Document downloaded from:

http://hdl.handle.net/10251/139663

This paper must be cited as:

Hurtado Oliver, LF.; González-Barba, JÁ.; Pla Santamaría, F. (14-0). Choosing the right loss function for multi-label Emotion Classification. Journal of Intelligent & Fuzzy Systems. 36(5):4697-4708. https://doi.org/10.3233/JIFS-179019

The final publication is available at

https://doi.org/10.3233/JIFS-179019

Copyright IOS Press

Additional Information

Choosing the right loss function for multi-label Emotion Classification

Lluís-F. Hurtado*, José-Ángel González and Ferran Pla
Departament de Sistemes Informàtics i Computació Universitat Politècnica de València Camí de Vera sn, València, Spain

AUTHOR COPY

Abstract. Natural Language Processing problems has recently been benefited for the advances in Deep Learning. Many of these problems can be addressed as a multi-label classification problem. Usually, the metrics used to evaluate classification models are different from the loss functions used in the learning process. In this paper, we present a strategy to incorporate evaluation metrics in the learning process in order to increase the performance of the classifier according to the measure we are interested to favor. Concretely, we propose soft versions of the Accuracy, micro-F_1, and macro-F_1 measures that can be used as loss functions in the back-propagation algorithm. In order to experimentally validate our approach, we tested our system in an Emotion Classification task proposed at the International Workshop on Semantic Evaluation, SemEval-2018. Using a Convolutional Neural Network trained with the proposed loss functions we obtained significant improvements both for the English and the Spanish corpora.

Keywords: Deep Learning, loss function, multi-label classification, Natural Language Processing, Emotion Classification

1. Introduction

In recent years, Deep Learning has become one of the most sprawling approaches in many areas of artificial intelligence. Many Natural Language Processing (NLP) tasks has been benefited from the advances in this area. A wide range of these tasks can be addressed following a classification framework. Sentiment Analysis [12, 28], Language Identification [27], Emotion Classification [17, 20], and Stance Detection [16, 19, 31], among others, are examples of NLP tasks addressed as classification problems.

A classification problem can be defined as the problem of learning a function capable of assigning to each sample one or more classes of the set of classes defined for the task. Different metrics to evaluate the performance of a classification model are proposed in the literature such as, Accuracy, Precision, Recall, or F_1 measure, with some average methods (macro-averaging and micro-averaring) [30]. Depending on the characteristics of the system that we want to build, we will be interested in favoring one or other measure. In this sense, it is interesting to consider some mechanisms that allow us to incorporate this criterion in the learning phase of the model.

In Deep Learning approaches, the loss function is used by the back-propagation algorithm to guide the parameter estimation process. Although any differentiable function can be used as loss function, a few numbers of loss functions are usually used, without considering the measure used to evaluate the specific task. Two of the most used loss functions in the literature are the Cross-Entropy (CE), in its binary (BCE) or categorical versions, and the Mean Squared Error (MSE) [7]. These functions are individually computed for each sample, which means that class-level

*Corresponding author. Lluís-F. Hurtado, Departament de Sistemes Informàtics i Computació Universitat Politècnica de València Camí de Vera sn, 46022, València, Spain E-mail: {lhurtado, jogonba2, fpla}@dsic.upv.es.

performance during the learning process is not taken into account.

Some recent works proposed different loss functions for further improving machine learning systems. A study of how particular choices of loss functions affect deep models and their learning dynamics can be consulted in [10]. The authors show the results will lead to a wider adoption of several losses in Deep Learning area, in particular, non-classical loss functions such as Tanimoto loss and Cauchy-Schwarz Divergence.

Other works are oriented to optimize F_1 measure for multinomial regression models [4]. This work introduced a novel plug-in rule algorithm that estimates all parameters required for a Bayes-optimal prediction.

The use of F_1 measure as training criterion for Deep Leaning systems to an image cleaning and enhancement binary task was previously studied [25]. We extended their work by defining commonly used evaluation metrics as loss functions to train neural networks for multi-class and multi-label classification problems.

In order to test different loss functions on a multilabel corpus, we selected the *E-c: Detecting Emotions Multilabel classification* corpus used in subtask 5 of Task 1 (*Affect in Tweets*) at the International Workshop on Semantic Evaluation, SemEval-2018 [20]. This subtask has high interest in the Natural Language Processing and Social Media Analytics areas. It consists on given a tweet, assign it one, or more, of eleven given emotions that best represent the mental state expressed in the text.

The rest of the paper is organized as follows. Section 2 presents our proposal to use evaluation metrics as loss function. Section 3 describes the main characteristics of the addressed task. Section 4 presents the designed system, concretely the preprocessing, the tweet representation, and the Convolutional Neural Network (CNN) architecture we used. Section 5 presents the experimental work conducted. Finally, section 6 presents some conclusions and the future work.

2. Evaluation metrics as loss functions

We propose some loss functions for training neural networks in order to address classification problems. These functions are approximations to the classical metrics used to evaluate classification models. The proposed functions have the advantage of being differentiable and consequently, they can be used as loss functions in the back-propagation algorithm.

Firstly, we formally define the classification problem and the metrics used for its evaluation. Next, we present the proposal of the new differentiable loss functions.

2.1. The classification problem

The classification problem can be defined as the problem of learning a function $f : \mathcal{X} \rightarrow \mathcal{Y}$ from a set of labeled samples D, where

- $\mathbb{C}$ is a finite set of classes $\mathbb{C} = \{c_1, \cdots, c_{|\mathbb{C}|}\}$, and $|\mathbb{C}| > 1$
- $\mathcal{X}$ is an input space
- $\mathcal{P}(\mathbb{C})$ is the label power set of $\mathbb{C}$
- $\mathcal{Y}$ is the set of considered labels, $\mathcal{Y} \subseteq \mathcal{P}(\mathbb{C})$, and
- $D = \{(x_1, \gamma_1), \cdots, (x_n, \gamma_n)\}$ is a data set of samples, where $x_i \in \mathcal{X}$ and $\gamma_i \in \mathcal{Y}$

When $|\mathbb{C}| > 2$, the problem is called multi-class classification and when a sample can be assigned to more than one class (i.e. $\mathbb{C} \subset \mathcal{Y}$) the problem is called multi-label classification.

In order to automatically evaluate the performance of the classifier we assume that a labeled test set is available. Let (X, Y) be a test set consisting on m samples (x_i, γ_i), $1 \leq i \leq m$, $(x_i \in \mathcal{X}, \gamma_i \in \mathcal{Y})$, with a set of classes $\mathbb{C}$, $|\mathbb{C}| = n$. Let $O = \{\theta_i : 1 \leq i \leq m\}$ be the set of predictions of the classifier on the test set, where $\theta_i \in \mathcal{Y}$ is the set of classes assigned to the sample x_i by the classifier.

