Sentiment Analysis on Maltese using Machine Learning

Alexiei Dingli and Nicole Sant

Department of Intelligent Computer Systems
Faculty of ICT
University of Malta
Email: alexiei.dingli@um.edu.mt, nicsant@gmail.com

***Abstract*—Sentiment analysis refers to the task of analysing a piece of text and classifying it according to the overall sentiment of the opinion being expressed. In this paper, we present a novel, supervised context based, machine learning system capable of performing such a task for text written in Maltese. Our system consists of two components both capable of performing classification of Maltese text at a context window level, yet while one follows the more traditional approach where features are hand-crafted and passed on for classification, the other performs unsupervised feature extraction and makes use of a deep learning algorithm. Through experimentation we determined that a Random Forest classifier in conjunction with 80% of our dataset for training and a four word context window achieved the best results, and were successful in achieving an accuracy of 62.3%.**

***Index Terms*—Sentiment Analysis; Maltese; Machine Learning; Deep Learning**

I. Introduction

Given today's easy accessibility to the web, the choice of medium most preferred for individuals who wish to express their opinion about one matter or another is that of online facilities, such as discussion forums or social media websites like Facebook. However, for entities such as companies conducting marketing research about their product, this introduces the problem of having to manually track, study and analyse the vast amount and diversity of people's opinions over the entire web, which requires a great deal of manual labour. The use of a sentiment analysis tool can help shoulder this burden since it can automatically classify any given piece of text, based on the polarity of the opinion expressed towards an entity being discussed within that text.

Due to the fact that in the recent years the Maltese population has adjusted and adapted to the use of online communication for sharing their experiences and opinions, this problem is now faced on a local basis. Motivation for this research stems from the observation that while numerous amounts of solutions have been developed to carry out the task of sentiment analysis for the English language, no studies have ever been carried out in an attempt to design such a system for Maltese. In light of this identified problem, and due to the fact that technology has become a dependable source of communication in our everyday lives, we decided to implement a Maltese based sentiment analysis tool in order to try and minimize the gap between our native language and technology. This system is capable of performing classification at a context window level by means of both traditionally used algorithms for the task at hand, which make use of hand-crafted features extracted from the text, as well as by utilizing a deep learning algorithm. Our main contributions by means of this research were the construction of a unique and novel system capable of performing sentiment analysis for text written in Maltese, as well as the composition of the first Maltese corpus consisting of manually labelled text.

This paper is structured in the following way. Section 2 highlights the aims and objectives behind this paper. The following section provides us with general information about the various approaches used for sentiment analysis. Details on the design and implementation of the artefact can be found in section 4, which is followed by an in-depth explanation of the tests carried out in order to evaluate our system in section 5. Finally, we provide a summary of the work carried out and ideas for possible improvements and future work in section 6.

II. Aims & Objectives

We aim to experiment with a number of different classifiers, particularly those commonly used throughout literature, in order to determine the most suited classification algorithm for Maltese text within our system. This shall be achieved through the design and implementation of a sound methodology, based on state-of-the-art solutions designed for English, consisting of the use of manually designed preprocessing techniques and extraction of hand-crafted features, when necessary, as well as that of third-party algorithms. Furthermore, we aim to establish the optimal parameters used to obtain the highest accuracies possible with our system by evaluating the chosen algorithms using different parameters, including context window size and training set size. Finally, since this is a first attempt at solving this problem, through the preparation and preprocessing of the dataset as well as the evaluation on the aforementioned parameters, we aim to find the best configuration which will help our system surpass a minimum accuracy of 34%, achieved by a random classifier, and possibly reach a target accuracy of 64%, which was obtained through the use of manually designed rules in [1].

III. Related Work

Sentiment analysis refers to the task of classifying a given piece of text, based on the polarity of the opinion expressed by the author within the text itself. This technique can be interpreted as a form of text classification, where the criterion of classification is the attitude expressed in the text rather than the content or topic [2]. The sentiment in question may be the authors overall judgement, mood or evaluation of the topic being discussed in the text [3]. Sentiment analysis is a context sensitive field of study [4] which requires various natural language processing, information retrieval and extraction tasks. It is said to be domain dependent, however generally, the majority of positive and negative opinions expressed by authors maintain a consistent meaning throughout various domains [4].

