Deep Learning Enhanced with Graph Knowledge for Sentiment Analysis

Fernando **Lovera**[a], Yudith **Cardinale**[a], Davide **Buscaldi**[b], Thierry **Charnois**[b] and Masun Nabhan **Homsi**[a,c]

[a]*Universidad Simón Bolívar, Caracas, Venezuela*
[b]*Institut Galilée, Université Paris 13, France*
[c]*Helmholtz Centre for Environmental Research, Germany*

Abstract

The traditional way to address the problem of sentiment classification is based in Machine Learning techniques; however, these models are not able to grasp all the richness of the text that comes from different social media, personal web pages, blogs, etc., ignoring the semantic of the text. Knowledge Graphs give a way to extract structured knowledge from images and texts, in order to facilitate their semantic analysis. In this work, we propose a new hybrid approach for Sentiment Analysis based on Knowledge Graphs and Deep Learning techniques, to identify the sentiment polarity (positive or negative) in short documents, particularly in tweets. We represent the tweets using graphs, then graph similarity metrics and a Deep Learning classification algorithm are applied to produce sentiment predictions. This approach facilitates the traceability and explainability of the classification results, since it is possible to visually inspect the graphs. We compare our proposal with character n-gram embeddings based Deep Learning models to perform Sentiment Analysis. Results show that our proposal is able to outperforms classical n-gram models, with a recall up to 89% and F1-score of 88%.

Keywords

Sentiment Analysis, Knowledge Graph, Long-Short Term Memory (LSTM), Graph Similarities

1. Introduction

Users of social networks, like Twitter, make use of such platforms to express opinions, as well as emotions on any topic. In this context, intelligent classification models, to perform Sentiment Analysis, have demonstrated efficiency to predict feelings in texts and to determine users' perception of aspects of everyday life [1]. In general, the idea is to predict the results or trends of a particular topic based on sentiment [2], for example in contexts of product preferences in the market [3], film preferences [4], or political opinions [5]. However, in Twitter, predicting sentiment is a challenging problem since most of tweets do not have a well-formed grammatical structure. Nowadays, there exists an increasing interest of improving Sentiment Analysis techniques in order to reach more accurate, traceable, and explainable results, as well as better performance in real-time applications [6]. Several computational techniques can be applied to solve the problem of predicting sentiment [7]. It varies from linear models to Deep Learning [8],

CEUR Workshop Proceedings (CEUR-WS.org)

and even some of research use Linked-Data [9]. For example, the use of Machine Learning allows defining a classifier that learns to differentiate positive and negative sentiments and then determines the polarity of new texts [10]. Knowledge Graphs provide a way to extract structured knowledge from images and texts, to facilitate the semantic analysis. This knowledge is applied to determine sentiment polarities based on similarity measurements between the text's graph and pre-determined polarities' graphs. Therefore it has broad application prospects in many areas, including health, computer networks, among others [11].

In this context, we propose a new hybrid approach that considers the words of the text connected to their definition. We address the problem of predicting sentiment in short pieces of texts, in particular in tweets, by treating the words of a tweet as entities that potentially are connected with other entities through the expansion of its Knowledge Graph representation. We represent the tweets using graphs, then graph similarity metrics (tweets' graphs vs. polarities' graphs) and a Deep Learning classification algorithm are applied to produce sentiment predictions. We compare our proposal with character n-gram embeddings based Deep Learning models to perform Sentiment Analysis. Results show that our proposal is able to outperforms classical n-gram models, reaching a recall up to 89% and the an F1-score of 88%. These results demonstrate that the use of Knowledge Graphs opens the opportunity of exploring the use of semantics in the task of Sentiment Analysis, as well as facilitating the tracebility and explainability of the classification results, since these graphs can be visually inspected. Moreover, the construction of Knowledge Graphs is not affected by the size of the text or the use of dialects.

2. Related Work

Classical approaches for Sentiment Analysis on social media are based on linear techniques. The study described in [12], presents a performance analysis of several tools (AlchemyAPI, Lymbix, ML Analyzer, among others) based on linear techniques for predicting sentiment, including Support Vector Machines (SVMs), Decision Trees, and Random Forest model. Authors combine different media sources (Twitter, reviews, and news) to build the datasets. In experiments, the best tools got an accuracy of 84%, with a Random Forest models. The study concludes that the accuracy tends to decrease when the texts get longer (this might be due that the model is unable to capture long-distance dependencies between words and also unable to detect other sentiments – i.e., neutral sentiment) and the combination of the tools with a meta-classifier can enhance the accuracy of the predictions.

