THE SPIN DOCTOR: YOU GIVE US A DOCUMENT, WE'LL REVERSE ITS SENTIMENT

GEORGE FEI, INDIRA PURI, AND CATHY WONG

Note to Nishith

This is a draft of our final paper. We know it goes over four pages, and so it's fine if you just wish to read the first four for milestone. We would very much appreciate feedback on whether this draft is sufficient for the final paper. Our group has put aside a lot of time for this project, and we are now hoping to spend time on other classes.

1. Introduction

It is March 2011, and the Arab Spring is spreading across the Middle East. You are Anna Wintour. Months ago, you commissioned a writer to do an expose on Syria's first lady. This expose was overwhelmingly positive, and its editor has tentatively named the article "A Rose in the Desert." But reports of human rights abuses in Syria have continued to mount, and *Vogue* might receive backlash for running the article. Do you publish the positive expose on Bashar Al-Assad and his wife, or not?

In reality, *Vogue* decided to go ahead with the publication of the article – much to their later regret. The magazine was pummeled for a positive portrayal of a family that increasing evidence showed were brutal, dictatorial rulers. *Vogue* immediately removed the expose, and today, only one copy of the original article is available online (unsurprisingly, the website hosting the article is Gawker).

Vogue could have seriously benefited from software that would take in an article, and, while maintaining its structure and the author's voice, re-written the article to be negative. Creating such software is the goal of our project. For example, given, say a movie review:

"Bambi was a horrible film. Hated it. Shame!"

We would like our software to output the movie review:

"Bambi was a wonderful film. Loved it. Wow!"

The transformation preserves the structure, punctuation, and formatting of the original review – it just makes the review positive. To create this code, we use sentiment analysis methods to classify the sentiment of our new review, thesauruses that have been trained on a large corpus (for example, the Python Natural Language Toolkit (i.e. Stanford's NLTK)) as well as feature extraction methods to determine, for example, how similar the revised review is in structure to the original. Because we write our own evaluation method, we also need to write and implement our own evaluation function.

To tackle the problem, we use the same corpus of movie reviews used for our second homework, "sentiment." We intend to take each review and reverse its sentiment. Our baseline is simply adding "not" in front of every adjective. Our oracle is a human re-writing the review to preserve the structure, but change sentiment. Our evaluation metric is (did sentiment change) * (closeness in structure) * (reward for using words similar to actual bad reviews). Intuitively, we want sentiment to change; we want closeness in structure; and we want the changed review to look like something a human could plausibly write. To understand the last goal, suppose we have a movie review

"The Matrix was mind-numbing in its awesomeness."

A re-written movie review of the form:

"The Matrix was mind-enhancing in its dullness."

may count as a change in sentiment, and certainly maintains structure, but it doesn't look like anything a human would ever write. Therefore if, as part of our score, we include some evaluation of how similar the words used in our changed review are to actual negative reviews, we discourage outputs like the above.

We defer a precise definition of the evaluation metric to the methodology section. Our baseline is simply adding "not" in front of every adjective and adverb. Our oracle is a human re-writing 20 reviews to switch their sentiment. The baseline receives a relatively low mean evaluation score of 0.343 (median score: 0.318 and standard deviation: 0.257), and the oracle receives evaluation score 0.544 (median score: 0.587 and standard deviation: 0.310). Because there is a large gap between these, finding a clever way to re-write a review to reverse sentiment is non-trivial.

We try three different methods for reversing sentiment, each more sophisticated than the next. First, we try intelligent greedy adjective replacement. This algorithm targets all adjectives and adverbs in the document with sentiment opposite to our target sentiment, and replaces them with an antonym. Second, we try rule-based replacement, in which a human writes part-of-speech based replacement rules that the computer follows. Third, we a try similarity-based replacement algorithm inspired by machine translation techniques for translation in the absence of labeled parallel corpora, in which we replace an the original adjective/adverb with a word that frequently appears in documents of the target sentiment, and whose surrounding words are similar to those surrounding the original adjective/adverb. Evaluated against a test set of IMDB movie review examples, intelligent greedy adjective replacement achieved a mean evaluation score of 0.592 (with a median score of 0.686, and standard deviation of 0.31), rule-based replacement achieved a mean evaluation score of 0.590 (with a median score of 0.685, and standard deviation of 0.31), and similarity-based replacement achieved a mean evaluation score of 0.376 (with a median score of 0.329, and standard deviation of 0.3). Notably, the greedy and rule-based algorithms actually outperform oracle results when evaluated on the test set using our context-specific evaluation function.