In order to perform sentiment analysis, we researched both lexicon based approaches as well as machine learning approaches. The former approach, which utilizes sentiment lexicons and scoring methods to perform classification, was used by the likes of Turney in [5] who achieved a 74% accuracy, as well as Dave et al. in [6] who increased this value to 76% with a similar system. The majority of solutions involve the use of machine learning techniques, such as those proposed in [1], [4], [7], and [8]. These systems were built using the traditional machine learning approach involving manual preprocessing and feature extraction methods, as well as algorithms including Naive Bayes, Maximum Entropy, SVMs and Decision Tree algorithms. The highest accuracy amongst these solutions was that of 87.4% achieved in [7] through the use of the Maximum Entropy classifier. Finally, we reviewed machine learning systems which utilize deep learning classifiers. The current state-of-the-art deep learning system is that found in [9], which by means of a Recursive Neural Tensor Network (RNTN) achieved an 88.5% accuracy. This was an improvement over the 85.4% accuracy achieved with the same classifier in [10]. Other researchers opted to use a Deep Belief Network classifier, such as in [11], where a 75.6% accuracy was achieved, while others implemented their own custom deep learning network, as done in [12] and [13].

IV. Methodology

In this section, we shall discuss the overall approach taken towards solving the problem for sentiment analysis in Maltese, as well as go into further detail regarding the system design and the two components which comprise our system.

A. Proposed Approach

As mentioned earlier, our solution is a supervised, machine learning context based system which performs classification at a context window level rather than at sentence or single word level. We opted for a context based approach rather than a Bag-of-Words approach due to the importance of context when determining the sentiment of specific words, since the surrounding words may change the overall polarity of the word itself. We incorporated context into our solution in two ways. Firstly, when required, we applied a Part-of-Speech (POS) tagger to the text as a whole before redundant data is removed, such that each word is assigned a POS tag within the context of its surrounding words. Secondly, we broke down each sentence of each piece of text to be classified into context windows, based on a predefined context window size, and trained our classifiers on these windows. For example, the sentence ”Jiena ma rridx niekol il-frott.” (Translated to ”I do not want to eat the fruit.”) and a context window size of three would produce the following context windows:

Jiena ma rridx
ma rridx niekol
rridx niekol il-frott.

The idea behind this decision was to be able to determine the different sentiments expressed within a sentence and classify them within the context of their surrounding words. While other researchers made use of predefined sentiment bearing expressions to identify the presence of an opinion, such as in [14], this was opted against since it would render our approach domain specific, while we are opting for a more general one. Therefore, the use of context windows was adapted from information extraction procedures such as in [15] and our classifiers were trained on a dataset composed of these context windows in order to learn representations of every possible pattern found within a sentence. The use of such a technique complements our idea of a general approach towards sentiment analysis since it is much easier for a classifier to generalise parts of a sentence rather than the sentence as a whole. Such an approach also helps reduce the risks brought about by sparseness and overfitting in the dataset.

B. System Design

Our system is comprised of two components, the Custom Feature Component (CFC) and the Unsupervised Feature Component (UFC). The CFC was designed such that it follows the traditional sentiment analysis process where it extracts hand-crafted features and passes them on to conventionally used algorithms for classification. On the contrary, the UFC was designed to perform unsupervised feature extraction for classification by a deep learning algorithm. The reason for including both components in our approach is to enable us to evaluate how one component fares against the other. The following sections shall provide an insight as to how each component was implemented.

C. Custom Feature Component

The first step within this process consists of parsing the provided dataset entry by entry from an XLSX file and passing each entry on for preprocessing. The preprocessing techniques we employed were chosen due to their success in previous work, such as in [1] [2] [16] [17] [4] [5]. We first tokenized each dataset entry into separate sentences and further passed each sentence on to the POS tagger, as mentioned previously. We made us of the MLSS POS tagger found in [18]. By parsing the POS tagger output we were able to further tokenize each sentence into separate word tokens. Finally, based on the resulting POS tag for each word, we removed uninformative word tokens including numbers, proper nouns, punctuation and determiners.

