Semantic Representations of Words and Automatic Keywords Extraction for Sentiment Analysis of Tourism Reviews

Mauricio Toledo-Acosta[0000−0002−8047−0189], Bolívar Martínez-Zaldivar[0000−0002−2153−4031], Alexandra Ehrlich-López[0000−0001−6880−8426], Eliseo Morales-González[0000−0002−3186−0308], David Torres-Moreno[0000−0001−7844−956X], and Jorge Hermosillo-Valadez[0000−0001−9040−767X]

Centro de Investigación en Ciencias, Universidad Autónoma del Estado de Morelos,
C.P. 62209, Cuernavaca, Morelos, México
{mauricio.toledo,jhermosillo}@uaem.mx

Abstract. In this paper, we describe the methods used to submit our results to the Rest-Mex Sentiment Analysis task of the Iberian Languages Evaluation Forum 2021. Our contribution is twofold. Firstly, we propose an unsupervised method for keyword extraction, in order to construct a list of prototypical words conveying a sentiment weight (pre-score). Secondly, we substantially improved a scoring system previously proposed by us. We emphasize here the match of the pre-scores of prototypical words with the labels of the texts where they appear. The classification task is done by a SVM applied to vector representations of text entities. These vectors are obtained as a partial mean of word representations, selecting the words with the highest absolute value of the score in each text entity.

Keywords: Tourism Sentiment Analysis · Unsupervised Keyword Extraction · Scored Word Embeddings.

1 Introduction

Tourism has become a crucial source of revenue worldwide. From a socioeconomic point of view, tourism has become one of the largest and fastest growing industries in the world, extending activity online in the most recent decade [5]. In Mexico, this phenomenon is no exception, accounting for 8.7% of the national GDP, generating around 4.5 million direct jobs. However, with the COVID-19 pandemic, which began in Mexico in mid-March 2020, tourism was one of the most affected sectors in this country [4].

IberLEF 2021, September 2021, Málaga, Spain.

In this context, the use of Artificial Intelligence (AI), and in particular, Natural Language Processing (NLP), could be of great help to identify problems based on the analysis of the semantic aspects of tourists' opinions. In the case of tourism, a significant number of users express their views and opinions regarding the experience of traveling to a certain place through social media. These opinions are subjective information that represents the user's feelings, and the user's assessment associated with that experience. Online customer reviews of hotels and restaurants for tourism play a key role in decision making. Text reviews on travel websites can potentially influence destination selection. Tourists use this information to satisfy their preferences. Similarly, managers of tourism services and public institutions dedicated to promoting tourism can use this information to improve customer service. In this way, tourism content shared through social networks has become a highly influential source of information that may impact tourism in many ways. Thus, mining the opinions of tourists in search of the polarity of this opinion could influence decision making throughout the value chain and support this industry.

In this paper, we describe the methods used to submit our results to the Rest-Mex Sentiment Analysis task of the Iberian Languages Evaluation Forum 2021 [4]. For this competition edition, the sentiment analysis problem is defined as follows: "*Given an opinion about a Mexican tourist place, the goal is to determine the polarity, between 1 and 5, of the text.*" The sentiment analysis sub-task is a classification task where the participating system has to predict the polarity of an opinion issued by a tourist who traveled to the most representative places of Guanajuato, Mexico. This collection was obtained from tourists who shared their opinion on TripAdvisor between 2002 and 2020.

Sentiment analysis task in tourist texts has gained relevance in the last decade [5]. Despite the fact that most of the efforts have focused on English, there are some studies that have focused on Spanish language from Spain, and few address Spanish spoken in Mexico. These approaches are typically applied to collections taken from social networks such as tweets so that tourist texts have not been directly addressed [4]. In this sense, we can say that there are practically no linguistic resources that are suitable for processing the Mexican Spanish language, and that have been applied to the direct study of tourism opinion texts. Our work overcomes this gap and seeks to be a contribution in this direction.

In order to train a classifier to predict the polarity of each entry, we first extract keywords exclusive to each class of text entities. These keywords will ideally carry the sentiment weight in each class, and their presence may hint which class the text entity would belong to. Then, we define and implement a scoring system that assigns a value to each word in the corpus, accounting for the sentiment polarity of the word. Once these word scores are calculated, we define vector representations for text entities based on these scores and their `word2vec` embeddings. These vectors are the text features that will be feed to the classifier. The last two steps of this proposed method is a follow up to the method presented in [20].

This paper is organized as follows: In Section 2 we describe the related work. In Section 3, we fully describe the keyword extraction method, the word scoring system and text representations. In Section 4, we describe the experimental setup and balancing strategies. In Section 5, we report and discuss the results of the experiments. Finally, we conclude in Section 6.

