature Functions

- 48 in total
- Component-specific:
  - Scores, ranks from different modules
    - Patterns. Stats, IR, even QA word overlap

Feature Functions

- 48 in total
- Component-specific:
  - Scores, ranks from different modules
    - Patterns. Stats, IR, even QA word overlap
- Redundancy-specific:
  - # times candidate answer appears (log, sqrt)

Feature Functions

- 48 in total
- Component-specific:
  - Scores, ranks from different modules
    - Patterns. Stats, IR, even QA word overlap
- Redundancy-specific:
  - # times candidate answer appears (log, sqrt)
- Qtype-specific:
  - Some components better for certain types: type+mod

Feature Functions

- 48 in total
- Component-specific:
  - Scores, ranks from different modules
    - Patterns. Stats, IR, even QA word overlap
- Redundancy-specific:
  - # times candidate answer appears (log, sqrt)
- Qtype-specific:
  - Some components better for certain types: type+mod
- Blatant ‘errors’: no pronouns, when NOT DoW

Experiments

- Per-module reranking:
  - Use redundancy, qtype, blatant, and feature from mod

Experiments

- Per-module reranking:
  - Use redundancy, qtype, blatant, and feature from mod
- Combined reranking:
  - All features (after feature selection to 31)

Experiments

- Per-module reranking:
  - Use redundancy, qtype, blatant, and feature from mod
- Combined reranking:
  - All features (after feature selection to 31)
- Patterns: Exact in top 5: 35.6% -> 43.1%
- Stats: Exact in top 5: 31.2% -> 41%
- Manual/knowledge based: 57%

Experiments

- Per-module reranking:
  - Use redundancy, qtype, blatant, and feature from mod

- Combined reranking:
  - All features (after feature selection to 31)

- Patterns: Exact in top 5: 35.6% -> 43.1%
- Stats: Exact in top 5: 31.2% -> 41%
- Manual/knowledge based: 57%
- Combined: 57%+