# Summary Report (11/06/2014)

## Overview

In this week, I have done the following things:

* Candidate phrase selection with Ngram

## Candidate Phrase Selection with Ngram

In previous method, we treat the candidate phrase selection as a segmentation problem.

Now, we just extract all ngram (up to 5) that are in the given dictionary.

The results are shown below: (R2 score for the Muddiest Point)

Only the best performance is show here

* Candidatengram-LSA: ngram selection with LSA similarity metric
* Candidatengramstemming-npsoft: ngram selection with npsoft similarity metric but the relax constraint that if the stemmed ngram appears in the given dictionary (also stemmed), then the ngram is a candidate phrase

Observation:

* The ngram model doesn’t gain better performance (actually, it is worse)
  + The possible reason due to the redundant issue. For example, for the given phrase “bond strength”, both the “bond strength” and “bond” will be extracted. However, “bond” is redundant.

## TF-IDF Phrase

Intuitively, rank the phrases just by their TF-IDF score doesn’t work well just like ranking them by TF because many of them appear just once.

Below is the results of tf-idf model.



### Observation:

* Surprising, unlike the fact that tf-idf of unigram model is worse than tf of the unigram model, the tf-idf of phrase model is better than the tf model. (ChunkNP-TFIDF is better than ChunkNP-Hard, and SyntaxNP-TFIDF is better than SyntaxNP-Hard)
* However, they are still not better than the Soft model and Clustering model.

## LexRank

The LexRank model is implemented in Mead as a feature function.

Thus, I used the PhraseMead + LexRank as the baseline (3 basic feature in Mead + 1 LexRank feature).

In addition, the length limitation is changed from the default value 9 to 1 so that all the phrases are considered.

The results of the LexRank model are really good. It wins on all the R2 scores. Therefore, instead of just looking at the f-measure of the Rouge scores, I also looked at the precision, recall of the rouge scores to see whether the proposed clustering model could be better.



\* best performance are highlighted.

\* Only the Rouge scores for Muddiest point are shown here

### Observation:

* The good news is that the overall performance gets better with the LexRank
* The bad news is it is even better than our proposed model (except the R1-Recall and RSU4-Recall).
  + One limitation of the clustering algorithm is that when K is small, the total number of words in the summary will be much shorter than 30, which will make the recall lower.

## Clustering + LexRank

Instead of just extracted the centroid phrase in a cluster, rank the phrases in the cluster with LexRank scores.

The idea is to use the clustering just as redundancy removing.

To get a better understanding about this model, a baseline just with the LexRank is implemented (Phrase\_syntax\_lexrank and Phrase\_chunk\_lexrank).



### Observation:

* This combined model becomes even better than the Mead+LexRank (It wins on R2 scores)
* LexRank alone is not better than Mead+LexRank
* Centroid Phrases might not be good due to the clustering noise

## Combining different Distance Metrics

TODO

One of the advantages of K-Medoid is that it is easy to combine different distance metrics. Thus, I assume combing different distance metrics could improve the performance because different metrics might be complementary with each other.

## Evaluation of the source measure

TODO

A difference between summarizing answers for open-ended question and other type of summarizations is the required output.

* Only the general summary with phrase/sentence doesn’t meet the needs. It only addressed the question “what are people talking about”, but we also like to see “how many people” think about it.

## Paper I read

Chan, W., Zhou, X., Wang, W., & Chua, T.-S. (2012). Community answer summarization for multi-sentence question with group L 1 regularization. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1* (pp. 582–591). Association for Computational Linguistics.

Hovy, E., & Lin, C.-Y. (1996). Automated text summarization and the SUMMARIST system. In *Proceedings of a workshop on held at Baltimore, Maryland October 13-15, 1998 -* (p. 197). Morristown, NJ, USA: Association for Computational Linguistics.

Lin, C.-Y., & Hovy, E. (2001). From single to multi-document summarization. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics - ACL ’02* (p. 457). Morristown, NJ, USA: Association for Computational Linguistics.

Liu, Y., Li, S., Cao, Y., Lin, C.-Y., Han, D., & Yu, Y. (2008). Understanding and summarizing answers in community-based question answering services. *Proceedings of the 22Nd International Conference on Computational Linguistics*, *1*, 497–504.

Ueda, Y., Oka, M., Koyama, T., & Miyauchi, T. (2000). Toward the “at-a-glance” summary. In *Proceedings of the 18th conference on Computational linguistics* (Vol. 2, p. 878). Morristown, NJ, USA: Association for Computational Linguistics.

<http://www.mendeley.com/share/document/invite/916bfe4263/?utm_medium=email&utm_source=transactional&utm_campaign=share%2Finvitation-document>