Some other traditional approaches supported on Machine Learning techniques, are based on language features. Pang and Lee [13], investigate the performance of various Machine Learning techniques, such as Naive Bayes, Maximum Entropy, and SVM in the domain of opinions on movies. From their analysis, they obtained 82.9% accuracy in their model, using SVM with unigrams. Normally, the features used for a sentiment classifier are extracted by Natural Language Processing (NLP). These NLP techniques are mainly based on the use of n-grams, nonetheless it is also popular the use of bag-of-words for this task. Many studies show relevant results using the bag-of-words as a text representation for object categorization [14, 15].

Researchers exploit the NLP topics and use them to implement Deep Learning models based on neural networks that have more than three layers. In general, in these studies, predicting sentiment is a task that is performed by Deep Learning models. These models include Con-

Table 1

Deep Learning techniques and results

Work	Model	Dataset	Result
Yanagimoto et al., 2013 [19]	DNN	T&C New	F-score of 90.8% of accuracy
Li et al., 2014 [20]	RNDM	2270 movie reviews from websites	Accuracy of 90.8%
Severyn and Moschitti, 2015 [16]	CNN	Semeval-2015	F-measure score sub-task A: 84.19% and sub-task B: 64.69%
Yanmei and Yuda, 2015 [17]	CNN	1000 micro-blog comments	Statistical model with average of 85.4% of accuracy
Silhavy et al., 2016 [21]	HBRNN	150,175 labelled reviews from 1500 hotels	HBRNN outperformed the rest of the methods.
Arras, Leila, Montavon, 2017 [18]	RNN	11,855 single sentences from movies review	Acurracy of 82.9% for binary classification (positive and negative).

volutional Neural Networks (CNN) [16] [17], Recursive Neural Networks (RNN) [18], Deep Neural Networks (DNN) [19], Recursive Neural Deep Model (RNDM) [20], and Deep Believe Networks. Different researches combine models in their study, and then these models are named Hybrid Neural Networks, for example the Hierarchical Bidirectional Recurrent Neural Network (HBRNN) [21] . In Table 1, we compare some relevant and recent works based on Deep Learning techniques, in terms of performance. As shown in Table 1, results in different studies demonstrate how good the Deep Learning techniques perform with different datasets. In particular, models based on RNN have obtained results above 80% of accuracy.

An emerging strategy to perform Sentiment Analysis, is based on using graphs as feature representation, which show an enhancement in text-mining applications. Nonetheless, one of the main challenges with the graphs is its structure and its semantic relations. Only few studies have approached this strategy. In [22], authors use a co-occurrence graph that represents relationships among terms of a document; they use centrality measures for extraction of sentiment words that can express the sentiment of the document; then, they use these words as features for supervised learning algorithms and obtain the polarity of the new document. In [23], authors use graphs to represent the sequences of words on a document. They apply graph similarity metrics and classification algorithms, such as SVM to predict sentiment. In [24], authors try to leverage a deep graph-based text representation of sentiment polarity by combining graphs and use them in a representation learning approach. Table 2 shows a comparative evaluation of these studies. Even-though the approach of graph-based techniques uses a different data structure for features and its learning pattern is different, these techniques reach to results as good as the techniques based on Deep Learning.

Our main contribution is related to Knowledge Graphs, whose main advantage is that their construction is not affected by the size of the text or the use of dialects and can be visually inspected. Results of previous studies suggest that the use of Deep Learning, for the task of sentiment prediction, produces accurate models. Therefore, combining Knowledge Graphs with Deep Neural Networks allows obtaining a powerful model, able to produce accurate, traceable, and explainable results in predicting sentiment labels.

3. Knowledge Graphs: Preliminaries

Knowledge Graphs (KG) are inspired in interlinking data, modeling unstructured information in a meaningful way. A KG is also known as a Knowledge Base (technology that stores

Table 2
Graph-based techniques and results

Work	Model	Dataset	Result
E. Castillo, O. Cervantes, 2015 [22]	Co-occurrence graphs	SemEval 2015	76% for positive and 68.04% for neutral classes
J. Violos, 2016 [23]	Word-graph model based	Twitter dataset	75.07% of accuracy.
K. Bijari, H. Zare, 2019 [24]	Sentence-level graph-based text representation	IMDB dataset IMDB dataset	88.31% for negative and 86.60% for positive classes
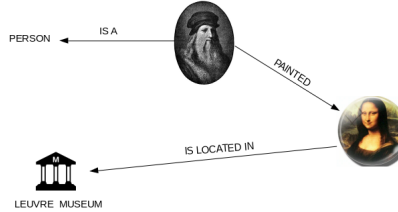
Figure 1: Extended graph with two entities and its relationship.

