# Knowledge Technology COMP90049 Report
# Project 1 partB
Lindong Li lindongl 655251

## 1. Introduction
The problem in this project is to identify film titles for each review by utilising approximate matching methods and then assess the quality of films — whether they are good or not. Datasets include around 7000 film titles, a set of reviews from IMDb (Maas, Daly, Pham, Huang, Ng, & Potts, 2011).

## 2. Overview of matching method
There are two approximate matching methods used in this project. One is the 2-gram and the other is Smith-Waterman algorithm.

### 2.1. 2-gram
#### 2.1.1. Changes
First change is that before splitting strings in to substrings，one percentage symbol will be added to the start and end of the strings. For example, for string “abc”, the substrings of it will be %a, ab, bc, c%. After applying this change, even though the length of string is less than 2, the algorithm still could work.

The second change is that in oerder to count the number of shared substrings, the algorithm would firstly take each substring of shorter string to check if there is a same substring in the longer string. Once it is detected, adding one to the number of shared substrings and start from the next substring of shorter string. This change could deal with situations where there are many same substrings in each strings. For example, for “aaaab”, substrings are %a, aa, aa, aa, ab b%. For “aaab”, substrings are %a, aa, aa, ab, b%. Counting the number of shared substrings would start from the shorter one “aaab”, thus there are 5 shared substrings.

#### 2.1.2. Multiple words titles process
For titles comprising multiple words, algorithm firstly splits the review into words by whitespace. Then according to the number of words the title contains, divide review into several strings which contain the same number of words as title. For example, for the title “12 Angry Men” and for a review “Evocative, funny look at four women stuck in temporary employment,”, the review will be divided to “Evocative, funny look”, “funny look at” …. Each string will be compared with the title. The highest score among them would be used as the score of this title for this review.

#### 2.1.3. Threshold
In this project, the threshold is set as 4. This threshold only allows one mismatch, including three situations as shown in Figure1:
1) Two strings match exactly
2) There is only one different character between two strings
3) The lengths of two strings are different by one and two strings are exactly same after removing that redundant character

#### 2.1.4. Which titles are returned?
For each review, all titles would be checked and each title would have a score for this review. Titles with the lowest score would be returned.

### 2.2. Smith-Waterman
#### 2.2.1. Score setting
The theory of the algorithm is the same as what we have learnt from the tutorials. The score for a match is 1, and for an insertion, deletion or replacement is -1.

#### 2.2.2. Multiple words titles process
For titles comprise multiple words, the algorithm treats the review as a string and compare it with each title.

#### 2.2.3. Threshold
For each title, threshold is the length of it minus two. This threshold only allows one insertion, one deletion or one replacement.

#### 2.2.4. Which titles are returned?
For each review, all titles would be checked and each title would have a score for this review. Titles with the highest score would be returned.

## 3. Effectiveness evaluation
In order to evaluate effectiveness of these two algorithms, I randomly select two sets of reviews. Each set contains 100 reviews. I have viewed these reviews and find titles by hand. Figure2 and Figure3 show results of these two algorithms. Figure 4 and Figure5 list the formal evaluation. Diagram 1, 2, 3 are also used to display evaluation information.

| Source | Target | 2-gram |
|---|---|---|
| abcd | abcd | 0 |
| abcd | abc | 3 |
| abcd | abed | 4 |

*Figure 1 Examples for three situations for threshold setting of 2-gram*

| Reviews | Number of titles identifies by hand | Number of titles returned by Smith-waterman | Number of matching titles |
|---|---|---|---|
| randomReviews1 | 41 | 120 | 20 |
| randomReviews2 | 48 | 135 | 16 |

*Figure 2 Results of Smith-Waterman on two sets of reviews*

| Reviews | Number of titles identifies by hand | Number of titles returned by 2-gram | Number of matching titles |
|---|---|---|---|
| randomReviews1 | 41 | 375 | 15 |
| randomReviews2 | 48 | 343 | 13 |

