University of Pisa

COMPUTER SCIENCE DEPARTMENT

Academic Year 2018/2019

Distant supervision for sentiment classification

Text Analytics Project

Daniela Occhipinti - 490548
Federica Trevisan - 568019

Contents

1 Sentiment140 dataset **2**
1.1 Introduction . . . 2

2 Data Understanding **3**
2.1 Training set . . . 3
2.2 Test set . . . 3
2.3 Validation set . . . 3

3 Data preparation **4**
3.1 Tweets length . . . 4
3.2 Cleaning data . . . 4

4 Weak labelling **5**
4.1 VADER algorithm . . . 5
4.2 SO-PMI algorithm . . . 5
4.3 VADER and SO-PMI evaluation . . . 6
4.4 Validation Set and threshold selection . . . 6

5 Classification **8**
5.1 Support Vector Machine . . . 8
5.2 Long Short-Term Memory . . . 9
5.2.1 LSTM Standard . . . 9
5.2.2 LSTM with embeddings . . . 10
5.3 Models comparison . . . 12
5.4 Conclusions . . . 13

1 Sentiment140 dataset

1.1 Introduction

Twitter is a microblogging service that suits well when developing a research of efficient techniques for sentiment analysis.
The dataset *Sentiment140* used for this project is available at the following link: `http://cs.stanford.edu/people/alecmgo/trainingandtestdata.zip`. It is composed of a training and a test set; both sets are isolated and distinct.
The dataset is a collection of tweets extracted in 2009, from April 6, until June 25, about various topics; each record of the dataset represents a specific tweet, that can have a "positive", "neutral" or "negative" sentiment depending on the content of the text.
Sentiment is defined as a personal negative or positive feeling. Sentiment analysis is the field of study that analyzes people's opinions, sentiments, emotions towards entities like products, services and organizations. It is widely applied in extracting insights of consumer feelings about a specific product and is useful for consumers who want to take decisions about a purchase, or companies that want to monitor the public sentiment of their brands.
The objective of this project is to compare two different weak labelling methods for implementing and evaluating a sentiment classifier.
Initially, after a proper preprocessing of the datasets, two methods for unsupervised sentiment classification have been compared, VADER (*Valence Aware Dictionary for sEntiment Reasoning*) and SO-PMI (*Semantic Orientation - Pointwise Mutual Information*). Both methods give as output a continuous score (e.g. from -1 to 1 for VADER) that ranks the polarity of the tweet.
The best performing one has been chosen and validated for discarding the neutral tweets from the dataset, since we decided to tackle the classification problem as a binary one (positive/ negative sentiment). Neutral tweets may have a negative impact on the models training performances and on the classification results on the test set.
The classification models that have been trained for this project are *Support Vector Machine* (SVM), *Long-Short Term Memory Neural Network* (LSTM), and *Long-Short Term Memory Neural Network* with embeddings (GloVe - *Global Vectors for Word Representation*).
The code of this project is fully available at the following Github repository: `https://github.com/federikovi/635AA-TextAnalytics`. The Python libraries Keras, NLTK, Sklearn have been used.

2 Data Understanding

There are no missing values in none of the three datasets.

2.1 Training set

The original training set is composed of 1.6 millions of records and 6 attributes:

- Polarity of the tweet (0 = negative, 2 = neutral, 4 = positive)
- ID of the tweet (e.g. 2087)
- Date of the tweet (e.g. Sat May 16 23:58:44 UTC 2009)
- Query (e.g. lyx). If there is no query, then this value is NO_QUERY.
- User that tweeted (e.g. robotickilldozr)
- Text of the tweet (e.g. Lyx is cool)

In this project it has been decided to keep only the columns *polarity* and *tweet*. At a first glance, it has been noticed that the polarity of the training set takes only values 0 and 4. Its distribution is uniform, 800000 tweets are positive and 800000 are negative. The neutral tweets (tweets with *polarity* = 2) are not present.
The training data has been extracted with an unique approach as described in the paper *Twitter sentiment classification using distant supervision* [1].
In the paper the training data has been created automatically, without any need of human annotation of the tweets sentiment. The approach for extracting tweets from the Twitter API used is the following: if any tweet has positive emoticons, like :), is positive while tweets with negative emoticons, like :(, are negative.
In this way, records extracted with a positive emoticon are labelled as positive, *polarity* = 4, while records with a negative emoticon are labelled as negative, *polarity* = 0.