2 Related work

Sentiment analysis is concerned with the automatic extraction of sentiment-related information from text. Traditionally, sentiment analysis has been concerned with opinion polarity (positive, negative, neutral), but in recent years there has been increasing interest in the affective dimension (angry, happy, sad, etc) [13]. The main types of sentiment analysis algorithms belong to one of these three classes: *Knowledge-based*; *Machine Learning*; and *Hybrid systems*.

Knowledge-based approaches perform sentiment analysis based on lexicons that typically incorporate sentiment word lists from many resources [9],[19]. To construct lexicons, one can use or compile Dictionaries (usually annotated by humans), or create lists of prototypical words that are further enriched with corpus data by seeking syntactic-semantic similarities.

Under the label of Machine Learning, we can find two kinds of approaches. The first can be viewed as a feature engineering problem, in which the objective is to find a suitable set of affect features in combination with an appropriate Machine Learning technique (e.g. Support Vector Machines, Classification Trees, Probabilistic models, etc.) [1], [2], [11], [5]. The second approach concerns the use of neural networks or deep learning architectures to learn *sentiment-specific word embeddings* [18], [12].

Hybrid systems combine both approaches, and our work belongs to this category. Here, we can cite [6], whose method uses basic NLP tools, a sentiment lexicon enhanced with the assistance of SentiWordNet [9, 7], and fuzzy sets to estimate the semantic orientation polarity and its intensity for sentences. More recently, [24] proposed a hybrid sentiment classification method for Twitter by embedding a feature selection method. The authors used principal component analysis (PCA), latent semantic analysis (LSA), and random projection (RP) as feature-extraction methods. They presented a comparison of the accuracy of the classification process using Support Vector Machine, Naïve Bayes, and Random Forest classifiers. They achieved performance rates on the order of 76%.

Similar to our approach, [23] use word embeddings from `word2vec` to compute the Sentiment Orientation [21] of a Weibo (Tweet). In order to attain their goal, the objective of [23] was to construct a Sentiment Dictionary, based on a basic dictionary for which each word was previously annotated by humans with its polarity and intensity. The Sentiment Dictionary was constructed by extracting the 100 words that were most similar to every word in a Weibo, using the cosine similarity measure over the embeddings of both the words of the Weibo and the words of the annotated basic dictionary. Then, the authors proposed scoring methods for computing the Sentiment Orientation for a Weibo. Another example using `word2vec` is [3], who analyzed short texts from Bengali micro-blogging websites by using a tagged corpus of comments and sentiment scoring formulae based on empirically (trial and error) tuned parameters to achieve the highest performance.

In this paper, we propose a method, based on `TextRank` [14] and frequency counting, to construct a lexicon made of keywords in documents belonging to each label. These keywords will ideally carry the sentiment weight in each class and their presence may hint which class the text entity would belong to. Contrary to the previous version of our method [20], in which the lexicon was constructed manually, in this paper, we extract the keywords in an unsupervised manner. Other keyword extraction methods can be found in the literature, such as Rapid Automatic Keyword Extraction (RAKE) [17], Degree of Fractality [16], C-Value/NC-Value [10]. Also, as an improvement of the method reported in [20], in this paper, we propose to emphasize the contribution to the score of the words of each review having congruent polarity, and decreasing their score in incongruent cases; that is, "positive"/"negative" reviews should reinforce the contribution of "positive"/"negative" words and diminish this same contribution in "negative"/"positive" reviews.

3 Methods

In this section we provide a full description of the pre-processing of the corpus, the implementation of the word scoring system and the text representations that will be used as features for the classifier.

The word scoring system is an improvement of our scoring system described in [20].

3.1 Pre-processing and Notation

First, we define the notation we will be using. The corpus is made up of entries with the following form

title$_1$	opinion$_1$	ℓ_1
...	...	...
title$_N$	opinion$_N$	ℓ_N

Table 1: The structure of the training corpus.

We denote by $\mathcal{W}$ the vocabulary appearing in the union of the fields $\{\text{title}_i\}$ and $\{\text{opinion}_i\}$. Each entity in the corpus has a label $\ell \in [1, ..., 5]$ indicating the polarity of the opinion. The labels are hugely unbalanced as shown in Figure 1. Later, in Subsection 4.1, we will describe how we deal with the unbalanced classes and give more details about the text entities.