*Figure 3 Results of 2-gram on two sets of reviews*

| | Precision | Accuracy | Coverage |
|---|---|---|---|
| randomReviews1 | 16.67% | 20% | 100% |
| randomReviews2 | 11.85% | 16% | 100% |
| Average | 14.26% | 18% | 100% |

*Figure 4 Evaluation of Smith-Waterman*

| Reviews | Precision | Accuracy | Coverage |
|---|---|---|---|
| randomReviews1 | 4% | 15% | 100% |
| randomReviews2 | 3.79% | 13% | 100% |
| Average | 3.90% | 14% | 100% |

*Figure 5 Evaluation of 2-gram*

### 3.1. Precision Analysis

For the precision, we could see that Smith-Waterman outperforms 2-gram a lot, even though the numbers of matching titles are mostly same. Obviously, this is caused by too many titles returned by 2-gram. This is because for 2-gram algorithm, for one review, titles with lowest score would be returned. If this review is pretty long, exact matches are pretty easy to happen because of some titles composing of common words. These titles would be definitely returned with a score of 0 due to which is the lowest. However, for the Smith-Waterman, the title with highest score would be returned. When exact matches happen because of common words, most of them would be discarded, since scores of them are not the highest. Thus the number of titles returned by 2-gram is much larger than Smith-Waterman. For example, in 16313.txt, 13 titles will be returned by 2-gram and only 1 title will be returned by Smith-Waterman as shown in Figure 6.

| Algorithm | Titles returned |
|---|---|
| 2-gram | Look;Harvey;Everything;The Power;Schindler's List;Praise;Lies;Feed;Mel;Knowing;The One;The Grey Zone;Believe |
| Smith-Waterman | Schindler's List |

*Figure 6 Titles returned by 2-gram and Smith-Waterman*

### 3.2. Accuracy Analysis

For the accuracy, we could see that they are both relatively low. For Smith-Waterman, one reason is that even though there is correct title in the review, other titles containing common words may be given higher scores than it. For example, in the 13344.txt, the correct title Roswell is in the review, however, another film title - Classified X attains higher score, so the correct title could not be returned. For 2-gram, one reason is that if the file title in the review is surrounded by quotations, 2-gram with a threshold of 4 would discard it because the score would be 6. For example, in the 14434.txt, even though the real title - The Killing of a Chinese Bookie has appeared in the review, 2-gram could not return it because it is surrounded by quotations.


*Diagram 1 Precision comparison between Smith-Waterman and 2-gram*

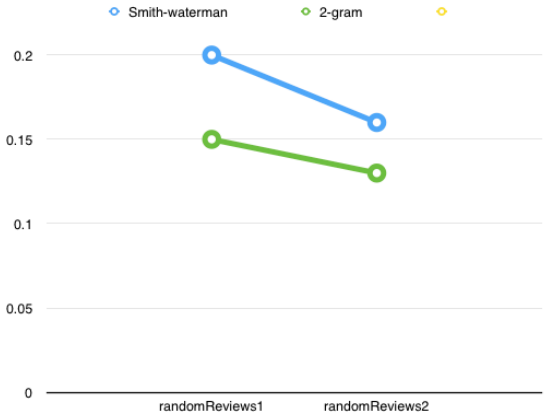


*Diagram 2 Accuracy comparison between Smith-Waterman and 2-gram*

### 3.3. Coverage Analysis

For the coverage, we could see that they are both 100%, even though the threshold has been set for each algorithm to only allow one mismatch. The reason for this I guess is that there are mostly 7000 titles and some reviews contain common words. Because for each review, all titles would be checked, no matter which algorithm we use, it is pretty possible to return a title consisting of common words which appear in the review. Besides, some reviews also contain film titles, but these reviews really talk about some other film, which may also improve the coverage. For example, in 28794.txt (in the appendix), it mentions Monsieur Verdoux, but it really talks about Unknown Chaplin.