2.2 Test set

The test data has 498 records and it has been manually collected from the Twitter API by the authors of the paper. It has six attributes, the same as the training set; also in this case the only columns we consider are *polarity* and *tweet*.
Differently from the training set, the polarity of each tweet has been manually marked, with a result of 177 negative tweets and 182 positive tweets. The remaining 139 tweets are neutral (*polarity* = 2), so they have been removed, reducing the test set to 359 records.
Since the polarity in the test set has been manually marked, they are treated as gold data.

2.3 Validation set

The validation set has been separately downloaded from the following link
`http://sentistrength.wlv.ac.uk/documentation/6humanCodedDataSets.zip`
with the main purpose of validation of the best weak labelling method and validation of the threshold for the elimination of neutral tweets.
SentiStrength reports two sentiment scores for positive and negative sentiment in short texts: -1 (not negative) to -5 (extremely negative), 1 (not positive) to 5 (extremely positive). The validation set has 4242 records and 3 features, that are:

- Mean positive: the mean of positive scores given by humans who voted the sentiment tweet
- Mean negative: mean of negative scores given by humans who voted the sentiment tweet
- Tweet: the text of the tweet

This dataset has been chosen also because the polarity has been manually marked, so it is gold as the test set.
The label marking has been assigned with a mechanical turk procedure and it has a human-level accuracy. The polarity of each tweet has been voted from a court of humans. For each tweet, each member classifies the tweet as "positive" or "negative". The sum of the votes accumulates the "positivity" or "negativity" score of that tweet. The difference between these two values returns the polarity score.
Finally, the *polarity* ≥ 2 has been converted to 1 (positive) while the *polarity* $\leq$ -2 to 0 (negative).
Discarding the neutral tweets also in this case, the validation records are 798: 503 of them are positive, 295 negative.

3 Data preparation

3.1 Tweets length

The *Sentiment140* training dataset has been collected in 2009, when the maximum length for tweet was 140 characters. As a preliminary check, we noticed that for some cases the length of the string "text" exceeds the 140 limit. Some tweets were longer only for HTML decoding, that has been removed with *Beautifulsoup*. Other tweets had another decoding and have been removed.

3.2 Cleaning data

The three datasets have been cleaned using the text processing tool *Ekphrasis* `https://github.com/cbaziotis/ekphrasis`.
The tool performs tokenization, word normalization and word segmentation (for splitting hashtags), using word statistics from 2 big corpora (english Wikipedia, Twitter - 330mil english tweets).
In this way, we handled many unique properties of the Twitter language model reducing the feature space.

1. **Usernames**: Twitter offers the possibility to include usernames in the tweets in order to direct the messages (e.g. @federikovi). The usernames starting with the symbol @ have been substituted with a unique token @USER.

2. **Links usage**: links are often included in tweets. We decided to convert them to the URL token.

3. **Elongated words**: Twitter can contain a casual language, like repeated letters inside a word that we decided to remove (e.g. *seeentiment* becomes *sentiment*).

4. **Text normalization**: the text has been normalized, switching all characters to the lowercase.

5. **Contractions**: the English language contractions have been unpacked (e.g. *can't* becomes *can not*).

6. **Eliminations**: we decided to eliminate emails, numbers, percentages, dates, time stamps, telephone numbers because not inline with the purpose of sentiment analysis.

7. **Tokenization**: the text has been segmented.

4 Weak labelling

Weak labelling is a way of labelling training data with a weakly-supervised or semi-supervised approach: data are not all validated by a human and may contain errors. As already mentioned above, for our study we decided to use VADER and SO-PMI.

4.1 VADER algorithm

VADER (*Valence Aware Dictionary for sEntiment Reasoning*) is a computational sentiment analysis engine that has good performances on social media texts. It is built on valence-based and human-curated *gold-standard* sentiment lexicon which has been attuned to microblog-like contexts. This lexicon, derived from other sentiment lexicons and produced using quantitative and qualitative methods, has been combined with grammatical and syntactical rules used by humans to emphasize sentiment intensity and for handling negations. VADER has been validated by humans: starting from a group of 20 human raters for sentiment of tweets, it has been proved that VADER perform as well as human raters [2].
When applied to tweets, VADER checks if the analyzed tweet contains any of the words present in the sentiment lexicon and produces four metrics, *Positive*, *Negative*, *Neutral* and *Compound*: the first three represent the proportion of the text that falls into those categories; the last one is the sum of all lexicon ratings which have been standardized to range between -1 and 1.

