An Analysis of Machine Learning Techniques to Prioritize Customer Service Through Social Networks

P. R. P. Amora[1], E. M. Teixeira[1], M. I. V. Lima[1], G. M. Amaral[1], J. R. A. Cardozo[2], J. C. Machado[1]

[1] LSBD - Department of Computer Science, Universidade Federal do Ceará, Fortaleza CE, Brazil
{paulo.amora,elvis.teixeira,isabel.lima,gabriel.maia,javam.machado}@lsbd.ufc.br
[2] Digitro Tecnologia SA, Brazil

Abstract. The large amount of opinionated data made available by social networks allows the extraction of valuable information for a variety of applications. Sentiment analysis is a powerful tool in this sense, allowing to identify and classify opinions in texts according to the predominant polarity exposed in them. An interesting use of this technique is for companies to rank the messages from their clients in order to identify and attend the most dissatisfied ones first, thus improving customer service. In this work, we evaluate the application of a range of different machine learning techniques (including two deep learning ones) to the sentiment analysis of tweets in Brazilian Portuguese, aiming customer service prioritization. Our results show that the deep learning models are able to classify tweets more efficiently in this context, compared to traditional machine learning ones.

Categories and Subject Descriptors: I.2.7 [**Artificial Intelligence**]: Natural Language Processing—*Text Analysis*; *Language parsing and understanding*; I.2.6 [**Artificial Intelligence**]: Learning—*Concept Learning*

Keywords: Sentiment Analysis, Deep Learning, LSTM, Social Networks, Customer Service

1. INTRODUCTION

Billions of users worldwide make use of social networks to express their thoughts, emotions and opinions and share all sorts of content. A huge amount of data is generated out of these actions, which can be collected and mined for different purposes. For example, companies that use social networks as a communication channel with their clients may wish to analyze the feedback provided by reviews of their businesses to measure the level of satisfaction related to specific services or products.

An extremely useful tool in the process of analyzing opinionated content is the use of sentiment analysis. Sentiment analysis (also referred to as opinion mining) is a known technique used to classify texts according to their prevalent polarity, in order to infer from texts the sentiment of their author [Liu 2015]. It may be suitable, for example, to identify if certain comments have a negative or positive connotation. Businesses may take advantage of this to discover the public sentiment towards their brand and make strategic decisions based on such analysis.

Another scenario is one that involves companies which are overwhelmed with hundreds of daily messages through social networks with questions, complaints and issues, making it difficult to provide efficient customer service. A possible application of sentiment analysis, in this context, is to classify the incoming messages according to their level of relevance, and prioritize customer service to give support to the most critical cases first (i.e. angry clients, suing threats and urgent problems). For service prioritization, it makes sense to determine the level of relevance of a message by the displayed

This publication is the result of a project promoted by the Brazilian Informatics Law (Law No. 8248 of 1991 and subsequent updates) and was developed under the Cooperation Agreement 01/2016 between Federal University of Ceara and Digitro Tecnologia SA. This research was also partially supported by LSBD/UFC.

feeling of dissatisfaction. Dissatisfied customers are most likely to lose their temper, make threats or cancel services, so it is important to make sure these cases are attended before less pressing ones.

The problem with relying solely on the polarity of a sentence to prioritize customer service is that it may be an inefficient approach due to the fact that there generally are many more negative messages than positive ones. This behavior reflects the fact that social network users are more likely to make complaints than write recommendation or satisfaction notes and, consequently, the majority of the cases analyzed may end up being classified as critical and the goal of prioritization would not be achieved. Therefore, it may be suitable, in this case, to break down the negative comments into two or more categories following a dissatisfaction scale.

In this work, we present a study of different machine learning techniques applied to the sentiment analysis of customer feedback in social media, including a deep learning method (LSTM) and a variation of it (bidirectional LSTM). We analyze a corpus of collected tweets in Brazilian Portuguese that mention two major telecommunications operators from Brazil, which are classified using three labels: Positive, Negative and Critical. Finally, we also test three different classification approaches using combinations of the labels and compare the results between them, using K-Fold as the validation method.

2. THEORETICAL BACKGROUND

2.1 Long Short-Term Memory Units (LSTM)

Recurrent neural networks (RNNs) assume the existence of correlations between different inputs to the network, be them semantic, causal, logical or simply temporal connections. The basic architecture for a RNN is the Simple Recurrent Network (SRN), proposed by Elman et al. [Elman 1990], consisting of a modification of a backpropagation network. In the SRN, the input is partitioned into a stream and the first element is fed into the input layer. The hidden layers take this information and also the contents of a context layer, which represents the internal state of the network. Results from the hidden layers are then used to update the contents of the context layer and more data is fetched from the input stream into the input layer. As in backprop networks, SRN relies on back-propagation algorithms to update weights and biases, including from context layer to hidden layer.