One of the most used metrics to evaluate classifiers is Accuracy. In mono-label classification tasks, i.e. only one class per sample, Accuracy is defined as the percentage of correctly classified samples. For multi-label classification tasks, it is necessary to introduce an extension of the Accuracy metric. Equation (1) shows the formulation of multi-label **Accuracy** (Acc).

$$Acc = \frac{1}{m} \sum_{i=1}^{m} \frac{|\gamma_i \cap \theta_i|}{|\gamma_i \cup \theta_i|} \tag{1}$$

where the numerator represents the number of correctly predicted classes for sample i, normalized by the size of the union of the predicted and correct class sets for sample i. Note that this metric is equivalent

AUTHOR COPY

to Jaccard index per sample, averaged across all the samples.

The other metrics we study in this work are Precision, Recall, and F_1 measure. Moreover, there are two points of view to calculate a value for these metrics considering the complete test set, micro-averaging and macro-averaging.

Following the micro-averaging approach, we can define the micro-Precision, or just **Precision** (P), as the fraction of all classes generated by the classifier that have been correctly predicted; and the micro-Recall, or just **Recall** (R), as the fraction of all correct classes that have been correctly predicted by the classifier. Both metrics are formally defined in Equations (2) and (3).

$$P = \frac{\sum_{i=1}^{m} |\gamma_i \cap \theta_i|}{\sum_{i=1}^{m} |\theta_i|} \tag{2}$$

$$R = \frac{\sum_{i=1}^{m} |\gamma_i \cap \theta_i|}{\sum_{i=1}^{m} |\gamma_i|} \tag{3}$$

Micro-F_1 (m-F_1) is a particular case of micro-F_β measure where $\beta = 1$, that is, the harmonic mean of Precision and Recall. Equation (4) shows the formulation of micro-F_1.

$$\text{m-F}_1 = \frac{2 \cdot P \cdot R}{P + R} = 2 \cdot \frac{\frac{\sum_{i=1}^{m} |\gamma_i \cap \theta_i|}{\sum_{i=1}^{m} |\theta_i|} \cdot \frac{\sum_{i=1}^{m} |\gamma_i \cap \theta_i|}{\sum_{i=1}^{m} |\gamma_i|}}{\frac{\sum_{i=1}^{m} |\gamma_i \cap \theta_i|}{\sum_{i=1}^{m} |\theta_i|} + \frac{\sum_{i=1}^{m} |\gamma_i \cap \theta_i|}{\sum_{i=1}^{m} |\gamma_i|}} = 2 \cdot \frac{\sum_{i=1}^{m} |\gamma_i \cap \theta_i|}{\sum_{i=1}^{m} |\theta_i| + \sum_{i=1}^{m} |\gamma_i|} \tag{4}$$

Additionally, we can compute the Precision, Recall and F_1 per class. Eqs. (5), (6) and (7) show the definition of these metrics for a specific class c. Note that, we use the Iverson bracket notation $[c \in \gamma_i \cap \theta_i]$ which has the value 1 if c bellows to $\gamma_i \cap \theta_i$ and 0 otherwise.

$$P_c = \frac{\sum_{i=1}^{m} [c \in \gamma_i \cap \theta_i]}{\sum_{i=1}^{m} [c \in \theta_i]} \tag{5}$$

$$R_c = \frac{\sum_{i=1}^{m} [c \in \gamma_i \cap \theta_i]}{\sum_{i=1}^{m} [c \in \gamma_i]} \tag{6}$$

$$F_{1,c} = 2 \cdot \frac{\sum_{i=1}^{m} [c \in \gamma_i \cap \theta_i]}{\sum_{i=1}^{m} [c \in \theta_i] + \sum_{i=1}^{m} [c \in \gamma_i]} \tag{7}$$

Following the macro-averaging approach, we can compute the **macro-$F1$** (M-F_1) as the arithmetic mean of F_1 per class. Equation 8 shows the definition of macro-F_1.

$$\text{M-F}_1 = \frac{2}{|\mathbb{C}|} \cdot \sum_{c \in \mathbb{C}} \frac{\sum_{i=1}^{m} [c \in \gamma_i \cap \theta_i]}{\sum_{i=1}^{m} [c \in \theta_i] + \sum_{i=1}^{m} [c \in \gamma_i]} \tag{8}$$

In this case, all classes equally contribute to the global measure regardless of the number of samples.

2.2. Soft metrics

In this section, we present some differentiable functions that are approximate versions of the averaged metrics introduced above. These functions can be used as loss functions to train neural networks for multi-class and multi-label classification problems.

When neural networks are used to address classification problems, the last layer of the network needs as many neurons as classes. Let o_i be the output layer of the network when processing sample i, and o_i^j the value of the jth neuron of o_i, that is, the value assigned to class c_j. Let y_i be the set of correct classes of sample i represented as a vector, thus $y_i^j = 1$ if sample i belongs to class c_j, otherwise $y_i^j = 0$.

In order to determine, from o_i, the classes assigned to the sample i by the model, it would be necessary to set a threshold and to select those classes for which the value of o_i^j is greater than that threshold. This would make the resultant function non-differentiable.

AUTHOR COPY

To solve this problem, in this work, we propose the use of the following approximations:

a) $|\theta_i| \approx \sum_{j=1}^{n} o_i^j$. The number of classes in the prediction for sample i, $|\theta_i|$, is approximated as the sum of the values of all the neurons in the output layer of the network, $\sum_{j=1}^{n} o_i^j$.

b) $|\gamma_i| = \sum_{j=1}^{n} y_i^j$. The number of classes in the reference for sample i, $|\gamma_i|$, is computed as the sum of the values in y_i, $\sum_{j=1}^{n} y_i^j$.

c) $|\gamma_i \cap \theta_i| \approx \sum_{j=1}^{n} y_i^j \cdot o_i^j$. The number of correctly predicted classes for sample i, $|\gamma_i \cap \theta_i|$, is approximated as the weighted sum of the output layer and the vector of correct classes for sample i, $\sum_{j=1}^{n} y_i^j \cdot o_i^j$.

d) $|\gamma_i \cup \theta_i| \approx \sum_{j=1}^{n} \left(y_i^j + o_i^j - y_i^j \cdot o_i^j \right)$. Applying the set theory for calculating the cardinality of a union set, the normalization factor of the Accuracy, $|\gamma_i \cup \theta_i|$, is computed using the three previous approximations.

e) $\sum_{i=1}^{m} [c_j \in \theta_i] \approx \sum_{i=1}^{m} o_i^j$. The number of samples for which the class c_j is predicted, $[c_j \in \theta_i]$, is approximated as the sum of the *jth* component of the output layer of the network for all samples, $\sum_{i=1}^{m} o_i^j$.

f) $\sum_{i=1}^{m} [c_j \in \gamma_i] = \sum_{i=1}^{m} y_i^j$. The number of samples with class c_j, $\sum_{i=1}^{m} [c_j \in \gamma_i]$, is computed as the sum of the *jth* component of the vector of correct classes for all samples, $\sum_{i=1}^{m} y_i^j$.

g) $\sum_{i=1}^{m} [c_j \in \gamma_i \cap \theta_i] \approx \sum_{i=1}^{m} y_i^j \cdot o_i^j$. The number of samples with class c_j correctly predicted, $\sum_{i=1}^{m} [c_j \in \gamma_i \cap \theta_i]$, is approximated as the weighted sum of the *jth* component of the output layer and the vector of correct classes for all the samples, $\sum_{i=1}^{m} y_i^j \cdot o_i^j$.