4.2 SO-PMI algorithm

SO-PMI (*Semantic Orientation - Pointwise Mutual Information*) is an unsupervised learning algorithm for classifying tweets as positive or negative comparing their similarity to words having positive or negative meanings. In this way, this algorithm provides an estimate of the semantic orientation of tweets by giving them a numerical rating which represents the Pointwise Mutual Information between tweets and positive or negative words. This numerical rating also indicates the strength of the semantic orientation.
The Pointwise Mutual Information between two words is defined as follows:

$$PMI(word_1, word_2) = \log_2 \left[\frac{p(word_1 \cap word_2)}{p(word_1 word_2)} \right]$$

where at the numerator there is the probability that the two words occur together and at the denominator the probability that they occur together if they are statistically independent. The ratio measures the independence between the two words.
The Semantic Orientation of a phrase is calculated as follows:

$$SO(phrase) = PMI(phrase, word_{positive}) - PMI(phrase, word_{negative})$$

SO is positive if the phrase is more statistically associated with the positive word, negative otherwise. [3]

4.3 VADER and SO-PMI evaluation

Due to computational complexity issues, we decided to apply SO-PMI to a sample dataset composed of 20000 tweets obtained by shuffle and compare its performances with VADER's ones on the same sample. We compared the labels given by both weak labeling methods and the *polarity* labels originally given.

	SO-PMI	**VADER**
Accuracy	0.499	0.668
Precision	0.508	0.617
Recall	0.071	0.892
F1 score	0.124	0.729
Pearson Correlation	0.004	0.375

Table 1: VADER and SO-PMI metrics on 20000 tweets.

Looking at the table, it is possible to notice that, on the sample, VADER's scores are higher than SO-PMI's ones. For this reason, we decided to proceed in our study using VADER algorithm and discarding SO-PMI.
After this we applied VADER on the full training set and compared its scores with the original *polarity* scores.
The resulting metrics are shown in Table 2.

	VADER
Accuracy	0.656
Precision	0.662
Recall	0.638
F1 score	0.650
Correlation Pearson	0.313

Table 2: VADER measures on full training set.

4.4 Validation Set and threshold selection

Since in the training set there are no tweets having neutral polarity originally given, VADER scores can be used to detect and discard neutral tweets with the aim to reduce the training set and focus our attention only on positive and negative tweets.

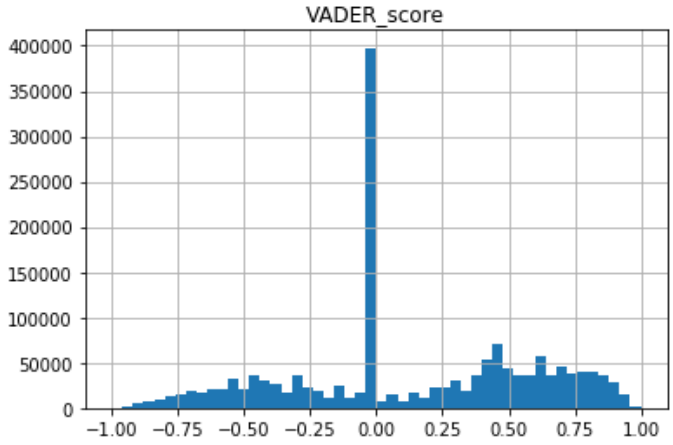


Figure 1: VADER scores distribution on full training set.

After observing the distributions of VADER scores, we decided to choose as thresholds the following values:

- VADER score = 0
- -0.25 < VADER score < 0.25
- -0.35 < VADER score < 0.35
- -0.50 < VADER score < 0.50

Since the polarity scores originally given for the training set are not gold, we decided to validate VADER thresholds on the *SentiStrength* validation set.
Comparing the metrics we obtained for each threshold, we notice that the best metrics are the ones for |0.5| threshold.

Validation set	**Vader = 0**	**-0.25 <V <0.25**	**-0.35 <V <0.35**	**-0.5 <V <0.5**
Accuracy	0.8638	0.8908	0.9033	0.9399
Precision	0.8616	0.8960	0.9129	0.9579
Recall	0.9392	0.9475	0.9532	0.9610
F1 Score	0.8987	0.9210	0.9327	0.9595

Table 3: VADER metrics on thresholds - Validation Set.

