

Text Technologies Assignment

Data

The data used in these experiments is taken from the AOL query logs (released August 2006). The training data comprises roughly 250MBs and consists of over 328,000 query sessions - consisting of over 18,000,000 queries (approximately 55 queries per session), just under 1/3 of which are single terms.

Similar to the data used in Analysis of a Very Large Web Search Engine Query Log_{3}, there is an average of a little over 2 query terms per query.

Ngram Models

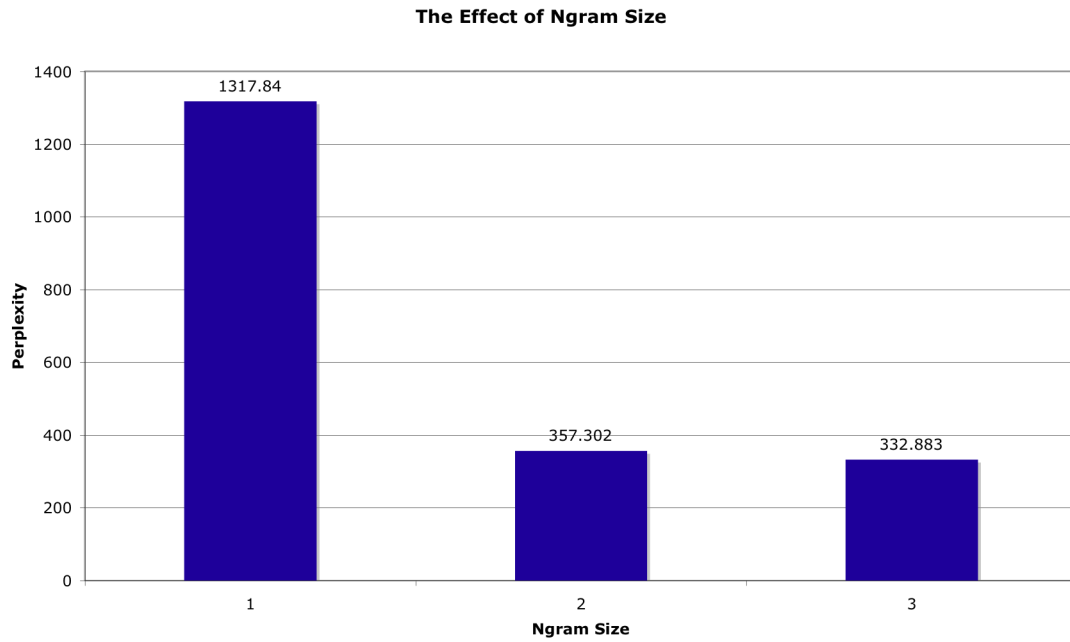
The SRI Language Model Toolkit_{1} was used to create ngram-models for all training set queries.

The means used to measure the effectiveness of the language models will be perplexity. This is a measure of "surprise" caused by the test sample when a probability distribution is generated from a training set_{5}.

This yielded a baseline perplexity of 332.883, on which we will aim to improve.