An RNN, then, possesses directed loops, allowing information about data already processed by the network to be fed backwards and used as input. Maintaining an internal state grants it a dynamical reaction to any series of inputs, where past information plays a part in determining the RNN's reaction to incoming information. However, such traditional RNNs find problems in their gradient-based updates, in particularly the vanishing gradient problem, as evidenced by Hochreiter et al. [Hochreiter et al. 2001]. The Long Short-Term Memory (LSTM) network, proposed by Hochreiter et al., [Hochreiter and Schmidhuber 1997], avoids such gradient problems. LSTM is a deep learning system, consisting of a cascading number of recurrent network units, each specialized in "remembering" information from past units, be it for a short or long duration.

LSTM units have an inner state vector and gates. A gate is a vector, whose values range from 0 to 1, which controls the flow of information inside the unit. They are computed through logistic functions, and then multiplied by the flowing information. The inner state vector tracks which values from the input stream are to be remembered or forgotten. All vectors have weights and biases associated to them, which are the only parameters of the LSTM and are trained with ”backpropagation through time” [Mozer 1995].

Figure 1, adapted from [Olah 2015], illustrates the LSTM unit utilized. The unit t consumes the word t (x_t), along with the output and inner state of unit $t-1$ (h_{t-1} and c_{t-1}), and calculates the values of three gate vectors f_t, i_t and o_t, represented in said order by yellow σs, which are then used with a hyperbolic tangent function ($tanh$) and sum ($+$) and Hadamard product (x) operations to calculate its new inner state vector c_t and its output vector h_t.

Gates i_t, the input gate, and f_t, both regulate the data flow into the unit's inner state. While i_t controls how much of it comes from input, f_t controls how much comes from the last unit's hidden state. The output gate o_t decides this state's influence in the output.

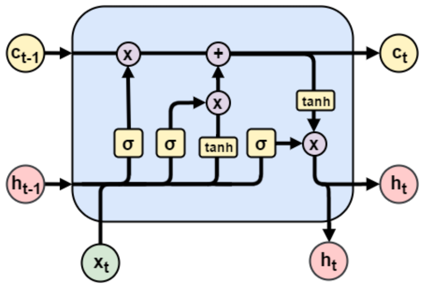

Fig. 1: Flow and structure of a LSTM unit.

2.2 Word Embedding

An embedding layer was employed to transform the textual inputs into numerical features. This layer is also trainable for better representing the words. Assuming that the layer is pre-trained to optimally represent the language's great majority of known and valid words, and allowing for new word entries, it then trains itself to better understand new vocabulary. This allows it to evade errors coming from the use of language in the Twitter platform, such as typos, colloquialism, sarcasm and other variants carried with sentimental value over syntactic.

After embedded into numeric vectors, words are given one-at-a-time to LSTM layers consisting of a recurrent unit each, which unfolds through time into a sequence of units. In addition, all LSTM layers apply the dropout technique to avoid overfitting, as described in [Hinton et al. 2012].

The sequence of output vectors of each LSTM layer is given to the next one as input vectors. The last LSTM layer is followed by a series of densely-connected regular neural network layers. Each neuron i in a dense layer processes a matrix X of all the layer's input into an output y_i through a function $y_i = \sigma(W_i X + b_i)$, where σ is a softmax function, and W_i and b_i are trainable parameters. The last dense layer is composed of n neurons, where n is the number of classes, each outputting a value $z_i \in [0, 1]$ for the tweets belonging to class i. We finally take the highest value to represent the tweet's class.

2.3 Bidirectional LSTM

Although LSTMs are good at exploiting long-range context, they are only able to take advantage of the previous context of the input. However, using bidirectional LSTMs (BLSTMs), it is possible to also exploit future context from the current state, which is done by processing the data in opposite directions using two hidden layers, and then connecting them to the same output layer [Graves et al. 2013]. This way, the output layer is able to access information from both past and future states. The most notable applications of BLSTMs include speech and handwritten recognition and part-of-speech tagging [Plank et al. 2016].

3. COMPLAINT PRIORITIZATION

When customers publicly manifest their issues with service providers, it may harm a company or product/service image. Besides customer service centrals, users can also manifest their dissatisfaction and problems with the service provided through social networks, which have a much wider range of influence and affect the business directly, allowing their critiques to be read and heard by a considerable amount of their customer base. Moreover, even if it is not always being used as a standard form of communication with companies, social media are, at an accelerating speed, being used as platforms to bring these companies, their brands, and their customers together. On those grounds, companies today have teams of customer service for social media, with the mission of answering those complaints in able time to preserve the company image and to ensure a better interaction between brand and public.

However, these strategies may become unfeasible when the number of complaints and comments directed at a company is so large that the people assigned to respond them are not enough. Take Twitter for example, which produces hundreds of millions of tweets per day. A great portion of them is addressed to user profiles owned by companies, all waiting for a reply. Therefore, some kind of screening is necessary in order to allow the customer service teams to perform well. To do this screening and further prioritization, we propose the use of sentiment analysis. However, the prioritization is not straightforward because:

—The number of complaints in most social media far outweighs the number of compliments and neutral inputs.

—Given that we have several serious complaints, which ones seem more urgent? How to best create an order to address them?

