Instituto Superior Técnico

Natural Language - Mini Project 2 - Group 72

2023/2024

Group and relative contribution (%)

Tiago Miranda, no. 93416 ..... 50%
Nuno Machado, no. 94021 ..... 50%

We chose to divide the project where each student developed, and tested, different methods. The work wasn't performed in parallel however, both students helped each other with every detail of this project.

1 Models

1.1 Pre-Processing

We performed the data's pre-processing in the *review.py* file. Firstly we extracted the reviews and labels from the *.txt* file to *numpy* arrays. We then created a python function named $pp_x(X, stop\_word\_rm = boolean, lemma = boolean, max\_words = 380, emb = boolean)$ where we could set if we wanted to do stop word removal, lemmatization and apply an embedding, tokenizations were always applied. We opted to use the GloVe word embedding from torchtext. Padding according to the largest review was also done. We transformed the labels into one *numpy* array with two columns, both with 0's and 1's. The first column for the realness (truthful=1) and the other for the sentiment (positive=1). Even though we included the option to perform stop word removal, we opted not to apply it in order to preserve the full meaning of each review (important for our sequential model), which could be lost otherwise.

1.2 Bi-directional LSTM

To better capture the sequential relation of a review, while not losing much information, we created a simple bi-directional model using the **keras** library. We used a single bi-directional layer, from which the last 64 neurons of each enter two dense layers of 24 and 2 neurons. One of the output neurons deals with the realness classification and the other with sentiment analysis. We also tried variations of this network with a single output neuron to test if the performance would change if the tasks were tackled separately.

1.3 Support Vector Classifier

SVM's find the optimal hyperplane in high-dimensional space that separates different classes of data points while maximizing the margin between them. Using a kernel trick makes the algorithm suitable for non-linearly separable data. We tested this method to predict the realness as well as both labels simultaneously. To compare the performance of different SVM implementations we used cross-validation using *sklearn*'s *RepeatedStratifiedKFold* function for n_splits=20 and n_repeats=10 and just presented the mean accuracies obtained. We tested each classifier in Table 2 for several regularization parameter (C) values and different kernels.

2 Experimental Setup and Results

We determined the padding needed for the tokenized reviews according to the largest number of tokens in a review from both the training and test sets. The largest one had 868 tokens. The reviews resulted in an average of ≈ 169 tokens with a median value of 146 tokens, meaning that an approach where some information was lost on the bigger reviews would probably yield decent results, while being computationally more efficient, however, we decided to preserve all the information.

The data was split into a training set (90%) and a validation set (10%). In order to properly assess and compare models we fixed a seed for all splits (seed=42).

We implemented an early stop callback in the *LSTM*, keeping track of the validation loss and restoring the weights to the best epoch. A reduce learning rate on *plateau* callback was also implemented in order to improve convergence. We selected a batch size of 32 for training. The loss was a *binary crossentropy* function and we chose the *Adam* optimizer, the activation function for the fully connected layers were *ReLu*'s and sigmoids for the last layer.

Training the LSTM for the whole task (output with one neuron for the truthfulness and another for the sentiment) we got an overall accuracy of ≈ 0.82, the results per label are on table 1. We also trained an LSTM only for the sentiment analysis task (a single output neuron), with an overall accuracy of ≈ 0.94, whose metrics are also on table 1.

We also tested a SVC with both labels together and obtained an average accuracy of ≈ 0.81.

The best model (the one with the highest average accuracy) is a SVM + BiLSTM model where the SVM (with TF-IDF) classifies the realness and the BiLSTM classifies the sentiment of the review. The final accuracy was ≈ 0.83.

Model	Label	Precision	Recall	F1-score
LSTM with 2 outputs	Truthfulness	0.88	0.90	0.89
	Sentiment	0.92	0.91	0.91
LSTM with a single output	Sentiment	0.95	0.95	0.95
SVC	Truthfulness	0.85	0.87	0.86

Table 1: Per label results of the LSTM with two outputs and a single output as well as a SVC only for realness classification

Classifier	Stop. and Lemma.	GLoVe	TF-IDF	Accuracy
SVM	×	✓	×	0.764
SVM	✓	×	✓	0.877

Table 2: Realness classification where "Stop. and Lemma." represents Stopword removal and lemmatization

		True labels: Deceptive	True labels: Truthful	Total
SVM	Deceptive	58	11	69
	Truthful	9	62	71
	Total	67	73	140

Table 3: Confusion Matrix for Realness classification

		True labels: Negative	True labels: Positive	Total
LSTM	Negative	73	1	74
	Positive	5	61	66
	Total	78	62	140

Table 4: Confusion Matrix for Sentiment analysis

3 Discussion

We found that the SVM + BiLSTM model makes 26 errors in 140 datapoints in the validation set without overlap of errors between the classifiers. This indicates that the hypothesis of independence between labels (realness and sentiment) seems to hold which supports our decision to develop different classifiers for each task.

From Tables 3 and 4 we see that the sentiment analysis is more accurate than the realness classification from the lower off-diagonal values of the confusion matrix of the first method. This could be explained by the BiLSTM being more suited to its task. However, we also argue that the realness classification task is inherently trickier. Looking at the labelled data we found the sentiment analysis task to be straightforward and both students agreed on most labels. This was not the case for the realness classification where, by nature, the reviews, just like deepfakes, are written with the intent of being unrecognizable from true reviews.

We analysed the Recall and Precision results for each method without prioritizing any prediction in favour of another. If the project's goal was to develop a classifier to incorporate it into a *Booking.com* style website, that is to display relevant hotel reviews, then the Realness precision is a more appropriate metric given that the priority would be to display truthful reviews first.

Taking a better look at the model's mistakes we find reviews that are very complimentary of the hotel's neighbourhood (review_id=1131 with several adjectives such as "great", "amazing" and "cool") and the BiLSTM labels the review as positive even if the hotel experience itself was unpleasant. This mistake is understandable. However other sentiment analysis mistakes are not. Looking at the realness classification mistakes it seems that vague descriptions of the hotel amenities and its location are often mislabelled as a positive user experience (with an example being review_id=1311 from where we retrived the following excerpt: "The Palmer House Hilton is an incredible break from the real world. The lobby is a throwback to the grand era of travel, and you can just imagine people stepping off the train with their steamer trunks. The lobby is luxe and gilded, with soaring ceilings and is staffed with friendly, accommodating employees." which was incorrectly labelled as a truthful review).

4 Future Work

It would be fruitful to test contextual word-embedding methods namely *DistilBERT* to enhance the model's understanding of sentiment and deception cues. Data augmentation techniques could also improve the model's robustness. An ensemble model with the LSTM and SVC could also be an interesting option to explore in order to better take advantage of the individual strengths of the developed models.