The contributions of this paper are therefore twofold: first, we offer three intelligent algorithms for sentiment reversal which score close to human re-writing. These algorithms successfully maintain structure while reversing sentiment, but are highly sensitive to the tools used to execute them, in particular the part-of-speech tagger used. Second, we construct a new scoring metric that can be used for sentiment reversal problems.

(George Fei) Department of Chemistry, Stanford Univeristy.
(Indira Puri) Department of Mathematics, Stanford University. Department of Economics, Stanford University.
(Cathy Wong) Department of Computer Science, Stanford University.
E-mail addresses: georgeqf@stanford.edu, puri@stanford.edu, catwong@stanford.edu.

1.1. **Related Literature.** While there is a very large corpus of work on improving sentiment analysis, to our knowledge this is the first paper on reversing the sentiment of a document.

Prior papers have looked at many facets of sentiment analysis, including accuracy [10] [1] [12] [4], domain adaption [2] [9] [13], and how sentiment analysis on various corpii relate to real-world events [3] [11].

Although we focus on sentiment analysis, our work is most closely related to the prior literature on translation. In particular, we may think of our problem as one of translating into a language whose every adjective and adverb has opposite sentiment from normal English. We may therefore utilize statistical machine translation techniques [5], which have been extensively researched. The basic approach behind statistical machine translation techniques is finding a semantic "chunk" of the phrase to be translated, and using Bayes rule with maximum likelihood to find its match in the target language. While the earliest implementation used single-word chunks, later work extended this to bigrams [14], and even more complex semantic structure [8] [6]. Our problem is easier than translation problems in that many of the words between English and sentiment-reversed English are the same; for example, the phrase with opposite sentiment to "the chair" is simply "the chair", because the phrase has neutral sentiment. This means that whereas a traditional translation problem would involve translating every word in a sentence, ours need not do so. We therefore eschew more complicated techniques in favor of easy-to-understand bigram-comparison and single word- reversal.

However, though we phrase our problem like as one of translation, we cannot use traditional automated methods of scoring translation algorithms, such as BLEU [15], for a variety of reasons. Most notably, metrics like BLEU use a body of pre-existing translated texts, but because sentiment reversal is not a traditional translation, we have no such corpus of human referenced sentiment reversals. Another problem is that metrics often compare every single n-gram between the two texts, and whereas that is a good strategy for texts of disparate languages and vocabularies, our texts will be in the same language and should thus have high word-for-word similarity between two "translated texts." Finally, BLEU suffers from several issues itself when it comes to machine translation evaluation. The most notable problem is BLEU's inability to capture syntactic features (true features of readability) of translated texts because most of its evaluation power stems from its n-gram model [16]. Our evaluation metric, developed independently from BLEU, improves on BLEU for this context by adding measures of readability and structural preservation.

2. Methodology

2.1. **Data.** Our corpus is the Large Movie Dataset, obtained from `http://ai.stanford.edu/~amaas/data/sentiment/` and originally developed by [7]. This dataset contains 50,000 movie reviews. It pre-classifies each movie review on a scale of 1 to 10, where 1 is very negative and 10 is very positive. The dataset also provides pre-processed train and test, so that the movies in each are disjoint. The provided train set is 25,000 reviews, and the provided test set is 25,000 reviews. We use 20% of the pre-processed test set for our dev set. This means that our train set has 25,000 reviews, our dev set 5,000, and our test set 20,000.

2.2. **Tools.** Our evaluation metric incorporates a number of off-the-shelf tools, chosen largely for robustness and ease of use while maintaining relative efficiency in run time over a large training and test set. To gauge shift in sentiment, we use the NLTK Vader sentiment classifier to score the transformed documents: while more sophisticated sentiment scorers, such as those backed by recursive neural networks, do exist and could easily be incorporated in future work, the Vader sentiment classifier was chosen for its relative robustness in handling corner cases such as negation and specific sentiment-laden idioms while still maintaining an acceptable runtime performance. Notably, the Vader sentiment classifier outputs normalized sentiment scores for a body of text that range from -1 (for more negative sentiment) to 1 (for more positive sentiment); as the labeled IMDB movie reviews and thus the evaluation metric described below assume a sentiment score range from 1 (for more negative) to 10 (for more positive), the Vader scores were mapped onto a 1 to 10 range before use. We also use the NLTK default tokenizers and part of speech taggers to analyze document similarities - again, chosen over alternatives for their relative robustness and ease of use. Finally, both the rule-based and intelligent greedy sentiment reversal algorithms described below rely on the antonym nets offered in the NLTK distribution of WordNet for replacement adjectives.

