HanSEL: Italian Hate Speech detection through Ensemble Learning and Deep Neural Networks

Marco Polignano
University of Bari Aldo Moro
Dept. Computer Science
via E. Orabona 4, 70125 Bari, Italy
marco.polignano@uniba.it

Pierpaolo Basile
University of Bari Aldo Moro
Dept. Computer Science
via E. Orabona 4, 70125 Bari, Italy
pierpaolo.basile@uniba.it

Abstract

English. The detection of hate speeches, over social media and online forums, is a relevant task for the research area of natural language processing. This interest is motivated by the complexity of the task and the social impact of its use in real scenarios. The task solution proposed in this work is based on an ensemble of three classification strategies, mediated by a majority vote algorithm: Support Vector Machine (Hearst et al., 1998) (SVM with RBF kernel), Random Forest (Breiman, 2001), Deep Multilayer Perceptron (Kolmogorov, 1992) (MLP). Each classifier has been tuned using a greedy strategy of hyper-parameters optimization over the "F1" score calculated on a 5-fold random subdivision of the training set. Each sentence has been pre-processed to transform it into word embeddings and TF-IDF bag of words. The results obtained on the cross-validation over the training sets have shown an F1 value of 0.8034 for Facebook sentences and 0.7102 for Twitter. The code of the system proposed can be downloaded from GitHub: https://github.com/marcopoli/haspeede_hate_detect

Italiano. *L'individuazione di discorsi di incitamento all'odio sui social media e sui forum on-line è una sfida rilevante per l'area di ricerca riguardante l'elaborazione del linguaggio naturale. Tale interesse è motivato della complessità del processo e dell'impatto sociale del suo utilizzo in scenari reali. La soluzione proposta in questo lavoro si basa su un insieme di tre strategie di classificazione mediate da un algoritmo di votazione per maggioranza: Support Vector Machine (Hearst et al., 1998) (SVM con kernel RBF), Random Forest (Breiman, 2001), Deep Multilayer Perceptron (Kolmogorov, 1992) (MLP). Ogni classificatore è stato configurato utilizzando una strategia greedy di ottimizzazione degli iper-parametri considerando il valore di "F1" calcolato su una suddivisione casuale in 5-fold del set di training. Ogni frase è stata pre-elaborata affinchè fosse trasformarta in formato word embeddings e TF-IDF. I risultati ottenuti tramite cross-validation sul training set hanno mostrato un valore F1 pari a 0.8034 per le frasi estratte da Facebook e 0.7102 per quelle di Twitter. Il codice sorgente del sistema proposto può essere scaricato tramite GitHub:* https://github.com/marcopoli/haspeede_hate_detect

1 Introduction and background

In the current digital era, characterized by the large use of the Internet, it is common to interact with others through chats, forums, and social networks. Common is also to express opinions on public pages and online squares. These places of discussion are frequently transformed into "fight clubs" where people use insults and strong words in order to support their ideas. The unknown identity of the writer is used as an excuse to fell free of consequences derived by attacking people only for their gender, race or sexual inclinations. A general absence of automatic moderation of contents can cause the diffusion of this phenomenon. In particular, consequences on the final user could be psychological problems such as depression, relational disorders and in the most critical situations also suicidal tendencies.

A recent survey of state of the art approaches for hate speech detection is provided by (Schmidt and Wiegand, 2017). The most common systems of speech detection are based on algorithms of text classification that use a representation of contents based on "surface features" such as them available in a bag of words (BOW) (Chen et al., 2012; Xu et al., 2012; Warner and Hirschberg, 2012; Sood et al., 2012). A solution based on BOW is efficient and accurate especially when n-grams have been extended with semantic aspects derived by the analysis of the text. (Chen et al., 2012) describe an increase of the classification performances when features such as the number of URLs, punctuations and not English words are added to the vectorial representation of the sentence. (Van Hee et al., 2015) proposed, instead, to add as a feature the number of positive, negative and neutral words found in the sentence. This idea demonstrated that the polarity of sentences positively supports the classification task. These approaches suffer from the lack of generalization of words contained into the bag of words, especially when it is created through a limited training set. In particular, terms found in the test sentences are often missing in the bag. More recent works have proposed word embeddings (Le and Mikolov, 2014) as a possible distributional representation able to overcome this problem. This representation has the advantage to transform semantically similar words into a similar numerical vector. Word embeddings are consequently used by classification strategies such as Support Vector Machine and recently by deep learning approaches such as deep recurrent neural networks (Mehdad and Tetreault, 2016). The solution proposed in this work reuse the findings of (Chen et al., 2012; Mehdad and Tetreault, 2016) for creating an ensemble of classifiers, including a deep neural network, which works with a combined representation of word embeddings and a bag of words.