complex unstructured or structured information/data), and it is defined as a *network* in which the nodes indicate entities and the edges indicate their relation. DBpedia[1], YAGO[2], SUMO[3] are three examples of huge KG that have been released on the past decade, since they have become an outstanding resource for NLP applications, such as Question-Answering (QA) [25]. Formally, the essence of KGs are triples, and its general form is explained in Def. 1.

Definition 1. ***Knowledge Graph*** *(KG). A knowledge graph, denoted as KG, is a triple $KG = (E, R, F)$, where $E = \{e_1, e_2, ..., e_n\}$ is a set of entities, $R = \{r_1, r_2, ..., r_n\}$ represents a set of binary relations, and $F \supseteq ExRxE$ represents the relationships between entities (fact triple set).*

Since the KG stores real-world information in RDF-style triplets[4], such as ($head, relation, tail$), it can be employed to support the representation of knowledge in different applications. For example, in the medical industry it can provide a clear visual representation of diseases as mentioned in [26]; in e-commerce, it maps clients' purchase intentions with sets of candidate products as mentioned in [27]. The construction of KG is based on its entities (also known as nodes) and their interrelation with other objects, organized in the form of a graph. Each entity is able to share its knowledge with other entities. Figure 1 shows an example of a KG representing the following triples: ('Da Vinci', 'is a', 'Person'), ('Da Vnci', 'painted', 'Mona Lisa'), ('Mona Lisa', 'is located in', 'Louvre').

4. Sentiment Analysis based on KG: Our Proposal

The proposed method is inspired in other works that are related to the representation of text as graphs [23]. We use updated techniques, such as Knowledge Graphs and Deep Learning, that

---

[1] http://aksw.org/Projects/DBpedia.html

[2] https://yago-knowledge.org/

[3] http://www.adampease.org/OP/index.html

[4] http://www.w3.org/TR/rdf11-concepts/

are different from the traditional n-gram representation with the aim of better understanding of the sentiment, which is related to the representation of text as graphs. KG and vector representation of text lead the Sentiment Analysis task in tweets.

Each tweet and the sentiment polarities (positive and negative) are represented as a KG (see Def. 1). They are built from a subset of the training set – i.e., one part of the training set is taken to produce the KG representing sentiment polarities (one for positive, one for negative) and the other produces KG for every tweet. The similarity comparison between the sentiment polarities and the individual KG produces vectors used to train a classifier model, in this case an LSTM network and a Bi-LSTM network. On the test set, each new tweet to be classified is transformed to a KG and then we measure its similarity to the polarity graphs.

Deep Learning models combined with the explicit semantic of texts represented by KG and the similarity metrics, facilitate the traceability and explainability of the classification results, since these graphs can be visually inspected, while the accuracy of the results is ensured.

4.1. Dataset

The dataset used for the implementation of our approach is Sentiment140[5], which contains 1,600,000 tweets in English, labeled positive, negative, and neutral, as well as meta-data describing each tweet. For the purposes of Sentiment Analysis, only the content of the tweet text along with its tag is required. However, the neutral tag is not considered in this work, because we are only interested in tweets that contain a polarity – i.e., that express a sentiment.

The dataset was divided in two pieces, one for training the model and the rest for testing it. The first 80% of the dataset was destined to training and the other 20% for testing. With this last 20%, we repeated the process of transforming the tweets to small KG and then measuring the similarities with the same metrics explained in Section 4.4. Then, we ran the model with these new similarities as input and we obtained a predicted label as an output. With these predicted labels we computed the $F1$ score of the model to measure its efficiency.

4.2. Pre-processing

The pre-processing of the raw text involves *removal of characters* that do not help to detect Sentiment, such as HTML characters, elements that contain the special character "@", URLs and hashtag; *transformation to lower case*, since the analysis is sensitive to case letters; *expansion of English contraction*, such as don't, won't; *elimination of long words*, we eliminate tweets that contain words longer than 45 characters; *elimination of stop words*, such as the, a, this. Nonetheless, it does not include the lemmatization, since we do not require the normal form of the words, but the inflected form of the words.