2.3. **Evaluation Metric.** Our evaluation score is

$$\begin{aligned}\text{Evaluation Score (revised document)} = &\text{ Sentiment Change Score (revised document, original document)}\\ &* \text{Closeness Score (revised document, original document)}\\ &* \text{Uses Typical Language Score (revised document, original document).}\end{aligned}$$

The components of each are listed below.

2.3.1. *Sentiment Change Score.*

$$\text{Sentiment Score Change} = \frac{|\text{Sentiment (revised document) - Sentiment (original document)}|}{\max(\text{Sentiment (original document)} - 1, 10 - \text{Sentiment (original document)})}$$

Intuitively, because our sentiment scores range from 1 to 10, we would like to take any positive review and re-write it to be as negative as possible; and to take any negative review and re-write it to be as positive as possible.

Note that Sentiment Change Score may only take on values between 0 and 1.

2.3.2. *Closeness Score.* We first need to define what is meant by "structure" of a document. When we say that two articles are similar in format and author's voice, we mean that:

(1) The documents contain the same proper nouns.
(2) The documents contain the same number of sentences.
(3) The documents contain the same number of each part of speech (ex. same number of nouns, same number of adjectives, same number of verbs, same number of pronouns).

Therefore a "closeness" score between two documents A and B is

$$\begin{aligned}\text{Closeness}(A, B) = &\text{Proper Noun Score} * \text{Number of Sentences Score}\\ &* \text{Part of Speech Score}\end{aligned}$$

where

(1) Proper noun score $= \frac{|\{\text{proper noun} \mid \text{proper noun} \in A \text{ and proper noun} \in B\}|}{\max(\text{number proper nouns in A, number proper nouns in B})}$.

(2) Number of sentences score $= \min\left(\frac{\text{number of sentences in A}}{\text{number of sentences in B}}, \frac{\text{number of sentences in B}}{\text{number of sentences in A}}\right)$.
(3) Part of speech score is

$$\frac{\sum_{\{x_i|\ x_i \text{ is a part of speech ex. noun, adjective, verb,...}\}} |\{b|b \text{ is an } x_i \text{ and } b \in A, B\}|}{\max(\text{Number of words in } A, \text{Number of words in } B)}$$

Note that this means the Closeness Score will always be between 0 and 1.

2.3.3. *Typical Language Score.* In our training corpus, reviews are have sentiment scores 1-10, with 1 denoting very negative, and 10 denoting very positive. If the original review has sentiment 6-10, then define "Relevant Sentiments" to be the range 1-5. If our original review has sentiment 1-5, then define "Relevant Sentiment" to be the range 6-10. Intuitively, because we wish to rewrite the original review to have the opposite sentiment, the relevant sentiments will be those sentiments opposite to the sentiment of the original review.

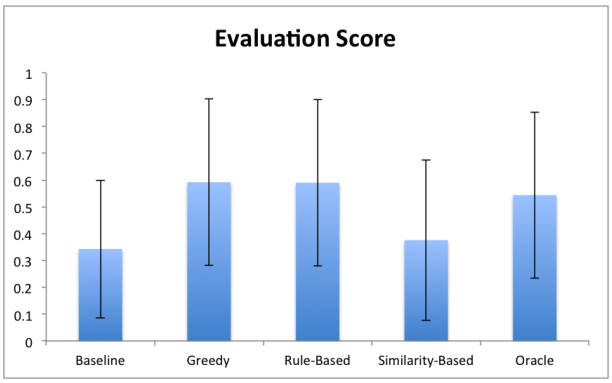
For each training document t_j, let Bigrams(t_j) denote the set of all bigrams in document t_j. Denote by "Real Bigrams" the set $\cup_{\{t_j|t_j \text{ has a sentiment} \in \text{Relevant Sentiments}\}}$Bigrams($t_j$). For example, if our original document had positive sentiment (was rated 6-10), Real Bigrams would be the set of all bigrams in negative sentiment (1-5 rated) training set reviews; and if our original document had negative sentiment, Real Bigrams would be the set of all bigrams in positive sentiment training set reviews. Then

$$\text{Typical Language Score} = \frac{\sum_{\text{All bigrams } b_i \text{ in the revised document}} \mathbb{1}\{b_i \in \text{Real Bigrams or } b_i \in \text{Original document}\}}{\text{Number of bigrams } b_i \text{ in the revised document}}.$$

Intuitively, each bigram should be something a real person has written, either in the original review, or in our known set of reviews whose sentiment we are trying to match.