Using these approximations, soft versions of the evaluation metrics can be defined. Equations 9, 10 and 11 present soft versions of Accuracy (SAcc), micro-F_1 (Sm-F_1) and macro-F_1 (SM-F_1). Note that the sign has been inverted because they are loss functions that should be minimized.

$$SAcc = -\frac{1}{m} \sum_{i=1}^{m} \frac{\sum_{j=1}^{n} \left(o_i^j \cdot y_i^j \right)}{\sum_{j=1}^{n} \left(o_i^j + y_i^j - o_i^j \cdot y_i^j \right)} \tag{9}$$

$$\text{Sm-F}_1 = -2 \cdot \frac{\sum_{i=1}^{m} \sum_{j=1}^{n} \left(o_i^j \cdot y_i^j \right)}{\sum_{i=1}^{m} \sum_{j=1}^{n} \left(o_i^j + y_i^j \right)} \tag{10}$$

$$\text{SM-F}_1 = -\frac{2}{n} \cdot \sum_{j=1}^{n} \frac{\sum_{i=1}^{m} \left(o_i^j \cdot y_i^j \right)}{\sum_{i=1}^{m} \left(o_i^j + y_i^j \right)} \tag{11}$$

These functions are approximations of Acc, m-F_1 and M-F_1 with the advantage of being able to work with continuous values of o_i^j. Nevertheless, when $o_i^j \in \{0, 1\}$, the evaluation metrics and their soft versions are equivalent.

In Section 4, devoted to the experimental work, a study of how the soft functions approximate the evaluation metrics is presented.

Soft functions have been defined to satisfy that they were differentiable functions. Equations (12), (13) and (14) show the derivatives of the three soft metrics (SAcc, Sm-F_1 and SM-F_1) with respect to o_k^l, $1 \leq k \leq m$, $1 \leq l \leq n$.

$$\frac{\partial \, \textbf{Acc}}{\partial \, o_k^l} = -\frac{1}{m} \cdot \frac{y_k^l \cdot \sum_{j=1}^{n} \left(o_k^j + y_k^j \right) - \sum_{j=1}^{n} \left(o_k^j \cdot y_k^j \right)}{\left(\sum_{j=1}^{n} \left(o_k^j + y_k^j - o_k^j \cdot y_k^j \right) \right)^2} \tag{12}$$

$$\frac{\partial \, \textbf{Sm-F}_1}{\partial \, o_k^l} = -2 \cdot \frac{y_k^l \cdot \sum_{i=1}^{m} \sum_{j=1}^{n} (o_i^j + y_i^j) - \sum_{i=1}^{m} \sum_{j=1}^{n} (o_i^j \cdot y_i^j)}{\left(\sum_{i=1}^{m} \sum_{j=1}^{n} (o_i^j + y_i^j) \right)^2} \tag{13}$$

AUTHOR COPY

$$\frac{\partial\, \textbf{SM-F}_1}{\partial\, o_k^l} = -\frac{2}{n} \cdot \frac{y_k^l \cdot \sum_{i=1}^{m}(o_i^l + y_i^l) - \sum_{i=1}^{m}(o_i^l \cdot y_i^l)}{\left(\sum_{i=1}^{m}(o_i^l + y_i^l)\right)^2} \quad (14)$$

Note that M-F_1 and m-F_1 are computed over a set of samples. We decided to use mini-batch training mode [2] in order to compute SM-F_1 and Sm-F_1 over all the samples of a given batch. However, in the mini-batch mode, it is necessary to assign a scalar for each sample in order to update the weights of the model by using the back-propagation algorithm. Therefore, we used the value of the loss function computed over a full batch as loss value for all the samples in the batch. Although it is not required, we used the same strategy for SAcc. The size of the batch is a relevant factor in the learning process. In Section 4 a study of how the size of the batch influences the performance of the model is presented.

3. Task description

In this section, we present the main characteristics of the subtask 5 of Task 1: *"E-c: Detecting Emotions Multilabel classification"* [20] proposed at the International Workshop on Semantic Evaluation, SemEval-2018[1]. In this work, we address the task for the English and the Spanish languages. This task can be formalized as a multi-class/multi-label classification problem, that is, given a text, we classify it in one, or more, of the eleven given emotions that best represents the mental state expressed in the text. The corpus supplied by the organizers is a collection of tweets tagged with a set of emotions tags.

Table 1 shows the details of this corpus both for English and Spanish languages. It can be seen the number of samples per emotion in the partitions of training, development, and test that participating teams must use for their machine-learning systems. It is also showed the total number of tweets. From these figures, it can be inferred that the average number of labels per tweet is about 2.3 for English and 1.7 for Spanish.

The official competition metric used for ranking the teams in E-c task was multi-label accuracy (or Jaccard index) as defined in Equation 1. Since this is a multi-label classification task, each tweet can have one or more gold emotion labels, and one or more predicted emotion labels. Apart from the official competition metric (multi-label accuracy), we also report micro-averaged F-score and macro-averaged F-score in order to validate our proposal.

[1] http://alt.qcri.org/semeval2018/index.php?id=tasks

4. System description

In this section, we present the main characteristics of the system developed in this work. Concretely, we discuss the tweet preprocessing, the external resources used to add polarity/emotion information to the model, the tweet representation, and the Deep Learning architecture.

We applied a tweet preprocessing that consisted in a tokenization process by means of the TweetMotif [11] package. Then, we applied a normalization step consisted in lowercasing the words, and in addition, for the Spanish language we removed some language-specific characters, for example, accents, dieresis, specific language characters (i.e. "$\tilde{n}$" character in Spanish language), etc.

Moreover, we carried out a normalization process over unicode emoticons. This process can be useful due to the great variability of the emoticons. In addition, the embedding models used in this work did not included these symbols. To solve this, we replaced the unicode emoticons by their short name extracted from the Unicode Common Locale Data Repository, which contains a textual description of the emoticon's shape, e.g. ☺ → "Slightly Smiling Face". It's important to notice that similar emoticons have similar textual descriptions, consequently, it makes possible to establish relationships among them by using their descriptions.