2 Task and datasets description

The hate speech detection strategy proposed in *HAnSEL* has been developed for HaSpeeDe (Hate Speech Detection) task organized within Evalita 2018 (Caselli et al., 2018), which is going to be held in Turin, Italy, on December 12th-13th, 2018 (Bosco et al., 2018). HaSpeeDe consists in the annotation of messages from social networks (Twitter and Facebook) with a boolean label (0;1) that indicates the presence and absence of hate speeches. The task is organized into three subtasks, based on the dataset used for training and testing the participants' systems:

- **Task 1: HaSpeeDe-FB**, where only the Facebook dataset can be used to classify the Facebook test set
- **Task 2: HaSpeeDe-TW**, where only the Twitter dataset can be used to classify the Twitter test set
- **Task 3: Cross-HaSpeeDe**, which can be further subdivided into two sub-tasks:
  1. **Task 3.1: Cross-HaSpeeDe_FB**, where only the Facebook dataset can be used to classify the Twitter test set
  2. **Task 3.2: Cross-HaSpeeDe_TW**, where only the Twitter dataset can be used to classify the Facebook test set

The Facebook and Twitter datasets released for the task consist of a total amount of 4,000 comments/tweets, randomly split into development and test set, of 3,000 and 1,000 messages respectively. Data are encoded in a UTF-8 with three tab-separated columns, each one representing the sentence id, the text and the class (Fig. 1).

id	text	hs
8	Io votero NO NO E NO	0
36	Matteo serve un colpo di stato.	1

Table 1: Examples of annotated sentences.

3 Description of the system

The system proposed in this work is HanSEL: a system of Hate Speech detection through Ensemble Learning and Deep Neural Networks. We decided to approach the problem using a classic natural language processing pipeline with a final task of sentences classification into two exclusive classes: hate speech and not hate speech. The data provided by task organizers are obtained crawling social network, in particular, Facebook and Twitter. The analysis of the two data sources showed many possible difficulties to face in the case of using approaches based on Italian lexicons of hate speeches. In particular, we identified the following issues:

- **Repeated characters**: many words includes characters repeated many times for emphasizing the semantic meaning of the word. As an example, the words "nooooo", "Grandeeeee", "ccanaleeeeeeeeeeeeeeee" are found in the training of Facebook messages.
- **Emoji**: sentences are often characterized by emoji such as hearts and smiley faces that are often missing in external lexicons.
- **Presence of links, hashtags and mentions**: this particular elements are typical of the social network language and can introduce noise in the data processing task.
- **Length of the sentences**: many sentences are composed by only one word or in general, they are very short. Consequently, they are not expressive of any semantic meaning.

The complexity of the writing style used in hate speech sentences guided us through the idea to do not use an approach based on standard lexicons and to prefer supervised learning strategies on the dataset provided by the task organizers.

Sentence processing.
We decide to represent each sentence as a concatenation of a 500 features word embedding vector and a 7,349 size bag of words for Facebook messages and 24,866 size bag of words for Twitter messages. In particular, the word-embedding procedure used is word2vec introduced by Mikolov (Mikolov et al., 2013). This model learns a vector representation for each word using a neural network language model and can be trained efficiently on billions of words. Word2vec allows being a very efficient data representation in text classification due to its capability to create very similar vectors for words strongly semantically related. The Italian word embeddings used in this work are provided by Tripodi (Tripodi and Li Pira, 2017). The author trained the model on a dump of the Italian Wikipedia (dated 2017.05.01), from which only the body text of each article is used. The corpus consists of 994,949 sentences that result in 470,400,914 tokens. The strategy of the creation of word embeddings is CBOW with the size of the vectors equal to 500, the window size of the words contexts set to 5, the minimum number word occurrences equal to 5 and the number of negative samples set to 10.