The first point refers to a known problem in classification tasks, which is unbalanced classes. An unbalanced dataset can provide very misleading results and a model unfit for the application. Here, it could give us a model that is excessively inclined to label tweets as complaints, which would provide no help to customer services. Solutions for unbalanced datasets include increasing data collection, changing performance metrics, synthetic sampling, different algorithms, a change of perspective, and others. In this work, we opted for a change in metrics and perspective. For the metrics, mere accuracy would be wrongfully swayed by the larger class. Instead, Precision, Recall, and F-score were used; they were calculated for each class detected in the problem, and the values averaged. This gives us a much clearer idea of our model's performances even under uneven grounds.

As for the perspective change, we eased the imbalance by, given good, neutral and bad comments, uniting good and neutral comments under the same class and dividing the remaining class of bad comments into two classes based on severity, resulting in three more balanced classes. Specifically, instead of the traditional Positive/Negative classes, we propose a new label, called Critical, in which the most pressing complaints are classified. We also define Positive as the class containing both positive and neutral tweets. Making a binary classification problem into a multi-class one has its drawbacks when regarding accuracy, which is why two alternative approaches to the problem will be taken into account: in one (which we shall call "Critical vs Non-Critical", we change the problem back into binary, and in another (named "double binary") we try mitigating the accuracy loss by dividing the multi-class problem into two binary ones. Also, the Critical label is semantically very close to the negative class, with the main difference consisting either in the context or intensity associated with the words, like swearing and threatening. A clear advantage of this division is that we now have a class to discriminate which complaints should be prioritized.

To tackle this classification problem, we decided to use LSTM and BLSTM networks along with more traditional machine learning approaches, such as Support-Vector Machines, Naive-Bayes, Decision Trees, Random Forests, K-Nearest Neighbors and an Ensemble Voting classifier composed of all these approaches besides LSTM/BLSTM. The main reason for this is to observe if these deep learning methods' ability to encode and interpret the context of each word used in a sentence will give them an upper hand in differentiating between Negative and Critical tweets when compared to the other approaches, even when they are combined.

The first step is gathering the data. Given the context of our application, we decided to create the dataset, making use of the Twitter API to collect data from telecommunication companies. The kind of data that we are most interested in are mentions from other users to the company profile, because that is where the complaints come from. We decided to monitor two Brazilian companies from 12/16/2016 to 01/11/2017 to obtain this data. From that, 15,991 tweets were collected and 3,000 were randomly selected to compose our corpus.

These 3,000 tweets were then manually labeled by one person, according to our three-label class system (Positive, Negative or Critical). As mentioned before, the Critical class is used for the most pressing tweets, which we define as the tweets that might damage more the company's image. The criteria to consider a tweet critical involved mentions of out-of-service times, rage, cursing, threats of reporting the companies, among others, and combinations of those. From that, we are able to identify the most helpless clients and, from this characterization, direct customer support to them.

Table I: Labeling system

Label	**Characteristics**	**Example (translated)**
Positive	appreciation; neutral	"Thank you for fixing my Internet, just don't do it again"
Negative	complaint; impatience	"Hey, are you really not going to answer my DM??"
Critical	threat; curse; anger	"I'm so angry, I just want to cancel my account, don't make me sue you"

Afterwards, we need to prepare the data for training. For this, we used known language pre-processing techniques, such as tokenization, stop words removal, case lowering and filtering of irrelevant information (URLs, Twitter mentions, retweet indicators, etc.). Stemming was also considered as part of the pre-processing, but its inclusion did not present a relevant impact on the overall performance. 861 tweets were labeled as Positive, 1,413 as Negative and 685 as Critical. In Table I, the main characteristics we used to guide the labeling process are listed, along with an example tweet for each label. Of course the labels given to the messages, especially the Critical one, are completely dependent on the application domain, and thus a different set would have been considered critical if the domain had been different from telecommunications.

We then vectorize the data, since machine learning algorithms cannot deal with natural language in the form of strings of characters as well as when encoded by numbers. For LSTM and BLSTM, we used a word embedding layer - created through a word2vec model - which is a special input layer, that is fed each tokenized tweet. It converts each word token into its embedded form, reading the tweet as a stream of vectors. It offers a faithful representation of the words' semantics through its location inside the vector space, boosting the accuracy of sentimental analysis tasks [Socher et al. 2013] by letting it calculate directly on a word's meaning, while the LSTM/BLSTM network itself is able to reason with the text structure.

For the traditional machine learning approaches, a process where a relevance score is assigned to each word in the corpus using TF-IDF (Term Frequency-Inverse Document Frequency) was applied. For each document – a tweet in our case – in the corpus, a vector of TF-IDF scores corresponding to the words contained in it is computed and used as the set of features for training and testing the classifiers.

It is important to notice a difference between the vectorization used in the LSTM and BLSTM and that used in the other more traditional machine learning approaches. Because the Deep Learning techniques are fed a word at a time, the vectorization is done word-by-word, as opposed to the other models, which attempt to classify the vectorization of whole tweets. Hence, it were chosen different models that would draw an appropriate representation of such data for each method; the word2vec model for the deep learning approaches and the TF-IDF model for the others.

