# Question Processing: Formulation & Expansion

Ling573
NLP Systems and Applications
May 2, 2013

# Deeper Processing for Query Formulation

- MULDER (Kwok, Etzioni, & Weld)
- Converts question to multiple search queries
  - Forms which match target
  - Vary specificity of query
    - Most general bag of keywords
    - Most specific partial/full phrases

# Deeper Processing for Query Formulation

- MULDER (Kwok, Etzioni, & Weld)
- Converts question to multiple search queries
  - Forms which match target
  - Vary specificity of query
    - Most general bag of keywords
    - Most specific partial/full phrases
  - Subsets 4 query forms on average
- Employs full parsing augmented with morphology

# Question Parsing

- Creates full syntactic analysis of question
  - Maximum Entropy Inspired (MEI) parser
    - Trained on WSJ

# Question Parsing

- Creates full syntactic analysis of question
  - Maximum Entropy Inspired (MEI) parser
    - Trained on WSJ
- Challenge: Unknown words
  - Parser has limited vocabulary
    - Uses guessing strategy
      - Bad: “tungsten”

# Question Parsing

- Creates full syntactic analysis of question
  - Maximum Entropy Inspired (MEI) parser
    - Trained on WSJ
- Challenge: Unknown words
  - Parser has limited vocabulary
    - Uses guessing strategy
      - Bad: “tungsten” → number
- Solution:

# Question Parsing

- Creates full syntactic analysis of question
  - Maximum Entropy Inspired (MEI) parser
    - Trained on WSJ
- Challenge: Unknown words
  - Parser has limited vocabulary
    - Uses guessing strategy
      - Bad: “tungsten” → number
- Solution:
  - Augment with morphological analysis: PC-Kimmo
  - If PC-KIMMO fails?

# Question Parsing

- Creates full syntactic analysis of question
  - Maximum Entropy Inspired (MEI) parser
    - Trained on WSJ
- Challenge: Unknown words
  - Parser has limited vocabulary
    - Uses guessing strategy
      - Bad: “tungsten” → number
- Solution:
  - Augment with morphological analysis: PC-Kimmo
  - If PC-KIMMO fails? Guess Noun

# Question Classification

- Simple categorization:
  - Nominal, numerical, temporal
  - Hypothesis: Simplicity → High accuracy
    - Also avoids complex training, ontology design

# Question Classification

- Simple categorization:
  - Nominal, numerical, temporal
  - Hypothesis: Simplicity → High accuracy
    - Also avoids complex training, ontology design
- Parsing used in two ways:
  - Constituent parser extracts wh-phrases:
    - e.g. wh-adj: how many

# Question Classification

- Simple categorization:
  - Nominal, numerical, temporal
  - Hypothesis: Simplicity → High accuracy
    - Also avoids complex training, ontology design
- Parsing used in two ways:
  - Constituent parser extracts wh-phrases:
    - e.g. wh-adj: how many → numerical; wh-adv: when, where

# Question Classification

- Simple categorization:
  - Nominal, numerical, temporal
  - Hypothesis: Simplicity → High accuracy
    - Also avoids complex training, ontology design
- Parsing used in two ways:
  - Constituent parser extracts wh-phrases:
    - e.g. wh-adj: how many → numerical; wh-adv: when, where
    - wh-noun: type?

# Question Classification

- Simple categorization:
  - Nominal, numerical, temporal
  - Hypothesis: Simplicity → High accuracy
    - Also avoids complex training, ontology design
- Parsing used in two ways:
  - Constituent parser extracts wh-phrases:
    - e.g. wh-adj: how many → numerical; wh-adv: when, where
    - wh-noun: type? → any
      - what height vs what time vs what actor