4.3. Graph construction

The automatic construction of KG demands finding a way to recognize the entities and their relationships. Following steps describe the KG construction process:

1. Sentence segmentation: Split the text (a tweet, in this case) into sentences. Therefore, we achieve to have exactly one object and one subject per sentence.

2. Entities extraction: An entity is a Noun or a Noun Phrase (NP). We can extract a single word entity from a sentence with the help of Parts of Speech (PoS) tags. For example, in the sen-

---
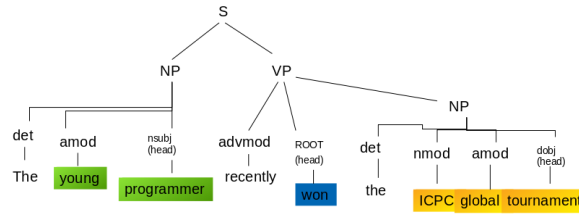
Figure 2: Tree parser representation of the sentence: "The young programmer recently won the ICPC global tournament".
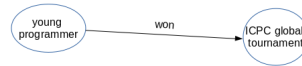

Figure 3: Knowledge Graph representation of the sentence: "The young programmer recently won the ICPC global tourmanet"

tence "Rafael won the first prize", the PoS tags classify "Rafael" as a Nominal Subject (`nsubj`), and "prize" as a Direct Object (`dobj`), both of them are syntactic dependency tags that contain the information needed for the formation of the KG entities. For most of the sentences, the use of PoS tags alone are almost enough. Nonetheless, for some sentences the entities span in multiple words; therefore, the syntactic dependency tags are not sufficient. For example, in the sentence "The 42-year-old won the prize", "old" is classified as the `nsubj`; nonetheless, we would like to extract "42-year-old" instead. The "42-year" is classified as adjectival modifier (`amod`) – i.e., it is a modifier of "old". Something similar happens with the `dobj`. In this case, we do not have modifiers but compound words (collection of words that form a new term with different meaning); for example, instead of having the word "prize" in our phrase, we have "ICP global tournament". The PoS tags only retrieve "tournament" as the `dobj`; however, we want to extract the compound words. These words are: "ICP" and "global", as well. Hence, we need to extract the subjects and objects *along with* its punctuation marks, modifiers, and also the compound words. Therefore, we parse the dependency tree of the sentence. For this, we extract the modifier of the subject (`amod` in the dependency tree). For example, for the sentence "The young programmer recently won the ICPC global tournament", its parse tree can give information of the PoS tags, defining the determiners, verbs, modifiers, subjects, and grouping them in verb phrases (VP) and noun phrases (NP) sub-trees, as shown in Figure 2. In this example, the entity "young programmer" is identified as a subject and "ICPC global tournament" as the object of the sentence. Therefore, for these cases, it is needed to include modifiers and adverbs in the entities of the KGs.

3. Relationships extraction: To extract the relation between nodes, we assume that it refers to the main verb of the sentence. Therefore, the main verb represents the relationship between two entities. In the sentence in Figure 2, the predicate is "won", which is also tagged as "ROOT" or main verb.

4. Building the knowledge graph: In order to build the KG, it is necessary to work with a network in which the nodes are the entities and the edges between the nodes represent the relations between the entities. It needs to be a directed graph, which means that the relation between two nodes is unidirectional.

In Figure 3, we show a KG representation of the parser tree presented in Figure 2. The entities
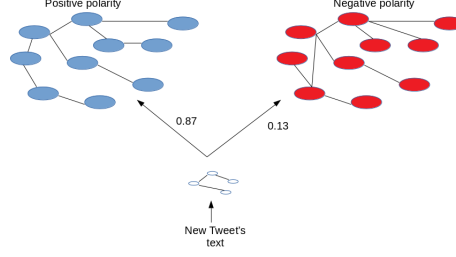
Figure 4: Similatity measurements

of this small KG are given by "young programmer" and "ICPC global tournament", the relation between the entities is the main verb, in this case the verb "won". In this way, we construct the KG given a sentence as an input. The two polarity KG are constructed from positive tagged tweet sets and another negative tagged set, respectively, taken from the training dataset. Then, with the other part of the training dataset a KG is generated for each tweet.

