Document Interpolation and Integration

CS 333: Natural Language Processing
Nathan Teetor

January 3, 2014

1 Introduction

1.1 Motivation

At the college level students are expected to copy down lecture slides, professors' comments, and draw connections between class topics. If a student spends all of their time organizing course notes then they have little time to learn the course material. In order to give students more time we propose an accomodation for students that happens as they take notes. By copying and inserting lecture notes back into previously taken course notes a system could help students draw connections between lectures and maintain a single organized file of notes for a course.

1.2 Clarifying, Simplfying the Task

Given a combined set of what were once individual documents and an additional document (sentence, paragraph, token, etc.) could we intelligently integrate that new document into the larger body of texts? However, to consider a system such as this we need a data set of course notes for a variety of topics taken by a diverse group of students. Because this critical component was unavailable in the time alloted, our goal needed to be simplified. Therefore, we proposed a more general system designed to integerate any text into a larger body of text.

Specifically, this paper focuses on three approaches for text integration. The most basic approach uses exact word overlap. This approach has been tried previously (Clough et al. 2002; Qiu et al. 2006; Zhang and Patrick, 2005) and serves as the baseline for the two additional methods. The second two methods make use of the WordNet database and move beyond exact word matching. They are outlined in section 4 and are based on the work of Mihalcea, Corely, & Strappavara and Fernando and Stevenson, respectively (Mihalcea et al. 2006; Fernando et al. 2008).

2 Related Work

This project was inspired by the research outlined in the paper *Topic Modeling of Research Fields: An Interdisciplinary Perspective* by Michael Paul and Roxana Girju. What Paul and Girju investigate is the strength of relations between fields of research. Broadly, they propose that if academics in certain fields of research are writing papers with strong enough similarities those fields of research could be combined into a third field of research. Furthermore, they suggest that systems in one field may be translated to another field if the the two fields are strongly connected. In the closing paragraphs of the paper Paul and Girju suggest their topic modeling systems could be used for an automatic note-taking system.

As mentioned in the previous section the work in this paper is also based largely on the work outlined in *A Semantic Similarity Approach to Paraphrase Detection* by Samuel Fernando and Mark Stevenson and in *Corpus-based and Knowledge-based Measures of Text Semantic Similarity* by Rada Mihalcea, Courtney Corley and Carlo Strapparava. Both papers presented functions which measure the similarity of texts. This project uses adaptations of those two functions to improve text integration.

3 Data

The data used for this project came from a collection of Wikipedia articles. The final number of documents in the project dataset was 16,724 each with a minimum word count of 600. While this minimum size cut down the total number of documents the minimum also gave us some certainty that each document would have multiple paragraphs and would be content rich. These two characteristics were necessary for splitting documents and ensuring the partial success of our baseline function. For further development actual notes from students or professional settings would be

necessary.

4 Approach

4.1 Paragraph Combination Probability

The baseline approach for the project is a simple measure of word overlap. Using select parts of speech this approach determines a similarity score between three paragraphs. For now the baseline looks at only nouns and years. Nouns were chosen because they are the subjects of sentences and collectively representative of the subject of a paragraph. Years were included under the following assumption: if two texts were to include the same year they are more related.

Given the list of paragraphs the PCP function uses the tokenizers and part of speech taggers found in the *nltk* Python module to tokenize and part of speech tag each paragraph. A list C of all pair-wise combinations of paragraphs is constructed. The total relatedness of these pairs is the $intersectScore$, defined as the following where the intersection of two paragraphs is the number of nouns and years they have in common,

$$intersectScore = \sum_{(p,p^*)\in C} |p \cap p^*|$$

However, this score does not take into account the total number of nouns and years in each paragraph. Similar to an inverse document frequency weight (idf-weight) we included a count of the total number of unique tokens times 3. This weight was included to boost the scores for smaller paragraphs with fewer nouns and years. Therefore, *pcp* similarity is defined as,

$$pcp(p_1, p_2, p_3) = \frac{intersectScore}{3 \times |unique(p_1 \cap p_2 \cap p_3)|}$$