We follow the same step of pre-processing applied by Tripodi (Tripodi and Li Pira, 2017) to transform the sentence of the task datasets into word embeddings. In particular, we applied the following Natural Language Processing pipeline:

- Reduction of repeated characters: we scan each sentence of the datasets (both training and test). For each sentence, we obtain words merely splitting it by space. Each word is analyzed, and characters repeated three times or more are reduced to only two symbols, trying to keep intact word that naturally includes doubles.
- Data cleansing: we transformed the words is lowercase and following we removed from each sentences links, hashtags, entities, and emoji

The normalized sentences are consequently tokenized using the TweetTokenizer of the NLTK library [1]. For each sentence we averaged the word2vec vectors correspondent of each token, removing during each sum the centroid of the whole distributional space. This technique is used for mitigating the problems of loss of information due to the operation of averaging the semantic vectors.

The two bags of words (Facebook and Twitter) are, instead, created directly on the sentences without any pre-processing, also if during the tuning of the architecture we had tried some configurations that include bag of words without stop words, with lowercase letters and processed by Snowball stemmer algorithm [2] without obtaining breaking results. The n-gram size considered for the construction of the bag is in the range of 1 to 3. The final representation of each sentence of the dataset is consequently obtained concatenating the word2vec vector and the correspondent bag of words. Sentences too shorts that cannot be transformed into word2vec as a consequence of the absence of all the tokens of the sentence have been classified using only the bag of words representation.

Classification strategy.
HAnSEL is based on a classification process that uses three different classification strategies mediated by a hard majority vote algorithm. A stacking of classifiers with a Random Forest blender

[1] https://www.nltk.org/data.html
[2] http://snowball.tartarus.org/texts/quickintro.html

has also been considered during the design of HAnSEL architecture but, the internal evaluation runs on 5-fold cross-validation of the training set showed us low performances of the approach. This analysis is not detailed more in this work due to the page limitations of it. In order to design the ensemble, we analyzed the performances of some of the most popular classification algorithms for the text categorization task. In particular, we considered:

- Logistic regression with stochastic gradient descendent training (SGD). It has the advantage to be very efficient with large datasets considering, during the training, one instance per time independent by others. It uses the gradient descendent as optimization function for learning the optimal weight of the separation function of the distributional space of items. In literature, it has been successfully used for tasks of text classification, especially with binary classification problems.

- C-Support Vector Classification (SVC). It is the standard Support Vector Machine algorithm applied for the classification task. It is a powerful approach that supports linear and non-linear classification function. Moreover, through the C parameter, it is possible to decide how much the margin of classification could be significant and consequently sensitive to outliers. The implementation is based on libsvm, and we evaluated different configurations of the algorithm: polynomial function with 2 and 3 degree, RBF kernel and different values of the C parameters.

- K-nearest neighbors vote (KNN). This classic and versatile algorithm is based on the concept of similarity among items according to a distance metric. In particular, for an unseen item, the k most similar items of the training set are retrieved, and the class, provided as output, is obtained by the majority vote of the neighborhoods. Despite the simplicity of the algorithm it is often used in tasks of text classification.

- A decision tree classifier (DT). This approach is another popular strategy of classification used especially when it is required to visualize the model. The DT algorithm works splitting items into a different path of the tree according to their feature values. In order to classify an unseen item, the three is navigated until reaching the leaf and then the ration of training items of class k in that leaf is used as a class probability.

- Random forest classifier (RF). It is an ensemble of Decision Trees trained on different batches of the dataset that uses averaging to improve the predictive accuracy and control over-fitting. A typical parameter is the number of threes to use in order to balance the precision of the algorithm and the randomness to obtain a good level of generalization of the model.

- Multi-layer Perceptron classifier (MLP). This model is a classical architecture of a deep neural network. It is composed by one layer of inputs, one layer of linear threshold units (LTU) as output and many hidden layers of an arbitrary number of LTU plus one bias neuron fully connected each other. The weights learned by each neuron (perceptron) are updated through back-propagation using a gradient descendent strategy. Important parameters to configuring are the number of hidden layers, the number of training epochs and the L2 penalty (regularization term) parameter.