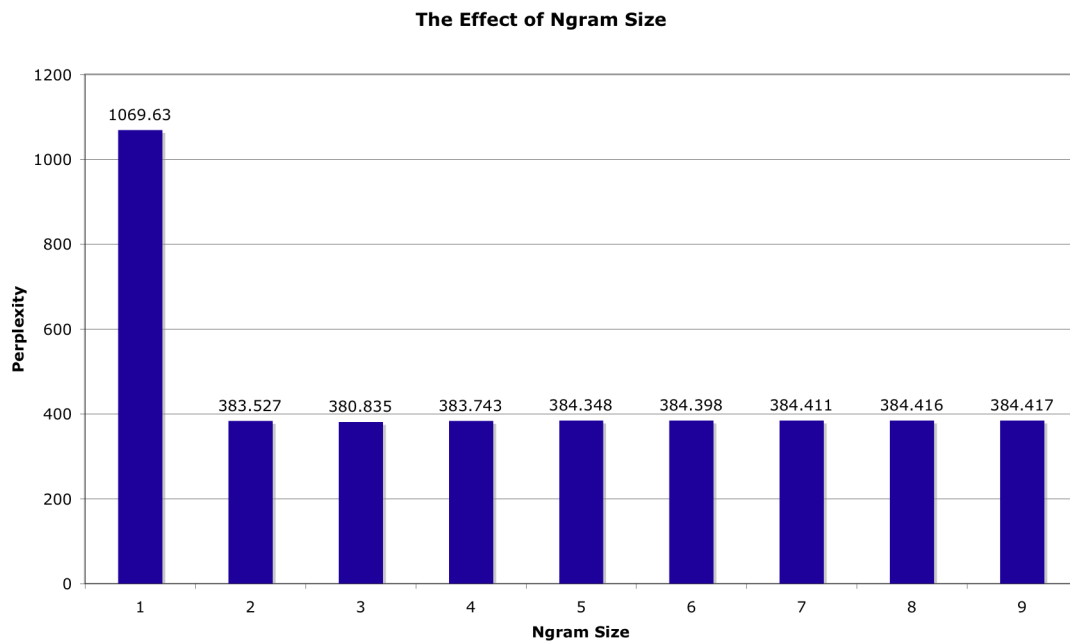
Ngram Sizes

Ngrams are sequences of words of length n . For example, a unigram is a 1-gram, a bigram a 2-gram and a tri-gram a 3-gram. A standard language model contains orders of ngrams 1 to 3. The graph below shows the effect of order on perplexity:



Here, we see that using order 3 has lower perplexity than order 2, and both of them yield a far lower perplexity than using order 1.

By creating a model of order 9 on 20% of the test set (trying to use the full test set required too much memory to run) perplexities for orders 1 through 9 were calculated as shown in the graph below:



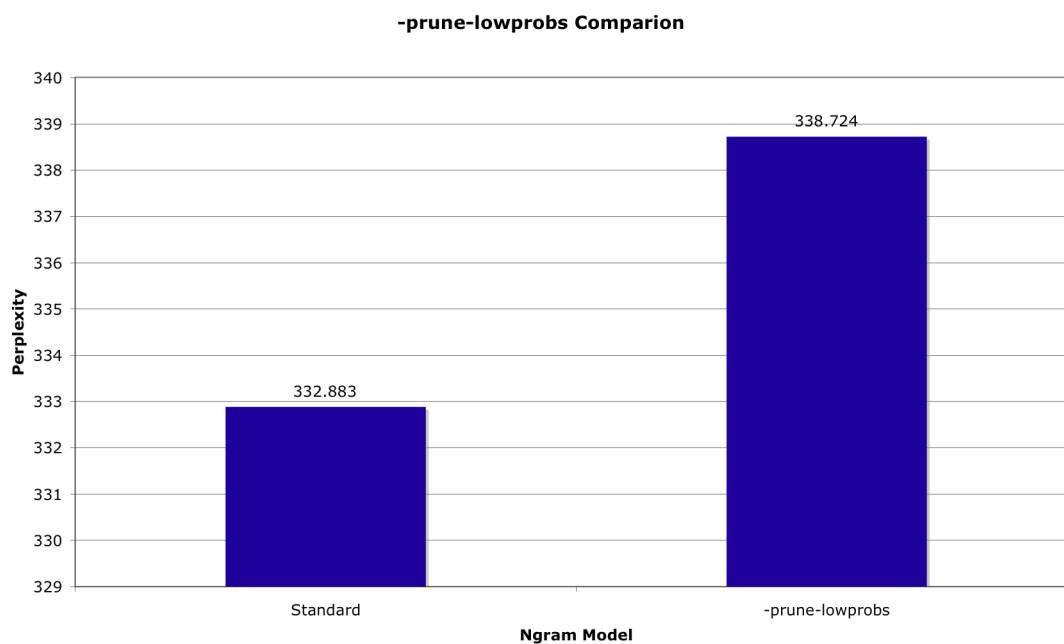
Here, we can see that there is not a significant difference (less than 4) in the perplexities calculated

from orders of ngram above 1, however 3 is marginally better. This is the default value.

Smoothing

Smoothing is the manipulation of data in order to eliminate noise and fill in missing data_{6}.