# Question Classification

- Simple categorization:
  - Nominal, numerical, temporal
  - Hypothesis: Simplicity → High accuracy
    - Also avoids complex training, ontology design
- Parsing used in two ways:
  - Constituent parser extracts wh-phrases:
    - e.g. wh-adj: how many → numerical; wh-adv: when, where
    - wh-noun: type? → any
      - what height vs what time vs what actor
  - Link parser identifies verb-object relation for wh-noun
    - Uses WordNet hypernyms to classify object, Q

# Syntax for Query Formulation

- Parse-based transformations:
  - Applies transformational grammar rules to questions

# Syntax for Query Formulation

- Parse-based transformations:
  - Applies transformational grammar rules to questions
  - Example rules:
    - Subject-auxiliary movement:
      - Q: Who was the first American in space?

# Syntax for Query Formulation

- Parse-based transformations:
  - Applies transformational grammar rules to questions
  - Example rules:
    - Subject-auxiliary movement:
      - Q: Who was the first American in space?
      - Alt: was the first American…; the first American in space was
    - Subject-verb movement:
      - Who shot JFK?

# Syntax for Query Formulation

- Parse-based transformations:
  - Applies transformational grammar rules to questions
  - Example rules:
    - Subject-auxiliary movement:
      - Q: Who was the first American in space?
      - Alt: was the first American…; the first American in space was
    - Subject-verb movement:
      - Who shot JFK? => shot JFK
    - Etc

# Syntax for Query Formulation

- Parse-based transformations:
  - Applies transformational grammar rules to questions
  - Example rules:
    - Subject-auxiliary movement:
      - Q: Who was the first American in space?
      - Alt: was the first American…; the first American in space was
    - Subject-verb movement:
      - Who shot JFK? => shot JFK
    - Etc

# More General Query Processing

- WordNet Query Expansion
  - Many lexical alternations: ‘How tall’ → ‘The height is’
  - Replace adjectives with corresponding ‘attribute noun’

# More General Query Processing

- WordNet Query Expansion
  - Many lexical alternations: ‘How tall’ → ‘The height is’
  - Replace adjectives with corresponding ‘attribute noun’

- Verb conversion:
  - Morphological processing
    - DO-AUX .... V-INF → V+inflection
    - Generation via PC-KIMMO

# More General Query Processing

- WordNet Query Expansion
  - Many lexical alternations: ‘How tall’ → ‘The height is’
  - Replace adjectives with corresponding ‘attribute noun’
- Verb conversion:
  - Morphological processing
    - DO-AUX .... V-INF → V+inflection
    - Generation via PC-KIMMO
- Query formulation contributes significantly to effectiveness

# Machine Learning Approaches

- Diverse approaches:
  - Assume annotated query logs, annotated question sets, matched query/snippet pairs

# Machine Learning Approaches

- Diverse approaches:
  - Assume annotated query logs, annotated question sets, matched query/snippet pairs
- Learn question paraphrases (MSRA)
  - Improve QA by setting question sites
  - Improve search by generating alternate question forms

# Machine Learning Approaches

- Diverse approaches:
  - Assume annotated query logs, annotated question sets, matched query/snippet pairs
- Learn question paraphrases (MSRA)
  - Improve QA by setting question sites
  - Improve search by generating alternate question forms
- Question reformulation as machine translation
  - Given question logs, click-through snippets
    - Train machine learning model to transform Q -> A

# Query Expansion

- Basic idea:
  - Improve matching by adding words with similar meaning/similar topic to query

# Query Expansion

- Basic idea:
  - Improve matching by adding words with similar meaning/similar topic to query
- Alternative strategies:
  - Use fixed lexical resource
    - E.g. WordNet

# Query Expansion

- Basic idea:
  - Improve matching by adding words with similar meaning/similar topic to query
- Alternative strategies:
  - Use fixed lexical resource
    - E.g. WordNet
  - Use information from document collection
    - Pseudo-relevance feedback

# WordNet Based Expansion

- In Information Retrieval settings, mixed history
  - Helped, hurt, or no effect
  - With long queries & long documents, no/bad effect