We evaluated the performance of the algorithms just described using a default configuration and a 5-fold cross validation over the Facebook training set. Moreover, we set the random seed equal to 42 for obtaining at each run always the same folder subdivision.

Tab. 3 shows the results obtained by the different classification algorithms during their preliminary analysis considering the macro F1 score as in the task specifications. The values obtained do not point out significant statistical differences among the approaches, but we decided to investigate more the top three scored algorithms: SVM with an RBF kernel, Random Forest with 300 trees, MLP with 2,000 hidden layers. In general, we observed that linear algorithms obtain a high score for the task supporting our idea that linguistic features are enough for defining a clear separation among the sentences of hate and not hate speeches. In order to identify an optimal configuration of the algorithms, we trained our models using a greedy search approach. For each algorithm, we performed 100 training runs with pa-

	Not HS			HS				
	Precision	**Recall**	**F1-score**	**Precision**	**Recall**	**F1-score**	**Macro F1**	**Pos.**
Task 1	0,6981	0,6873	0,6926	0,8519	0,8581	0,855	*0,7738*	7
Task 2	0,7541	0,8801	0,8122	0,6161	0,4012	0,4859	*0,6491*	14
Task 3.1	0,7835	0,2677	0,3991	0,3563	0,8456	0,5013	*0,4502*	11
Task 3.2	0,3674	0,8235	0,5081	0,7934	0,3234	0,4596	*0,4838*	8

Table 2: Final scores obtained during the HASpeeDe challenge

Algorithm	**Macro F1 score**
LR	0.780444109
SVC-rbf - C= 1	*0.789384136*
SVC-poly 2 C=1	0.758599844
SVC-poly 3 C=1	0.667374386
KNN - 3	0.705064332
KNN - 5	0.703990867
KNN - 10	0.687719117
KNN - 20	0.663451598
DT	0.68099986
RF-50	0.75219596
RF-100	0.764247578
RF-300	*0.787778421*
RF-500	0.768494151
MLP-1000	0.766835616
MLP-2000	*0.791230474*
MLP-3000	0.76952709

Table 3: Classification algorithms on Facebook training set using 5-fold cross validation.

rameters randomly selected from a range of values preliminary defined. Each run has been evaluated, considering the macro F1 score, on the training set using the same strategy of cross-validation already described before. At the end of the 100 runs the model that achieve the best results has been stored and later used in our final ensemble of classifiers. The final configurations obtained for the three strategies are the following:

- SVC(C=1.76800710431488, gamma=0.1949764030136127, kernel='rbf')
- RandomForestClassifier(bootstrap=False, max_depth=30,max_features='sqrt', min_samples_leaf=2,min_samples_split=2, n_estimators=200, warm_start=False)
- MLPClassifier(alpha=0.5521952082781035, early_stopping=False, hidden_layer_sizes=2220, learning_rate_init=0.001, max_iter=184,solver='adam', warm_start=False)

The models are consequently used in a voting classifier configured for using a hard majority vote algorithm. The ensemble obtains an F1 value of 0.8034 for Facebook sentences and 0.7102 for Twitter using the 5-fold subdivision of the training sets. The implementation of the system has been realized into Python language and using the scikit-learn 0.20 machine learning library [3].

4 Results and discussion

HanSEL has been used for classifying the data provided as a test set for each of the three specialized tasks of HaSpeeDe competition. Tab. 2 shows the final results obtained by our system in the challenge. It is possible to observe that the system well performed for Task 1 and Task 3.2 which involve the classification of Facebook messages. In particular, it emerges that HanSEL performs better fot hate speeches sentences than for not hate speeches probably a consequence of the presence of many clear hate words used in this type of messages such as "sfigati" and "bugiardo" in that category of textual sentences. A symmetrical situation is obtained for Task 2 and Task 3.2 that involves Twitter messages. In this scenario, the significant use of specific hashtags, irony, and entities instead of clear hate words has made difficult the identification of hate speeches. The cross-classification task has, moreover, stressed the generalization of the system. It has been observed that the writing style of the two social networks strongly influences the classification performance, especially when the models are trained on a small training set, as in our case. Finally, the optimization of the models inside the ensemble has been stressed more on the Facebook dataset consequently overfitting on the characteristics of that type of messages. The outcomes achieved for the challenge

[3] http://scikit-learn.org/stable/

allow us to deduce important consideration for further developments of the system. In particular, we consider essential to mix the two datasets in order to allow the models to generalize better considering the two different sources of data. Moreover, extra features regarding hashtags, entities, and links can be helpful for obtaining better results with Twitter messages.

