A Knowledge-based Weighted KNN for Detecting Irony in Twitter

Delia Irazú Hernández Farías[1], Manuel Montes-y-Gómez[1], Hugo Jair Escalante[1], Paolo Rosso[2], and Viviana Patti[3]

[1] Instituto Nacional de Astrofísica, Óptica y Electrónica (INAOE), Tonantzintla, Mexico {dirazuherfa,mmontesg,hugojair}@inaoep.mx
[2] PRHLT Research Center, Universitat Politècnica de València, Valencia, Spain prosso@dsic.upv.es
[3] Dipartimento di Informatica, University of Turin, Turin, Italy patti@di.unito.it

Abstract. In this work, we propose a variant of a well-known instance-based algorithm: WKNN. Our idea is to exploit task-dependent features in order to calculate the weight of the instances according to a novel paradigm: the Textual Attraction Force, that serves to quantify the degree of relatedness between documents. The proposed method was applied to a challenging text classification task: irony detection. We experimented with corpora in the state of the art. The obtained results show that despite being a simple approach, our method is competitive with respect to more advanced techniques.

Keywords: Instance-based algorithm · WKNN · Irony Detection.

1 Introduction

Social media are nowadays an important communication channel where people express their opinions, thoughts, and ideas. Analyzing such kind of content is the main aim of Sentiment Analysis (SA). An important challenge to address in SA is the presence of figurative language devices such as irony [12]. The most common definition of irony refers to an utterance by which the words are used with the intention of communicating the opposite of what is literally said [10].

During the last years the interest in detecting the presence of ironic content in social media has grown significantly especially on Twitter. Several computational linguistics approaches have been proposed to deal with irony detection. Mainly, they take advantage of different features attempting to capture the presence of ironic content together with machine learning algorithms. For further details on state-of-the-art irony detection methods see [14]. Moreover, detecting the presence of irony has been the aim of some shared tasks in both languages English [7, 22] and Italian [1, 2].

In this paper, we are proposing an approach for addressing irony detection that exploits a variation of WKNN. Our proposal is about a new schema for calculating the weights. We took advantage of knowledge-based features together with a novel paradigm called Textual Attraction Force (that allows us to calculate the relatedness between two documents) to assign the weights. With the purpose of evaluating the performance of the proposed model, we decided to apply it for detecting irony in Twitter. The performance of the proposed approach was assessed over a set of state-of-the-art corpora. The obtained results outperform those from well-known classifiers validating the usefulness of our method. It is worth mentioning that in spite of the fact that the proposed model is simple, the obtained results are quite encouraging when compared with both traditional classifiers and more sophisticated techniques such as deep learning.

Summarizing, the main contributions of this paper are: (i) We propose a new variant of WKNN that exploits knowledge-based features by means of the Textual Attraction Force in order to classify unlabeled instances according to a weighted voting KNN; and, (ii) We evaluated our approach on irony detection. A set of corpora of the state of the art was used for experimental purposes. The obtained results demonstrate the usefulness of our approach for irony detection.

The paper is organized as follows. Section 2 introduces the basis of the proposed approach. Section 3 describes the knowledge-based features exploited to characterize irony. Section 4 describes the experimental setting and results. Finally, Section 5 draws some conclusions.

2 A Knowledge-based Weighted KNN

kNN (*k-Nearest Neighbors*) is one of the oldest and simplest classification approaches based on the use of the nearest neighbor rule. It assumes that the class of an unlabeled instance is assigned by a majority voting between the classes of its k nearest neighbors. kNN belongs to the family of the instance-based learning algorithms [15]; besides, it has several advantages: it is simple, effective, intuitive, and it has a competitive classification performance in many domains.

With the aim of improving kNN, different variations have emerged, where the main idea is to incorporate a weighting schema, namely WKNN. One approach, often known as *Distance-based WKNN (DWKNN)*, consists in weighting close neighbors more heavily according to their distances to the test instance [6, 9].

