Learning a Monolingual Language Model from a Multilingual Text Database

Rayid Ghani & Rosie Jones
School of Computer Science
Carnegie Mellon University
What is a LM?

- Probability distribution over terms in the language
- Most common LMs are n-gram models
- Unigram Model:
  - For a doc $d$ that is composed of words $w_1, w_2, \ldots, w_n$:
  
  $$P(d) = \prod_{i=1}^{n} P(w_i)$$
How to create a LM?

- Acquire a (representative) Corpus in the language
- Apply standard language modeling methods (Chen & Goodman 1996)
What corpora are available?

- Explicit, marked up corpora: Linguistic Data Consortium -- 20 languages [Liebermann and Cieri 1998]

- Search Engines -- implicit language-specific corpora, European languages, Chinese and Japanese
  - Excite -- 11 languages
  - Google -- 12 languages
  - AltaVista -- 25 languages
  - Lycos -- 25 languages
Motivating Question

- Can we quickly get a LM of contemporary usage for some minority language e.g. Tagalog? Slovene? Persian?

(Tagalog is spoken in the Phillipines by 39 million people; Slovene is spoken in Slovenia by 2 million people)
Our Solution

- Acquire Construct a Corpus of the language efficiently with minimum human effort
- Apply standard language modeling methods
How to construct our corpus?

- Expensive Method
  - Spider all of the web (~1.5 Billion Pages) and use a language filter

- Less Expensive Method
  - Selective Spidering
Task

- **Given:**
  - 1 Document in Target Language
  - 1 Document in Other Language
  - Access to a Web Search Engine

- **Create a Language Model of the Target Language quickly with no human effort**
Experimental Data

- 498 documents in Tagalog
  - 133 geocities.com
  - 29 maxpages.com
  - 29 bhepatitis.com
- 16,000 distractors in English and Brazilian Portuguese
- Common vocabulary
  - “at”: Tagalog for “and”
  - “the”, “homepage”: Internet English
Algorithm

Seed Docs

LM Builder

Query Generator

WWW

Language Filter
Algorithm

1. Select one seed document from each of M and O.
2. Build language models for M' (LM_{M'}) and O' (LM_{O'}) from the seed documents.
3. Sample a document from the database.
4. Use the language filter to decide whether to add the new document to the list of documents in M, or those in O.
5. Update the language models for LM_{M'} and LM_{O'}.
6. If the stopping criterion has not been reached, go back to Step 3
Methodology -- Adaptive Query-based Sampling

- Query-based sampling [Callan et al. 99]
- Adaptive Sampling -- used in oceanography [Curtin et al. 1993], environmental monitoring [Thompson and George 1995]
  - Target class is rare
  - Sampling is expensive
  - Target examples clustered together
Language Identification

- Simple filter – uses the current LMs
- Does not require any more prior knowledge than the seed documents
- Improves with time
## Query Methods

<table>
<thead>
<tr>
<th>Query Method</th>
<th>Sample Query</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random</td>
<td></td>
</tr>
<tr>
<td>Most-Frequent</td>
<td>+sa</td>
</tr>
<tr>
<td>Unigram</td>
<td>+kanyang</td>
</tr>
<tr>
<td>Most-Frequent-exclude-Most-Frequent</td>
<td>+sa –the</td>
</tr>
<tr>
<td>Unigram-exclude-Most-Frequent</td>
<td>+kanyang –the</td>
</tr>
<tr>
<td>Unigram-exclude-Unigram</td>
<td>+kanyang -more</td>
</tr>
</tbody>
</table>
Evaluation

- Average of three runs
- True Unigram Model constructed from all 498 Tagalog documents
- Take sampled collection and compare to True Unigram Model

Metrics
- Kullback Leibler divergence (relative entropy, distance between probability distributions)
- Percent vocabulary coverage
- Cumulative term frequency
Percentage of vocabulary covered

Unigram

Unigram-exclude-unigram

Unigram-exclude-most-frequent

Most-frequent

Random

Percent vocabulary coverage vs. Total all documents sampled
Conclusions

- Can automatically build a corpus and thus a LM for a minority language
- Short queries successful
- Mostly, single document adequate starting point
Future Work

- Stopping Criteria
- Synthetic Data
- More sophisticated filtering
- Other languages
- Bigger Data sets (Web)
- Incorporate hyper-link information
- Use a Reinforcement Learning framework and learn the query parameters
- Release code
- URL for this presentation
  - http://www.cs.cmu.edu/~rayid/talks

- URL for the project
  - http://www.cs.cmu.edu/~TextLearning