# WordNet Based Expansion

- In Information Retrieval settings, mixed history
  - Helped, hurt, or no effect
  - With long queries & long documents, no/bad effect
- Some recent positive results on short queries
    - E.g. Fang 2008
  - Contrasts different WordNet, Thesaurus similarity
  - Add semantically similar terms to query
    - Additional weight factor based on similarity score

# Similarity Measures

- Definition similarity: $S_{def}(t_1,t_2)$
  - Word overlap between glosses of all synsets
    - Divided by total numbers of words in all synsets glosses

# Similarity Measures

- Definition similarity: $S_{def}(t_1,t_2)$
  - Word overlap between glosses of all synsets
    - Divided by total numbers of words in all synsets glosses
- Relation similarity:
  - Get value if terms are:
    - Synonyms, hypernyms, hyponyms, holonyms, or meronyms

# Similarity Measures

- Definition similarity: $S_{def}(t_1,t_2)$
  - Word overlap between glosses of all synsets
    - Divided by total numbers of words in all synsets glosses
- Relation similarity:
  - Get value if terms are:
    - Synonyms, hypernyms, hyponyms, holonyms, or meronyms
- Term similarity score from Lin's thesaurus

# Results

- Definition similarity yields significant improvements
  - Allows matching across POS
  - More fine-grained weighting than binary relations

# Managing Morphological Variants

- Bilotti et al. 2004
- “What Works Better for Question Answering: Stemming or Morphological Query Expansion?”

# Managing Morphological Variants

- Bilotti et al. 2004
- "What Works Better for Question Answering: Stemming or Morphological Query Expansion?"
- Goal:
  - Recall-oriented document retrieval for QA
    - Can't answer questions without relevant docs

# Managing Morphological Variants

- Bilotti et al. 2004
- "What Works Better for Question Answering: Stemming or Morphological Query Expansion?"
- Goal:
  - Recall-oriented document retrieval for QA
    - Can't answer questions without relevant docs
- Approach:
  - Assess alternate strategies for morphological variation

# Question

- Comparison
  - Index time stemming
    - Stem document collection at index time
    - Perform comparable processing of query
    - Common approach
      - Widely available stemmer implementations: Porter, Krovetz

# Question

- Comparison
  - Index time stemming
    - Stem document collection at index time
    - Perform comparable processing of query
    - Common approach
      - Widely available stemmer implementations: Porter, Krovetz
  - Query time morphological expansion
    - No morphological processing of documents at index time
    - Add additional morphological variants at query time
      - Less common, requires morphological generation

# Prior Findings

- Mostly focused on stemming
- Mixed results (in spite of common use)
  - Harman found little effect in ad-hoc retrieval: Why?

# Prior Findings

- Mostly focused on stemming
- Mixed results (in spite of common use)
  - Harman found little effect in ad-hoc retrieval: Why?
    - Morphological variants in long documents
    - Helps some, hurts others: How?

# Prior Findings

- Mostly focused on stemming
- Mixed results (in spite of common use)
  - Harman found little effect in ad-hoc retrieval: Why?
    - Morphological variants in long documents
    - Helps some, hurts others: How?
      - Stemming captures unrelated senses: e.g. AIDS → aid
  - Others:
    - Large, obvious benefits on morphologically rich langs.
    - Improvements even on English

# Prior Findings

- Mostly focused on stemming
- Mixed results (in spite of common use)
  - Harman found little effect in ad-hoc retrieval: Why?
    - Morphological variants in long documents
    - Helps some, hurts others: How?
      - Stemming captures unrelated senses: e.g. AIDS → aid
  - Others:
    - Large, obvious benefits on morphologically rich langs.
    - Improvements even on English
    - Hull: most queries improve, some improve a lot