In order to definitively choose the threshold, we compared on the full traning set the measures obtained by applying VADER on the same thresholds to check if |0.5| is the best threshold to discard neutral tweets.

Training set	**Vader = 0**	**-0.25 <V <0.25**	**-0.35 <V <0.35**	**-0.5 <V <0.5**
Accuracy	0.7151	0.7414	0.7549	0.7806
Precision	0.6624	0.6992	0.7177	0.7512
Recall	0.8586	0.8872	0.9091	0.9287
F1 Score	0.7479	0.7820	0.8022	0.8306

Table 4: VADER metrics on thresholds - Full training set.

In both cases, the best metrics are the ones obtained for |0.5| threshold. For this reason we decided to discard tweets for which VADER score is between -0.5 and 0.5, removing 993609 tweets.

5 Classification

As anticipated in the introduction, 6 models have been built for the binary classification: 3 have been trained on the original polarity of the training set (SVM, LSTM and LSTM with GloVe) and the other 3 have been trained on the VADER score obtained from the previous phase (SVM, LSTM and LSTM with GloVe).
This has been chosen in order to make a comparison between the polarity scores given by our weak labelling method and the polarity scores of the original training set.
For metrics of all the classifiers we refer the reader to the Table 9.

5.1 Support Vector Machine

Support Vector Machine is a supervised learning method used for classification. In this algorithm, each item of the dataset is a point in a *multidimensional space* or *hyperplane* with the value of each feature being the value of a particular coordinate. Support Vectors represents the coordinates of each observation. Classification task is performed by finding the hyperplane that better differentiate the classes.
For SVM, we decided to apply features selection according chi-squared stats, which measure dependence between variables: we selected only features with the highest k scores. For choosing the best k value, we decided to test five different k values: 100, 5000, 10000, 20000 and 25000.
In Table 5 we compare SVM metrics for each k value: the highest are for $k = 5000$, so we decided to consider only SVM with $k = 5000$.

	k=100	**k=5000**	**k=10000**	**k=20000**	**k=25000**
Accuracy	0.719	**0.833**	0.825	0.825	0.825
Precision	0.667	**0.811**	0.799	0.808	0.802
Recall	**0.890**	0.874	0.874	0.857	0.868
F1 score	0.762	**0.841**	0.835	0.832	0.834

Table 5: SVM metrics for different k values.

Then, we trained SVM also on VADER scores in order to compare both results. The following tables represent the confusion matrices for SVM trained on the original polarity and SVM trained on the VADER scores.

SVM	**Prediction: Negative**	**Prediction: Positive**
Real: Negative	140	37
Real: Positive	23	159

SVM VADER	**Prediction: Negative**	**Prediction: Positive**
Real: Negative	133	44
Real: Positive	22	160

Table 6: Confusion matrix - SVM polarity (left), Confusion matrix - SVM VADER (right).

Regarding the SVM trained on the polarity, the positive class is represented as "positive tweet". Considering the application of the model, we evaluate as best model the one with less false positive (negative tweets labelled as positive). On the other hand, predictions error on the "negative" class (positive tweets labelled as negative) have a minor impact.
Between the two SVM models, the best performing one is the SVM trained with the original polarity, with 37 negative tweets labelled as positive (Table 6, left).

5.2 Long Short-Term Memory

Long Short-Term Memory (LSTM) units are an extension of a *Recurrent Neural Network* (RNN). An RNN is a neural network designed to recognize patterns in sequences of data, such as text, spoken words, numerical times series data, etc. In RNN connections between units form a directed cycle. Recurrent networks take as input not just the current input example they see, but also what they have perceived previously in time. A common LSTM unit is composed of a cell, an input gate, an output gate and a forget gate.
Gates act on the signals they receive and block or pass information based on its strength, which they filter with their own sets of weights. Cells learn when to allow data to enter, leave or be deleted in an iterative way, making guesses and adjusting weights through gradient descent.

We decided to use a Bidirectional LSTM because it connects two hidden layers of opposite directions to the same output so that the output layer can get past and future information at the same time. That is, while in Recurrent Neural Networks the future information is not reachable from the current state, preserving only past information, in Bidirectional Recurrent Neural Networks both future and past information is preserved. Using also information from the future may help the network learning in a better way.
We used the *sigmoid function* as activation function and *Adam*, an extension to stochastic gradient descent, as optimizer. We applied two different implementations of LSTM: a standard LSTM and a LSTM based on GloVe word embeddings. As SVM, both implementations were trained on the original polarity on one side and on VADER score on the other. All LSTMs are trained and validated through the holdout method. The training set was splitted in the following way: 80% for training and 20% for validation. Finally all LSTMs have been tested on the test set.