Note that the Typical Language Score will always be between 0 and 1.

3. Results



3.1. **Baseline.** Our baseline algorithm simply transforms the document by adding the word "not" in front of any adjectives and adverbs identified using the NLTK tagger. This achieves an abysmal mean score of 0.343, but a median score of 0.318. An example can be found in Appendix A.

3.2. **Oracle.** Our oracle measures human performance on this task, using 20 reviews that were handwritten by a team member in an attempt to preserve document similarity while reversing review sentiment. These 20 reviews achieve a mean evaluation score of 0.519 and median of 0.742. An example can be found in Appendix A.

3.3. **Intelligent Reversal Algorithms.** Each of these involves replacing words with antonyms. For the first two algorithms, we choose an antonym for a given word as follows: we input our word into the NLTK antonyms tool, and pick uniformly at random an antonym with the same part of speech.
In deciding which words to replace with antonyms, we noted that if a word already had sentiment aligned with the target sentiment, there is no need to replace the word. For example, in switching the sentiment of the review

This film was acceptable, but exhausting.

if we switched the sentiment of all adjectives, we would obtain

This film was horrible, but invigorating.

On the other hand, if we notice that exhausting already has negative sentiment, we need only switch "acceptable" to create a strongly negative review:

This film was horrible and exhausting.

Because our goal is to create a strongly positive or negative review, for the first two algorithms below, we only replace those words whose sentiment is opposite to the target sentiment. For the third algorithm, we deviate from this convention and give the rationale for doing so in that subsection. We will specify which parts of speech we are replacing in each algorithm.
For Greedy and Rule-Based replacement below, once we have a word, we choose an antonym as follows:

(1) Look up the word in the NLTK antonym tool.
(2) If there is an antonym with the same part of speech as the word, replace the world with that antonym. If there are multiple such antonyms, pick any of them as a replacement for the word.
(3) If there is no antonym with the same part of speech as the word, simply insert 'not' in front of the word.

3.3.1. *Greedy Replacement.* Given an original document with (negative, positive) sentiment, we replace all those adverbs and adjectives with (positive, negative) sentiment by their antonyms.

3.3.2. *Rule-Based Replacement.* For this algorithm, we manually write replacement rules that we have the computer implement on any given review.
The rules are as follows:
We notice that a phrase containing a negative adjective has negative sentiment unless:

(1) There is a "not" in front of the adjective.
(2) There is an adverb with positive sentiment in front of the adjective (ex. wonderfully absurd, engagingly stupid, etc.)

We further note that a negative conjunction (but, yet) is a qualifier on a previous. We want the changed review to have as strong a sentiment as possible, so it makes sense to get rid of qualifiers. Our rule-based changes for adjectives are therefore:

- Loop over all bigrams.
- Once we see an adjective,
  (1) Change the adjective to its antonym unless:
    (a) There is a "not" in front of the adjective. In this case, remove the "not."
    (b) There is an adverb with positive sentiment in front of the adjective. In this case, keep the adjective the same but find an antonym for the adverb.
    (c) There is a "but" or "yet" in front of the adjective. In this case, change the "but" or "yet" to an "and" and switch the adjective to its antonym.
  (2) Skip over the next bigram, since we have attended to the adjective the first time we see it.

We also notice that an adverb can contribute to negative sentiment. For example, consider:

The movie was well written but badly executed.