In this paper, we propose a variation of the WKNN called Knowledge-based Weighted KNN (hereafter KBWKNN). We attempt to exploit a novel paradigm: the Textual Attraction Force (TAF), in order to calculate the weight of the instances. The idea behind TAF is to emulate the Newton's gravitational force where the attraction between two objects depends on their masses and the distance between them. The higher their masses and the lower their distance, the greater the attraction force between them is.

Current text classification approaches are based on finding similar documents, considering that all the instances have the same relevance for building a given classifier. TAF represents a change in the way by which the relationship between documents is calculated. By using TAF for text classification, we could evaluate the relationship between two documents by considering not only their similarity (for example in terms of vocabulary) but also their relevance (or mass). Therefore, defining the mass function is crucial for identifying similar documents. The mass (or relevance) of an instance refers to a subjective and dependent aspect of the problem in hand. In this paradigm, the main hypothesis is that there are differences regarding the relevance of the objects, i.e., some objects are more relevant than others. Therefore, relevant objects have the greatest influence during the classification phase. Given two documents (t_i and t_j), each one having a mass ($mass(t)$) associated with a particular aspect. The TAF between them is calculated as:

$$TAF(t_i, t_j) = \frac{mass(t_i) * mass(t_j)}{d(t_i, t_j)} \tag{1}$$

where: $mass(t)$ Is a mass function related to a particular aspect.
$d(t_i, t_j)$ Is a distance metric between the documents.

As mentioned before, we are proposing KBWKNN, a variant of WKNN where the weights are determined by the TAF. We are using the term "knowledge-based" due to the fact that in this approach the information regarding the problem in hand is used in order to calculate the TAF between instances, and finally to determine the class of a given instance by considering a weighting schema.

The KBWKNN algorithm is as follows: Given a set of training instances $< t_i, f(t_i) >$, an unlabeled instance t_q, and the set of the k *nearest neighbors* to t_q (denoted as knn) in the training set, the class of t_q is determined as follows[4]:

$$f(t) \leftarrow \underset{c \in C}{\text{argmax}} \sum_{i=1}^{k} TAF(t_q, knn_i) * \partial(c, f(knn_i)) \tag{2}$$

Figure 1 shows a schematic example of the representation of an instance (symbolized as a star) to be classified by kNN in (a) and by using KBWKNN in (b). In both cases k takes as values 3 and 5. By kNN, the assigned class will be the one of the gray circles since in both cases it represents the majority class in the neighborhood. On the other side, in Fig. 1-(b) each instance has a different size that represents its relevance in terms of a given particular aspect. The higher the magnitude of a circle, the greater the mass is. Then, the unlabeled instance will be assigned according to the class of the examples having a higher weight, i.e., the one represented by darker circles.

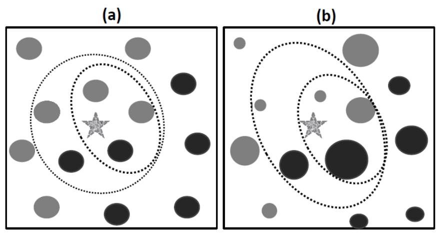

Fig. 1. Schematic representation of an unlabeled instance with its nearest neighbors.

[4] Where $\partial(c, f(knn_i))$ returns 1 if $c = f(knn_i)$ and 0 otherwise.

3 KBWKNN for Detecting Irony in Twitter

Irony detection is considered as an special case of text classification, where the aim is to identify ironic texts from non-ironic ones. As mentioned before, different approaches have been proposed to deal with such a complex task. Here, we are proposing to perform irony detection by using the approach described in Section 2. It is important to highlight that this is the first time that the Textual Attraction Force paradigm is applied for detecting irony.

Let us to introduce an example[5]:

tw *I absolutely LOVE moving house*[6].