4.2 Maximum Word Similarity (MCS)

This approach is adapted from the greedy method based on word-to-word similarities proposed by Mihalcea, Corely, & Strappavara (Mihalcea et al. 2006). Given three paragraphs the function returns a score in the range [0,1] of how similar the paragraphs are. Similar to the baseline MCS does this by computing the similarity of all pair-wise combinations of the given paragraphs. Due to the limitations of the *nltk* Python module paragraphs were tagged using the Penn Tree Bank tags. Then, to accomodate the *wup* similarity function, these tags were translated into WordNet tags. The resulting set of nouns, verbs, adverbs, and adjectives was used to compute word to text similairities. For two texts MCS defines their similarity as,

$$sim(T_1, T_2) = \frac{1}{2}\Big(mean\big(\sum_{w\in\{T_1\}} maxSim(w, T_2)\big) + mean\big(\sum_{w\in\{T_2\}} maxSim(w, T_1)\big)\Big)$$

where $maxSim$ computes the maximum word-to-word similarity of word w and all words in text T. $maxSim$ makes use of the *wup* metric (Wu and Palmer, 1994) when computing word-to-word similairties.

For the three given paragraphs the MCS similarity score is the mean score for all possible pair-wise paragraph combinations.

4.3 Complete Word Similarity (FST)

This approach uses the work of Fernando and Stevenson (Fernando et al. 2008) whose work was in part to improve upon the word-to-word approach of Michalcea et al. (Michalcea et al. 2006). Fernando and Stevenson suggested using all word-to-word similarities when computing text similarity rather than the $maxSim$ funciton outlined above. As with the prevous to approaches FST computes a similarity score for three paragraphs. Again, FST returns a mean score based on the scores of all the pair-wise paragraph combinations for the three paragraphs.

FST computes the similarity of two texts T_1 and T_2 as follows. For all texts construct a single list L of the combined nouns, verbs, adjectives, and adverbs (all parts of speech within WordNet). Then an L by L matrix M is constructed (in the Python implementation this is handled by lists of lists). For each cell (i, j) of M for word w_i and word w_j a *wup* metric is calculated. Because the axes of the matrix are symmetric the cells along the diagonal (starting at 0,0) will have a score of 1 i.e. $w_i = w_j$. If there is no similarity between w_j and w_i a 0 is entered into the cell. Once the matrix is filled in a selective mean is computed i.e. the mean of all cells with value greater than 0 in M. This selective mean is returned as the similarity score between the three paragraphs.

5 Experiments

To begin, select information was pulled from each file within our dataset. This information included

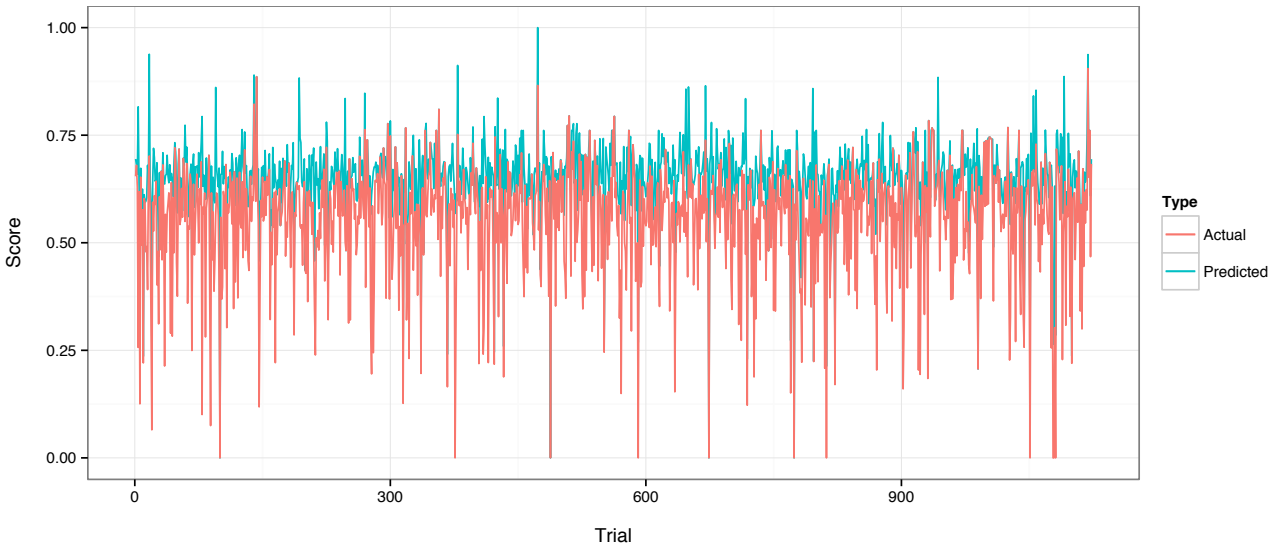

Figure 1: MCS Score Comparison

the file name, total word count, the number of paragraphs, the average paragraph length, the number of nouns, verbs, adjectives, and adverbs, and the sum of the previous four numbers. Using this information were able to get a better understanding of syntactic relationships and how they were affecting our similarity functions.