Both "well" and "badly" are adverbs. Which should we change and why? We can replicate the rules we used for adjectives. Therefore our final rule-based replacement algorithm is:

- Loop over all bigrams.
- Once we see an adjective OR adverb:
  (1) If at least one adjective/adverb in the bigram has sentiment opposite to the target sentiment:
    (a) Change the adjective/adverb to its antonym unless:
      (i) The bigram is (adverb, adjective). In this case,
        (A) If the adverb has sentiment opposite to target sentiment and the adjective has the same sentiment as the target (ex. "irritatingly good" and target sentiment is positive). In this case, keep the adjective the same but find an antonym for the adverb (ex. "wonderfully good").
        (B) If the adverb has sentiment neutral or in line with target sentiment and adjective has opposite sentiment to target (ex. "really bad" and target sentiment is positive), then keep the adverb the same but switch the adjective to an antonym.
        (C) If the adverb and adjective both have sentiment opposite to target sentiment (ex. "wonderfully awesome" and target sentiment is negative), switch both adverb and adjective to antonyms.
      (ii) There is a "not" in front of the adjective/adverb. In this case, remove the "not."
      (iii) There is a "but" or "yet" in front of the adjective. In this case, change the "but" or "yet" to an "and" and switch the adjective/adverb to its antonym.
    (b) If the bigram is (anything NOT an adjective or adverb, adjective/adverb), skip over the next bigram since we have dealt with the adjective/adverb.
    (c) If the current bigram is (anything, adverb), and the next bigram is (adverb, adjective) where adjective has the same sentiment as the target sentiment, skip over the next two bigrams because switching the adverb's sentiment already switches the sentiment of the (adverb, adjective) bigram.

3.3.3. *Similarity-Based Replacement.* Recall that in our training corpus, reviews are have sentiment scores 1-10, with 1 denoting very negative, and 10 denoting very positive. If the original review has sentiment 6-10, then define "Relevant Sentiments" to be the range 1-5. If our original review has sentiment 1-5, then define "Relevant Sentiment" to be the range 6-10. Intuitively, because we wish to rewrite the original review to have the opposite sentiment, the relevant sentiments will be those sentiments opposite to the sentiment of the original review.

For each training document t_j, let Words(t_j) denote the set of all words in document t_j. Denote by "Real Words" the set

$$\cup_{\{t_j | t_j \text{ has a sentiment} \in \text{Relevant Sentiments}\}} \text{Words}(t_j).$$

For example, if our original document had positive sentiment (was rated 6-10), Real Words would be the set of all bigrams in negative sentiment (1-5 rated) training set reviews; and if our original document had negative sentiment, Real Words would be the set of all bigrams in positive sentiment training set reviews.

We give each word in Real Words an associated score. If Relevant Sentiments is 6-10, the score for word X is $\text{Score}(X) = \sum_i$ Sentiment score of document containing ith occurrence of X. If Relevant Sentiments is 1-5, this score is $\text{Score}(X) = -\sum_i$ Sentiment score of document containing ith occurrence of X. For example, if Relevant Sentiments is 6-10, and "was good" appears once in a document with sentiment 6, twice in a document with sentiment 7, and once in another document with sentiment 7, the score of "was good" would be $6 + 7 + 7 + 7 = 27$. Intuitively, a word scores higher if it appears more frequently in documents with strong sentiment.

For a given adverb/adjective, consider its neighboring bigrams b_1 and b_2 and unigrams u_1 and u_2:

- Let W be the set of all words in Real Words with the same neighboring bigrams. Replace the adverb/adjective with the word in W with the highest absolute score.

- If no unigram with the same part of speech as the original word and the same neighboring bigrams exists, let W be the set of all words in Real Words with the same neighboring unigrams .Replace the adverb/adjective with the word in W with the highest absolute score.

- If no words with identical neighboring bigrams or neighboring unigrams exists, let W be the set of all words in Real Words with one identical neighboring bigram .Replace the adverb/adjective with the word in W with the highest absolute score.
- If none of the above finds a replacement, default to using the NLTK antonym tool in the method described above to find a suitable antonym.

4. Discussion

Our greedy and rule-based algorithms are similar in that both use the same antonym dictionary (WordNet), and in the rule-based replaces many of the same adjectives and adverbs greedy does. In running greedy and rule-based, there were two takeaways: first, that the quality of output is highly dependent on the quality of the antonym dictionary. For example, there were cases when greedy and rule-based would change "This was a solid movie" into "this was a gaseuous movie." In the previous sentence, the discrepency is between general antonyms for "solid" and antonyms for "solid" as the word is used in movie reviews; having a domain-specific antonyms dictionary would have been helpful.

The second takeaway is that the quality of the algorithm is highly dependent on the quality of the part of speech scorer. For example, for the simple movie review "Brilliant and moving," both greedy and rule-based failed to return a changed review. This is because "brilliant" and "moving" were erroneously tagged as nouns by our part of speech tagger.