Nearest neighbors[7]

$tw1$ *#elf #why not moving http://t.co/ZcrJaOwPqZ* **
$tw2$ *the_angry_ranga UberFacts For the masses. I love it.* **
$tw3$ *#forgive #others #because they #deserve #forgiveness #you #peace love this from @i2imovement #words #wisdom ... http://t.co/MjXdZCZDud* **
$tw4$ *love is bliss...* **
$tw5$ *Absolutely love waking up to snow* *
$tw6$ *I just love the NHS* *
$tw7$ *I'm in love with the cocoa* 🍵 **

Provided that tw is the instance to be classified. First, its k-nearest neighbors were identified (represented as $tw1 - tw7$). Supposing that a bag-of-words based representation is used, then the instances in the example are within the same neighborhood because they share two noticeable features: having at least a term in common ("love" or "moving") and they are composed by less than six tokens[8]. According to a kNN approach, the class of tw is "Nonirony", because of the majority voting between the nearest neighbors. Conversely, taking advantage of a paradigm such the TAF it is possible to capture further information that allows to provide each instance with a given relevance, therefore, it is possible to discard those instances that are only similar at a surface level. For instance, being able to capture potential clues such as the use of "LOVE" to stress a subjective opinion in an indirect way could help to identify ironic content. In this case, by exploiting KBWKNN the assigned label to tw is "Irony", since the instances expressing an opinion in an indirect way are weighted more heavily.

In order to exploit KBWKNN for irony detection, it is needed to harness domain-related knowledge that allows us to capture specific aspects of the use of irony in Twitter. Attempting to take advantage of various factors that could be useful for characterizing irony in tweets, we defined a set of six different mass functions described below.

[5] This example is part of the obtained results when the aforementioned method was applied with a size of k=7. All the tweets were extracted from dataset developed by [22]
[6] This tweet was labeled with the class "Irony"
[7] The real classes of these tweets are denoted as follows: "*" represents the class "Irony" while "**" is denoted as "Nonirony".
[8] Hashtags, mentions, emoji, and url were not considered in the bag-of-words model.

3.1 Mass Functions for Detecting Irony

Our set of mass functions aim to cover different aspects related to such interesting linguistic phenomenon. Below we introduce each of the mass functions evaluated.

– *Structural* Unlike the case of spoken communication, where enunciation, stress and tone of voice help us to communicate effectively the sense of our utterances, in written communication it is needed to make use of lexical marks to point out the intended meaning of a text. In order to calculate the *mass_Structural* of a tweet t, we considered the frequency of five aspects: Str_i: Uppercase characters; Str_{ii}: Words in uppercase; Str_{iii}: Punctuation marks[9]; Str_{iv}: Hashtag, mention, emoticons, and emoji; and, Str_v: Internet Slang[10] terms. Once we have these frequencies, we determine the value of *mass_Structural* as follows:

$$sumStr = Str_i + Str_{ii} + Str_{iii} + Str_{iv} + Str_v$$

$$mass\_Structural(t) = \begin{cases} sumStr + 1 & \text{if } sumStr > 0 \\ 1 & otherwise \end{cases} \tag{3}$$

– *Sentiment* Irony can be used to reveal an evaluative judgment. Thus, the sentiment score of a tweet may help to characterize ironic instances. We attempt to capture such value focusing especially on the positive sense of each tweet. It has been recognized the important role of positive words for masking the ironic intention [11]. To determine the *mass_Sentiment* we used the Hu&Liu lexicon (HL) [13]. It is a well-known resource developed for opinion mining that includes more than six thousand of words divided into two groups: positive and negative. Formula 4 shows how to calculate the *mass_Sentiment*.

$$mass\_Sentiment(t) = 1 + \frac{Pos - Neg}{Len} \tag{4}$$

where:	Pos	It refers to the number of positive terms in the tweet t.
	Neg	It refers to the number of negative terms in t.
	Len	It refers to the length in words of t.

– *Emotions* Affective information plays a key role for irony comprehension-communication. We defined two mass functions attempting to capture information related to emotions:
  - *Categorical Model of Emotions* Several theories propose different sets of basic or fundamental emotions. In our approach, we adopted the eight basic emotions considered in the Plutchik model [18]: *anger*, *anticipation*, *disgust*, *fear*, *joy*, *sadness*, and *trust*. In particular, we took advantage of EmoLex [16], a lexical resource containing more than fourteen thousand

[9] We consider five different punctuation marks: ".", ",", ":", "!", and "?".