We next passed on the remaining word tokens for feature extraction, where we extracted four features per token. These are the unigram, the part-of-speech, the negation presence and the stem word. Once again the features used were chosen based on their performance in similar systems, particularly in [7], and include unigram value, that is, the value of the word token itself, the POS tag of the word extracted from the MLSS tagger, negation presence, which is a binary feature indicating whether a verb has been negated or not, and the stem word of each token. We made use of the stemmer found in [19]. Unfortunately, the sentiment was not included since we were unable to include a feature indicative of the sentiment associated with each word token due to the lack of a translator for the Maltese language, rendering us unable to use the SentiWordNet tool. Once all required features were extracted, we compiled the feature vectors in order to perform the final classification step.

As mentioned earlier, classification within our system takes place at a context window level and a feature vector is created to represent each context window within a sentence. To do this, we predefined a context window size and iterated through every sentence of every dataset entry, and built a feature vector for every context window within that sentence. Therefore, since a word token within a sentence is now a quadruple of features, a feature vector for a given context window size will consist of an amount of word tokens, hence an amount of quadruples, equivalent to that predefined context window size, as well as a sentiment value. Due to time restrictions we were unable to have each context window individually labelled by our annotators and so each context window took the sentiment label of its originating dataset entry, introducing a noise value of 33.2% in our approach.

Finally these feature vectors were passed on to classification algorithms for training. We opted to include Naive Bayes, Maximum Entropy, Support Vector Machine (SVM), a Decision Tree and Random Forest classifier. We ran these algorithms by means of the WEKA Explorer GUI. These classifiers were chosen due to their popularity throughout the literature reviewed as well as their successful performance, as can be seen in [4], [7], [3] and [8].

D. Unsupervised Feature Component

In order to implement this component, we made use of the Deep Learning for Java (DL4J) library [20] and based the design of this constituent on an example of sentiment analysis for an English corpus found on the library's website [21].

The first step in the UFC process consists of parsing the provided dataset, tokenizing each dataset entry down to context window level, and storing the resulting windows in a CSV file. Once again, time limitations restricted us from manually labelling each context window and so the 33.2% noise rate was re-introduced, with each context window taking the sentiment label of the overall dataset entry. The resulting dataset, consisting of the labelled context windows, was separated into training and test sets. Since deep learning algorithms handle preprocessing of text automatically, we did not carry out any preprocessing ourselves, but rather we created a data pipeline, which would be used to iterate through the latter dataset and pass on the data directly for feature vector creation. This pipeline, known as a label-aware sentence iterator, simply takes each context window within the dataset, together with its label, and passes it on for feature extraction.

The DL4J implementation of the Word2Vec algorithm was used to perform the unsupervised feature extraction. This algorithm is a neural network which processes text before handing it over to deep learning algorithms for training by creating feature vectors consisting of numerical values to represent the text. This algorithm works by creating a word vector for each individual word based on its context, usage and past appearances. A lookup table of these word vectors is constructed and used to compose feature vectors for phrases by averaging out the vectors of the individual words. The resulting feature vectors are finally passed on to our deep learning algorithm for training.

The algorithm chosen for use within this component is a Deep Belief Network (DBN), which we configured based on the example in [21] and similar networks used throughout literature. Our final configuration states that our DBN consists of three layers, each representing a Restricted Boltzmann Machine and each trained 50 times, as in [11]. We initialized the weights of the network using a uniform distribution, and used the ”tanh” function as the activation function for each node within each layer. We also took a number of measures to help the network avoid overfitting. These include constraining the gradient, using AdaGrad (a feature-specific learning rate optimization technique), integrating the L2 Regularization technique (reduces overfitting by adding a complexity penalty to the loss function), and performing Dropout (a bootstrapping technique which forces the network to learn different representations of the data by randomly dropping different features within a feature vector). Finally, we configured the final, output layer where we used the softmax function to produce the final classification. Once the network was completely configured, we used our data pipeline to pass in data from the training dataset in order to train our classifier.