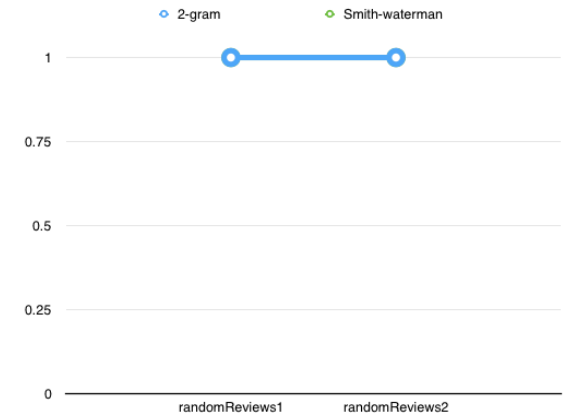


*Diagram 3 Coverage comparison between Smith-Waterman and 2-gram*

## 4. Access goodness of films

### 4.1. preparatory work

**1)** Create a dictionary which contains a lot of nouns or noun groups related to movies. These words would be considered as movie feature words.

2) Use adjectives as opinion words and identify the semantic orientation of each opinion word. Method proposed in (Hu & Liu, 2004) is used to identify the semantic orientation for each opinion word.

### 4.2. Process

There are fours steps to decide whether a film is good:

1) Split review into sentences and then use NLProcessor (NLProcessor–TextAnalysisToolkit, 2000) to parse each sentence to get part-of-speech tag for each word(identify the word is noun, adjective etc.).
2) Go through each sentence to check if it contains one or more movie feature words and one or more opinion words by using approximate matching methods. If it does, then identify its semantic orientation by comparisons the number of positive opinion words and negative opinion words. If positive opinion words are more, the sentence is considered as positive and vice versa. If the numbers are same, the sentence is considered as neutral. Besides, for a sentence followed by a but clause (starting from but, however etc.), firstly check if there are opinion words in but clause, if there are, using them to identify the semantic orientation of the whole sentence. If not, the opposite orientation of main sentence is used as orientation of the whole sentence.
3) If the review contains more positive sentences, then this is a positive indicator for the film and vice versa. If the numbers of positive sentences and negative sentences are same, then this is a neutral indicator for this film.
4) If the review collection contains more positive reviews for a film title than negative reviews, then this film is "good" and vice versa. If the numbers of positive reviews and negative reviews are same, then the film is just so so.

### 4.3. Example

In the 24774.txt , for the first sentence "I am a huge 'Dallas'-fan, but this movie is just not good enough", in the first subsentence, although there is an opinion word "huge", there is no movie feature word. In the second subsentence, "movie" is a movie feature word and "not good" is an opinion word with negative semantic orientation. Thus the semantic of whole sentence is negative. Another example in the 16582.txt, "Surprisingly Rowlands wasn't that good at first--way too muted but she eventually got better", the semantic orientation is decided by the but clause, which is positive according to the "better".

### 4.4. Strategy evaluation

This is a better strategy compared with strategy which simply decide whether a review is positive according to the number of positive and negative words, because it introduces two useful factors (movie feature words and opinion words) to solve this problem and considers the semantic orientation based on sentences. However, there are also some problems. For example, creating a relatively complete dictionary related to movies is hard. Some sentences in one review may discuss about other films, but this strategy could not tell it.

## 5. Conclusion

In conclusion, for the problem of searching for approximate matches to film titles in reviews, the performance of 2-gram and Smith-waterman are both relatively poor. Smith-waterman outperforms 2-gram slightly reflected by precision and accuracy. Automatically determining which films are "good" is a much harder knowledge problem because it involves nature language process.

## Reference

Hu, M., & Liu, B. (2004). Mining and Summarizing Customer Reviews. New York, NY, USA.

Maas, A. L., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., & Potts, C. (2011). Learning Word Vectors for Sentiment Analysis. The 49th Annual Meeting of the Association for Computational Linguistics (ACL 2011) , 142-150 .

NLProcessor–TextAnalysisToolkit. (2000). Retrieved from http://www.infogistics.com/textanalysis.html