Preprocessing Methods

In this chapter, we present which methods we applied to pre-process the data and explain for each our reasoning behind it.

*# Explain main ideas of what might improve deduction*

# Negation Replacements

This method replaces some simple negations with their opposite meaning. Moreover, it takes some of the most common negations and makes one word out of them. The transformations we used are the following:

|  |
| --- |
| *not good / bad 🡪 bad / good* |
| *not sad / happy 🡪 happy / sad* |
| *will not 🡪 willnot* |
| *won’t 🡪 willnot* |
| *is not 🡪 isnt* |
| *can not 🡪 cannot* |

To be concise, we have omitted the examples in which the “*not”* is shortened to its “*n’t”* form.

**Reasoning:** We thought that strongly indicative words such as “happy” and “sad” might mislead our classifiers when in the context of a negation. This method is supposed to avoid that and make it easier to make use of the words “good” and “bad”.

Additionally, we thought that because getting rid of the “not” will change the *context window* and better the word associations because it will bring closer the words that are meaningfully connected.

The last four replacements were added after we read in [***Citation***] that they improved classification score. In fact, mapping several words of the same meaning onto one reduces our embeddings and theoretically improve accuracy.

**Flaws:** For one, this method is incomplete. It doesn’t consider past tense such as “wasn’t” nor does it consider other indicative words such as “beautiful / ugly” or “amazing / boring”. To address this issue, we considered to use a natural language library to automate the replacement of negations but figured it would be too risky as some opposites might be context dependent and thus don’t have an antonym that works for all cases. Not only that, but we also had no guarantee as to the completeness of the word library, so only a small fraction of negations could have been replaced.

# Word Cancellations

Simply put, this method takes what we assumed some of the most common words and removes them. Examples include: “i, you, am, have, were, ur, an”. For the complete list, please check the appendix [**Reference]**.

**Reasoning:** Like before, the goal was to create better context windows. By removing common personal pronouns and the verbs “are” and “have” we hoped to achieve more precise word associations. An English sentence, usually, has a structure where removing those words brings nouns, adjectives, adverbs, and verbs closer together. In doing so, we aimed to have those words come together in the context window.

**Flaws:** This method is based on our own hypothesis of what words we deemed empty of meaning and there is no rigid rule behind our word choices. One could therefore argue that our list is incomplete and that other words carry little to no sentimental information, like “maybe”, “some”, “or”, and “thing”. Aware of this (legitimate) critique, we implemented a method that addresses this issue. More on that in [**Chapter Cite**].

# Remove Tags

As its name suggests, this method removes the “<user>” and “<url>” tags from each tweet.

**Reasoning:** Our motivation is the same as we had for “Word Cancellation”. We removed what we assumed to be meaningless. We would have liked to have the original URLs to replicate the results of the paper [**Cite**] which found that keeping them increases accuracy. However, with only a tag, all we could do was to see how removing them affects our score.

# (Dis)Connect Negation

These two methods are very simple. By “connecting a not” we mean that “n’t” becomes “nt”. “Disconnecting a not”, on the other hand, transforms “n’t” into “ not “.

**Reasoning:** Inspired by the paper [**The Role of…**] we decided to see if we can replicate their results when we explicitly write out the “not”. Moreover, we wanted to take their connecting transformations of “won’t” and “can’t” one step further and connect all negations, not just those two. We ought it worthwhile to compare the two options. We assumed that connecting will reduce the number of embeddings because the twitter environment is filled with shortened words due to the character limit of a tweet. This means that e.g. a tweet likely contains “didnt” and therefore our transformation of “didn’t” will map to it. Disconnecting on the other hand will add more words to the context of “not”, probably reducing its value of information.

# Remove Short Words

If a word consists of two characters or less, it is removed from the tweet.

**Reasoning:** One reason for this, as so often, was to reduce dimensionality and improve word associations by moving meaningful words together into the context window. The other reason is that two character words are too few to be descriptive; they often are pronouns, articles or conjunctions. We therefore considered them safe to remove.

# Remove Same-Letter Sequence

In this step, every word in a tweet is checked and if it contains a letter more than twice in a row, that sequence is shortened down to two characters. For instance, the word “lol” - shorthand for “laughing out loud” – is often encountered with many o’s to emphasise how funny something is. Occurrences like “loool” are thus shortened to “lool”.

**Reasoning:** The shortening greatly reduces the number of word embeddings created.

# Keep Specific Word Types

With a natural language library, we categorised every word and filtered them. We only kept words that belonged to one of these types: noun, verb, adverb, adjective. This list includes verbs of all time tenses and comparatives and superlatives.

**Reasoning:** For this method, we assumed that only aforementioned types convey proper sentiment; all the other word types are neutral as they probably appear in both happy and sad statements equally often.

**Flaws:** Grammar and spelling both suffer from typing mistakes and abbreviations in online environments. Both these factors lead to words and sentence structures unknown to the machine which makes it difficult for it to label words correctly. A spell checker would probably ease that problem to some degree.

Pre-processing Evaluation

# Setup and Process

We used two classifiers to test the influence of our various pre-processing methods. One is a logistic classifier (LC), the other a recurrent neural network (RNN). The goal is to improve classification accuracy (fraction of correctly classified tweets over all tweets); this is therefore the metric used to evaluate our models. To compute a reasonable baseline, we ran each classifier on the original dataset and averaged the scores.