Our initial aim was to also include an RNTN classifier within this component, however the model within the DL4J library was being improved upon at the time of implementation and so we were unable to configure it correctly for use within our research.

V. Evaluation

In this section, we shall discuss the compilation process of the dataset as well as the various tests carried out to evaluate our system and the respective results obtained by each.

A. *Compilation of Dataset*

In order to compile our dataset, we extracted 900 microblogs written purely in Maltese, as is required by both components of our system, from various online Maltese gazettes, such as the Times of Malta and The Malta Independent amongst others. After collection, we proceeded to spell check each microblog in order for the tools used within our system, such as the POS tagger and the stemmer, to produce the optimal results possible. Finally, manual labelling of the dataset was carried out by three individuals, who were asked to label each microblog as positive, negative or neutral, based on the sentiment, if any, expressed, in an objective manner, irrelevant of their political opinions and by understanding the text in the most literal sense possible. The inter-annotator agreement value was 51.22% and the final distribution of the dataset is as shown in Fig. 1.

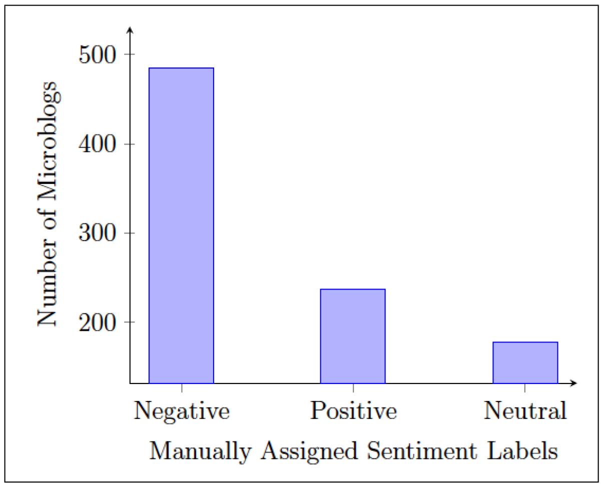


Fig. 1. Dataset Distribution

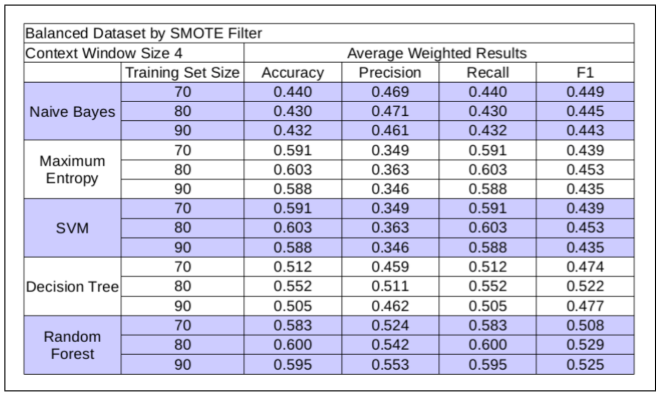
B. *Experiments*

In the experiments carried out, we followed the approach taken by [16], [4], [7], [3], and [8], where we evaluated algorithms from both components against each other using standard information retrieval measures, including accuracy, precision, recall and F-measure, also used in the above mentioned literature. Due to the fact that the DL4J library did not permit us to evaluate the deep learning model using the cross-fold validation technique, we used a separate test dataset for evaluation so as to directly compare algorithms in the CFC with that in the UFC. Through experimentation we evaluated our dataset, as well as the effects of increasing both the context window size as well as the training set size, and finally the chosen algorithms.

1) Evaluation on Dataset Distribution: As can be seen in Fig. 1 above, a bias towards the negative class is present within our dataset. We attempted to balance out this distribution by both reducing the amount of negative examples as well as applying the Synthetic Minority Over Sampling Technique (SMOTE) filter, which reduces bias by under-sampling the majority class and over-sampling the minorty class through the use of synthetic examples. We carried out all experiments mentioned above on all three datasets, and concluded that both attempts to reduce bias were futile, since none of the chosen algorithms were able to overcome the bias, and furthermore always performed better when trained using the original dataset. This dataset was therefore used for the remaining tests.

