Finding topic-descriptive segments in a document using language modeling

Myung-ha Jang

March 23, 2014

1 Motivation

We extracted and observed a correlation between frequent terms from Wikipedia articles and terms with high TF-IDF in each document set [1]. Although the list contained generally relevant keywords, it still contained noisy terms. We also observed that the teaching documents intrinsically consist of a few text segments that directly discuss the topics and other text segments that do not contain topic-indicative keywords. The examples of such irrelevant segments are course administrative content, examples that use analogous language, or other formatting such as copyrights. To identify highly relevant segments, or to filter out non-relevant segments, we applied language modeling to the document corpus. For simplicity, we set a segment as a fixed window of k terms, and computed a segment probability given a document. As a preliminary check-up, we visualized each segment's probability to see if a set of high-scored segments contain good keywords and if a set of low-scored segments contain words to be abandoned for clustering.

2 Language Modeling

To overcome the sparsity of the collection documents, we used Google n-gram data upto trigrams as as the collection probability for smoothing. All models use Dirichlet Smoothing with parameters $\mu_1 = \mu_2 = \mu_3 = 2000$.

2.1 Unigram model

$$logP(s|d) = \sum_{w \in s} logP(w|d) = \frac{c(w;d) + \mu_1 \dot{P}(w|C)}{|d| + \mu_1}$$

$$P(w|C) = \frac{google\_ngram\_count(w)}{\sum_{w' \in C} google\_ngram\_count(w')}$$

where s is a segment (i.e., a fixed window of terms), $|d|$ is a length of the document.

[1] Results available http://cs.umass.edu/~mhjang/table.html

2.2 Bigram model

$$logP(s|d) = \sum_{w_i, w_{i-1} \in s} logP(w_i|w_{i-1}d) = \frac{c(w_i, w_{i-1}; d) + \mu_2 \dot{P}(w_i|w_{i-1}, C)}{c(w_{i-1}; d) + \mu_2}$$

$$P(w_i|w_{i-1}, C) = \frac{google\_ngram\_count(w_i, w_{i-1})}{google\_ngram\_count(w_{i-1}) + 1}$$

2.3 Trigram model

$$logP(s|d) = \sum_{w_i, w_{i-1}, w_{i-2} \in s} logP(w_i|w_{i-2}, w_{i-1}; d) = \frac{c(w_i, w_{i-1}, w_{i-2}; d) + \mu_3 \dot{P}(w_i|(w_{i-1}, w_{i-2}), C)}{c(w_{i-1}, w_{i-2}; d) + \mu_3}$$

$$P(w_i|w_{i-1}, C) = \frac{google\_ngram\_count(w_i, w_{i-1}, w_{i-2})}{google\_ngram\_count(w_{i-1}, w_{i-2}) + 1}$$

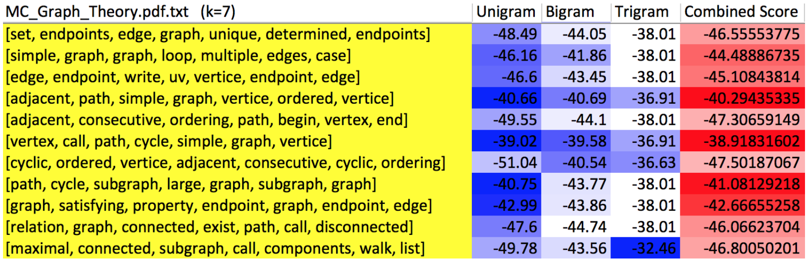
Then the three models can be combined using two parameters, $0 \leq \alpha, \beta \leq 1$:

$$score(s) = \alpha\lambda_{unigram}(s) + \beta\lambda_{bigram}(s) + (1 - \alpha - \beta)\lambda_{trigram}(s)$$

where $\lambda_{N-gram}(s)$ refers to the log probability $P(s|d)$ in the corresponding N-gram model. Although we empirically set $\alpha = 0.7, \beta = 0.2$ for the current experiment, finding the right parameters should be investigated later.

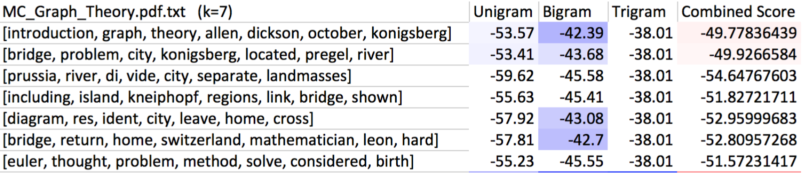
3 Preliminary Results

The whole segment data visualized by their score is available at the shared Google drive/results/languagemodeling.xlsx. The examples of relevant and irrelevant segments are shown in Figure 1 and 2.



MC_Graph_Theory.pdf.txt (k=7)	Unigram	Bigram	Trigram	Combined Score
[set, endpoints, edge, graph, unique, determined, endpoints]	-48.49	-44.05	-38.01	-46.55553775
[simple, graph, graph, loop, multiple, edges, case]	-46.16	-41.86	-38.01	-44.48886735
[edge, endpoint, write, uv, vertice, endpoint, edge]	-46.6	-43.45	-38.01	-45.10843814
[adjacent, path, simple, graph, vertice, ordered, vertice]	-40.66	-40.69	-36.91	-40.29435335
[adjacent, consecutive, ordering, path, begin, vertex, end]	-49.55	-44.1	-38.01	-47.30659149
[vertex, call, path, cycle, simple, graph, vertice]	-39.02	-39.58	-36.91	-38.91831602
[cyclic, ordered, vertice, adjacent, consecutive, cyclic, ordering]	-51.04	-40.54	-36.63	-47.50187067
[path, cycle, subgraph, large, graph, subgraph, graph]	-40.75	-43.77	-38.01	-41.08129218
[graph, satisfying, property, endpoint, graph, endpoint, edge]	-42.99	-43.86	-38.01	-42.66655258
[relation, graph, connected, exist, path, call, disconnected]	-47.6	-44.74	-38.01	-46.06623704
[maximal, connected, subgraph, call, components, walk, list]	-49.78	-43.56	-32.46	-46.80050201

Figure 1: A set of segments of relevant terms



MC_Graph_Theory.pdf.txt (k=7)	Unigram	Bigram	Trigram	Combined Score
[introduction, graph, theory, allen, dickson, october, konigsberg]	-53.57	-42.39	-38.01	-49.77836439
[bridge, problem, city, konigsberg, located, pregel, river]	-53.41	-43.68	-38.01	-49.9266584
[prussia, river, di, vide, city, separate, landmasses]	-59.62	-45.58	-38.01	-54.64767603
[including, island, kneiphopf, regions, link, bridge, shown]	-55.63	-45.41	-38.01	-51.82721711
[diagram, res, ident, city, leave, home, cross]	-57.92	-43.08	-38.01	-52.95999683
[bridge, return, home, switzerland, mathematician, leon, hard]	-57.81	-42.7	-38.01	-52.80957268
[euler, thought, problem, method, solve, considered, birth]	-55.23	-45.55	-38.01	-51.57231417

Figure 2: A set of segments of irrelevant terms

4 Future Directions

We will further investigate towards resolving below research questions:

- How do we use the language modeling score to represent the documents?
  :As as a starting point, we will use the terms ranked by summing over the score from each segment that the term appears

- Does the language modeling approach improve clustering result?
  :We will compare the clustering result with empirically set parameters as a baseline.

- How do we find the optimal parameters α, β?

  - Should we use extrinsic evaluation (i.e., clustering result) or intrinsic evaluation (i.e., extracted keyword quality) for parameter learning?