4.4. Graph similarities

The similarity between the graph that represents the tweet and the polarity graphs, expresses how the tweet is related to one polarity or to another. It indicates that if the graph of a tweet is more similar to the positive polarity graph, this tweet is considered as expressing a positive sentiment (as shown in Figure 4). Each similarity distance means the percentage of correlation between the new graph and the polarity graphs. To calculate the similarity of a tweet KG, in this work, we use three graph similarity metrics:

1. *Containment similarity measurement*: It expresses the percentage of common edges between the graphs, taking the size of the smaller graph as the factor of this measurement. Given KG_t as the KG of a tweet and KG_s the KG of a polarity, Eq. (1) calculates the containment similarity measurement between these two graphs, where e refers to the edges of the graph.

$$containment(KG_t, KG_s) = \frac{\sum_{e \in KG_t} \mu(e, KG_s)}{min(|KG_t|, |KG_s|)} \tag{1}$$

2. *Maximum common sub-graph similarity measurement*: Given two graphs, KG_1 and KG_2, the maximum common sub-graph of them, is a sub-graph of both graphs, such that there is no another sub-graph of KG_1 and KG_2 with more nodes [28]. The measurement of maximum common sub-graph is based on the sizes of common sub-graph between the two graphs. Detecting the maximum common sub-graph between two graphs with labeled nodes is a linear problem. Eq. (2) is used to calculate the maximum common sub-graph between KG_t and KG_s, where MCSN is a function that returns the number of nodes that are contained in the maximum common sub-graph of these graphs.

$$max\_subgraph\_nodes(KG_t, KG_s) = \frac{MCSN(KG_t, KG_s)}{min(|KG_t|, |KG_s|)} \tag{2}$$

3. *Maximum common sub-graph number of edges*: It takes into account the number of common edges that are contained in the maximum common sub-graph instead of the nodes in common. This is reflected on Eq. (3), where the Maximum Common Subgraph of Edges (MCSE) is the number of edges contained in the maximum common sub-graph.

$$max\_subgraph\_edges(KG_t, KG_s) = \frac{MCSE(KG_t, KG_s)}{min(|KG_t|, |KG_s|)} \tag{3}$$

The $MCSE(KG_t, KG_s)$ is defined as the quantity of edges contained in the maximum common sub-graph and maintain the same direction in both KG_t and KG_s. It is also possible to measure the similarity using the edges but not taking into account if the direction is maintained, this is what we measured in the Equation (4), where $MCSUE(G_t, G_s)$ represents the Maximum Common Subgraph of Undirected Edges (i.e., total number of edges contained in the maximum common sub-graph *regardless* their direction). Thus, we obtain eight similarity metrics (four for each polarity). These metrics form the vectors used to train a classifier and out predictions.

$$max\_subgraph\_edges(KG_t, KG_s) = \frac{MCSUE(KG_t, KG_s)}{min(|KG_t|, |KG_s|)} \qquad (4)$$

4.5. Dimensionality reduction

Typical Machine Learning methodologies imply feature selection or dimensionality reduction, in order to improve metrics, such as accuracy or the mean squared error [29]. In this work, the mutual information criteria for graphs is used. This criterion is applied after the creation of the KG for the sentiment classes and tweets in the training process. Dimensionality reduction discards edges so the graphs become smaller, therefore computational resources are optimized.

We use the edge filter with the containment similarity metric. The mutual information criterion between the term (t) (edge) and category (c) (sentiment class) is defined as in Eq. (5) [30], where A is the number of times that t exists in the graph of sentiment polarities and in the graphs of the tweets of the second part of the training process; B is the number of times that t does not exist in the graph of sentiment polarities but it exists in the graphs of the tweets of the second part of the training process; C is the number of times the second part of the training process expresses the sentiment of c and do not contain t; N is the total number of documents in the second part of the training process.

$$PI(t, c) \approx \log\left(\frac{A * N}{(A + C) * (A + B)}\right) \qquad (5)$$

Finally, we calculate the average factor, $I_{avg}(t, c)$, to know the contribution of the edge to the sentiment graphs. For every edge, Eq. (6) computes the global mutual information; where $P_r(S^i)$ is the percentage of tweets from the second part of the training process that express the sentiment S^i. Naturally, if this contribution for a certain edge is smaller than a certain threshold (chosen accordingly to the problem or criteria of the researcher), then the edge is discard.

$$I_{avg}(t, c) = \sum_{i \in (+,-)} P_r(S^i) * PI(t, c) \qquad (6)$$

4.6. Tweet's classification

Tweets have a limitation in length, thus they are short pieces of text. Therefore, it is necessary to use a classification model able to deal with this characteristic and also deal with the long dependencies in the words in a tweet (i.e., dependencies of words that are not strictly adjacent words or morphemes). In this sense, our work is based on LSTM and Bi-LSTM networks, that are able to manage these text conditions [31].