2) Evaluation on Context Window Size: We experimented with a number of different context window sizes in order to determine the effects of including more words, therefore more context, within a context window. We conducted the remaining experiments, that is, those regarding training set size and classifiers, using context window sizes of 3, 4, and 5 words. The results obtained allowed us to come to the conclusion that a context window size of 4 words was the optimal parameter within our system since it achieved the best results in the majority of the remaining tests.

3) Evaluation on Training Dataset Size: The aim of this test was to evaluate whether increasing the size of the training dataset improved performance of the classifiers. We experimented with training dataset sizes of 70, 80 and 90% of our original dataset, in conjunction with the optimal 4 word context window size. Results in Fig. 2 showed that using 80% of our dataset for training yielded the best performance for most classifiers.



Balanced Dataset by SMOTE Filter						
Context Window Size 4		Average Weighted Results				
	Training Set Size	Accuracy	Precision	Recall	F1
Naive Bayes	70	0.440	0.469	0.440	0.449
	80	0.430	0.471	0.430	0.445
	90	0.432	0.461	0.432	0.443
Maximum Entropy	70	0.591	0.349	0.591	0.439
	80	0.603	0.363	0.603	0.453
	90	0.588	0.346	0.588	0.435
SVM	70	0.591	0.349	0.591	0.439
	80	0.603	0.363	0.603	0.453
	90	0.588	0.346	0.588	0.435
Decision Tree	70	0.512	0.459	0.512	0.474
	80	0.552	0.511	0.552	0.522
	90	0.505	0.462	0.505	0.477
Random Forest	70	0.583	0.524	0.583	0.508
	80	0.600	0.542	0.600	0.529
	90	0.595	0.553	0.595	0.525

Fig. 2. Results of Evaluation on Balanced Dataset by the Synthetic Minority Over-Sampling TEchnique (SMOTE) Filter with Context Window Size 4

4) Evaluation on Classifier: In light of the above conclusions drawn, we evaluated the performance of the classifiers based on the results obtained when using 80% of the dataset for training and a 4 word context windows. The results are shown in Table I below.

Compared to their performance throughout literature, the algorithms used within both the CFC and the UFC performed very poorly when trained on a Maltese corpus, and were unsuccessful in overcoming the bias present within the dataset. It was also noticed that algorithms used within the CFC always outperformed our DBN, allowing us to conclude that the algorithms which required hand-crafted features were more successful in classifying Maltese text and therefore more suitable for use within our system. Finally, it was observed that the Random Forest classifier achieved the best results, rendering it the most suited classifier for our corpus with a maximum accuracy of 62.3%, and allowing us to surpass the 34% baseline accuracy while also coming close to reaching the 64% target.

TABLE I. RESULTS OF CLASSIFIERS

Classifier	Accuracy	Precision	Recall	F1
Naive Bayes (NB)	0.603	0.363	0.603	0.453
Maximum Entropy (ME)	0.603	0.363	0.603	0.453
SVM	0.603	0.363	0.603	0.453
Decision Tree (DT)	0.546	0.489	0.546	0.507
Random Forest (RF)	0.623	0.579	0.623	0.547
DBN	0.285	0.200	0.335	0.250

5) Evaluation on Features in CFC: By means of WEKA's attribute evaluator, we were able to determine which features used within the CFC were most helpful during classification for each context window size. From the obtained results, we concluded that unigram and stem word values were the most informative, particularly the value of the last unigram in the window, followed by that of the first word in the window. POS tags and negation presence proved to be uninformative in all cases.

VI. Conclusion & Future Work

In this research, we implemented a novel machine learning, context based system capable of performing sentiment analysis for text written in Maltese. Through experimentation we concluded that a Random Forest classifier was the best classifier for use within our system when used in conjunction with the optimal parameters, that is, a 4 word context window and 80% of our dataset for training. By means of this setup we were able to achieve a maximum accuracy of 62.3%, surpassing the 34% baseline. Future work for this research includes increasing the dataset and manually labelling each context window separately, while also attempting to split sentences on conjunctions rather than in context windows. The incorporation of a feature indicative of a word's sentiment value is also a worthwhile improvement, together with the experimentation of alternative deep learning classifiers, such as an RNTN.