From the set of Wikipedia articles for each file F, F was split into paragraphs P. Of the paragraphs in $P = \{p_1, p_2, \ldots, p_n\}$ one was selected at random from $\{p_2, \ldots, p_{n-1}\}$ and removed. This limitation was to ensure any randomly chosen paragaph p^* would, in its original position, have two neighboring paragraphs. From the remaining paragraphs in P a list of consectutive paragraph pairs was constructed. Using this list of pairs a new list of triples each including p^* was created,

$$T^* = \{(p_1, p_2, p^*), (p_2, p_3, p^*), \ldots, (p_{n-1}, p_n, p^*)\}$$

So, for every triple $t \in T^*$ for each file F, t was input into PCP, MCS, and FST. Each function calculated a similarity score and the max score for all t was calculated. Based on this score a position at which to integrate p^* was calculated and checked against the true position of p^*. Additionally, the similarity score of p^* with its original neighboring paragraphs was calculated. All results for each function were written to a file and analyzed using R.

6 Results

Table 1 shows the results of each of the three similarity methods plus a fourth random method where the position was chosen at random. *Percentage* is the percentage of total files for which the randomly chosen paragraph was properly integrated back into the text.

Approach	Percentage
random	5.34
PCP	8.55
MCS	12.11
FST	7.57

Table 1: Comparison of Approach Success

These percentages are small, however the next table offers us insight as to why our results were so poor. Along with the score *Predicted* for the predicted position we also kept track of the score *Actual* for the original position of each removed paragraph. The following table demonstrates an underlying flaw with each of the functions,

Approach	Mean Predicted	Mean Actual
PCP	0.858	0.553
MCS	0.650	0.551
FST	0.884	0.810

Table 2: Predicted Scores vs Actual Scores

From the Table 1 data and visualizations like Figure 1 we discovered that the functions were consitently returning predicted scores higher than the actual scores. This indicated to us that our functions were either 1) too smart or 2) not smart enough. Based on our understanding of the functions we believe they are not smart enough. These

functions have no understanding of paragraph or article structure and therefore make no adjustment for the types of paragraphs they are considering. So, although our results are not what we wanted or expected we feel confident about the next steps to take in order to improve our results.

7 Conclusion and Future Work

This paper presented an idea for assisting students and the beginnings of that idea, a method of text integration. Our method involved both the Penn Tree Bank and WordNet and was, in part, based on previous work done in this area of natural language processing. And while our results were poor we understand the reasons why.

To improve our methods we would implement a machine learning paragraph classifier. This classifier would classify paragraphs as *intro*, *body*, or *conclusion* in order for the similairy functions to adjust their scoring methods. For this we would need a dataset of labeled paragraphs. Furthermore, it would be necessary for us to research the common syntactical structures of intro and conclusion paragraphs to better understand how our scoring methods need to be adjusted.

References

Paul Clough, Robert Gaizauskas, Scott Piao, and Yorick Wilks. 2002. METER: MEasuring TExt Reuse. In *Proceedings of the 40th Anniversary Meeting for the Assocation for Computational Linguistics (ACL-02)*, pages 152-159, Pennsylvania, PA

Samuel Fernando and Mark Stevenson. 2008. A semantic similarity appraoch to paraphrase detection, In *Computational Linguistics UK (CLUK 2008) 11th Annual Research Colloquium*

Rada Mihalcea, Courtney Corley, and Carlo Strapparava. 2006. Corpus-based and Knowledge-based Measures of Text Semantic Similarity. In *Proceedings of the American Association for Artificial Intelligence (AAAI 2006)*, Boston, July.

Long Qiu, Min-Yen Kan, and Tat-Seng Chua. 2006. Paraphrase recognition via dissimilarity significance classification. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 18-26, Sydney, Australia, July. Association for Computational Linguistics.

Yitao Zhang and Jon Patrick. 2005. Paraphrase identification by text canonicalization. In *Proceedings of the Australasian Language Technology Workshop 2005*, pages 160-166, Sydney, Australia, December.