In Figure 5, we illustrate the steps for the construction of the training set and the respective KG. At the 1st step, we have the training set (a percentage of the whole dataset). At the 2nd
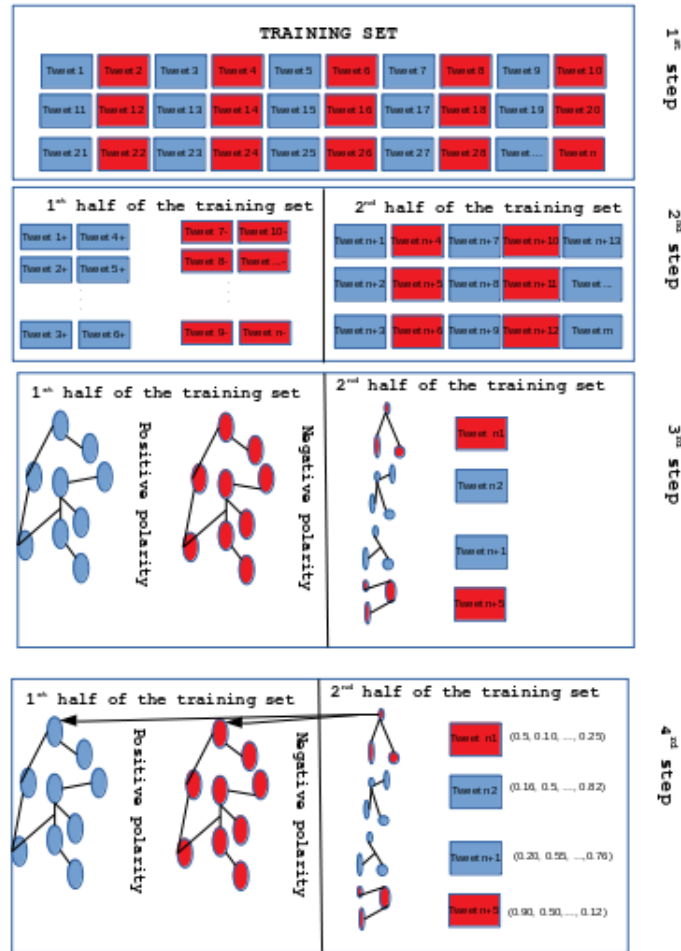
Figure 5: Transformation stages of a dataset to its vector form in the Knowledge Graph Method.

step, we divide it into two parts. The first part is used to build the KG that represents the sentiment classes (positive and the negative polarities); we obtain one positive polarity KG and one negative polarity KG, derived from tweets tagged as positive and negative, respectively (3rd step). With the second part, in the 3rd step, we create a KG for every tweet following the process explained in Section 4.3. Then, we measure the similarity between every KG representing the tweet with the positive and negative polarities, resulting in a vector of eight dimensions (four measurements against the positive KG and four against the negative KG – see Section 4.4) and a label of a sentiment (4rd step). Naturally, we take similar steps as in Figure 5, to create a test set. The classifier's input vectors contain the eight different similarity measurements.

5. Results

In order to evaluate our approach, we implemented an LSTM enhanced with the KG version (LSTM with KG) and a Bi-LSTM enhanced with the KG model (Bi-LSTM with KG). We also implemented an LSTM (Character n-gram based LSTM) and a Bi-LSTM (Character n-gram based

Table 3
Results for experiments.

Model	$F1 - score$	Precision	Recall
LSTM with KG	**0.884**	**0.880**	**0.890**
Bi-LSTM with KG	0.757	0.690	0.840
Character n-gram based LSTM	0.849	0.840	0.860
Character n-gram based Bi-LSTM	0.852	0.851	0.856

Bi-LSTM) classical versions as baselines, representing state of the art techniques for learning long distance dependencies. For the traditional character n-gram embeddings versions, the original input is the sentence, and then using max-pooling the more important n-grams are extracted. Combining the maximum values of each layer of the network in only one vector to finally use a `Sigmoid` function to make the prediction.

For the implementation of the four versions, we use Python 3 with the Keras library and TensorFlow, using GoogleGPUs[6]. GoogleGPUs is a free cloud based version of a Jupyter notebook, that allows optimizations for big arrays in Colab Notebooks. The hardware available in these notebooks is a 12GB NVIDIA Tesla K80 GPU, that can be used up to 12 hours continuously. The hyperparameters for all versions were chosen empirically and they are: (i) Batch size is 500, *Max length* of the input sequence is 280, since 280 is the maximum length of a tweet, the embedding matrix is initialized to 0; (ii) Embedding layer created within the Sequential model given by Keras, (iii) SpatialDropout1D layer to delete 1D feature maps, promoting independence between features; and (iv) LSTM and Bi-LSTM layer given by Keras.