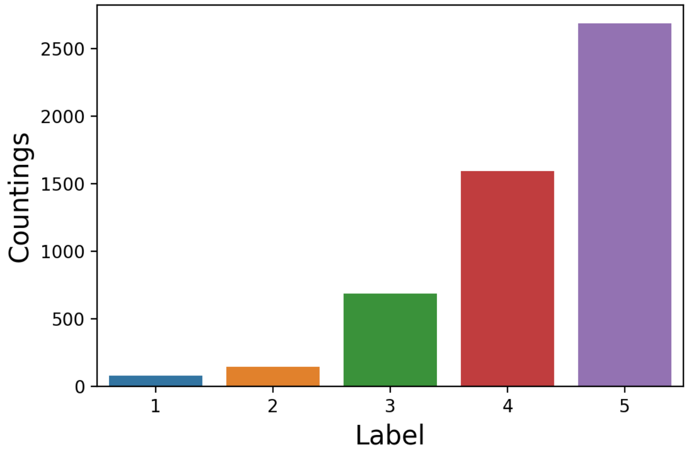

Fig. 1: Labels counting. The classes have 80, 145, 686, 1595, 2689 instances respectively.

We remove stopwords, numbers and punctuation marks from the text entries. We consider each text entity i as the concatenation of title$_i$ and opinion$_i$. We define a normalized label $\overline{\ell} \in [-1, 1]$ for each text entity, depending on the label ℓ of the entity, given by

$$\overline{\ell} = \frac{1}{2}(\ell - 3).$$

3.2 Prototypical words

We first compile a list of prototypical words W_0, which should ideally carry certain *sentiment weight*, so that their presence in a text may help to predict the label of said text. Instead of considering a manually defined list of words as in [20], we now use a non-supervised method to extract these prototypical words. Each of these words $z \in W_0$ will have a *pre-score* $\tilde{s}(z) \in [-1, 1]$.

We obtain the list W_0 using a hybrid approach. On the one hand, we use `TextRank` [14] which assigns an index $\theta(w)$ to each word w in a document. This index accounts for the importance of this word in the document. We apply `TextRank` to the each of the 5 documents made up of all text entities in each of the 5 label classes; thus, we obtain 5 lists of words Ψ_ℓ, $\ell = 1, ..., 5$. Each list consists of the 20 words with the highest index $\theta(w_j)$ in each label. If a word appears in more than one list Ψ_ℓ, we keep the word only in the list where it has the highest index. These lists contain the most important words in each label, according to `TextRank`.

Also, we apply `TextRank` to the whole collection of text entities to extract the 20 most important words in the whole corpus, we denote this set of words by Ψ^G. It is worth noting that the lists Ψ_ℓ may contain words that are *important* in the corpus as a whole; i.e. words in Ψ_ℓ that also belong to Ψ^G. Hence, we will take the following subset of each Ψ_ℓ containing words that are only relevant to each specific label

$$W_0^{\mathrm{TR}} = \bigcup_{\ell=1}^{5} \Psi_\ell \setminus \Psi^G, \tag{1}$$

We denote by $L(w) = \ell$ the index of the unique list Ψ_ℓ to which the word $w \in W_0^{\mathrm{TR}}$ belongs to.

Recall that, in each label, each word $w \in \Psi_\ell$ has a `TextRank` index given by $\theta(w)$. Let $f : [\theta_{\min}, \theta_{\max}] \to [0, 1]$ be the function applying the min-max normalization. We also could have used another keyword extraction method for this previous step, such as RAKE [17], Degree of Fractality [16] or C-Value/NC-Value [10]. However, `TextRank` showed the best performance in the classification task compared to the other methods.

On the other hand, we consider the following frequency-based pre-scoring of words. For each word $w \in \mathcal{W}$, we define $F_i(w)$ as the frequency of w in the collection of text entities with label $\ell = i$, where $i = 1, ..., 5$. We define

$$\tilde{t}(w) = \frac{1}{F(w)} \sum_{i=1}^{5} \overline{\ell} \cdot F_i(w),$$

where $F(w)$ is the frequency of w in the corpus. This function $\tilde{t}(w)$ assigns, to every word w, a weighted mean of the labels where it appears; the weights are the frequencies in each label. In order to dampen the effect of the frequencies, we define

$$t(w) = \left(1 - e^{1-F(w)}\right) \tilde{t}(w) \tag{2}$$

We consider the 30 words in $\mathcal{W}$ with the largest $|t(w)|$, and we denote this list of words by W_0^t.

Finally, the list of prototypical words W_0 is given by

$$W_0 = W_0^t \cup W_0^{\mathrm{TR}}, \tag{3}$$

and their prescores are given by

$$\tilde{s}(w) = f(\theta(z))L(w) + t(w), \tag{4}$$

where $w \in W_0$.