[10] We used a list of terms defined in https://en.wiktionary.org/wiki/Appendix:English_internet_slang

words labeled according to the Plutchik's model of emotions. First, we compute *eCatScore* that captures how many words in the tweet t are associated to an emotion category in EmoLex, then *mass_EmotCat* is calculated as:

$$mass\_EmotCat(t) = \begin{cases} \log_{10}(eCatScore) + 1 & \text{if } eCatScore > 0 \\ 1 & Otherwise \end{cases} \tag{5}$$

- *Dimensional Model of Emotions* We employ information regarding dimensional models of emotions by taking advantage of SenticNet 3 (SN3) [5]. It contains 30,000 concepts associated with the four dimensions of the Hourglass of Emotions: Pleasantness (Pl), Attention (At), Sensitivity (Sn), and Aptitude (Ap). In [4], the authors propose a formula for calculating a polarity measure in terms of the affective dimensions in the Hourglass of Emotions. We decided to take advantage of this formula in order to calculate *mass_EmotDim*:

$$mass\_EmotDim(t) = \sum_{i=1}^{n} \frac{Pl(c_i) + |At(c_i)| - |Sn(c_i)| + Ap(c_i)}{3N} \tag{6}$$

where:	c_i	Is an input concept
	N	Is the total number of concepts of the tweet t

– *Lexical Cohesion* Comprehending irony involves getting the literal sense of the words and then understanding the figurative intention behind them [8]. Often, a way to achieve an ironic sense in an utterance is to use words that are semantically unrelated. We attempt to quantify the degree of lexical cohesion (*mass_LexicalCohesion*) in a tweet by exploiting word embeddings-based[11] similarity scores between words in a sentence. We calculated the similarity score for each pair of words in a tweet, then the maximum value is kept. The final value is determined as follows:

$$mass\_LexicalCohesion(t) = 1 + \max_{i,j \in tweet}(sim(w_i, w_j)) \tag{7}$$

– *Cognitive Aspects* Understanding irony involves different cognitive processes such as the ones devoted to text comprehension. Several factors influence such process. One of them is related to lexical sophistication. A measurable aspect related to this issue is "Word Concreteness", which can be calculated for each term in a sentence to indicate a ratio of how abstract or concrete a word is. In [21] Skalicky and Crossley, analyzing different features to identify satirical[12] and non-satirical reviews, found that the language in satirical reviews was more concrete than in non-satirical ones, i.e., words with more

[11] We used the embeddings pre-trained on the Google News corpus. https://code.google.com/archive/p/word2vec/

[12] Satire is strongly related to verbal irony, providing a detailed definition of such a concept is beyond of the scope of this work.

specific meaning were used in satirical reviews. In other words, they observed that satirical sentences have higher levels of word concreteness. Inspired by those findings, we decided to experiment with such feature for detecting irony in tweets. We exploited a lexical resource developed by Brysbaert et al. [3]. First, we measure the overall word concreteness (denoted as *tweet-Concreteness*) of a tweet as the sum of all the ratings of the words contained on it. Then, the *mass_CognitiveAspects* is calculated as follows:

$$mass\_CognitiveAspects(t) = 1 + tweetConcreteness \tag{8}$$

4 Experiments and Results

4.1 Evaluation Datasets

For evaluation purposes, we took advantage of four different corpora for irony detection covering different aspects such as collection criteria and balance degree. Below we briefly introduce each of the corpora we used:

– *TwMohammad2015*. Mohammad et al. [17] collected a set of tweets labeled with hashtags pertaining to the 2012 US presidential elections. Those data where manually annotated considering several aspects such as: sentiment, emotions, purpose and style; the latter being the one including the presence of irony. The distribution of classes in *TwMohammad2015* is 532 ironic and 1,397 non-ironic tweets.
– *TwRiloff2013*. A set of more than three thousands of tweets compose the dataset created by Riloff et al. [20]. They followed a mixed approach considering first a set of tweets tagged with the #sarcasm[13] and #sarcastic hashtags. Then, after removing the aforementioned hashtags, those tweets were manually annotated on the presence of sarcastic content. *TwRiloff2013* contains 474 ironic tweets and 1,689 nonironic ones.
– *TwReyes2013*. Reyes et al. [19] retrieved a set of tweets by using four hashtags: #irony, #education, #humor, and #politics. The first hashtag was used in order to get ironic instances relying on the idea that the author of a tweet self-annotate her ironic intention. *TwReyes2013* is composed by 40,000 tweets equally distributed in four classes[14].
– *TwVanHee2018*. In the framework of the *SemEval-2018 Task 3: Irony Detection in English tweets*[15] shared task, a dataset containing more than four thousand tweets was developed. Van Hee et al. [22] collected a set of tweets labeled with a set of hashtags: #irony, #sarcasm and #not. As a second step, the tweets were manually annotated attempting to minimize the noise. A total of 2,222 ironic and 2,396 nonironic tweets composed this dataset.

[13] In computational linguistics, irony is often considered as an umbrella term that covers also sarcasm.

[14] We performed three different binary classifications by combining each of the non-ironic classes with the ironic one. From now on, these experiments will be referred as *TwReyes2013-Edu*, *TwReyes2013-Hum*, and *TwReyes2013-Pol*.

[15] https://competitions.codalab.org/competitions/17468

4.2 Experimental Setting

A preprocessing phase was applied to the data attempting to reduce the high dimensionality of the feature space. We filtered out all the mentions, hashtags, url, emoticons, emoji, and stop words for each tweet. Additionally, all data were converted to lowercase. Finally, each instance was represented as a binary bag-of-words vector. The vocabulary of each dataset was built according to a single criterion: we kept the words with a minimum frequency. In the case of *TwMohammad2015*, *TwRiloff2013*, and *TwVanHee2018* we considered all terms that appear more than twice in the training set. While in the *TwReyes2013* the minimum frequency was fixed as twenty.

In order to calculate the mass functions defined in Section 3.1, we used the original content of each tweet, i.e. any kind of preprocessing was applied during this phase. In this case it is crucial to avoid losing important information that could be discarded. For example by converting the text to lowercase, some of the factors considered for calculating *mass_Structural* cannot be captured.

We experimented with the mass functions previously defined but adding a criterion before calculating the TAF. The idea is to compensate the imbalance degree by assigning a greater weight for neighbors belonging to the minority class. For each mass described above we applied the following criterion: if the instance belongs to the minority class, the mass is recalculated as follows[16]:

$$\textit{massFuction-Modified(t)} = e^{\textit{massFunction(t)}} \tag{9}$$

For the sake of the readability, the same acronyms defined in Section 3.1 will be used for introducing the obtained results.

In addition, we decided to combine all the masses together into a single one *mass_Combination* by means of the sum of all the masses *sumMass(t)* considering also the imbalance degree between the classes by means of the amount of instances per class (denoted as *nClass*).

$$mass\_Combination = \begin{cases} sumMass(t) * \frac{nClass_i * 100}{nClass_i + nClass_j} & \text{if } t \in Class_j \\ sumMass(t) * \frac{nClass_j * 100}{nClass_i + nClass_j} & \text{if } t \in Class_i \end{cases} \tag{10}$$

We experimented with three variations of the k-nearest neighbor classifier: kNN, DWKNN (we used the inverse of the distance for calculating the weights), and KBWNN. Concerning the size of k, we assessed the performance of our proposed approach with three different values: 3, 5, and 7. As distance function for finding the neighbors we used the Cosine Distance calculated as $D(t_i, t_j) = 1 - simCos(t_i, t_j)$.

Furthermore, for comparison purposes we also experimented with standard classifiers previously used in irony detection. The Scikit-learn implementation of Naïve Bayes (NB), Decision Tree (DT), and Support Vector Machine (SVM)[17] was used. All the experiments were carried out in a 5-fold cross-validation setting.

[16] Where *massFunction* can be any of the functions defined in Section 3.1

[17] The default configuration of parameters in the classifiers was applied.

4.3 Results

In Table 1 we present the results achieved by applying different classification approaches. Underlined values are used to point out that the achieved outcome is higher than using one of the standard classifiers. Bold values highlight the best obtained result for the "Irony" class in each dataset.