Regarding to the external resources, we used distributed representations of words, concretely, embeddings obtained using Word2Vec [13, 14] models, in order to consider the semantic/contextual information of each token. Moreover, we used several lexicons to consider polarity/emotion information. This polarity/emotion information was combined with the embeddings of the words.

On the one hand, for the English task, we used the following lexicons: AFFIN [24], Bing Liu's Opinion [8], MPQA [34], Sentiment140 [5], SentiWordnet [1], NRC Emotion Lexicon [18], NRC Hashtag Emotion Lexicon [15] and LIWC2007 [26]. As word embeddings, we used the pretrained model by [6] with more than 400 million English tweets. On the other hand, for the Spanish task, we used

AUTHOR COPY

Table 1
Data set distribution of the Emotion Classification task at Semeval-2018

	English				Spanish			
Emotion	Train	Dev	Test	$\sum$	Train	Dev	Test	$\sum$
Anger	2544	315	1101	3960	1155	209	919	2283
Anticipation	978	124	425	1527	415	94	321	830
Disgust	2602	319	1099	4020	521	98	423	1042
Fear	1242	121	485	1848	373	74	298	745
Joy	2477	400	1442	4319	1087	201	873	2161
Love	700	132	516	1348	261	55	245	561
Optimism	1984	307	1143	3434	378	66	278	722
Pessimism	795	100	375	1270	578	115	495	1188
Sadness	2008	265	960	3233	845	143	644	1632
Surprise	361	35	170	566	169	37	122	328
Trust	357	43	153	553	175	31	122	328
$\sum$	16048	2161	7869	26078	5957	1123	4740	11820
#samples	6838	886	3259	10983	3561	679	2854	7094

the following lexicons: ELHPolar [29], ISOL [21], MLSenticon [3] and the Spanish version of NRC Emotion Lexicon. As word embeddings, we trained a skip-gram model, with 300 dimensions for each word, from 87 million Spanish tweets collected for the experimental work presented in this paper.

We represented each tweet $\boldsymbol{x}$ as a matrix $\boldsymbol{S} \in \mathbb{R}^{n\times(d+v)}$, where n is the maximum number of words per tweet, d is the dimensionality of word embeddings and v is the dimensionality of the polarity/emotion features. In order to obtain this representation, we use an embedding model $\mathbf{h(w)} \in \mathbb{R}^d$ and a set of lexicons $\boldsymbol{h'(w)} = [h'_1(w), h'_2(w), ..., h'_l(w)] \in \mathbb{R}^v$ (note that all the features extracted from lexicons, for the word w, are concatenated).

Therefore, given a tweet x with n tokens, $x = w_1, w_2, ..., w_n$, we represented it as a matrix S in which, each row i is the concatenation of the embedding of w_i ($h(w_i)$) and a vector with the polarity values of w_i in each lexicon ($h'(w_i)$), $S = [h(w_1)|h'(w_1), h(w_2)|h'(w_2), ..., h(w_n)|h'(w_n)]$. If a word w_i is out of vocabulary for the embedding models, we replace its embedding by the embedding of the word "unknown", $h(w_i) = h(\text{“}unknown\text{"})$. Similarly, if w_i is not included in any lexicon, $h'(w_i) = [0, 0, ..., 0] \in \mathbb{R}^v$. Given a tweet, it can belong to several classes, from the set of classes $\mathbb{C}$, in an independent way i.e. $y = \{y_1, y_2, ..., y_{|\mathbb{C}|} : y_i \in \{0, 1\}\}$ with $|\mathbb{C}| = 11$.

Moreover, due to the variable length of the tweets, we used zero padding at the start of a tweet if it does not reach the maximum specified length. Otherwise, if the length of a tweet is greater than the maximum, we truncated the tweet at the end. In the English task the average length of words per tweet is $n_{avg} = 19$, and the maximum length is $n_{max} = 85$. We decided to set the length n as the mean of n_{avg} and n_{max}. In the Spanish task, $n_{avg} = 16$, $n_{max} = 82$, therefore $n = \frac{n_{avg}+n_{max}}{2} = 49$.

Regarding to the Deep Learning system, we used a Convolutional Neural Network (CNN) architecture inspired in [35], which has shown competitive results in several text classification tasks such as Customer Review Dataset [8], Opinion Polarity [33] and Irony Detection [32]. We used this Deep Learning system for all the experiments conducted.

Following the CNN architecture, we applied one-dimensional convolutions with variable height filters in order to extract the temporal structure of the tweet over several region sizes. Note that the width of the filters is constant and equal to d and we only modify the height of the filters.

Fig. 1 summarizes the model configuration and the hyper-parameters that we used in all the experimental work. As can be seen, we used 6 different region sizes (the filter height ranges from 1 to 6) and 128 filters for each region size. A total of 768 different filters were used in this architecture. After applying the filters, we obtained 128 output feature maps for each region size. Note than the output feature maps have length n due to we used a “same padding" scheme.

In order to extract the most salient features for each region size, we applied 1D Global Max Pooling to the feature maps of each region size. Therefore, we obtained 6 vectors with 128 components, that were concatenated and used as input to a fully-connected layer which performs the multi-label classification decision. We used sigmoid activation functions to model the probability of each class independently of the probability of the other classes.