3.3 Scoring

Now, we describe the word scoring system. This system assigns a value to each word, accounting for the sentiment content of the word. A first version of the approach may be found in [20]. Here, we propose a substantial improvement by emphasizing the match of the pre-scores of prototypical words with the labels of the texts where they appear; in other words, we define new label-dependent scores. To achieve this, we will consider the prototypical pre-scores of Equation (4) in order to define new word scores depending on the match between the sign of the label of the text, and the sign of the a prototypical pre-score, according to a similarity criterion between words.

We denote by $\mathbf{w}$ the embedding of the word w. We consider a list of neighbours of words in W_0, denoted by W_1, defined as

$$W_1 = \{z \in \mathcal{W} \mid \exists w \in W_0,\ \mathrm{sim}(\mathbf{z}, \mathbf{w}) > \alpha\}$$

The hyper-parameter $\alpha \geq 0$ defines a closeness threshold. In other words, it defines how similar two word embeddings have to be in order to consider them as neighbours. We use $\alpha = 0.5$.

For each word $z \in W_1$, there exists another word $\sigma(z) \in W_0$ such that

$$\mathrm{sim}(z, \sigma(z)) = \max_{\zeta \in W_0} \mathrm{sim}(z, \zeta).$$

We define a label-dependent score for each word w, denoted by $\tilde{s}_\ell(\mathbf{w})$. Let $\tilde{s}_\ell(\mathbf{w}) = 0$ whenever $w \notin \mathcal{W}_0 \cup \mathcal{W}_1$.

For each word $w \in \mathcal{W}_1$ in a text entity with label ℓ consider the term $\overline{\ell} \cdot \tilde{s}(\sigma(\mathbf{w}))$. We have two cases:

1. If $\overline{\ell} \cdot \tilde{s}(\sigma(\mathbf{w})) \geq 0$, then

   $$\tilde{s}_\ell(w) = \tanh\left(\beta_1 x\right),$$

   where $x = \overline{\ell} + \mathrm{sim}(\mathbf{w}, \sigma(\mathbf{w})) \cdot \tilde{s}(\sigma(\mathbf{w}))$

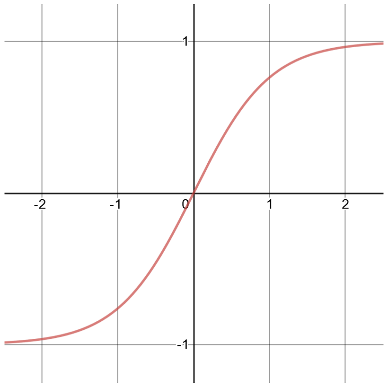

Fig. 2: The function producing the label-dependent score of a word w appearing in a text entity with the same polarity as $\sigma(\mathbf{w})$.

2. If $\overline{\ell} \cdot \tilde{s}(\sigma(\mathbf{w})) < 0$, we define

$$x = \overline{\ell} - \mathrm{sim}(\mathbf{w}, \sigma(\mathbf{w})) \cdot \tilde{s}(\sigma(\mathbf{w})),$$

and then

$$\tilde{s}_\ell(w) = \begin{cases} e^{\beta_2 x}, & \text{if } x \leq 0 \\ -e^{-\beta_2 x}, & \text{if } x > 0. \end{cases}$$

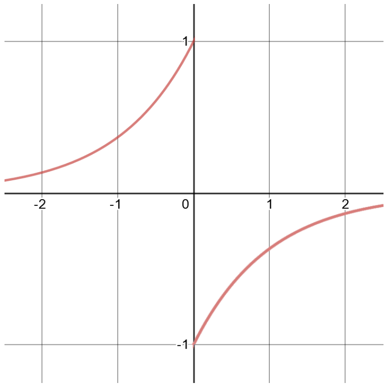

Fig. 3: The function producing the label-dependent score of a word w appearing in a text entity with opposite polarity to $\sigma(\mathbf{w})$.

These definitions yield $-1 \leq \tilde{s}_\ell \leq 1$ for any word in the corpus.

Finally, the score of the word $w \in \mathcal{W}$ is given by

$$s(w) = \frac{1}{F(w)} \sum_{\ell=1}^{5} F_\ell(w) \tilde{s}_\ell(w), \tag{5}$$

where $F(w)$ is the frequency of w in the corpus and $F_\ell(w)$ the frequency of this word in text entities with label ℓ.