On the other hand, both greedy and rule-based have high scores, very close to the oracle. What this tells us is that our greedy and rule-based algorithms do preserve the structure of the document, and do reverse sentiment. The last lingering issue is human readability, which can be improved by improving the quality of our tools.

In an effort to mitigate the problems with part-of-speech tagging and non-domain specific antonyms identified above, we created the similarity-based algorithm. This algorithm replaces an adjective or adverb with a word in our training corpus, if one exists, whose nearby words are the same. If there is no word in the training corpus with similar neighbors to the adjective under consideration, the similarity-based algorithm defaults to using the antonym dictionary. The hope, here, was that such a technique that drew its inspiration from the realm of machine translation would somehow be able to learn the oppositely sentimented word from to their surrounding contexts. The power of this algorithm should lie its sensitivity to the surrounding text of a given word and determine what would should substitute it in order to negate the sentiment. However, this sensitivity turned out to be one of its major flaws, and we observed that our algorithm performed noticeably worse than even our baseline. Because our algorithm does not actually have a proper understanding of antonyms and relies solely on the context of the translation source to identify and choose among candidate replacement words, the surrounding words of a word to be translated have a great deal of influence on the translation that the machine picks out. Our model of the context, however, was too naive, making our model too sensitive local features of the text. While the algorithm was able to generate readable translations in a local context, the translation ended up being wrong, often hilariously so. For example, for one review, it took the word "different" in "different relationships" and turned that to "sexual," giving us "sexual relationships." We can see that though it did learn that "sexual relationships" is indeed a common phrase, it wasn't sensitive enough to the context of the review as a whole to determine that such a replacement was inappropriate. True machine translation approaches to parallel corpora often draw on more sophisticated measures of contextual similarity, ranging from complex part of speech measures to even deep-learning and statistical based metrics of contextual similarity - and future work could incorporate these techniques to improve choice among candidate antonyms.

5. Conclusion

Our intelligent spin doctor algorithms, particularly greedy and rule-based, successfully preserve the structure of a document while reversing the sentiment. These algorithms are highly sensitive to the quality of the part-of-speech tagger and antonym dictionary being used.

The similarity-based replacement algorithm, while promising in theory, faced pitfalls both because the surrounding context chosen to determine similarity and infer potential antonyms proved to be insufficient - that is, neighboring bigrams and unigrams did not provide enough context to accurately choose true antonyms - and because attempts to use an automated part-of-speech tagger in order to constrain potential antonyms to those that matched the desired replacement part of speech were restricted by inaccuracies in the tagger itself; the tagger frequently failed to identify parts of speech correctly, making it inefficient and impossible to use this as a better heuristic in choosing potential among potential word-replacement candidates. Ultimately, this meant that the similarity based metric preserved neither the general structure of the essay, nor often provided a true reversal of sentiment through well-chosen antonyms.

Finally, our scoring metric provides a reasonable way to combine measurements of readability, sentiment, and structure.

Future work could include: constructing better part of speech taggers and constructing domain specific dictionaries, as well as applying sentiment reversal to domains other than movie reviews.

Appendix A. Examples

A.1. **Baseline Example.** As stated above, our baseline algorithm attempts to reverse sentiment simply by adding the word 'not' before every identified adjective and adverb in a POS-tagged review.

For example, consider the positive original review:

> If you had asked me how the movie was throughout the film, I would have told you it was great! However, I left the theatre feeling unsatisfied. After thinking a little about it, I believe the problem was the pace of the ending. I feel that the majority of the movie moved kind of slow, and then the ending developed very fast. So, I would say the ending left me disappointed.I thought that the characters were well developed. Costner and Kutcher both portrayed their roles very well. Yes! Ashton Kutcher can act! Also, the different relationships between the characters seemed very real. Furthermore,I thought that the different plot lines were well developed. Overall, it was a good movie and I would recommend seeing it.In conclusion: Good Characters, Great Plot, Poorly Written/Edited Ending. Still, Go See It!!!