# Prior Findings

- Mostly focused on stemming
- Mixed results (in spite of common use)
  - Harman found little effect in ad-hoc retrieval: Why?
    - Morphological variants in long documents
    - Helps some, hurts others: How?
      - Stemming captures unrelated senses: e.g. AIDS → aid
  - Others:
    - Large, obvious benefits on morphologically rich langs.
    - Improvements even on English
    - Hull: most queries improve, some improve a lot
    - Monz: Index time stemming improved QA

# Overall Approach

- Head-to-head comparison
- AQUAINT documents

# Overall Approach

- Head-to-head comparison
- AQUAINT documents
- Retrieval based on Lucene
  - Boolean retrieval with tf-idf weighting

# Overall Approach

- Head-to-head comparison
- AQUAINT documents
- Retrieval based on Lucene
  - Boolean retrieval with tf-idf weighting
- Compare retrieval varying stemming and expansion

# Overall Approach

- Head-to-head comparison
- AQUAINT documents
- Retrieval based on Lucene
  - Boolean retrieval with tf-idf weighting
- Compare retrieval varying stemming and expansion
- Assess results

# Improving a Test Collection

- Observation: (We've seen it, too.)
  - # of known relevant docs in TREC QA very small

# Improving a Test Collection

- Observation: (We've seen it, too.)
  - # of known relevant docs in TREC QA very small
  - TREC 2002: 1.95 relevant per question in pool
  - Clearly many more
- Approach:

# Improving a Test Collection

- Observation: (We've seen it, too.)
  - # of known relevant docs in TREC QA very small
  - TREC 2002: 1.95 relevant per question in pool
  - Clearly many more
- Approach:
  - Manually create improve relevance assessment
  - Create queries from originals

# Improving a Test Collection

- Observation: (We've seen it, too.)
  - # of known relevant docs in TREC QA very small
  - TREC 2002: 1.95 relevant per question in pool
  - Clearly many more
- Approach:
  - Manually create improve relevance assessment
  - Create queries from originals
    - Terms that "must necessarily" appear in relevant docs
  - Retrieve and verify documents
  - Found 15.84 relevant per question

# Example

- Q: What is the name of the volcano that destroyed the ancient city of Pompeii?"  A: Vesuvius
- New search query:

# Example

- Q: What is the name of the volcano that destroyed the ancient city of Pompeii?"  A: Vesuvius
- New search query: "Pompeii" and "Vesuvius"
- In A.D. 79, long-dormant Mount Vesuvius erupted, burying the Roman cities of Pompeii and Herculaneum in volcanic ash."

# Example

- Q: What is the name of the volcano that destroyed the ancient city of Pompeii?" A: Vesuvius
- New search query: "Pompeii" and "Vesuvius"
- Relevant: In A.D. 79, long-dormant Mount Vesuvius erupted, burying the Roman cities of Pompeii and Herculaneum in volcanic ash."
- Pompeii was pagan in A.D. 79, when Vesuvius erupted.

# Example

- Q: What is the name of the volcano that destroyed the ancient city of Pompeii?" A: Vesuvius
- New search query: "Pompeii" and "Vesuvius"
- Relevant: In A.D. 79, long-dormant Mount Vesuvius erupted, burying the Roman cities of Pompeii and Herculaneum in volcanic ash."
- Unsupported: Pompeii was pagan in A.D. 79, when Vesuvius erupted.
- Vineyards near Pompeii grow in volcanic soil at the foot of Mt. Vesuvius

# Example

- Q: What is the name of the volcano that destroyed the ancient city of Pompeii?" A: Vesuvius
- New search query: "Pompeii" and "Vesuvius"
- Relevant: In A.D. 79, long-dormant Mount Vesuvius erupted, burying the Roman cities of Pompeii and Herculaneum in volcanic ash."
- Unsupported: Pompeii was pagan in A.D. 79, when Vesuvius erupted.
- Irrelevant: Vineyards near Pompeii grow in volcanic soil at the foot of Mt. Vesuvius

# Stemming & Expansion

- Base query form: Conjunct of disjuncts
  - Disjunction over morphological term expansions