3.4 Representations of Words and Text

In this subsection, we define the representations of words and text entities, using the word scoring defined in Equation (5). In the case of word representations, these are a transformation of the original `word2vec` embeddings. More precisely,

$$r(w) = s(w) \cdot \mathbf{w} \in \mathbb{R}^d \tag{6}$$

Using (6), we compute a vector representation for each text entity in the corpus. This representation is obtained as the mean of the representations of the $k = 5$ words with the most positive or most negative scores in the text. Thus, for each text entity $m = \{w_1, ..., w_n\}$ we first sort these words as

$$w_{j_1}, ..., w_{j_n}$$

such that

$$|s(w_{j_1})| > ... > |s(w_{j_n})|.$$

We compute the vector $R(m) \in \mathbb{R}^d$ defined as:

$$R(m) = \frac{1}{k} \sum_{i=1}^{k} r(w_{j_i}). \tag{7}$$

This is the representation of text entities we will use as features for the classifier.

4 Experiments

4.1 Dealing with the Unbalanced Classes

In this subsection, we describe how we deal with the unbalanced classes. Our approach is twofold: first, we modify the text entities by taking combinations of titles and opinions in order to create new instances of the first three classes; and second, we use SMOTE [8] (Synthetic Minority Oversampling Technique) to completely balance the classes. SMOTE is an algorithm that creates new instances of unbalanced classes by taking points lying in the line segments between points of unbalanced classes, thus, preserving the convex envelope of the unbalanced class.

title_{i_1}	ℓ_{i_1}
$\text{title}_{i_1} + \text{opinion}_{i_1}$	ℓ_{i_1}
...	...
title_{i_p}	ℓ_{i_p}
$\text{title}_{i_p} + \text{opinion}_N$	ℓ_{i_p}

Table 2: Text instances for classes with label 1,2.

The counting of labels is shown in Figure 1. We denote by $i_1, ..., i_p$ the indices of the entries with label $\ell = 1, 2$, and we define new text instances as shown in Table 2. In Tables 2 and 3 the addition sign denotes concatenation of strings. We denote by $j_1, ..., j_q$ the indices of the entries with label $\ell = 3, 4, 5$, we define new text instances as shown in Table 3.

$\text{title}_{j_1} + \text{opinion}_{i_1}$	ℓ_{i_1}
...	...
$\text{title}_{j_q} + \text{opinion}_N$	ℓ_{j_q}

Table 3: Text instances for classes with label 3,4,5.

Our set of text entities will be given by Tables 2 and 3. Figure 4 shows the label counting at this stage.

This set of instances is passed to SMOTE to obtain a completely balanced set of classes, each class now has 2689 instances.

4.2 Experimental setup: Training Phase

We start with the corpus, as described by Table 1. Then, we train a `word2vec` model [15] on the pre-processed text, which is the list of tokens in each text entity. We denote this trained model $\mathbf{M}_1$ with vector size $d = 100$. We also consider the pre-trained word2vec model $\mathbf{M}_2$ [22] with vector size $d = 300$.

For a word w, we denote by $\mathbf{w}$ the word embedding given by $\mathbf{M}_1$ and $\mathbf{w}'$ the word embedding given by $\mathbf{M}_2$.

We calculate the sets W_0 and W_1, and the words scoring as described in subsections 3.2 and 3.3, using the model $\mathbf{M}_1$.

We apply the balancing strategy described in Subsection 4.1 to the corpus and obtain a set of $K = 160 + 290 + 686 + 1595 + 2689 = 5420$ labeled text entities as follows

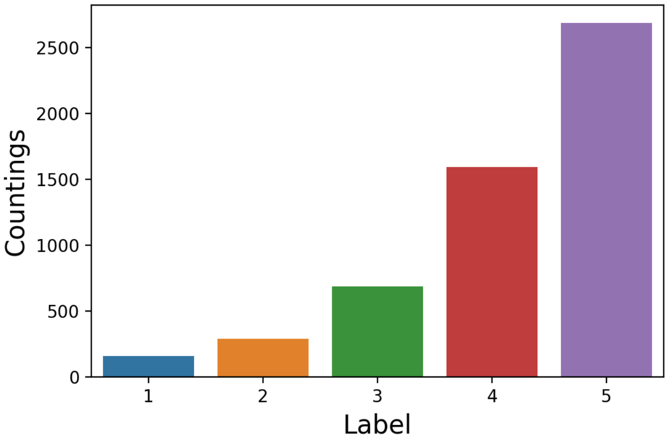

Fig. 4: Labels counting after taking the set of text entities given by Tables 2 and 3. The classes now have 160, 290, 686, 1595, 2689 instances respectively

$$\begin{array}{r|l} \text{text}_1 & \ell_1 \\ \ldots & \ldots \\ \text{text}_K & \ell_K \end{array}$$

Now, using the model $\mathbf{M}_2$, we apply the text representations described in Subsection 3.4 to obtain the following set of features

$$\begin{array}{r|l} R(m_1) & \ell_1 \\ \ldots & \ldots \\ R(m_K) & \ell_K \end{array}$$

We split this data set into training and test (validation) sets using a 4:1 ratio. We feed the training set to a support vector machine with Gaussian kernel.