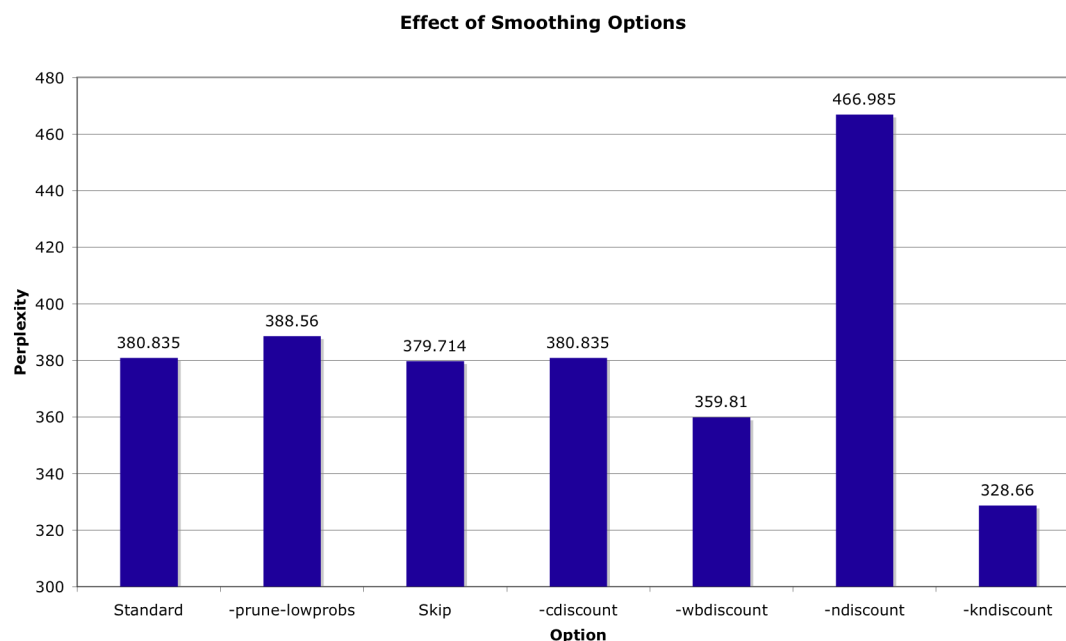
The most basic way to eliminate noise is to remove low probabilities (for example mis-spellings and common “stop words” like *the*) from the Language Models. The SRILMT provides the means for doing this (as detailed in the manual pages_{2}) with the `-prune-lowprobs` option. This is shown comparative to the baseline perplexity in the graph below:



Here we can see that removing low probability ngrams has actually made the perplexity marginally worse (by a little under 6 points) rather than improving it.

Clearly smoothing is a complex problem, in order to explore it more thoroughly, a reduced dataset consisting of 20% of the training data will be used. The baseline perplexity for this is 380.835.

The SRILMT provides various smoothing options_{2}, the results of which are shown in the graph below:



Here, we can see that smoothing does not necessarily result in the perplexity of the test set being reduced. The `-ndiscout` option, for example, gave a worse result than our baseline. Creating a skip ngram reduced the perplexity only very marginally (a little over 1 point). However both the `-wbdiscout` and the `-kndiscout` options gave an improved perplexity, with the `-kndiscout` giving the most improvement (over 50 points).