Additionally, if one attempted to use the word2vec to vectorize whole tweets - by taking an average or a sum of the vectorized words, for example - one would have to study the viability of this altered embedded model, which is out of this work's scope. The same thing would apply if one tried to use the TF-IDF vectorization of the tweet to infer the embedding of singular words. Finally, trying to modify the traditional models to analyze the sentiment based on the classification of singular words would not be as realistic as to how these models are effectively employed. Hence, the embedding was kept different for both types of approaches.

The next step after vectorization is to train the machine learning models and the LSTM/BLSTM networks. As mentioned before, we do this with a more classic approach and also in two other ways to try mitigating the impact of the problem being multi-classed. For the first approach, we train the models with the entire dataset as they are originally classified, into the three aforementioned classes. The output of these models, for each tweet in the testing partition, is either Positive, Negative or Critical. A secondary approach turns the multi-class problem back into a binary classification by merging the Positive and Negative classes. The whole dataset is revised and every tweet is relabeled into a class 0, if it is originally Positive or Negative, or class 1, if it is originally Critical. The models are trained according to this new labeling and the outputs are binary, either Non-Critical or Critical. A third approach is taken, where we divide the multi-class problem into two sequential binary problems. For this, every model is replaced by two models. The first model is trained with a relabeled version of the dataset where Negative and Critical are combined into a class 1, while the Positive is a class 0. The second model is trained with a relabeled and edited version of the dataset where Positive tweets are removed, Negatives become class 0 and Criticals become class 1. This means that the first model is trained to perceive a tweet as being Positive or not, while the second model is trained to perceive if a tweet, being Non-Positive, is Critical or not. In this approach, the output for a testing tweet is two binary values. If the first output is 0, the tweet is considered Positive no matter the second output. Otherwise, the tweet is considered Negative or Critical whether the second output is 0 or 1, respectively.

Finally, the output provided by the trained classifiers on real data, whatever approach is chosen, is used as a decision support mechanism on what clients are the most impacted and in need of immediate assistance. Specially, for models like LSTM, or any probability based model, the outputs for a specific tweet can be given as a value between zero and one for each possible class, describing the probability of said tweet belonging to that class. This means that, if we calculate each tweet's probability of belonging to the Critical class, we can provide the decision support mechanism with a ranking of Critical tweets, which gives a recommended order of tweets to prioritize first.

4. EXPERIMENTAL EVALUATION

We now present how the LSTM and its bidirectional variant perform against the other, more traditional machine learning approaches. We also show how different ways of addressing the problem showed different results, along with a respective example of estimated criticality ranking.

4.1 Setup

The experiments were conducted on an Intel Core I5 3.1GHz running Ubuntu 16.04 LTS, with 8GB of RAM. To implement the traditional machine learning approaches, we used Python 3.4 with SciKit Learn 0.18.1. For the implementation of both the LSTM and the bidirectional LSTM we relied on Keras [Chollet 2015].

4.2 Experiments

For validation purposes, a K-Fold cross-validation, with K being 3, is used for all three classification approaches - the original three-classed, the Critical vs Non-Critical and the two sequential binary classifications. This gives us four versions of the problem, and four sets of metrics to analyze further ahead.

After the pre-processing steps mentioned, we applied the dataset to the models. It was observed that some models performed poorly on default parameters. For that reason, some parameters were adjusted. For example, in K-Nearest Neighbors, the parameter K was adjusted to select the optimal value, by using a validation set and testing different values, comparing the F-Score. On SVM, the different kernel types (Linear, RBF, Polynomial and Sigmoid) were tested and the *gamma* factor was varied, the same way as KNN. Decision Tree had the node split criterion changed between Information Gain and Gini Impurity. The amount of trees was changed in Random Forest to one that optimized performance.

The LSTM network's optimal architecture, concerning quantities of layers, units and nodes, was decided mainly through trial and comparison. Dropout rates were obtained similarly. The comparison was made by getting the average metrics of the LSTM model when used in all of the three established approaches. The original multi-class approach was run twice, once with the holdout and another time with the K-Fold, while the other two approaches were run with K-Fold only. The decided architecture, after this process, consisted of three LSTM layers, each with 200 units and dropout rate of 0.3, and one dense layer with three neurons.

Figure 2 depicts the architecture employed for the original three-classed approach. Initially, the tokenized tweet is fed to the word embedding layer (WE). Each word w_i is thus embedded in an input vector x_i. The inputs are sequentially processed by the LSTM layers, whose output passes through a dense activation layer. The three final outputs are the tweet's pertinence levels to each class. For the other two approaches, only the output is changed, and these changes are illustrated in figures 3 and 4. For the Critical vs Non-Critical problem, output was turned into binary. As for the double binary classification, the output of the model was turned and two copies of the model were trained.