We applied each pre-processing method individually and compared the scores to the baseline. After this step, we ran all combinations of the methods that improved the score on the original data.

# Results and Discussion

## Logistic Regression

### Evaluation

A screenshot of a cell phone

Description generated with very high confidenceA screenshot of a cell phone

Description generated with very high confidenceIn table [**Cite**] we show the change each method has compared to the baseline score. Most notable are the decreases caused by removing sequences of characters and filtering out specific word types. They worsen accuracy by 0.56 and 0.74 respectively. A remarkable positive change can be observed when tags are removed. It adds 0.6 to the baseline. On second place is “connecting nots” which scores an additional 0.44.

In table [**Cite**] the results of applying two pre-processing methods are shown. To save space, only the combinations that achieved a change greater than 0.4 are displayed. For the full results, please refer to the appendix. It is worth noting that combining method R with others decreases its standalone contribution. Seemingly not interfering with each other are the combinations MD and DS which are close to the sum of their single-score values.

### Discussion

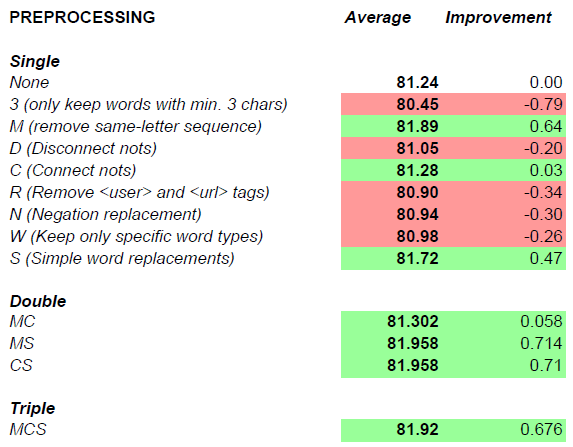
It comes as a surprise to us that the methods 3 and W decrease the score. They remove most words that we thought neutral but apparently that has a negative effect. We assume that this might be caused by creating “weaker bonds” between words. If we only keep words with sentiment, then their context window more rarely encounters a word that they have already seen. This leads to more infrequent word associations and therefore their strength to indicate towards a specific sentiment weakens.

The other methods all improved accuracy, likely because they all reduce the number of embeddings and create indicative word associations. The method C worked better than D; this is as we expected – given the findings in [**Cite**]. The impact of N is almost negligible. This is most likely because it does change very little and only rarely. A more sophisticated replacer could potentially contribute much more. Lastly, it would be interesting to figure out why applying R had such a big impact. It is a relatively big improvement for the small changes it makes.

When looking at the cases in which two pre-processing methods were applied, the pairs involving R improved accuracy the most. The methods are detached and it is unclear to us yet why the accuracy worsens compared to R alone. Contrary to pairs involving R, the remaining combinations improved nearly by the sum of their single-scores. We assign this effect, again, to the reduced word embeddings and increase in common word associations. The latter happens because (dis)connecting negations creates a standard way of writing them that is independent of spelling. Still, it would be interesting to find out why R is the only example that did not improve at all with other methods, despite all of them not influencing each other.

## Recurrent Neural Network

### Evaluation

Table [**Cite**] contains information on how a Recurrent Neutral Network is influenced by our various pre-processing methods. One quickly notices that, unlike in the case of a logistic regression, in many cases pre-processing the tweets leads to a decrease in accuracy.

Highest loss is caused by removing words of two characters or less. The best improvement is achieved by shortening the sequences of equal letters. Connecting negations is negligible whereas disconnecting them has a strong negative impact. Methods R and N have almost as much influence.

In the “Double”-Section, we see that the MC combination is insignificant but MS and CS bring about equal improvement. Finally, using all three methods that helped when used alone still provided a better accuracy when used together.

### Discussion

The nature of neural networks makes it difficult to find reasons about why the methods affect it the way they do. We therefore resort to often compare the RNN to the logistic regression (LR) model.

Our first guess as to what worsens the score was the removal of complete words. However, this is contradicted by method S. Not only that but removing the tags comes with a higher decrease than filtering word types.

Another noteworthy difference between the two models is that method N greatly affects the RNN whereas the logistic regression barely notices it. Similarly, method C greatly helped LR but is negligible for RNN. The same effect can be observed with method R, only that is negative for RNN instead of close to zero.

In our opinion, the most astounding effect occurs when we combine methods M and C. The former was a big winner when applied alone. However, in combination with C it is almost negated even though using only C slightly improves the score. In the Appendix, one can observe the same effect for the logistic regression case, though not as extremely. Surprising on this note is that method C significantly helps method S. Together, they achieve the same score as M and S together.

In conclusion, we can say that handling same-letter sequences is a safe improvement, given that both LR and RNN seem to benefit from it. They also agreed on connecting negations, although care should be taken when combining it with other pre-processing method. Lastly, for hard to explain reasons, our method to remove simple, common words also helps either model. The insight is limited and we hope that future research finds more patterns as to what words are safe to remove. Also, we think that our approach of word type filtering a worthwhile endeavour; at least when pursued in a more formal environment where grammar and spelling are guaranteed to be correct.