We compared the performance of our approach against to standard classifiers. There are many cases where KBWKNN outperforms at least one classifier in terms of Macro F-score. Often, the obtained results with our method overcome those of DT. Following the evaluation schema of the SemEval-2018 Task 3, we also present the results in terms of the class of interest, i.e., the ironic one. All the experiments involving each individual mass in KBWKNN outperforms the outcomes achieved by kNN and DWKNN in most of the corpora used for evaluation purposes. On the other hand, some of these results also improve the performance of the other classifiers.

Concerning to *TwMohammad2015* and *TwVanHee2018*, the best performance for the class of interest achieved by exploiting the *mass_CognitiveAspects*. In the case of *TwRiloff2013*, and *TwReyes2013*, the best performance was obtained when all the masses were combined into a single one. Overall, the best improvement in terms of the ironic class is observed on the experiments involving data where a crowd-sourcing process is part of the corpora construction. Similarly, a difference was observed in [11], where the proposed irony detection model was evaluated with different state-of-the-art corpora featured by different collection and annotation methodologies, and the performance was different depending on the methodology applied for developing the corpora. Such difference represents an interesting aspect that deserves to be further investigated. It is possible that the annotation methodology exploited for developing corpora for irony detection affects the consistency of data, especially when we compare, on the one hand, corpora developed via self-tagging, where irony-related hashtags used by ironists to express their intention to be ironic are taken as class labels, and, on the other hand, manually annotated corpora, which involve external annotators tagging the ironic intention of tweets written by others.

We are also interested in comparing the performance of the proposed approach with the state of the art. Table 2 shows such information. Previous results by exploiting an irony detection model (called "emotIDM") on the *TwMohammad2015*, *TwRiloff2013*, and *TwReyes2013* corpora are found in [11][18]. The proposed approach, KBWKNN outperforms the results of *TwMohammad2015* and *TwRiloff2013*. Conversely, in the case of the experiments related to the *TwReyes2013*, our proposed approach did not achieve higher performance than "emotIDM". Regarding the comparison with *TwVanHee2018*, we reported the three best official results achieved during the shared task.

[18] The authors reported the performance of their model in terms of F-measure considering both classes together. Attempting to compare our results, we carried out experiments by exploiting the aforementioned model but instead of considering an overall performance, we are reporting only the performance in terms of the ironic class.

Table 1. Results obtained in Macro F-score and F-score for the "Irony" class.