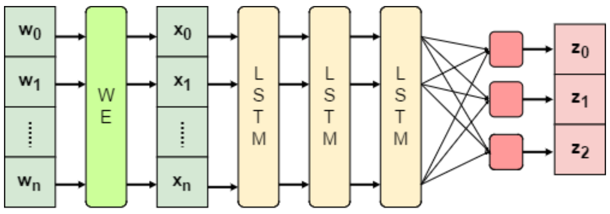

Fig. 2: Architecture of the LSTM network employed

The embedding layer of the network was initialized by a generator model, trained with the aid of the set of words encountered in the Portuguese Wikipedia. The adopted procedure was the skip-gram model proposed in [Bojanowski et al. 2016], which takes word morphology into account. The skip-gram is an instance of the Word2vec models, a group of embedding generating models which focuses on spatially approximating words with common contexts [Mikolov et al. 2013].

The optimally initialized embedding layer was then given freedom to learn the embedding of new words encountered specifically inside our model's domain, while also changing the vectorization of known words if needed.

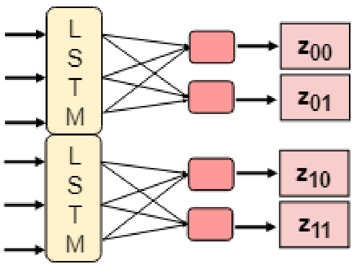

Fig. 3: Double binary approach

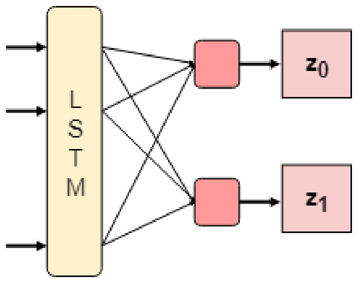

Fig. 4: Critical vs Non-Critical approach

4.3 Results

In order to properly evaluate the obtained results, we relied on the following metrics: overall accuracy; precision, recall and F-score of the Critical class. For the specific problem presented in this article, the results of the Critical class are more important than correctly classifying all classes, because we are interested in prioritizing service exclusively to the customers in more pressing situation.

Table II: Results on the critical class for the original approach using K-Fold

	Precision	**Recall**	**F-score**
Naive-Bayes	0.690089	0.648190	0.628975
KNN	0.650930	0.615068	0.596376
Decision Tree	0.650864	0.642446	0.643054
SVM	0.724625	0.692803	0.682781
Random Forest	0.702691	0.686492	0.686336
Ensemble	0.675867	0.671960	0.669733
LSTM	**0.940357**	**0.937918**	**0.937829**
BLSTM	0.87551288	0.86817094	0.86730047

Table III: Results on the critical class for the Critical vs Non-Critical approach using K-Fold

	Precision	**Recall**	**F-score**
Naive-Bayes	0.89720827	0.38849018	0.54100611
KNN	0.94941077	0.19597855	0.32341771
Decision Tree	0.76151713	0.71689671	0.73642363
SVM	0.8983253	0.54976422	0.68139584
Random Forest	0.88542634	0.61842913	0.72791425
Ensemble	**0.97229611**	0.15891284	0.2730753
LSTM	0.949345	**0.9442913**	**0.94636127**
BLSTM	0.92703123	0.9263132	0.92633846

Table IV: Results on the critical class for the double binary approach using K-Fold

	Precision	**Recall**	**F-score**
Naive-Bayes	0.70535362	0.63805229	0.60710304
KNN	0.69205499	0.61169476	0.57087591
Decision Tree	0.66042623	0.64030709	0.64329833
SVM	0.72597711	0.68401594	0.66945266
Random Forest	0.7013863	0.68885345	0.68701587
Ensemble	0.69526861	0.63478614	0.60928012
LSTM	**0.93717836**	**0.93633896**	**0.93641974**
BLSTM	0.87868543	0.87256512	0.87258475

The results regarding the classification of the Critical class for all approaches are shown in tables II to V. Because the F-score is the harmonic mean of precision and recall, methods with good precision but low recall, like Naive-Bayes, end up having a lower F-score compared to methods which have these two metrics more balanced out. SVM, a method that tends to produce the best results in related works, had real close performance metrics to Random Forest, and was surpassed by both LSTM and BLSTM. As expected, the LSTM and the BLSTM far outperform the traditional machine learning approaches with an F-score of, at least, approximately 0.86 for the BLSTM.

The overall accuracy of each classification method, under each classification problem, can be seen on Figures 5 through 7. KNN, despite all efforts in adjusting the number of neighbors to an optimal number, still performed poorly compared to the other methods. SVM and Random Forest once again presented good results, but with LSTM outperforming both in accuracy in the original problem.