4.3 Experimental setup: Test Phase

Once the models were trained, the test data, which were not labeled, were received. Thus, we describe now the experimental setup used to produce the test results. The test corpus consists of entries of the following form

title_1	opinion_1
...	...
$\text{title}_{N'}$	$\text{opinion}_{N'}$

Table 4: The test corpus.

We removed stopwords, numbers and punctuation marks from each text entry. We consider each text entity i as the concatenation of title and opinion as shown in Table 5. This will be test corpus.

$\text{title}_1 + \text{opinion}_1$
$\cdots$
$\text{title}_{N'} + \text{opinion}_{N'}$

Table 5: Text instances (test corpus) for the test phase of the classification task.

As the texts in the test data set brought a large amount of unknown words (not previously seen during training), we decided to resort to the model $\mathbf{M}_2$ to build the test vocabulary. We denote by $\mathcal{W}^T$ the corpus resulting from the intersection of the $\mathbf{M}_2$ model vocabulary and the test corpus (Table 5). In order to obtain the word representations of the words in $\mathcal{W}^T$, we take a word $w \in \mathcal{W}^T$, and consider two cases:

1. If $w \in \mathcal{W}$, then $r(w)$ is already defined.
2. If $w \notin \mathcal{W}$, the representation of w is given by $r(\gamma(w))$, where

$$\gamma(w) = \underset{z \in \mathcal{W}}{\operatorname{argmax}} \operatorname{sim}(\mathbf{w}', \mathbf{z}') \,.$$

Now, we have a word representation for every word in $\mathcal{W}^T$. Therefore, we can obtain the text entities for every instance of Table 5. These vector representations will be used as features of each instance, and they will be fed to the pre-trained classifier in order to obtain the test label predictions.

5 Results and Discussion

In this section, we report and discuss the results obtained after performing the experiments described in Section 4. The classifier performance, on the validation (training phase), and test subsets is shown in Table 6. In Figure 5 we show the confusion matrix of the classification task for the validation phase.

The Mean Absolute Error (MAE) was the primary metric used to determine the overall ranking of participants. With respect to this metric, our results were ranked in the 7th place out of the 14 different runs.

Looking at Table 6, we can see that, compared to the baseline (Majority Class), our model performs 11% better with respect to MAE but 4.5% below with respect to Accuracy. This can be explained by the large imbalance in the data set, which would also explain the poor performance of the baseline in terms of recall. Compared to the best result of the competition, our performance is 35%

Metric	**Validation** (training phase)	**Test phase**	**Baseline** (Majority Class)	**Best result**
MAE	–	0.642	0.724	0.475
Accuracy	80.81 %	49.05 %	51.35 %	56.72 %
Recall	80.27 %	32.7 %	10.27 %	49.92 %

Table 6: Prediction performance.

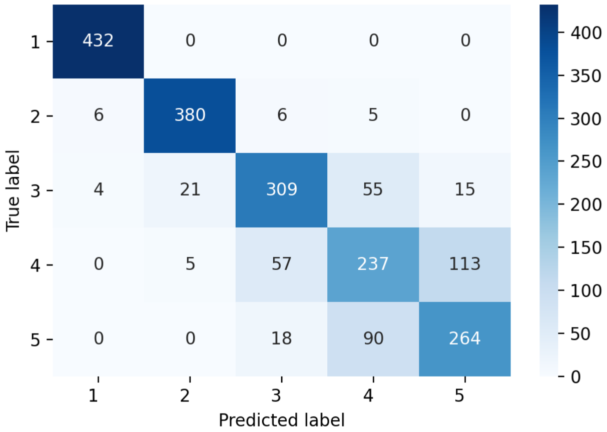

Fig. 5: Confusion matrix for the classification task during the validation phase.

and 13.5% below in terms of MAE and Accuracy respectively. Our recall is also nearly 35% below the best result. In all metrics (except for the accuracy), our method performed better than average. This is shown in Figure 6. This figure was obtained by taking the distribution of the results of all participants, in each metric reported by the competition organizers. The figure shows violin plots depicting the distribution of the results, where the average, the best result and our performance are depicted on this plot.