		TwMohammad2015		*TwRiloff2013*		*TwVanHee2018*		*TwReyes2013-Edu*		*TwReyes2013-Hum*		*TwReyes2013-Pol*	
		F-score	F_{Irony}	F-score	F_{Irony}	F-score	F_{Irony}	F-score	F_{Irony}	F-score	F_{Irony}	F-score	F_{Irony}
Naive Bayes		0.529	0.257	0.637	0.403	0.612	0.597	0.865	0.867	0.851	0.858	0.895	0.896
Decision Tree		0.522	0.279	0.577	0.33	0.582	0.565	0.83	0.824	0.821	0.818	0.884	0.882
Support Vector Machine		0.539	0.311	0.645	0.435	0.601	0.57	0.862	0.863	0.858	0.861	0.907	0.909
	k												
kNN	3	0.519	0.242	0.597	0.339	0.574	0.53	0.82	0.814	0.83	0.823	0.884	0.882
	5	0.511	0.211	0.589	0.314	0.584	0.546	0.825	0.82	0.832	0.825	0.884	0.882
	7	0.505	0.18	0.594	0.319	0.594	0.557	0.83	0.825	0.834	0.828	0.885	0.883
DWKNN	3	0.522	0.247	0.596	0.338	0.578	0.534	0.827	0.822	0.839	0.832	0.887	0.886
	5	0.511	0.212	0.585	0.309	0.588	0.55	0.833	0.828	0.839	0.833	0.887	0.886
	7	0.505	0.181	0.588	0.311	0.596	0.558	0.835	0.831	0.839	0.833	0.884	0.884
KBWKNN	3	0.49	0.387	0.576	0.415	0.534	0.641	0.781	0.793	0.798	0.789	0.859	0.87
	5	0.432	0.415	0.534	0.425	0.478	0.654	0.733	0.759	0.758	0.746	0.826	0.847
mass_CognitiveAspects	7	0.373	**0.419**	0.479	0.415	0.441	**0.655**	0.69	0.73	0.732	0.718	0.8	0.83
KBWKNN	3	0.497	0.388	0.58	0.412	0.543	0.641	0.8	0.807	0.815	0.805	0.871	0.879
	5	0.532	0.348	0.617	0.428	0.568	0.638	0.816	0.821	0.819	0.812	0.88	0.886
mass_Sentiment	7	0.527	0.384	0.595	0.42	0.535	0.652	0.801	0.809	0.808	0.798	0.87	0.878
KBWKNN	3	0.497	0.374	0.59	0.419	0.54	0.622	0.794	0.802	0.807	0.798	0.865	0.874
	5	0.455	0.404	0.577	0.436	0.502	0.645	0.772	0.788	0.785	0.776	0.845	0.86
mass_Structural	7	0.421	0.406	0.55	0.436	0.473	0.652	0.745	0.77	0.765	0.756	0.828	0.848
KBWKNN	3	0.492	0.385	0.581	0.415	0.543	0.64	0.802	0.809	0.816	0.806	0.871	0.88
	5	0.526	0.349	0.618	0.432	0.567	0.638	0.814	0.82	0.818	0.81	0.879	0.885
mass_LexicalCohesion	7	0.517	0.39	0.595	0.427	0.353	0.652	0.8	0.808	0.807	0.798	0.87	0.879
KBWKNN	3	0.505	0.377	0.59	0.422	0.555	0.641	0.803	0.811	0.819	0.811	0.875	0.882
	5	0.518	0.374	0.612	0.432	0.558	0.637	0.81	0.817	0.817	0.809	0.877	0.884
mass_EmotCat	7	0.519	0.372	0.599	0.426	0.547	0.654	0.803	0.813	0.812	0.804	0.872	0.879
KBWKNN	3	0.496	0.387	0.583	0.418	0.541	0.641	0.8	0.808	0.815	0.805	0.871	0.879
	5	0.532	0.348	0.618	0.428	0.567	0.637	0.816	0.821	0.82	0.812	0.88	0.886
mass_EmotDim	7	0.525	0.389	0.603	0.43	0.538	0.653	0.8	0.809	0.808	0.798	0.87	0.878
KBWKNN	3	0.513	0.38	0.586	0.416	0.574	0.533	0.827	0.823	0.839	0.833	0.885	0.884
	5	0.522	0.371	0.597	**0.439**	0.582	0.545	0.835	0.832	0.839	0.834	0.888	0.888
mass_Combination	7	0.517	0.372	0.583	0.428	0.586	0.55	0.836	**0.835**	0.842	**0.837**	0.887	**0.888**

Deep learning-based approaches were exploited by the three best ranked systems[19]. As it can be noticed, our three best results are higher than the one obtained by the 3^{rd} best ranked system. It serves to validate that even if our method is simple, it is able to obtain competitive results against more sophisticated techniques.

Table 2. Comparison of our results with the state of the art.

	emotIDM	Our approach		
Dataset	SVM	Mass1	Mass2	Mass3
TwMohammad2015	0.011	0.419	0.415	0.406
TwRiloff2013	0.134	0.439	0.436	0.432
TwReyes2013-Edu	0.892	0.835	0.832	0.823
TwReyes2013-Hum	0.89	0.837	0.834	0.833
TwReyes2013-Pol	0.888	0.888	0.886	0.885

TwVanHee2018			Our approach	
Ranking Position	F_{Irony}		Mass	F_{Irony}
1st	0.705		Mass1	0.655
2nd	0.671		Mass2	0.654
3rd	0.650		Mass3	0.653