5.2.1 LSTM Standard

Analyzing the training and validation accuracy plot (Figure 2, left) it can be noticed that the maximal accuracy on the validation set is at the 7th epoch.
Regarding the loss function (Figure 2, right), it reaches the minimum on the validation set at the 6th epoch with 0.360.

Figure 2: Training and validation accuracy (left), training and validation loss (right) LSTM trained on polarity.

Considering the accuracy of the training and validation accuracy plot of LSTM trained with VADER (Figure 3, left) the maximum of the accuracy on the validation set is reached at the 5th epoch; regarding the loss on the validation the minimum values is reached at the 6th epoch with 0.052.

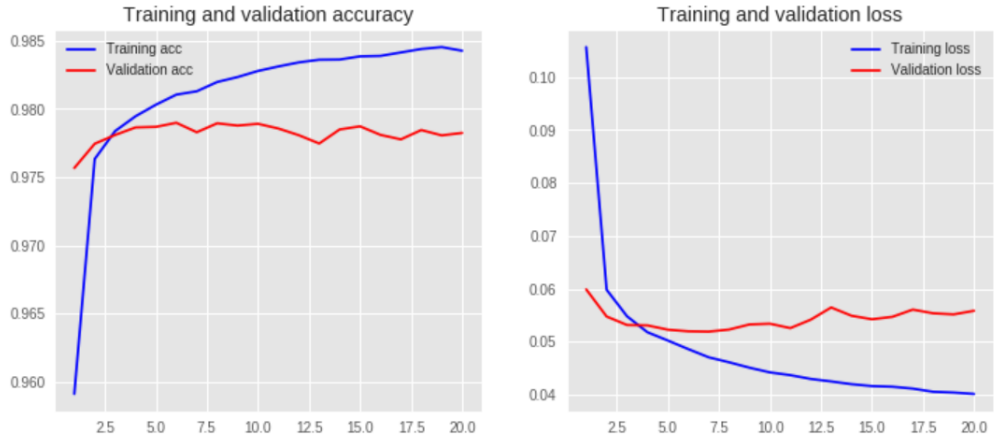


Figure 3: Training and validation accuracy (left), training and validation loss (right) LSTM trained on VADER scores.

Considering the two LSTM models, the best performing one is the LSTM trained with the original polarity, with 43 negative tweets labelled as positive (Table 7 left).
It can be noticed that it is also the best of the two models in terms of metrics (Table 9).

LSTM	**Prediction: Negative**	**Prediction: Positive**
Real: Negative	134	43
Real: Positive	27	155

LSTM VADER	**Prediction: Negative**	**Prediction: Positive**
Real: Negative	119	58
Real: Positive	21	161

Table 7: Confusion matrix - LSTM polarity (left), Confusion matrix - LSTM VADER (right).

5.2.2 LSTM with embeddings

An embedding is a low-dimensional space into which can be translated high-dimensional vectors. Embeddings make it easier to do machine learning on large inputs like sparse vectors representing words. An embedding captures some of the semantics of the input by placing semantically similar inputs close together in the embedding space.
For the training of the LSTM we used word embeddings from GloVe (*Global Vectors for Word Representation*). [4]
GloVe is a model for the unsupervised learning of word representations that outperforms other models on word analogy, word similarity, and named entity recognition tasks. It was trained on a dataset of one billion tokens with a vocabulary of 400 thousand words. GloVe has embedding vector sizes, including 50, 100, 200 and 300 dimensions.
For our purposes, we have chosen the 100-dimensional word vectors pre-trained on aggregated global word-word co-occurrence statistics from Twitter (2B tweets, 27B tokens, 1.2M vocab).
As depicted in the training and validation accuracy plot trained with polarity (Figure 4, left), the accuracy of the validation set reaches the maximum at the 5th epoch; while the loss function on the validation set (Figure 4, right) reaches the minimum at the 4th epoch with 0.343.
The loss function grows increasingly with the progress of the algorithm.