Finally, it is worth noting the performance drop between the train (validation) and test metrics (see Table 6). We briefly discuss the possible reasons behind this drop in performance and some strategies to improve our method.

Recall from Subsection 3.4 that features used to characterize the text entities are means of vector representations of words. As a consequence of the underlying vector addition involved, many vector representations of text end up being close to the origin, thus, making the classification task more difficult. In order to avoid this effect, the initial vectors must be *close* to each other. Their closeness will determine a threshold which is key to better understand and solve this problem.

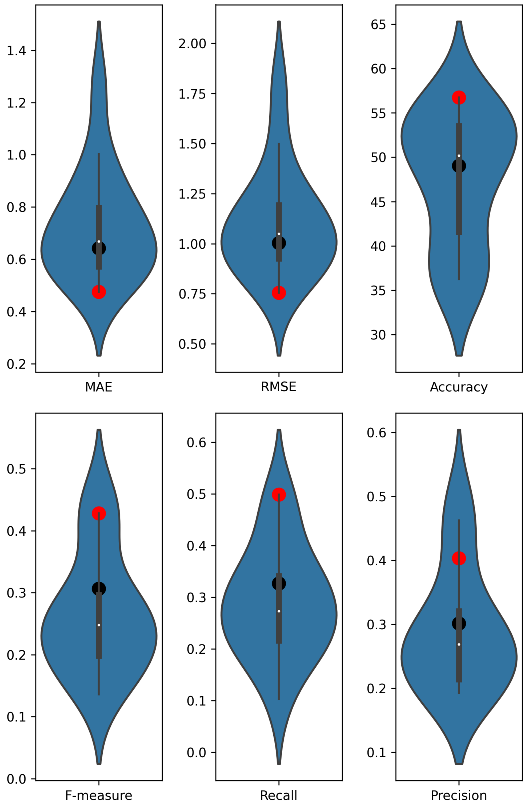

Fig. 6: Performances of all the participating runs in each of the metrics of the competition. The black dot in each metric is the performance of our method. The red point is the best performance in each metric. The white point is the average of the performances.

One possible bound is the following: If $r_0 = \min_{\mathbf{w}\in\mathcal{W}} |\mathbf{w}|$ and $\frac{1}{2} > r > 0$, then for any two vectors $\mathbf{u}, \mathbf{v} \in \mathcal{W}$, such that $\text{sim}(\mathbf{u}, \mathbf{v}) > \beta = 2\frac{r^2}{r_0^2}$, it holds $\frac{1}{2}|\mathbf{u} + \mathbf{v}| > r$, where $|\mathbf{z}|$ denotes the Euclidean norm.

On the other hand, there are some strategies to possibly improve the performance of our method when it faces unlabeled documents. As a first approach, we can restrict our attention to specific parts of speech, such as adjectives and adverbs, since these words might carry more sentiment weight than verbs or nouns.

6 Conclusions

In this paper, we presented the methods we used to address the Rest-Mex Sentiment Analysis task, of the Iberian Languages Evaluation Forum 2021 [4]. We firstly proposed an unsupervised method for keyword extraction using `TextRank` [14]. Then, we proposed a substantial improvement to our word scoring system [20]. This improvement consisted in emphasizing the contribution to the score of the words of each review having congruent polarity, and decreasing their score in incongruent cases; that is, "positive"/"negative" reviews reinforced the contribution of "positive"/"negative" words. The polarity of the words was calculated based on their similarity to prototypical words obtained from the training corpus in a unsupervised manner.

The classification task is done by a SVM applied to vector representations of text entities, obtained as a partial mean of word embeddings. The results obtained were ranked in the 7th place out of the 14 different runs. They outperformed the majority class baseline in MAE and Recall, and performed slightly better than the average final test results. Better performance may be obtained by improving the text representations, and possibly focusing on specific parts of speech, such as adjectives and adverbs. This is because these classes of words might carry more sentiment weight than verbs or nouns.

Funding

Research was partially funded by CONACYT Project A1-S-24213 of Basic Science and CONACYT grant 28268.