Using our baseline algorithm, which relies on (an inherently imperfect) default part of speech tagger to identify adjectives and adverbs, we produce the following transformed document, which receives a score of 0.650 by our evaluation metric:

> If you had asked me how the movie was throughout the film , I would have told you it was not great ! not However , I left the theatre feeling not unsatisfied . After thinking a not little about it , I believe the problem was the pace of the ending . I feel that the majority of the movie moved kind of slow , and not then the ending not developed not very not fast . not So , I would say the ending left me disappointed.I thought that the characters were not well developed . Costner and Kutcher both portrayed their roles not very not well . Yes ! Ashton Kutcher can act ! not Also , the

> not different relationships between the characters seemed not very not real . Furthermore , I thought that the not different plot lines were not well developed . not Overall , it was a not good movie and I would recommend seeing not it. In conclusion : not Good Characters , Great Plot , Poorly Written/Edited Ending . not Still , Go See It ! ! !

A.2. **Oracle Example.** As our oracle algorithm, we asked a member of our team to manually rewrite 20 reviews, preserving the original document structure as much as possible in line with our defined evaluation metric while still attempting to reverse the sentiment of the document to a highly polarized, opposite sentiment result. This, it should be mentioned, turned out to be a surprisingly difficult task in many cases - there is enormous variation and complexity in how moviegoers express their (often deeply-felt) disappointment and delight, and many of these sentiments have not easy analogue when reversed.
Using the same original review above, we created the following oracle-rewritten review, which received a score of: 0.727

New review:

> If you had asked me how the movie was throughout the film, I would have told you it was terrible! I left the theatre feeling unsatisfied. After thinking a little about it, I believe the problem was the pace of the ending. I feel that the majority of the movie moved kind of slow, and then the ending developed very fast. So, I would say the ending left me disappointed.I thought that the characters were poorly developed. Costner and Kutcher both portrayed their roles horribly. No! Ashton Kutcher cannot act! Also, the different relationships between the characters seemed implausible. Furthermore,I thought that the different plot lines were poorly developed. Overall, it was a bad movie and I would not recommend seeing it.In conclusion: Bad Characters, Terrible Plot, Poorly Written/Edited Ending. Don't Go See It!!!

A.3. **Intelligent Greedy Algorithm Example.** As described above, our intelligent greedy algorithm attempts to replace adjectives with their antonyms, but attempts to do so greedily without excessively risking changing the structure of the review by replacing only adjectives with a non-neutral sentiment score, and then negating adjectives it could not locate antonyms for in the WordNet lexicon. This is evident in the example below, which again attempts to transform the same positive review above and receives an evaluation score of 0.762.

New review:

> If you had asked me how the movie was throughout the film , I would have told you it was not great ! However , I left the theatre feeling unsatisfied . After thinking a little about it , I believe the problem was the pace of the ending . I feel that the majority of the movie moved kind of slow , and then the ending developed very fast . So , I would say the ending left me disappointed.I thought that the characters were badly developed . Costner and Kutcher both portrayed their roles very badly . Yes ! Ashton Kutcher can act ! Also , the different relationships between the characters seemed very real . Furthermore , I thought that the different plot lines were badly developed . Overall , it was a bad movie and I would recommend seeing it.In conclusion : bad Characters , Great Plot , Poorly Written/Edited Ending . Still , Go See It ! ! !

A.4. **Rule-based Replacement Algorithm Example.** Much like the intelligent greedy algorithm, as described above, the rule-based replacement algorithm attempts to replace sentiment-laden adjectives with antonyms; however, rule-based replacement uses a series of predefined rules that draw from prior knowledge of syntactical and lexical sentence construction in order to determine when and how to reverse more complex lexical constructions, such as lists of adjectives, adverb-adjective pairings, and negations. In most cases, this yields a result that is similar, if not identical, to the greedy algorithm, but the algorithm does attempt to intelligently handle certain sentiment-specific constructions. When run on the same example above, the rule-based algorithm creates the following review, which receives an evaluation score of 0.817:

New review:

> If you had asked me how the movie was throughout the film , I would have told you it was not great ! However , I left the theatre feeling unsatisfied . After thinking a little about it , I believe the problem was the pace of the ending . I feel that the majority of the movie moved kind of slow , and then the ending developed very fast . So , I would say the ending left me disappointed.I thought that the characters were badly developed . Costner and Kutcher both portrayed their roles very badly . Yes ! Ashton Kutcher can act ! Also , the different relationships between the characters seemed very real . Furthermore , I thought that the different plot lines were badly developed . Overall , it was a bad movie and I would recommend seeing it.In conclusion : bad Characters , Great Plot , Poorly Written/Edited Ending . Still , Go See It ! !