AUTHOR COPY

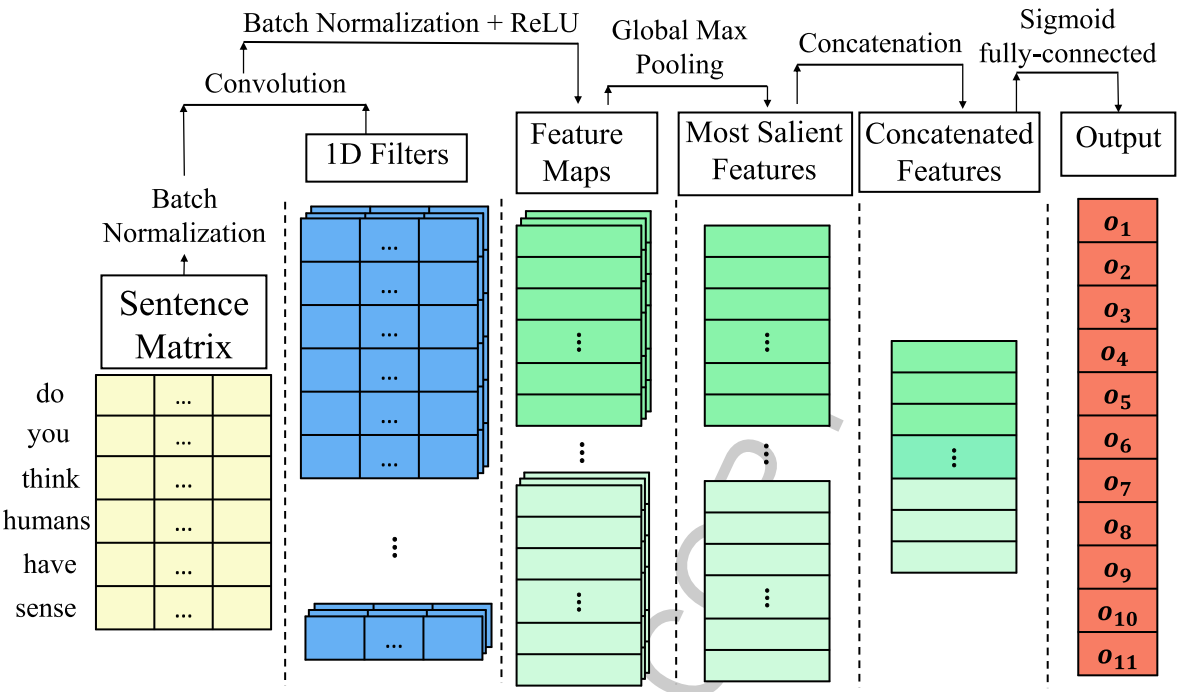

Fig. 1. CNN architecture for multi-label classification.

Moreover, with the aim of improving the generalization and speed up the model convergence, we used BatchNormalization [9] after each convolution and after the input layer. To achieve non-linearity after each convolution, we used ReLU [23] activation functions. As optimization algorithm, we used RMSprop to learn the parameters of the network with respect to the proposed loss functions.

Given the previous convolutional network architecture, $f : \mathbb{R}^{n\times(d+v)} \rightarrow \mathbb{R}^{\mathbb{C}}$, the steps to assign a set of classes to a tweet $\boldsymbol{x}$ are the following: first we obtain its representation matrix $\boldsymbol{S}$; second, we make a forward pass in order to obtain the probability that $\boldsymbol{x}$ belongs to each class independently, $\boldsymbol{f(x)} = \{o_1, o_2, ..., o_n\}$; and finally, we consider that the tweet $\boldsymbol{x}$ belongs to class j if $\boldsymbol{f(x)}_j \geq 0.5$.

5. Experimental results

In this section, we present the experimental work conducted in order to evaluate the performance of the loss functions proposed in Section 2. We study the impact of these loss functions on the overall results on the Emotion Classification task proposed at SemEval2018 competition.

A parameter that has a high impact on the calculation of the loss functions is the batch size. Therefore, we first studied the behavior of the functions by varying the batch size.

This study was carried out on the development sets for English and Spanish defined in Section 3 (see Table 1). We considered 30 training epochs and the SAcc, SM-F_1 and Sm-F_1 loss functions. Fig. 2 shows the achieved results for English corpora. It can be seen the values of the evaluation metric per epoch on the development set for the CNN trained with the SAcc, SM-F_1, and Sm-F_1 loss functions, respectively, by varying the batch size $b \in \{16, 32, 64, 128\}$.

As it can be seen in Fig. 2, in all cases, lower values of b produces faster convergence, that is, the maximum value of the evaluation metric is obtained in initial epochs. We hypothesized that, the faster convergence is because more updates are made in each training epoch. Moreover, in all cases, the maximum value of the evaluation metric is obtained with the minimum batch size considered, $b = 16$. It can be also observed that generally, the batch sizes that obtained the best results were $b = 16$ and $b = 32$.

In addition, we can also observe that with largest batch sizes, $b = 128$, the model takes more time to converge, but when it becomes stable, the results were similar compared to those obtained with smaller batch sizes.

We performed a similar experimentation for the Spanish corpora and we have observed a similar

AUTHOR COPY

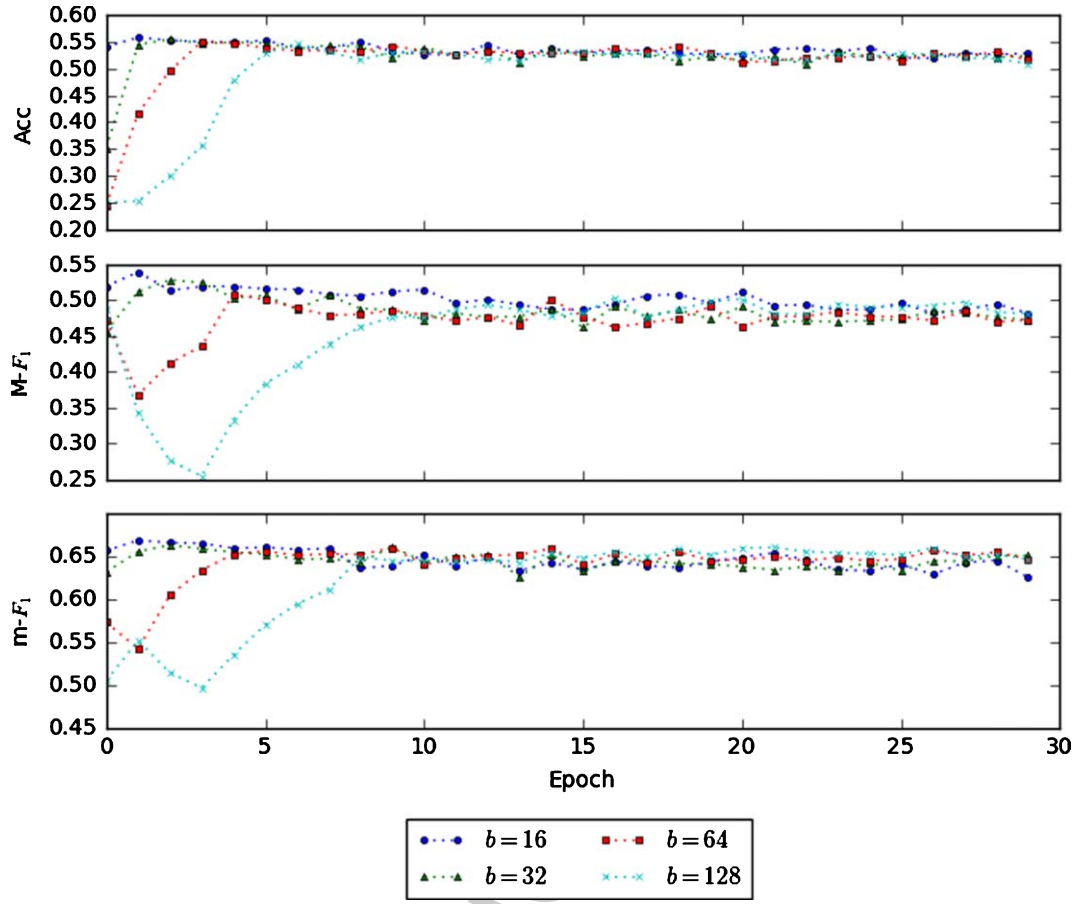

Fig. 2. Results of the evaluation metrics per epoch for CNN+SAcc, CNN+SM-F_1 and CNN+Sm-F_1 models varying the batch size (English development set).