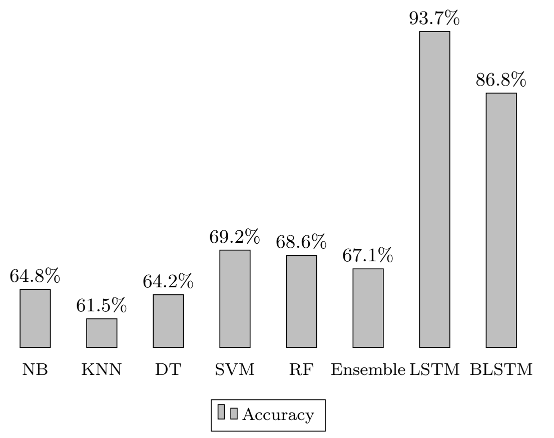

Fig. 5: Overall accuracy for the original three-classed approach using K-Fold

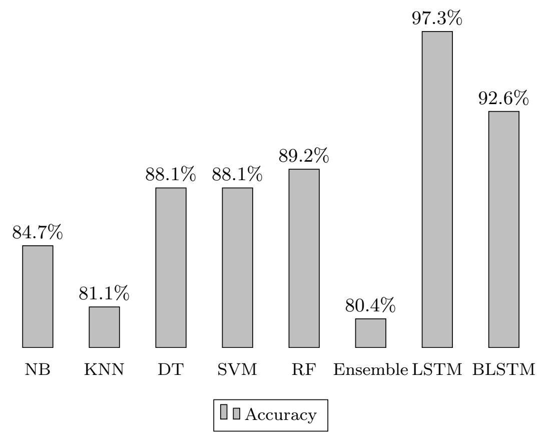

Fig. 6: Overall accuracy for the Critical vs Non-Critical approach using K-Fold

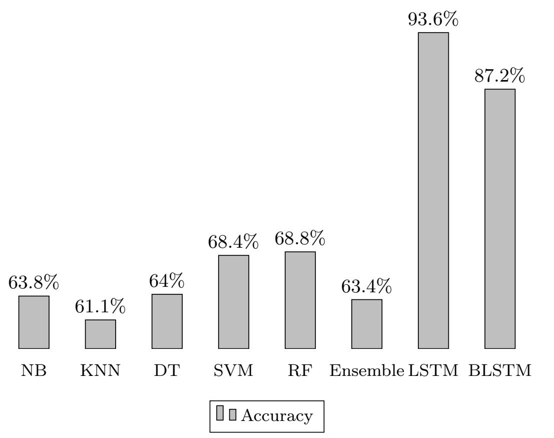

Fig. 7: Overall accuracy for the double binary approach using K-Fold

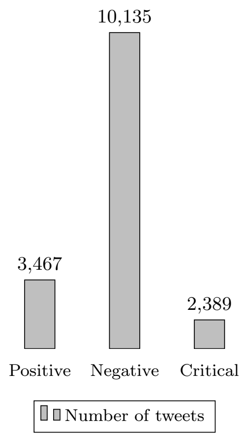

Fig. 8: Class dispersion

Also, for further analysis, a new model with all labeled tweets was trained and the tweets from the original dataset were tested against it. In Figure 8, a chart illustrates the distribution of the classified tweets over the three classes. As expected, the number of negative tweets is considerably higher than the positive ones, which validates our initial claim that customers are more inclined to write complaints rather than compliments on social networks. More importantly, the number of critical tweets is about 25% of the negative ones, which is much more realistic to manage from the customer service point of view. This means that the creation of the Critical class has made it possible to break down the large and unfeasible to handle Negative class, thus allowing service to be directed to a smaller and more relevant set of clients.

Contrary to expectation, the BLSTM did not perform better than the unidirectional LSTM, even though it consistently displayed the second best results in all experiments. A possible reason for this is that the number of epochs used for training was not enough. Also, the network parameters used for the BLSTM were roughly the same as the ones used in the LSTM, in order to compare them. We believe, however, that properly adjusting these parameters would yield better results for the BLSTM.

Lastly, as mentioned, we are able to produce a ranking of critical tweets to provide a prioritizing order. In table V we can see a top 15 ranking of critical tweets as produced by the ternary-classed

LSTM model, trained with K-Fold. Cursing words were censored, and the tweets are in the original language, Brazilian Portuguese. We also provide along with each one a score, which is the probability calculated by the LSTM of said tweet being Critical. We can see that most of those tweets contain swears, heavy critiques, threats and even sarcasm.

Table V: Top 15 ranking of critical tweets according to the LSTM model, on the original approach using K-Fold