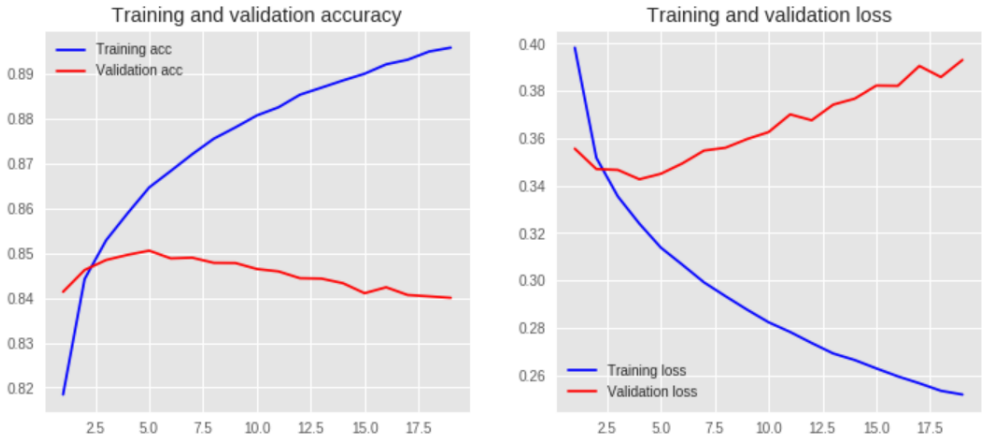


Figure 4: Training and validation accuracy (left), training and validation loss (right) LSTM Glove trained on polarity

Regarding the training and validation accuracy of LSTM GloVe trained with VADER score (Figure 5, left) the accuracy on the validation set has a steady high progress, reaching the maximum at the 8th epoch.
The loss function on the validation set has a steady progress with low values, taking the minimum pick at the 8th epoch with 0.011.

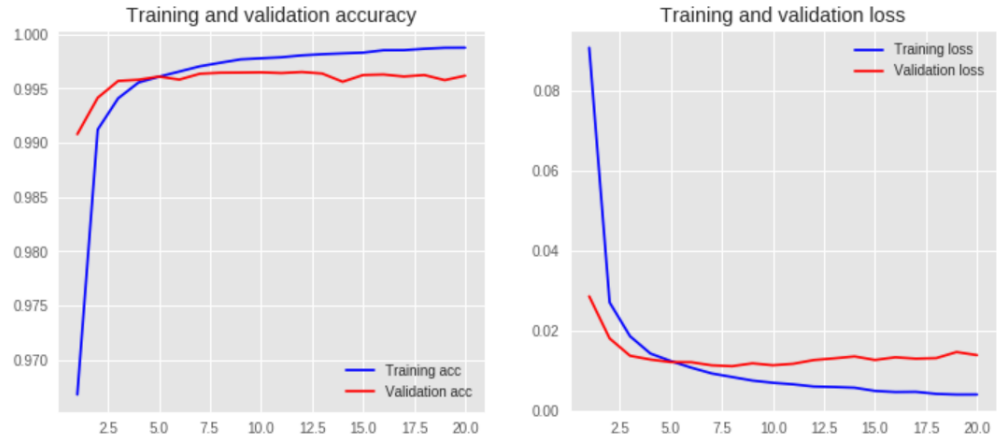


Figure 5: Training and validation accuracy (left), training and validation loss (right) LSTM Glove trained on VADER scores

Considering the two models, the best performing one is the one trained with the original polarity, with 36 negative tweets labelled as positive (Table 8 left).
It can be noticed that LSTM GloVe trained on polarity is the best model with the highest metrics.

LSTM GloVe	**Prediction: Negative**	**Prediction: Positive**
Real: Negative	141	36
Real: Positive	22	160

LSTM GloVe VADER	**Prediction: Negative**	**Prediction: Positive**
Real: Negative	124	53
Real: Positive	15	167

Table 8: Confusion matrix - LSTM GloVe polarity (left), Confusion matrix - LSTM GloVe VADER (right)