References

1. Abbasi, A., Chen, H.: Affect intensity analysis of dark web forums. 2007 IEEE Intelligence and Security Informatics pp. 282–288 (2007)
2. Agarwal, A., Xie, B., Vovsha, I., Rambow, O., Passonneau, R.: Sentiment analysis of twitter data. In: Proceedings of the Workshop on Languages in Social Media. p. 30–38. LSM '11, Association for Computational Linguistics, USA (2011)
3. Al-Amin, M., Islam, M.S., Uzzal, S.D.: Sentiment analysis of bengali comments with word2vec and sentiment information of words. In: 2017 International Conference on Electrical, Computer and Communication Engineering (ECCE). pp. 186–190. IEEE (2017)
4. Álvarez-Carmona, M.Á., Aranda, R., Arce-Cárdenas, S., Fajardo-Delgado, D., Guerrero-Rodríguez, R., López-Monroy, A.P., Martínez-Miranda, J., Pérez-Espinosa, H., Rodríguez-González, A.: Overview of rest-mex at iberlef 2021: Recommendation system for text mexican tourism. Procesamiento del Lenguaje Natural **67** (2021)
5. Anis, S., Saad, S., Aref, M.: Sentiment analysis of hotel reviews using machine learning techniques. In: International Conference on Advanced Intelligent Systems and Informatics. pp. 227–234. Springer (2020)
6. Appel, O., Chiclana, F., Carter, J., Fujita, H.: A hybrid approach to the sentiment analysis problem at the sentence level. Knowledge-Based Systems **108**, 110–124 (2016)
7. Baccianella, S., Esuli, A., Sebastiani, F.: Sentiwordnet 3.0: an enhanced lexical resource for sentiment analysis and opinion mining. In: Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10) (May 2010)

8. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: Smote: synthetic minority over-sampling technique. Journal of artificial intelligence research **16**, 321–357 (2002)
9. Esuli, A., Sebastiani, F.: Sentiwordnet: a high-coverage lexical resource for opinion mining. Evaluation **17**(1), 26 (2007)
10. Frantzi, K., Ananiadou, S., Mima, H.: Automatic recognition of multi-word terms:. the c-value/nc-value method. International journal on digital libraries **3**(2), 115–130 (2000)
11. Gautam, G., Yadav, D.: Sentiment analysis of twitter data using machine learning approaches and semantic analysis. In: 2014 Seventh International Conference on Contemporary Computing (IC3). pp. 437–442 (2014)
12. Li, Y., Pan, Q., Yang, T., Wang, S., Tang, J., Cambria, E.: Learning word representations for sentiment analysis. Cognitive Computation **9**(6), 843–851 (2017)
13. Mäntylä, M.V., Graziotin, D., Kuutila, M.: The evolution of sentiment analysis—a review of research topics, venues, and top cited papers. Computer Science Review **27**, 16–32 (2018)
14. Mihalcea, R., Tarau, P.: Textrank: Bringing order into text. In: Proceedings of the 2004 conference on empirical methods in natural language processing. pp. 404–411 (2004)
15. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in neural information processing systems. pp. 3111–3119 (2013)
16. Najafi, E., Darooneh, A.H.: The fractal patterns of words in a text: a method for automatic keyword extraction. PloS one **10**(6), e0130617 (2015)
17. Rose, S., Engel, D., Cramer, N., Cowley, W.: Automatic keyword extraction from individual documents. Text mining: applications and theory **1**, 1–20 (2010)
18. Tang, D., Wei, F., Qin, B., Yang, N., Liu, T., Zhou, M.: Sentiment embeddings with applications to sentiment analysis. IEEE transactions on knowledge and data Engineering **28**(2), 496–509 (2015)
19. Thelwall, M., Buckley, K., Paltoglou, G.: Sentiment strength detection for the social web. Journal of the American Society for Information Science and Technology **63**(1), 163–173 (2012). https://doi.org/10.1002/asi.21662, `https://onlinelibrary.wiley.com/doi/abs/10.1002/asi.21662`
20. Toledo-Acosta, M., Barreiro, T., Reig-Alamillo, A., Müller, M., Aroca Bisquert, F., Barrigon, M.L., Baca-Garcia, E., Hermosillo-Valadez, J.: Cognitive emotional embedded representations of text to predict suicidal ideation and psychiatric symptoms. Mathematics **8**(11), 2088 (2020)
21. Turney, P.D., Littman, M.L.: Measuring praise and criticism: Inference of semantic orientation from association. ACM Transactions on Information Systems (TOIS) **21**(4), 315–346 (2003)
22. Word2vec embeddings from sbwc (2021), `http://cs.famaf.unc.edu.ar/\actualtildeccardellino/SBWCE/SBW-vectors-300-min5.bin.gz`
23. Xue, B., Fu, C., Shaobin, Z.: A study on sentiment computing and classification of sina weibo with word2vec. In: 2014 IEEE International Congress on Big Data. pp. 358–363. IEEE (2014)
24. Zainuddin, N., Selamat, A., Ibrahim, R.: Hybrid sentiment classification on twitter aspect-based sentiment analysis. Applied Intelligence **48**(5), 1218–1232 (2018)