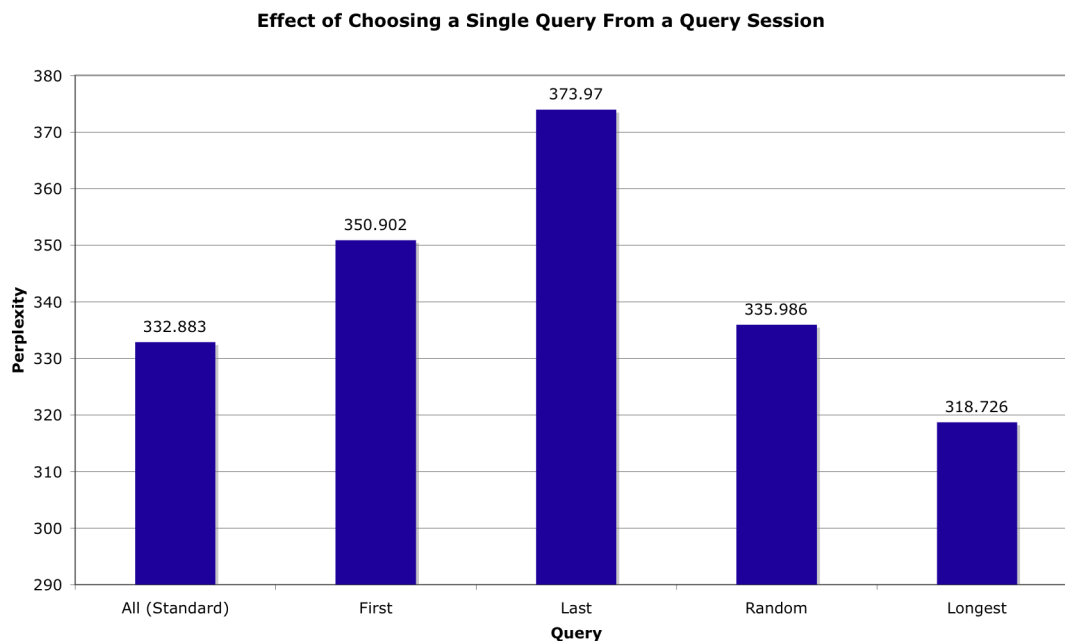
While both the `-wbdiscout` and the `-kndiscout` options can take an argument n , experimentation with this did not appear to improve the perplexity.

Selecting Specific Queries

In a query session, how do we determine which query, if any, is the most important? To determine this, single queries were taken from the sessions and a language model built from this subset. The single queries were selected as follows:

- The first query
- The last query
- A random query
- The longest query (according to number of words)

Results, compared to the baseline probability are shown in the graph below:

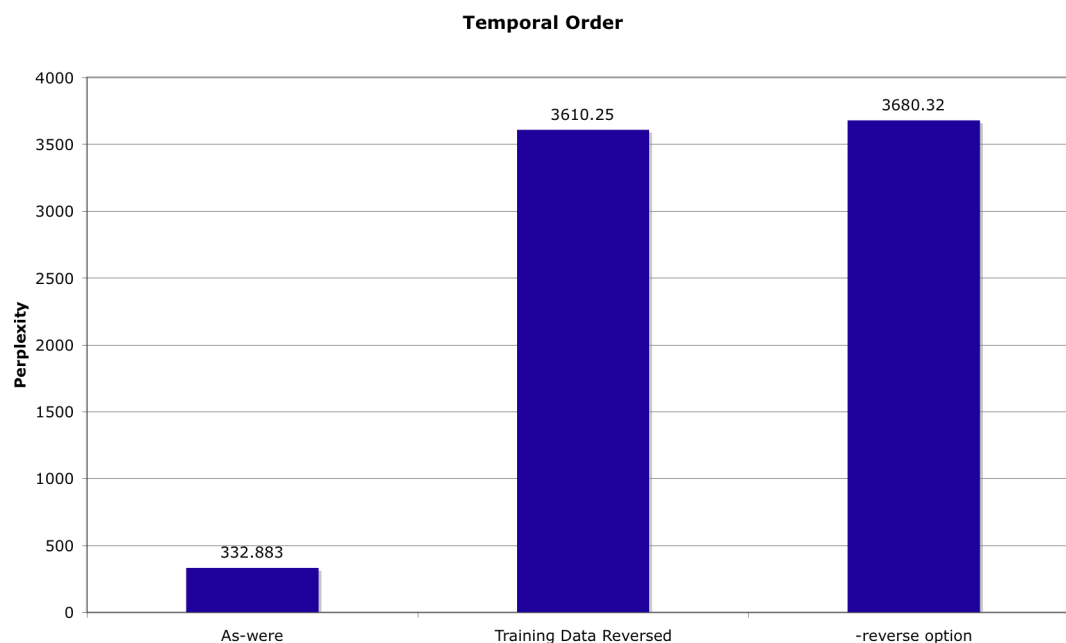


Here we can see that using a random query does not perform significantly worse than using all queries (with a difference in perplexity of around 3), but using the first query gives us significantly worse performance, and the last query worse still. By choosing the longest query, we achieve an improvement in perplexity of nearly 15.