Table 2
Results on the English test set

Loss	Acc	M-F_1	m-F_1
SAcc	**56.39***±**1.11**	49.72±1.15	67.32*±0.96
SM-F_1	54.85*±1.09	**54.73***±**1.02**	66.75*±0.97
Sm-F_1	55.44*±1.11	50.77±1.08	**67.60***±**0.95**
BCE	52.03±1.12	50.47±1.25	64.44±0.97
MSE	52.32±1.11	49.59±1.17	64.65±0.97

Table 3
Results on the Spanish test set

Loss	Acc	$M-F_1$	$m-F_1$
SAcc	**47.33***±1.43	42.06±1.64	54.82±1.31
SM-F_1	45.26±1.40	**45.25***±**1.57**	**55.10±1.30**
Sm-F_1	44.20±1.47	42.34±1.66	54.83±1.34
BCE	44.08±1.41	40.42±1.51	52.70±1.34
MSE	44.22±1.44	41.28±1.52	54.10±1.33

behavior than in the English corpora, that is, lower batch sizes produce faster convergence.

From the performed study of how influence batch size on loss function optimization, we can conclude that the best size is $b = 16$, both for the English and Spanish data sets.

Once the best value of the batch size was set, our objective was to compare the performance of the system trained using the loss functions proposed in this paper with the performance obtained using BCE and MSE as loss functions, on the test sets of the Emotion Classification task. To make this comparison more accurate, the confidence intervals of the three metrics used to evaluate the systems in the competition were computed.

In order to compute the confidence intervals, we used the Bootstrap Confidence Intervals [22] approach. First, from the set of hypotheses provided by the system that we want to evaluate, we generated up to 5000 resamples by sampling with replacement from this original set of hypotheses. Each resample had the same size of the original set. Next, the value

Table 4
Precision, Recall and F_1 per class, for English test set, with the models trained with SM-F_1 and Sm-F_1

	SM-F_1			Sm-F_1		
Emotion	P	R	F_1	P	R	F_1
Anger	72.41	81.29	76.59	68.29	85.10	75.78
Anticipation	26.80	38.59	31.63	35.67	14.35	20.47
Disgust	66.06	79.53	72.17	62.97	82.62	71.47
Fear	69.40	64.54	66.88	70.83	63.09	66.74
Joy	82.06	84.05	83.04	79.35	86.89	82.95
Love	49.67	73.06	59.14	57.76	56.98	57.37
Optimism	65.14	75.85	70.09	65.42	76.47	70.51
Pessimism	33.91	42.13	37.57	41.43	27.73	33.23
Sadness	64.57	71.56	67.89	61.78	72.92	66.89
Surprise	16.14	24.12	19.34	52.63	5.88	10.58
Trust	11.36	40.52	17.74	22.22	1.31	2.47
Macro Average	50.68	61.39	54.73	56.22	52.12	50.77
Micro Average	59.35	71.80	64.99	66.02	69.25	67.60

Table 5
Precision, Recall and F_1 per class, for Spanish test set, with the models trained with SM-F_1 and Sm-F_1

	SM-F_1			Sm-F_1		
Emotion	P	R	F_1	P	R	F_1
Anger	66.91	69.75	68.30	68.53	68.01	68.27
Anticipation	45.68	23.05	30.64	42.27	25.55	31.84
Disgust	46.96	38.30	42.19	48.81	34.04	40.11
Fear	64.15	45.64	53.33	60.09	43.96	50.78
Joy	80.30	72.85	76.40	81.19	71.71	76.16
Love	65.96	50.61	57.27	70.67	43.27	53.67
Optimism	32.02	29.14	30.51	43.81	16.55	24.02
Pessimism	38.94	19.19	25.71	38.52	18.99	25.44
Sadness	65.53	59.94	62.61	65.95	61.65	63.72
Surprise	25.81	26.23	26.02	37.14	10.66	16.56
Trust	25.00	24.59	24.79	33.33	9.84	15.19
Macro Average	50.66	41.75	45.25	53.67	36.75	42.34
Micro Average	60.12	50.57	54.93	63.90	48.02	54.83

AUTHOR COPY

of the evaluation measure is calculated for each of the resamples. Finally, we compute the 95% confidence interval using the bootstrap distribution.

Tables 2 and 3 show the results obtained by our systems both for the English and the Spanish corpora. An asterisk (*) on values means that these values are statistically significant.

It can be seen that the models trained with the proposed loss functions obtained the best results. This occurs both for English and Spanish test sets with significant improvements in all cases except when evaluating with m-F_1 on the Spanish test set. In addition, it can be observed how, generally, the model trained with the loss function that approximates the evaluation metric used, obtained the best results for that evaluation metric. This is true for all the cases studied, except for the m-F_1 measure for Spanish in which it achieved the second-best result (note that this is the only case where the improvements are not significant). Compared with the official results of the competition[2] our best result achieved the 1st position of 13 teams for Spanish and the 7th place of 34 teams for English.

Next, we present an analysis of the performance of our system at class level. Tables 4 and 5 show the results of P, R, and F_1 for all classes with the CNN trained using SM-F_1 and Sm-F_1 as loss function respectively.

From the results for the English corpus, it can be observed that the system trained with SM-F_1 achieved higher values of F_1 for the minority classes, e.g. Trust ($F_1 = 17.74$ on development and $F_1 = 24.79$ on test) and Surprise ($F_1 = 19.34$ on development and $F_1 = 26.02$ on test), compared to those obtained by the system trained with Sm-F_1 (for Trust: $F_1 = 2.47$ on development and $F_1 = 15.19$ on test