In our experiments we observed that using the Dimensionality Reduction helps to improve the precision of the method, since it decreases the amount of edges in the KG. Results are shown in Table 3 in terms of $F1$ score, precision, and recall. We can appreciate that our approach based on KG has a higher score with LSTM (LSTM with KG in Table 3) than with Bi-LSTM (BI-LSTM with KG in Table 3); nonetheless, this is a more complex and fast convergence model.

Comparing both versions of our model against the two classical Deep Learning models, we appreciate that the Character n-gram based Bi-LSTM model presents the worst results. We got better results with LSTM, for both KG and Character n-gram based LSTM. For both cases, $F1$ scores are quite similar, being results of LSTM with KG better than results of Character n-gram based LSTM; actually, precision and recall values of LSTM with KG are the best among the four models, meaning that we built a model able to make correct predictions and the ratio of correctly predicted values tends to be high as well. The results show that the combination of KG with a classifier is appropriate for detecting the feelings expressed in short texts.

Besides results are in the state-of-the-art for Sentiment Analysis, the advantages of our approach also lie in several aspects: (i) the facility of crating KG that can be visually inspected, conducting to more traceable and explainable classification results; (ii) the explicit semantic of texts represented by KG, which capture grammar, long-distance dependencies, and neologism, and are suitable for detecting sentiment in micro-blogging texts; and (iii) this approach represents a way of doing Sentiment Analysis that can be used in other contexts, for example to recognize topics based on the semantic provided by KG and expanded with the use of Linked-Data, which are indeed complex KG.

---

[6]https://colab.research.google.com/notebooks/intro.ipynb#recent=true

6. Conclusion

In this work, we propose an innovative Sentiment Analysis method, combining Knowledge Graphs with Deep Learning techniques. Knowledge Graphs can capture structural information of a tweet and also part of its meaning. Our proposed model is based on the use of graph techniques and similarity metrics between graphs. The result of the comparison of graphs (i.e., graph similarity measurements) is a vector, which is fed later to the neural networks, which recognize the polarity of the sentiment. The recognition of entities and their connections is crucial to our approach, the use of PoS tagging allows performing this operation. Results are comparative with the ones encountered on literature for the same problem. Our methodology is simple, suitable for detecting sentiment in micro-blogging texts, provides more informed and traceable Deep Learning algorithms that produce more accuracy scores, and with the potential to be expanded with the use of Linked-Data, thanks to the properties of the Knowledge Graphs.

References

[1] M. M. Mostafa, More than words: Social networks' text mining for consumer brand sentiments, Expert Syst. Appl. 40 (2013) 4241–4251.

[2] E. Cambria, Affective computing and sentiment analysis, IEEE Intelligent Systems 31 (2016) 102–107.

[3] M. A. Mirtalaie, O. K. Hussain, E. Chang, F. K. Hussain, Sentiment analysis of specific product's features using product tree for application in new product development, in: Internat. Conf. on Intelligent Networking and Collaborative Systems, 2017, pp. 82–95.

[4] E. Chu, D. Roy, Audio-visual sentiment analysis for learning emotional arcs in movies, in: IEEE Internat. Conf. on Data Mining, 2017, pp. 829–834.

[5] D. J. S. Oliveira, P. H. d. S. Bermejo, P. A. dos Santos, Can social media reveal the preferences of voters? a comparison between sentiment analysis and traditional opinion polls, Journal of Information Technology & Politics 14 (2017) 34–45.

[6] S. Rosenthal, N. Farra, P. Nakov, Semeval-2017 task 4: Sentiment analysis in twitter, in: Internat. Workshop on Semantic Evaluation, 2017, pp. 502–518.

[7] D. M. E.-D. M. Hussein, A survey on sentiment analysis challenges, Journal of King Saud University-Engineering Sciences 30 (2018) 330–338.

[8] Z. Jianqiang, G. Xiaolin, Z. Xuejun, Deep convolution neural networks for twitter sentiment analysis, IEEE Access 6 (2018) 23253–23260.

[9] J. F. Sánchez-Rada, M. Torres, C. A. Iglesias, R. Maestre, E. Peinado, A linked data approach to sentiment and emotion analysis of twitter in the financial domain., in: WaSABi-FEOSW@ ESWC, 2014.