# Stemming & Expansion

- Base query form: Conjunct of disjuncts
  - Disjunction over morphological term expansions
  - Rank terms by IDF
  - Successive relaxation by dropping lowest IDF term
- Contrasting conditions:

# Stemming & Expansion

- Base query form: Conjunct of disjuncts
  - Disjunction over morphological term expansions
  - Rank terms by IDF
  - Successive relaxation by dropping lowest IDF term
- Contrasting conditions:
  - Baseline: No nothing (except stopword removal)

# Stemming & Expansion

- Base query form: Conjunct of disjuncts
  - Disjunction over morphological term expansions
  - Rank terms by IDF
  - Successive relaxation by dropping lowest IDF term
- Contrasting conditions:
  - Baseline: No nothing (except stopword removal)
  - Stemming: Porter stemmer applied to query, index

# Stemming & Expansion

- Base query form: Conjunct of disjuncts
  - Disjunction over morphological term expansions
  - Rank terms by IDF
  - Successive relaxation by dropping lowest IDF term
- Contrasting conditions:
  - Baseline: No nothing (except stopword removal)
  - Stemming: Porter stemmer applied to query, index
  - Unweighted inflectional expansion:
    - POS-based variants generated for non-stop query terms

# Stemming & Expansion

- Base query form: Conjunct of disjuncts
  - Disjunction over morphological term expansions
  - Rank terms by IDF
  - Successive relaxation by dropping lowest IDF term
- Contrasting conditions:
  - Baseline: No nothing (except stopword removal)
  - Stemming: Porter stemmer applied to query, index
  - Unweighted inflectional expansion:
    - POS-based variants generated for non-stop query terms
  - Weighted inflectional expansion: prev. + weights

# Example

- Q: What lays blue eggs?
- Baseline: blue AND eggs AND lays
- Stemming: blue AND egg AND lai
- UIE: blue AND (eggs OR egg) AND (lays OR laying OR lay OR laid)
- WIE: blue AND (eggs OR egg$^{w}$) AND (lays OR laying$^{w}$ OR lay$^{w}$ OR laid$^{w}$)

# Evaluation Metrics

- Recall-oriented

# Evaluation Metrics

- Recall-oriented: why?
  - All later processing filters

# Evaluation Metrics

- Recall-oriented: why?
  - All later processing filters
- Recall @ n:
  - Fraction of relevant docs retrieved at some cutoff

# Evaluation Metrics

- Recall-oriented: why?
  - All later processing filters
- Recall @ n:
  - Fraction of relevant docs retrieved at some cutoff
- Total document reciprocal rank (TDRR):
  - Compute reciprocal rank for rel. retrieved documents
  - Sum overall documents
  - Form of weighted recall, based on rank