[2] https://competitions.codalab.org/competitions/17751#results

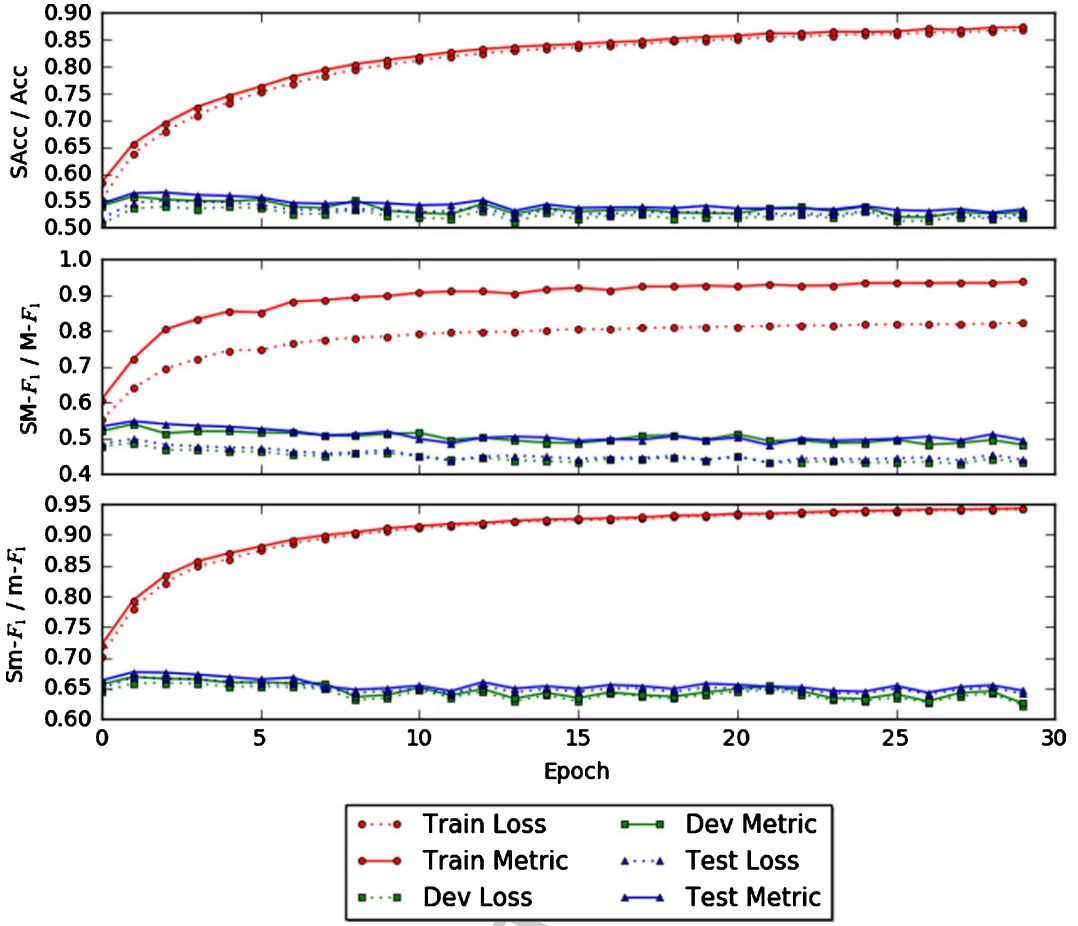

Fig. 3. Loss function and evaluation metric per epoch, on English training, development and test sets using CNN+SAcc, CNN+SM-F_1 and Sm-F_1, respectively.

and for Surprise: $F_1 = 10.58$ on development and $F_1 = 16.56$ on test). This occurs because of SM-F_1 consider the performance of all the classes in the same way, independently of the number of samples per class. In contrast, Sm-F_1 favors majority classes. A similar behavior can also be observed in the results for the Spanish corpus. This is an important aspect to keep in mind in the design of a classification system.

Finally, an analysis of how the loss functions approximate the evaluation metrics is presented. For the English corpus, Fig. 3 shows loss function and the evaluation metric on training, development and test sets for the CNN trained using SAcc, SM-F_1, and Sm-F_1, respectively.

It can be observed how, throughout all the training epochs, the proposed loss functions follow the same trend as the evaluation metrics. This seems to indicate that they are good approximations to the evaluation metrics and consequently, we can make decisions about the evaluation metrics by looking only at the values obtained by the loss functions.

Another interesting aspect to note is that the loss functions can be considered as a lower bound of the evaluation metrics. This can be observed in all the previous cases, since, for the different epochs, the loss functions never overestimate the values of the evaluation metric.

In addition, it draws attention how SM-F_1, although it correctly estimates the trend of the metric M-F_1 throughout the different epochs, obtains values much smaller than these ones. On the other hand, the SAcc and Sm-F_1 loss functions, follow the trend of the evaluation metrics to which they approximate and they obtain values more similar to those obtained by those metrics. A similar behavior was observed in all the partitions for Spanish language.

6. Conclusions and future work

In this paper we have proposed soft versions of the Accuracy, micro-F_1, and macro-F_1 measures that can be used as loss functions in the learning process of Deep Learning models.

The proposed method imposes minor changes in the training phase, only the change of the loss function is required in the training process.

We conducted an extensive experimental work on a multi-label text classification task in order to validate our approach. In all the experiments conducted on the *"E-c: Detecting Emotions Multilabel classification"* task proposed at Semeval-2018, when loss functions adapted to the metrics were used, the results were improved compared to those obtained using classical loss functions and this improvement was statistically significant at 95% confidence level.

These results for the Spanish language outperformed the best results published at Semeval-2018 competition. Competitive results for the English language were also obtained.

We verified that the convergence of the models trained with the proposed loss functions is faster, i.e. the best results are obtained with fewer epochs, in comparison with classical loss functions. Moreover, we observed that the proposed loss functions adequately approximates the evaluation metrics. This is interesting because more robust models can be obtained, using these loss functions, with respect to the evaluation metrics.

In this work we only addressed some evaluation metrics that are commonly used in multi-label text classification tasks. However, following the proposal of this work, it would be possible to extend this technique to other evaluation metrics as loss functions.

As future work, we plan to do a deeper study about the properties of the loss functions introduced in this work.

Firstly, because we softened the evaluation metrics with the aim of making them differentiable, there are differences between the softened version (loss function, before the discretization process) and the evaluation metric (after the discretization process). We plan to explore how to minimize these differences by using some kind of regularization on the predictions.

Finally, it is interesting to explore the capacity of our loss functions to address other tasks with different characteristics and difficulties such as extremely imbalanced classification problems.

AUTHOR COPY

Acknowledgments

This work has been partially supported by the Spanish MINECO and FEDER founds under project AMIC (TIN2017-85854-C4-2-R) and the GiSPRO project (PROMETEU/2018/176). Work of José-Ángel González is also financed by Universitat Politècnica de València under grant PAID-01-17.

References

[1] S. Baccianella, A. Esuli and F. Sebastiani, Sentiwordnet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining, *In in Proc of LREC*, 2010.

[2] J. Bilmes, K. Asanovic, C.-W. Chin and J. Demmel, Using phipac to speed error back-propagation learning, *In 1997 IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 5, 1997, pp. 4153–4156.