[10] M. Taboada, Sentiment analysis: An overview from linguistics, Annual Review of Linguistics 2 (2016).

[11] X. Wang, X. He, Y. Cao, M. Liu, T.-S. Chua, Kgat: Knowledge graph attention network for recommendation, in: Int. Conf. on Knowl. Discover & Data Mining, 2019, pp. 950–958.

[12] M. Cieliebak, O. Dürr, F. Uzdilli, Potential and limitations of commercial sentiment detection tools., in: ESSEM@ AI* IA, 2013, pp. 47–58.

[13] B. Pang, L. Lee, et al., Opinion mining and sentiment analysis, Foundations and Trends® in Information Retrieval 2 (2008) 1–135.

[14] J. Krapac, J. Verbeek, F. Jurie, Modeling spatial layout with fisher vectors for image categorization, in: Internat. Conf. on Computer Vision, IEEE, 2011, pp. 1487–1494.

[15] K. Sikka, T. Wu, J. Susskind, M. Bartlett, Exploring bag of words architectures in the facial expression domain, in: European Conference on Computer Vision, 2012, pp. 250–259.

[16] A. Severyn, A. Moschitti, Twitter sentiment analysis with deep convolutional neural networks, in: Int. Conf. on Research and Develop. in Infor. Retrieval, 2015, pp. 959–962.

[17] L. Yanmei, C. Yuda, Research on chinese micro-blog sentiment analysis based on deep learning, in: Internat. Symp. on Comput. Intelligence and Design, 2015, pp. 358–361.

[18] L. Arras, G. Montavon, K.-R. Müller, W. Samek, Explaining recurrent neural network predictions in sentiment analysis, arXiv preprint arXiv:1706.07206 (2017).

[19] H. Yanagimoto, M. Shimada, A. Yoshimura, Document similarity estimation for sentiment analysis using neural network, in: Int. Conf. Computer and Inf. Science, 2013, pp. 105–110.

[20] C. Li, B. Xu, G. Wu, S. He, G. Tian, H. Hao, Recursive deep learning for sentiment analysis over social data, in: Int. Con. on Web Intell. and Intelligent Agent Tech., 2014, pp. 180–185.

[21] R. Silhavy, R. Senkerik, Z. K. Oplatkova, P. Silhavy, Z. Prokopova, Artificial intelligence perspectives in intelligent systems, in: Computer Sc. On-line Conf., 2016, pp. 249–261.

[22] E. Castillo, O. Cervantes, D. Vilarino, D. Báez, A. Sánchez, Udlap: sentiment analysis using a graph-based representation, in: Int. Workshop SemEval, 2015, pp. 556–560.

[23] J. Violos, K. Tserpes, E. Psomakelis, K. Psychas, T. Varvarigou, Sentiment analysis using word-graphs, in: Int. Conf. on Web Intelligence, Mining and Semantics, 2016, pp. 1–9.

[24] K. Bijari, H. Zare, E. Kebriaei, H. Veisi, Leveraging deep graph-based text representation for sentiment polarity applications, arXiv preprint arXiv:1902.10247 (2019).

[25] X. Huang, J. Zhang, D. Li, P. Li, Knowledge graph embedding based question answering, in: Internat. Conf. on Web Search and Data Mining, 2019, pp. 105–113.

[26] M. Rotmensch, Y. Halpern, A. Tlimat, S. Horng, D. Sontag, Learning a health knowledge graph from electronic medical records, Scientific reports 7 (2017) 1–11.

[27] K. K. Teru, W. L. Hamilton, Inductive relation prediction on knowledge graphs, arXiv preprint arXiv:1911.06962 (2019).

[28] H. Bunke, On a relation between graph edit distance and maximum common subgraph, Pattern Recognition Letters 18 (1997) 689–694.

[29] S. Chagheri, S. Calabretto, C. Roussey, C. Dumoulin, Feature vector construction combining structure and content for document classification, in: Internat. Conf. on Sciences of Electronics, Technologies of Information and Telecommunications, 2012, pp. 946–950.

[30] Y. Xu, G. J. Jones, J. Li, B. Wang, C. Sun, A study on mutual information-based feature selection for text categorization, J. of Computat. Informat. Systems 3 (2007) 1007–1012.

[31] B. Wang, W. Liu, G. Han, S. He, Learning long-term structural dependencies for video salient object detection, IEEE Transactions on Image Processing 29 (2020) 9017–9031.