	Tweet	**Score**
1	RT @Plinio_Barboza: Eita *****!!! Sério isso @Vivoemrede ?	9.9999809265e-01
2	@Vivoemrede vergonha da *****, comprou a gvt e fez mais *****	9.9999809265e-01
3	@digaoi Vai tomar no **, cadê a ***** da minha internet ?????????	9.9999797344e-01
4	Eita *****!!! Sério isso @Vivoemrede ?	9.9999785423e-01
5	hoje tá **** pra ******* hein, @digaoi	9.9999785423e-01
6	sério @Vivoemrede vão se *****, enfia esse secretário eletronico no **. Serviço ***** o atendimento de vcs, só cai	9.9999773502e-01
7	@KidSimomCSGO @qepcsgo @Vivoemrede @NEToficial Aqui é uma ***** porque não tem concorrência. Mas essa semana tudo muda	9.9999773502e-01
8	@digaoi Pra que serve esta ***** de canal no twitter?	9.9999773502e-01
9	@beaaCRF @digaoi *****, fala não	9.9999773502e-01
10	RT @vanderlkkkwiel: vai toma no ** @Vivoemrede a internet de vcs é uma ***** igual vcs mano, toma no **	9.9999773502e-01
11	EU TE ODEIO DESGRAÇADA @Vivoemrede	9.9999773502e-01
12	RT @Billy_Ddevil: **** que pariu! Como eu odeio internet lenta, obrigado @digaoi pelo serviço de ***** que vcs prestam pra sociedade @Procon	9.9999773502e-01
13	**** ***** mano já comprovei a ***** do pagamento duas vezes nesse ******* e ainda consta que não paguei essa ****** @Vivoemrede	9.9999773502e-01
14	Que dia a @digaoi vai querer resolver essa ***** de wifi ?	9.9999773502e-01
15	@digaoi PIOR PROVEDOR DE INTERNET QUE EXISTE	9.9999761581e-01

5. RELATED WORKS

Sentiment analysis, as defined by [Liu 2015] and [Pang and Lee 2007], is the science that analyzes what are people's opinions and impressions towards a target product or service based on written texts, such as reviews or comments. In this context, social networks are an important source of such content, given that, nowadays, increasingly more businesses have an online presence and interact in real time with their customers.

In sentiment analysis, most works can be classified according to the method they are based on, which is usually either lexicon-based or using machine learning. [Taboada et al. 2011], [Balage Filho et al. 2013] and [Souza and Vieira 2012] are good examples of works that use a thesaurus (lexicon) to identify the polarity of texts written in Portuguese. The main challenge of the lexicon approach is that you have to design an algorithm considering the quality of the lexicon and the application domain. Machine learning approaches, such as ours, have the advantage of not having to concern themselves with how to interpret the words.

The applications of deep learning in the field of sentimental analysis, while new, are increasingly pertinent and promising. As shown by [Rojas-Barahona 2016], different deep networks may hold solid ground on polarity tasks when analyzing large texts, such as reviews. Timmaraju et al. [Timmaraju and Khanna 2015] evidences this potential by showing a model based on recurrent networks able to reach test accuracy levels close to those of featured-engineered models without the need for any handcrafted feature. Furthermore, Liu et al. [Liu et al. 2016] shows possible enhancements to these models, as it provides different architectures of deep recurrent networks yielding improved results through multi-task learning. Rojas et al.[Rojas-Barahona 2016] also notes its viable application to twitter data, should the network and the data be finely tuned and balanced. Indeed, the short length, complex language, abbreviated vocabulary and often implicit inflections, such as sarcasm, on text data are constraints that beg for cautious pre-processing of data, as remarked by [Yuan and Zhou 2015].

Works such as [Lee and Dernoncourt 2016] avoid the text length problem by laying tweets in time-sequences, grouped by user, and classifying tweets while holding information of the ones behind them. This approach manages state-of-the-art results.

Adversities emerging from the language, specifically Portuguese, due to its complexity, have been noted and treated by [Alves et al. 2016]. Even though they utilize Naive Bayes and SVM classifiers, their work demonstrates that sentimental analysis applied to tweets in such adversities can still yield favorable results if pre-processing is carefully taken care of. They manage it through the manual correction of grammar mistakes, unnecessary characters, abbreviations, slang words and non-words. Our approach shows effectiveness even when data is used as-is. Even the most basic pre-processing had no manual interference.

In [Dosciatti et al. 2013], Support Vector Machines are used to identify six different emotions, listed neutral, displeasure, joy, anger, fear, surprise and sadness, in texts from online newspapers. The results were compared with those of two other experiments, using K-Nearest Neighbors and Naive-Bayes.

The approach taken by [Lee and Dernoncourt 2016] falls short when the tweet sequence consists of quick replies with high variation of tone and subject, as happens in customer support exchanges, leaving us with single-tweet analysis. And, unlike [Alves et al. 2016], we focus on making the least amount of text correction possible while maintaining test accuracy levels. We chose instead to train our model to adapt to this new linguistic environment by letting the network's word embedding layer more flexible. We followed the conclusion of [Tang et al. 2014], which states that a word embedding concerned solely with syntactic context performs worse than one unifying syntax and sentiment information. We therefore let the language variations be perceived as sentiment information by the network and observe that it satisfactorily reduces hindrances such as slang words, typos and sarcasm.

The work [Amora et al. 2017] presents an evaluation of several machine learning techniques, including LSTM aiming to identify complaints on telecom operators' social media using three classes, Positive, Negative and Critical. The work uses hold-out for the training step of the models, missing the result depth obtained by other validation techniques, such as cross-validation. It also only compares the results using the same three-class approach, not making use of other classification, like binary classification or other technique, such as [Dosciatti et al. 2013].