5 Conclusion

The HaSpeeDe competition has been a perfect scenario for developing and testing solutions for the social problem of hate speeches on social media and, in particular, for them in the Italian language. In our work, we presented *HAnSEL* a system based on an ensemble of classifiers that includes the Support Vector Machine algorithm, Random Forests, and a Multilayers Perceptron Deep Neural Network. We formalize messages as a concatenation of word2vec sentence vectors and a TF-IDF bag of words. Results showed the efficacy of the solution in a scenario that uses clear offensive words such as Facebook messages. On the contrary, there is a large margin of improvements for the classification of Tweets. The future direction of the work will surely investigate the use of more data and semantic features for allowing classification methods to create a more general model.

References

Cristina Bosco, Felice Dell'Orletta, Fabio Poletto, Manuela Sanguinetti, and Maurizio Tesconi. 2018. Overview of the EVALITA 2018 HaSpeeDe Hate Speech Detection (HaSpeeDe) Task. In Tommaso Caselli, Nicole Novielli, Viviana Patti, and Paolo Rosso, editors, *Proceedings of the 6th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA'18)*, Turin, Italy. CEUR.org.

Leo Breiman. 2001. Random forests. *Machine learning*, 45(1):5–32.

Tommaso Caselli, Nicole Novielli, Viviana Patti, and Paolo Rosso. 2018. EVALITA 2018: Overview of the 6th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. In Tommaso Caselli, Nicole Novielli, Viviana Patti, and Paolo Rosso, editors, *Proceedings of Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2018)*, Turin, Italy. CEUR.org.

Ying Chen, Yilu Zhou, Sencun Zhu, and Heng Xu. 2012. Detecting offensive language in social media to protect adolescent online safety. In *Privacy, Security, Risk and Trust (PASSAT), 2012 International Conference on and 2012 International Confernece on Social Computing (SocialCom)*, pages 71–80. IEEE.

Marti A. Hearst, Susan T Dumais, Edgar Osuna, John Platt, and Bernhard Scholkopf. 1998. Support vector machines. *IEEE Intelligent Systems and their applications*, 13(4):18–28.

Věra Kolmogorov. 1992. Kolmogorov's theorem and multilayer neural networks. *Neural networks*, 5(3):501–506.

Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *International Conference on Machine Learning*, pages 1188–1196.

Yashar Mehdad and Joel Tetreault. 2016. Do characters abuse more than words? In *Proceedings of the 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 299–303.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Anna Schmidt and Michael Wiegand. 2017. A survey on hate speech detection using natural language processing. In *Proceedings of the Fifth International Workshop on Natural Language Processing for Social Media*, pages 1–10.

Sara Sood, Judd Antin, and Elizabeth Churchill. 2012. Profanity use in online communities. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1481–1490. ACM.

Rocco Tripodi and Stefano Li Pira. 2017. Analysis of italian word embeddings. *arXiv preprint arXiv:1707.08783*.

Cynthia Van Hee, Els Lefever, Ben Verhoeven, Julie Mennes, Bart Desmet, Guy De Pauw, Walter Daelemans, and Véronique Hoste. 2015. Detection and fine-grained classification of cyberbullying events. In *International Conference Recent Advances in Natural Language Processing (RANLP)*, pages 672–680.

William Warner and Julia Hirschberg. 2012. Detecting hate speech on the world wide web. In *Proceedings of the Second Workshop on Language in Social Media*, pages 19–26. Association for Computational Linguistics.

Jun-Ming Xu, Kwang-Sung Jun, Xiaojin Zhu, and Amy Bellmore. 2012. Learning from bullying traces in social media. In *Proceedings of the 2012 conference of the North American chapter of the association for computational linguistics: Human language technologies*, pages 656–666. Association for Computational Linguistics.