A.5. **Similarity Based Algorithm.** As described above, the similarity based algorithm attempts to use measures of context - in this case, the relatively naive measures of surrounding unigram and bigram context - in order to find similar candidate algorithms for identified adjectives in the original review, we search over similar contexts within the training data of the desired sentiment.
The pitfalls of this approach, however, are exemplified by the review below, which incorrectly identifies potential antonyms based on words that do in fact appear frequently in the given contexts, but are not actually antonyms for the desired word; this transformed review receives an evaluation score of 0.236:

New review:

> If you had asked me how the movie was throughout the film , I would have told you it was terrible !) , I left the theatre feeling unsatisfied . After thinking a movie about it , I believe the problem was the pace of the ending . I feel that the majority of the movie moved kind of slow , and then the ending is too good . no , I would say the ending left me disappointed.I thought that the characters were never developed . Costner and Kutcher both portrayed their roles are good . Yes ! Ashton Kutcher can act !) , the sexual relationships between the characters seemed to good . Furthermore , I thought that the main plot lines were not developed . however , it was a home movie and I would

recommend seeing it.In conclusion : repellent Characters , Great Plot , Poorly Written/Edited Ending . however , Go See It ! ! !

References

[1] Albert Bifet and Eibe Frank, *Sentiment Knowledge Discovery in Twitter Streaming Data*, Lecture Notes in Computer Science, 2010, pp. 1–15.

[2] Neil O'Hare, Michael Davy, Adam Bermingham, Paul Ferguson, Paraic Sheridan, Cathal Gurrin, and Alan F. Smeaton, *Topic-dependent sentiment analysis of financial blogs*, TSA '09 Proceedings of the 1st international CIKM workshop on Topic-sentiment analysis for mass opinion (2009), 9–16.

[3] Yu Jiang, Weiyi Meng, and Clement Yu, *Topic Sentiment Change Analysis*, Machine Learning and Data Mining in Pattern Recognition. Lecture Notes in Computer Science **6871** (2011), 443–457.

[4] Alistair Kennedy and Diana Inkpen, *Sentiment Classification of Movie Reviews Using Contextual Valence Shifters*, Computational Intelligence **22** (2006).

[5] P. and Knight Koehn K., *Statistical machine translation* (2009). US Patent 7,624,005.

[6] Yang Liu, Qun Liu, and Shouxun Lin, *Tree-to-string Alignment Template for Statistical Machine Translation*, Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics, 2006, pp. 609–616, DOI 10.3115/1220175.1220252.

[7] Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts, *Learning Word Vectors for Sentiment Analysis*, Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, 2011, pp. 142–150.

[8] Daniel Marcu and William Wong, *A Phrase-based, Joint Probability Model for Statistical Machine Translation*, Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing - Volume 10, 2002, pp. 133–139, DOI 10.3115/1118693.1118711.

[9] Robert P. Schumaker, Yulei Zhang, Chen-Neng Huang, and Hsinchun Chen, *Evaluating sentiment in financial news articles*, Decision Support Systems **53** (2012), 458–464.

[10] Richard Socher, Alex Perelygin, Jean Y. Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts, *Recursive Deep Models for Semantic Compositionally Over a Sentiment Treebank*, Conference on Empirical Methods in Natural Language Processing (2013).

[11] Mike Thelwall, Kevan Buckley, and Georgios Paltoglou, *Sentiment in Twitter events*, Journal of the Association for Information Science and Technology **62** (2011), 406–418.

[12] Sida Wang and Christopher D. Manning, *Baselines and bigrams: simple, good sentiment and topic classification*, ACL 2012 Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers **2** (2012), 90–94.

[13] Rui Xia, Chengqing Zong, Xuelei Hu, and Erik Cambria, *Feature Ensemble Plus Sample Selection: Domain Adaption for Sentiment Classification*, IEEE Intelligent Systems **28** (2013), 10–18.

[14] Richard Zens, Franz Josef Och, and Hermann Ney, *Phrase-Based Statistical Machine Learning*, Advances in Artificial Intelligence. Lecture Notes in Computer Science **2479** (2002), 18–32.

[15] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu, *BLEU: a Method for Automatic Evaluation of Machine Translation*, Proceedings for the 40th Annual Meeting of the Association of for Computational Linguistics **2479** (2002), 311–318.

[16] Jesus Gimenez and Lluis Marquez, *Linguistic measures for automoatic machine translation evaluation*, Machine Translation **24** (2010), 209-240.