6. CONCLUSION

We have presented a study case of several machine learning techniques to prioritize customer service through social networks using sentiment analysis. More specifically, we focus on the differentiation between more robust deep learning models and classic machine learning ones. Additionally, we have experimented with three different classification strategies: the original three-classed, Critical vs. Non-Critical and double binary.

The experiments show that both deep learning techniques employed (LSTM and BLSTM) presented the two best results in terms of accuracy and F-score, with the LSTM still being more efficient than the BLSTM. The visible discrepancy, in almost all cases, between the deep learning methods and the classic machine learning ones imply that their ability to make inferences from the context is useful even when considering small texts such as tweets.

Also, we notice that the original three-classed and the double binary approaches performed relatively similar to each other. The Critical vs Non-Critical strategy, however, worked best in this case, with the LSTM achieving a surprising accuracy of 97.3%. The results for the F-score metric were also slightly higher using this approach.

In the future, we plan on doing similar experiments using customer feedback from other social networks in which longer texts are supported, as the short length of tweets may present a problem for methods that rely on context. Also, we intend to explore different configurations for the deep learning methods (i.e. training with more epochs and modifying network parameters).

REFERENCES

Alves, A. L. F., de Souza Baptista, C., Firmino, A. A., de Oliveira, M. G., and de Paiva, A. C. A spatial and temporal sentiment analysis approach applied to twitter microtexts. *Journal of Information and Data Management* 6 (2): 118, 2016.

Amora, P., Teixeira, E., Lima, M., Amaral, G., Cardozo, J., and Machado, J. A deep learning approach to prioritize customer service using social networks, 2017.

Balage Filho, P. P., Pardo, T. A., and Aluísio, S. M. An evaluation of the Brazilian Portuguese LIWC dictionary for sentiment analysis. In *Proceedings of the 9th Brazilian Symposium in Information and Human Language Technology (STIL)*. pp. 215–219, 2013.

Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*, 2016.

Chollet, F. Keras, 2015. Accessed: 2017-05-22.

Dosciatti, M. M., Ferreira, L. P. C., and Paraiso, E. C. Identificando emoções em textos em português do brasil usando máquina de vetores de suporte em solução multiclasse. *ENIAC-Encontro Nacional de Inteligência Artificial e Computacional. Fortaleza, Brasil*, 2013.

Elman, J. L. Finding structure in time. *Cognitive science* 14 (2): 179–211, 1990.

Graves, A., Mohamed, A.-r., and Hinton, G. Speech recognition with deep recurrent neural networks. In *Acoustics, speech and signal processing (icassp), 2013 ieee international conference on.* IEEE, pp. 6645–6649, 2013.

Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. R. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

Hochreiter, S., Bengio, Y., Frasconi, P., and Schmidhuber, J. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies, 2001.

Hochreiter, S. and Schmidhuber, J. Long short-term memory. *Neural computation* 9 (8): 1735–1780, 1997.

Lee, J. Y. and Dernoncourt, F. Sequential short-text classification with recurrent and convolutional neural networks. *arXiv preprint arXiv:1603.03827*, 2016.

Liu, B. *Sentiment Analysis - Mining Opinions, Sentiments, and Emotions.* Cambridge University Press, 2015.

Liu, P., Qiu, X., and Huang, X. Recurrent neural network for text classification with multi-task learning. *arXiv preprint arXiv:1605.05101*, 2016.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

Mozer, M. C. A focused backpropagation algorithm for temporal. *Backpropagation: Theory, architectures, and applications*, 1995.

Olah, C. Understanding lstm networks, 2015. [Online; accessed 2017-04-26].

Pang, B. and Lee, L. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval* 2 (1-2): 1–135, 2007.

Plank, B., Søgaard, A., and Goldberg, Y. Multilingual part-of-speech tagging with bidirectional long short-term memory models and auxiliary loss. *arXiv preprint arXiv:1604.05529*, 2016.

Rojas-Barahona, L. M. Deep learning for sentiment analysis. *Language and Linguistics Compass* 10 (12): 701–719, 2016.

Socher, R., Perelygin, A., Wu, J. Y., Chuang, J., Manning, C. D., Ng, A. Y., Potts, C., et al. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the conference on empirical methods in natural language processing (EMNLP)*. Vol. 1631. Citeseer, pp. 1642, 2013.

Souza, M. and Vieira, R. Sentiment analysis on twitter data for portuguese language. In *International Conference on Computational Processing of the Portuguese Language.* Springer, pp. 241–247, 2012.

Taboada, M., Brooke, J., Tofiloski, M., Voll, K. D., and Stede, M. Lexicon-based methods for sentiment analysis. *Computational Linguistics* 37 (2): 267–307, 2011.

Tang, D., Wei, F., Yang, N., Zhou, M., Liu, T., and Qin, B. Learning sentiment-specific word embedding for twitter sentiment classification. In *ACL (1)*. pp. 1555–1565, 2014.

Timmaraju, A. and Khanna, V. Sentiment analysis on movie reviews using recursive and recurrent neural network architectures. *Semantic Scholar*, 2015.

Yuan, Y. and Zhou, Y. Twitter sentiment analysis with recursive neural networks, 2015.