From these results it appears that people are less specific about what they are looking for towards the end of a query session than they are at the beginning of a query session, and where they are most specific, by having the longest query, is where we achieve the best perplexity.

Temporal Order

In order to calculate the effect of temporal order, two methods were used. The first trained a language model on the full training set, with all query terms in reverse order. The second calculated the perplexity of the test set using the `-reverse` option in the SRILMT ngram program. The results are shown in the graph below:



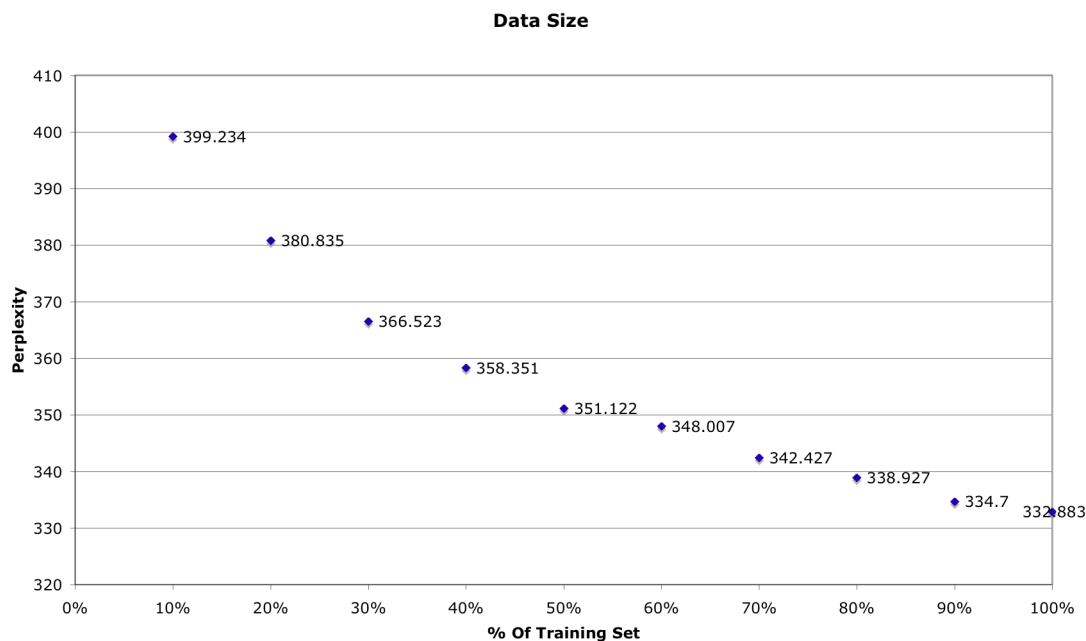
Here we can see that there is a massive (more than ten times) increase in perplexity when query terms are reversed. The reversal of the training data has a marginally lower perplexity (a difference of around 70 between the two reversed options).

This suggests that when people write queries they use more specific terms early in the query. If later terms are more general, going backwards, we will have a greater number of possible options available to follow them. This would explain the massive increase in perplexity.

Further research would be necessary to determine whether this is the case, however if it is, it would suggest that terms that occur early in query should be given higher weights than those which occur later.

Sample Size and Perplexity

The training set used to generate 9 files, one containing 10% of the data, 20%, 30% and so on through to 90% (100% is the training set itself). Language models were built for the different percentages of data, and the results of this are shown in the graph below:



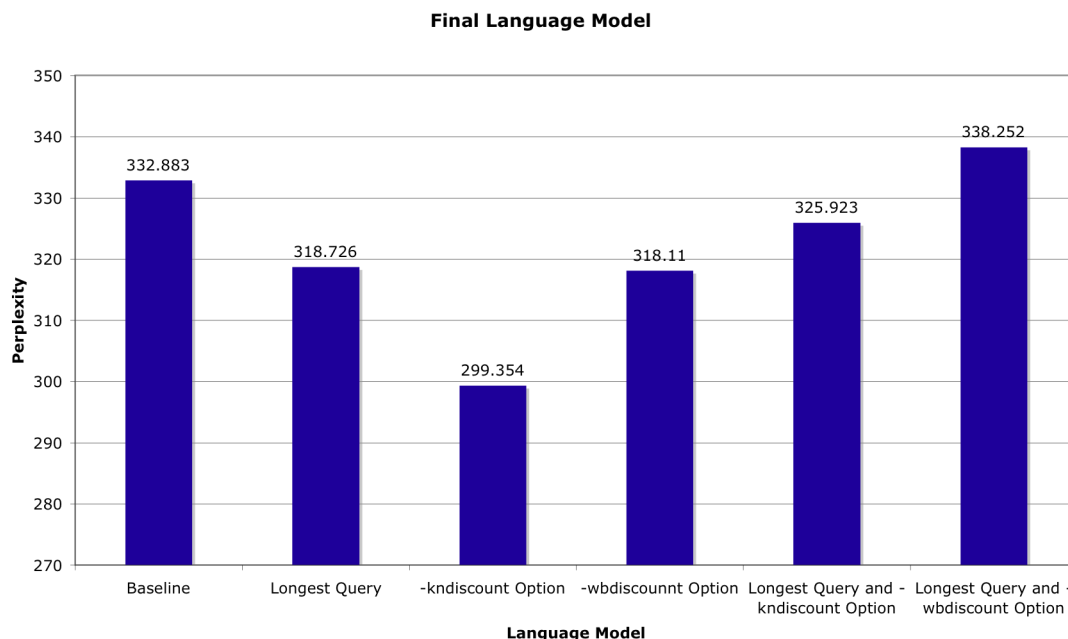
Here, we can see that the more data the set is trained on the more the perplexity is reduced. The difference an extra 10% of the data makes is reduced as we get to higher percentages of the test set. This is to be expected, in going from 10% to 20% of the training set we double the amount of data we have to train on and reduce the perplexity by nearly 20 points. By the time we increase the amount of data from 80% to 100% however this is only an increase of $\frac{1}{4}$, and a decrease of only 6 points in perplexity.

Conclusions

From the results above, we can select the best characteristics from each section. An ngram size of 3 (standard), the `-kndiscount` smoothing option, the longest ngram from each query session, maintain the temporal order of the queries, and use as much data as is available.

This Language Model yielded a perplexity of 325.923. Whilst an improvement on our baseline this is actually not as good as the perplexity when the longest query was used from each session (smoothing omitted), or when just the smoothing was done on the full set.

This is shown in the graph below:



Here, it can be seen that combining two improvements that have worked well on their own may not always be successful.

Taking the longest query and using the `-wbdiscout` option yielded similar improvements in perplexity from the baseline, however combining the two yielded a worse perplexity than the baseline.

The best perplexity produced was therefore using the `-kndiscount` smoothing on an otherwise standard model.

In order to improve this it might be necessary to take into account the terms themselves (rather than the replacing them with numbers). We could then consider:

- Spelling correction
- Splitting URLs
- Case insensitivity (typical in modern search engines)
- Removing stop words (like *the*)

Problems Encountered

The SRILMT is not particularly well documented and hence it was difficult to work out how to use it, and it took a long time to get started. Language Models took a long time to create and some of the more complex ones could not be created on my machine, as they required more than

1GB of RAM to build. In all, I created around 50 Language Models, however not all of these were useful! In the end, I used the reduced training set for some sections as it meant that I could see what was effective in a shorter time frame and creating a Language Model that yielded the same results as a baseline model was not quite as frustrating!

References

1. SRILMT Home
<http://www.speech.sri.com/projects/srilm/>
2. SRILMT Manual
<http://www.speech.sri.com/projects/srilm/manpages/>
3. Analysis of a Very Large Web Search Engine Query Log, Craig Silverstein and Monika Henzinger
4. Probabilistic Query Expansion Using Query Logs, Hang Cui, Ji-Rong Wen, Jian-Yun Nie, Wei-Ying Ma
5. Wikipedia <http://en.wikipedia.org/wiki/Perplexity>
6. Chapter 6: Smoothing Data, Filling Missing Data and Nonparametric Fitting,
documents.wolfram.com/applications/eda/SmoothingDataFillingMissingDataAndNonparametricFitting.html