# Results

| Limit | Experiment | Recall relevant | Recall Δ | Recall both | Recall Δ | TDRR relevant | TDRR Δ | TDRR both | TDRR Δ |
|---|---|---|---|---|---|---|---|---|---|
| 100 | unstemmed | 0.2720 | | 0.2595 | | 0.6403 | | 0.6673 | |
| | stemmed | 0.2589 | −4.82% | 0.2460 | −5.20% | 0.5869 | −8.33% | 0.5987 | −10.28% |
| | expanded | 0.2748 | +1.03% | 0.2612 | +0.66% | 0.5752 | −10.16% | 0.5968 | −10.56% |
| | w. expanded | 0.2944 | +8.24% | 0.2798 | +7.82% | 0.6094 | −4.82% | 0.6305 | −5.52% |
| 250 | unstemmed | 0.3738 | | 0.3584 | | 0.6509 | | 0.6790 | |
| | stemmed | 0.3626 | −3.00% | 0.3474 | −3.07% | 0.5995 | −7.90% | 0.6122 | −9.84% |
| | expanded | 0.3682 | −1.50% | 0.3533 | −1.42% | 0.5863 | −9.93% | 0.6090 | −10.31% |
| | w. expanded | 0.3776 | +1.02% | 0.3618 | +0.95% | 0.6185 | −4.98% | 0.6406 | −5.67% |
| 500 | unstemmed | 0.5393 | | 0.5123 | | 0.6596 | | 0.6879 | |
| | stemmed | 0.5364 | −0.54% | 0.5097 | −0.51% | 0.6086 | −7.74% | 0.6216 | −9.65% |
| | expanded | 0.5467 | +1.37% | 0.5182 | +1.15% | 0.5957 | −9.69% | 0.6186 | −10.08% |
| | w. expanded | 0.5551 | +2.93% | 0.5258 | +2.64% | 0.6279 | −4.81% | 0.6501 | −5.50% |
| 750 | unstemmed | 0.5981 | | 0.5689 | | 0.6614 | | 0.6899 | |
| | stemmed | 0.5934 | −0.79% | 0.5638 | −0.90% | 0.6103 | −7.72% | 0.6234 | −9.63% |
| | expanded | 0.6093 | +1.87% | 0.5799 | +1.93% | 0.5976 | −9.65% | 0.6207 | −10.03% |
| | w. expanded | 0.6112 | +2.19% | 0.5816 | +2.23% | 0.6296 | −4.81% | 0.6520 | −5.49% |
| 1000 | unstemmed | 0.6196 | | 0.5917 | | 0.6618 | | 0.6904 | |
| | stemmed | 0.6131 | −1.05% | 0.5824 | −1.57% | 0.6111 | −7.67% | 0.6238 | −9.64% |
| | expanded | 0.6290 | +1.52% | 0.5993 | +1.28% | 0.5980 | −9.65% | 0.6211 | −10.03% |
| | w. expanded | 0.6290 | +1.52% | 0.5993 | +1.28% | 0.5980 | −9.65% | 0.6211 | −10.03% |

# Overall Findings

- Recall:

# Overall Findings

- Recall:
  - Porter stemming performs WORSE than baseline
    - At all levels

# Overall Findings

- Recall:
  - Porter stemming performs WORSE than baseline
    - At all levels
  - Expansion performs BETTER than baseline
    - Tuned weighting improves over uniform
  - Most notable at lower cutoffs

# Overall Findings

- Recall:
  - Porter stemming performs WORSE than baseline
    - At all levels
  - Expansion performs BETTER than baseline
    - Tuned weighting improves over uniform
  - Most notable at lower cutoffs
- TDRR:
  - Everything's worse than baseline
  - Irrelevant docs promoted more

# Observations

- Why is stemming so bad?

# Observations

- Why is stemming so bad?
  - Porter stemming linguistically naïve, over-conflates
    - police = policy; organization = organ; European != Europe

# Observations

- Why is stemming so bad?
  - Porter stemming linguistically naïve, over-conflates
    - police = policy; organization = organ; European != Europe
  - Expansion better motivated, constrained

# Observations

- Why is stemming so bad?
  - Porter stemming linguistically naïve, over-conflates
    - police = policy; organization = organ; European != Europe
  - Expansion better motivated, constrained
- Why does TDRR drop when recall rises?

# Observations

- Why is stemming so bad?
  - Porter stemming linguistically naïve, over-conflates
    - police = policy; organization = organ; European != Europe
  - Expansion better motivated, constrained
- Why does TDRR drop when recall rises?
  - TDRR – and RR in general – very sensitive to swaps at higher ranks
    - Some erroneous docs added higher

# Observations

- Why is stemming so bad?
  - Porter stemming linguistically naïve, over-conflates
    - police = policy; organization = organ; European != Europe
  - Expansion better motivated, constrained
- Why does TDRR drop when recall rises?
  - TDRR – and RR in general – very sensitive to swaps at higher ranks
    - Some erroneous docs added higher
- Expansion approach provides flexible weighting