5.3 Models comparison

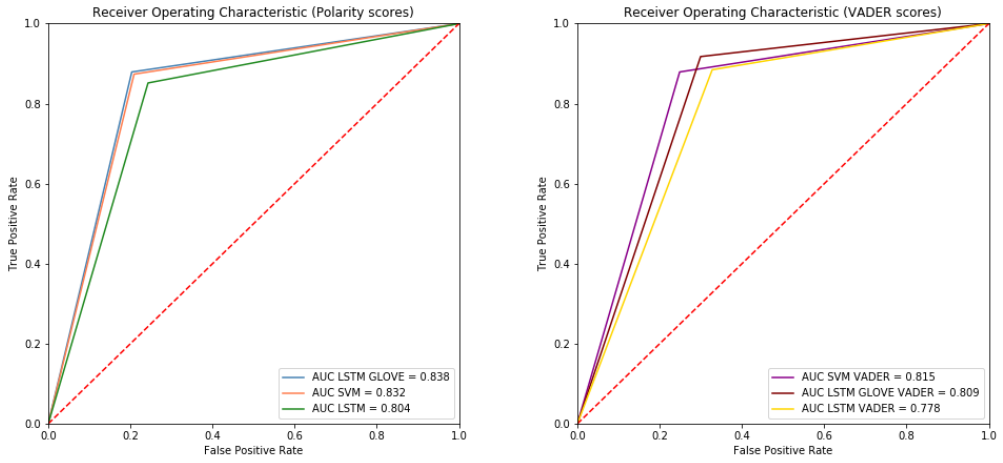
The final objective of this project is to compare the different classifiers evaluating the best one in terms of accuracy in the label prediction.
As analyzed before, in terms of false negative the best classifiers were SVM polarity, LSTM polarity, LSTM GloVe polarity.

	Polarity scores			**VADER scores**		
	SVM	**LSTM**	**LSTM GloVe**	**SVM**	**LSTM**	**LSTM GloVe**
Accuracy	0.833	0.805	**0.838**	0.816	0.780	0.811
Precision	0.811	0.783	**0.816**	0.784	0.735	0.759
Recall	0.874	0.852	0.879	0.879	0.885	**0.918**
F1 score	0.841	0.816	**0.847**	0.829	0.803	0.831
AUC	0.832	0.804	**0.838**	0.815	0.778	0.809

Table 9: Classifiers' metrics

The table above represents the metrics for each classifier trained. As highlighted in the table, LSTM GloVe has the highest value of accuracy, precision, F1 score and Area Under the Curve and the lowest number of False positive (36); it can be clearly defined as the best model for our purpose.

Considering only the "trained with VADER" models, the best one in terms of accuracy, precision and area under the curve is SVM, while the best one in terms of recall is LSTM GloVe. In fact, on the Figure 6 at the right representing the ROC curves, the purple one identifies SVM and is the most left one.



(a) ROC curve Polarity scores

(b) ROC curve VADER scores

Figure 6: ROC Curve

As a final remark, from the ROC curve at the left (Figure 9) it can be noticed that the model LSTM GloVe is the best in terms of Area Under the Curve.

5.4 Conclusions

Using distant supervision it has been possible to add another phase to the classification problem: the selection of the weak labelling method.
After comparing the performance analysis of the different classifiers, the best one in terms of prediction is the LSTM GloVe, in the case of training with polarity label. The best performing model trained with VADER score is SVM.
In our case study this means that the best weak labelling method seems to be the one proposed in the original paper, *A parsimonious rule-based model for sentiment analysis of social media text.*
Nonetheless, the classifiers trained on VADER score have produced interesting results in terms of metrics.
For a future development, we could tackle this problem as a multiclass classification (positive/neutral/negative) or repeat the same experiment in another language (e.g. Italian).

References

[1] Go, Alec Bhayani, Richa Huang, Lei. *Twitter sentiment classification using distant supervision.* CS224N Project Report, Stanford, 2009. Available at `https://cs.stanford.edu/people/alecmgo/papers/TwitterDistantSupervision09.pdf`.

[2] Gilbert, CJ Hutto Eric. *Vader: A parsimonious rule-based model for sentiment analysis of social media text.* Eighth International Conference on Weblogs and Social Media (ICWSM-14). Available at (20/04/16) `http://comp.social.gatech.edu/papers/icwsm14.vader.hutto.pdf`, 2014.

[3] Turney, Peter D. *Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews.* Association for Computational Linguistics, Proceedings of the 40th annual meeting on association for computational linguistics, pages 417-424, 2002. Available at `http://www.aclweb.org/anthology/P02-1053.pdf`

[4] Pennington, Jeffrey and Socher, Richard and Manning, Christopher. *Glove: Global vectors for word representation.* Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), pages 1532-1543, 2014. Available at `https://nlp.stanford.edu/pubs/glove.pdf`