[3] F.L. Cruz, J.A. Troyano, B. Pontes and F.J. Ortega, Building layered, multilingual sentiment lexicons at synset and lemma levels, *Expert Systems with Applications* **41**(13) (2014), 5984–5994.

[4] K. Dembczynski, A. Jachnik, W. Kotlowski, W. Waegeman and E. Huellermeier, Optimizing the F-Measure in Multi-Label Classification: Plug-in Rule Approach versus Structured Loss Minimization. In S. Dasgupta and D. McAllester, editors, *Proceedings of the 30th International Conference on Machine Learning volume 28 of Proceedings of Machine Learning Research*, Atlanta, Georgia, USA, PMLR, 2013, pp. 1130–1138.

[5] A. Go, R. Bhayani and L. Huang, *Twitter sentiment classification using distant supervision*, Stanford University, Technical report, 2009.

[6] F. Godin, B. Vandersmissen, W. De Neve and R. Van de Walle, Multimedia lab @ ACL W-NUT NER sharedtask: Named entity recognition for Twitter microposts using distributed word representations, *ACL-IJCNLP* **2015** (2015), 146–153.

[7] I. Goodfellow, Y. Bengio and A. Courville, Deep Learning, MIT Press, http://www.deeplearningbook.org (2016).

[8] M. Hu and B. Liu, Mining and summarizing customer reviews, In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '04*, New York, NY, USA, ACM, 2004, pp. 168–177.

[9] S. Ioffe and C. Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift, *CoRR, abs/1502.03167* (2015).

[10] K. Janocha and W.M. Czarnecki, On loss functions for deep neural networks in classification, *CoRR, abs/1702.05659* (2017).

[11] M. Krieger and D. Ahn, Tweetmotif: Exploratory search and topic summarization for twitter, *In Proc of AAAI Conference on Weblogs and Social*, 2010.

[12] B. Liu, Sentiment Analysis and Opinion Mining, A Comprehensive Introduction and Survey. Morgan & Claypool Publishers, 2012.

[13] T. Mikolov, I. Sutskever, K. Chen, G. Corrado and J. Dean, Distributed representations of words and phrases and their compositionality, *CoRR, abs/1310.4546* (2013a).

[14] T. Mikolov, K. Chen, G. Corrado and J. Dean, Efficient estimation of word representations in vector space, CoRR, abs/1301.3781, 2013b.

[15] S. Mohammad, #emotional tweets, In **SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the Main Conference and the Shared Task and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, Montréal, Canada. Association for Computational Linguistics, 2012, pp. 246–255.

[16] S. Mohammad, S. Kiritchenko, P. Sobhani, X. Zhu and C. Cherry, Semeval-2016 task 6: Detecting stance in tweets, In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, 2016, pp. 31–41.

[17] S.M. Mohammad and F. Bravo-Marquez, WASSA-2017 shared task on emotion intensity, CoRR, abs/1708.03700, 2017.

[18] S.M. Mohammad and P.D. Turney, Crowdsourcing a word-emotion association lexicon, *Computational Intelligence* **29**(3) (2013), 436–465.

[19] S.M. Mohammad, P. Sobhani and S. Kiritchenko, Stance and sentiment in tweets, *ACM Trans Internet Technol* **17**(3) (2017), 26:1–26:23.

[20] S.M. Mohammad, F. Bravo-Marquez, M. Salameh and S. Kiritchenko, Semeval-2018 Task 1: Affect in tweets, In *Proceedings of International Workshop on Semantic Evaluation (SemEval-2018)*, New Orleans, LA, USA, 2018.

[21] M.D. Molina-González, E. Martínez-Cámara, M-T Martín-Valdivia and J.M. Perea-Ortega, Semantic orientation for polarity classification in spanish reviews, *Expert Systems with Applications* **40**(18) (2013), 7250–7257.

[22] D. Moore, G. McCabe, W. Duckworth and S. Sclove, *The Practice of Business Statistics Companion Chapter 18: Bootstrap Methods and Permutation Tests*, W. H. Freeman, 2003.

[23] V. Nair and G.E. Hinton, Rectified linear units improve restricted boltzmann machines, In *Proceedings of the 27th International Conference on International Conference on Machine Learning, ICML'10*, USA, 2010, pp. 807–814. Omnipress

[24] F.Å. Nielsen, AFINN, 2011.

[25] J. Pastor-Pellicer, F. Zamora-Martínez, S. España Boquera and M.J. Castro Bleda, F-Measure as the Error Function to Train Neural Networks, In *IWANN Proceedings*, 2013.

[26] J. Pennebaker, C. Chung, M. Ireland, A. Gonzales and R. Booth, The development and psychological properties of liwc2007, 2014.

[27] F. Pla and L.-F. Hurtado, Language identification of multilingual posts from twitter: A case study, *Knowledge and Information Systems* **51**(3) (2017), 965–989.

[28] S. Rosenthal, N. Farra and P. Nakov, SemEval-2017 task 4: Sentiment analysis in Twitter, In *Proceedings of the 11th International Workshop on Semantic Evaluation*, SemEval '17, Vancouver, Canada, Association for Computational Linguistics, 2017.

[29] X. Saralegi and I. San Vicente, Elhuyar at tass 2013, In *XXIX Congreso de la Sociedad Espaola de Procesamiento de Lenguaje Natural, Workshop on Sentiment Analysis at SEPLN (TASS2013)*, 2013, pp. 143–150.

[30] F. Sebastiani, Machine learning in automated text categorization, *ACM Comput Surv* **34**(1) (2002), 1–47.

[31] M. Taulé, M. Martí, F. Rangel, P. Rosso, C. Bosco and V. Patti, Overview of the task of Stance and Gender Detection in Tweets on Catalan Independence at IBEREVAL 2017, *In Notebook Papers of 2nd SEPLN Workshop on Evaluation of Human Language Technologies for Iberian Languages (IBEREVAL)*, Murcia (Spain). CEUR Workshop Proceedings. CEUR-WS.org, 2017, 2017.

[32] B.C. Wallace, D.K. Choe, L. Kertz and E. Charniak, Humans require context to infer ironic intent (so computers probably do, too), *In ACL (2)*, The Association for Computer Linguistics, 2014, pp. 512–516.

[33] J. Wiebe, T. Wilson and C. Cardie, Annotating expressions of opinions and emotions in language, *Language Resources and Evaluation* **1**(2) (2005).

[34] T. Wilson, J. Wiebe and P. Hoffmann, Recognizing contextual polarity in phrase-level sentiment analysis, In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing, HLT '05*, Stroudsburg, PA, USA, 2005, pp. 347–354. Association for Computational Linguistics.

[35] Y. Zhang and B. Wallace, A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification, In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 2017, pp. 253–263. Asian Federation of Natural Language Processing.

AUTHOR COPY