References

[1] E. Spertus, "Smokey: Automatic recognition of hostile messages," in *AAAI/IAAI*, 1997, pp. 1058–1065.

[2] M. Gamon, "Sentiment classification on customer feedback data: noisy data, large feature vectors, and the role of linguistic analysis," in *Proceedings of the 20th international conference on Computational Linguistics*. Association for Computational Linguistics, 2004, p. 841.

[3] J. Khairnar and M. Kinikar, "Machine learning algorithms for opinion mining and sentiment classification," *International Journal of Scientific and Research Publications*, vol. 3, no. 6, pp. 1–6, 2013.

[4] A. Kamal and M. Abulaish, "Statistical features identification for sentiment analysis using machine learning techniques," in *Computational and Business Intelligence (ISCBI), 2013 International Symposium on*. IEEE, 2013, pp. 178–181.

[5] P. D. Turney, "Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews," in *Proceedings of the 40th annual meeting on association for computational linguistics*. Association for Computational Linguistics, 2002, pp. 417–424.

[6] K. Dave, S. Lawrence, and D. M. Pennock, "Mining the peanut gallery: Opinion extraction and semantic classification of product reviews," in *Proceedings of the 12th international conference on World Wide Web*. ACM, 2003, pp. 519–528.

[7] E. Boiy and M.-F. Moens, "A machine learning approach to sentiment analysis in multilingual web texts," *Information retrieval*, vol. 12, no. 5, pp. 526–558, 2009.

[8] B. Pang, L. Lee, and S. Vaithyanathan, "Thumbs up?: sentiment classification using machine learning techniques," in *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*. Association for Computational Linguistics, 2002, pp. 79–86.

[9] L. Dong, F. Wei, M. Zhou, and K. Xu, "Adaptive multi-compositionality for recursive neural models with applications to sentiment analysis," in *Twenty-Eighth AAAI Conference on Artificial Intelligence*, 2014.

[10] R. e. a. Socher, "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, 2013, p. 1642.

[11] D. Tang, B. Qin, T. Liu, and Z. Li, "Learning sentence representation for emotion classification on microblogs," in *Natural Language Processing and Chinese Computing*. Springer, 2013, pp. 212–223.

[12] C. N. dos Santos and M. Gatti, "Deep convolutional neural networks for sentiment analysis of short texts." in *COLING*, 2014, pp. 69–78.

[13] D. Tang, F. Wei, B. Qin, T. Liu, and M. Zhou, "Coooolll: A deep learning system for twitter sentiment classification," in *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, 2014, pp. 208–212.

[14] J. Ramanathan and R. Ramnath, "Context-assisted sentiment analysis," in *The 25th Annual ACM Symposium on Applied Computing*, 2010, pp. 404–413.

[15] F. Ciravegna, "2, an adaptive algorithm for information extraction from web-related texts," in *In Proceedings of the IJCAI-2001 Workshop on Adaptive Text Extraction and Mining*, 2001, p. organization=Citeseer.

[16] A. Kamal, "Subjectivity classification using machine learning techniques for mining feature-opinion pairs from web opinion sources," *arXiv preprint arXiv:1312.6962*, 2013.

[17] W. Jin, H. H. Ho, and R. K. Srihari, "Opinionminer: a novel machine learning system for web opinion mining and extraction," in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2009, pp. 1195–1204.

[18] U. of Malta, "MLSS: Maltese Language Software Server, Part of Speech Tagger for Maltese," http://metanet4u.research.um.edu.mt/services/MtPOS?wsdl., 2015, [Retrieved: September, 2016].

[19] K.Sultana, "Kurt Sultana - Research: Basic Stemmer for Maltese," http://www.kurtsultana.com/, 2015, [Retrieved: September, 2016].

[20] Skymind, "DL4J - Deep Learning for Java," http://www.deeplearning4j.org/, 2015, [Retrieved: September, 2016].

[21] ——, "Movie Review Sentiment Analysis With Word2Vec, DBNs and RNTNs," http://deeplearning4j.org/sentiment-analysis-word2vec.html, 2015, [Retrieved: September, 2015].