5 Conclusions

In this paper, we introduce a variation of WKNN by exploiting knowledge-based information together with a novel paradigm, called Textual Attraction Force. The proposed approach was evaluated in an special case of text classification: irony detection. This is the first time that such a complex task is addressed by exploiting this kind of approach. We have performed several experiments over a set of state-of-the-art corpora. Across most of the experiments carried out, it can be concluded that using knowledge-based information for calculating the TAF between two instances (and then using this value as the weight of the instances in KBWKNN), despite being a simple model exploiting a traditional representation (bag-of-words) together with domain-dependent features not only improves the classification performance of well-known machine learning algorithms for irony detection, but also validates the usefulness of using novel paradigms (more intuitive and easier to interpret) to find similar documents in text classification related tasks. As future work, it could be interesting to further explore different ways to calculate the mass functions as well as comparing our results against deep learning techniques.

Acknowledgments

This research was funded by CONACYT project FC 2016-2410. The work of P. Rosso has been funded by the SomEMBED TIN2015-71147-C2-1-P MINECO research project. The work of V. Patti was partially funded by Progetto di Ateneo/CSP 2016 (IhatePrejudice, S1618_L2_BOSC_01).

[19] For further details on the shared task see [22].

References

1. Barbieri, F., Basile, V., Croce, D., Nissim, M., Novielli, N., Patti, V.: Overview of the Evalita 2016 SENTIment POLarity Classification Task. In: Proc. of Third Italian Conf. on Computational Linguistics. vol. 1749. CEUR-WS.org (2016)
2. Basile, V., Bolioli, A., Nissim, M., Patti, V., Rosso, P.: Overview of the Evalita 2014 SENTIment POLarity classification task. In: Proc. of the First Italian Conf. on Computational Linguistics. pp. 50–57 (2014)
3. Brysbaert, M., Warriner, A.B., Kuperman, V.: Concreteness Ratings for 40 Thousand Generally Known English Word Lemmas. Behav Res Met **46**(3) (2014)
4. Cambria, E., Hussain, A.: Sentic Computing: A Common-Sense-Based Framework for Concept-level Sentiment Analysis, vol. 1. Springer (2015)
5. Cambria, E., Olsher, D., Rajagopal, D.: SenticNet 3: A Common and Common-Sense knowledge base for cognition-driven sentiment analysis. In: Proceedings of AAAI Conference on Artificial Intelligence. pp. 1515–1521 (2014)
6. Dudani, S.A.: The Distance-Weighted k-Nearest-Neighbor Rule. IEEE Transactions on Systems, Man, and Cybernetics **SMC-6**(4), 325–327 (April 1976)
7. Ghosh, A., Li, G., Veale, T., Rosso, P., Shutova, E., Barnden, J., Reyes, A.: Semeval-2015 Task 11: Sentiment Analysis of Figurative Language in Twitter. In: Proc. of the 9th Int. Workshop on Semantic Evaluation. pp. 470–478 (2015)
8. Giora, R., Fein, O.: Irony: Context and Salience. Metaphor and Symbol **14**(4), 241–257 (1999)
9. Gou, J., Du, L., Zhang, Y., Xiong, T.: A New Distance-weighted k-Nearest Neighbor Classifier. Journal of Inform. and Comp. Science **9**(6), 1429–1436 (2012)
10. Grice, H.P.: Logic and Conversation. In: Cole, P., Morgan, J.L. (eds.) Syntax and Semantics: Vol. 3: Speech Acts, pp. 41–58. Academic Press, San Diego, CA (1975)
11. Hernández Farías, D.I., Patti, V., Rosso, P.: Irony Detection in Twitter: The Role of Affective Content. ACM Trans. Internet Technol. **16**(3), 19:1–19:24 (2016)
12. Hernández Farías, D.I., Rosso, P.: Irony, Sarcasm, and Sentiment Analysis. Chapter 7. In: Pozzi, F.A., Fersini, E., Messina, E., Liu, B. (eds.) Sentiment Analysis in Social Networks, pp. 113–127. Morgan Kaufmann (2016)
13. Hu, M., Liu, B.: Mining and Summarizing Customer Reviews. In: Proc. of the 10th SIGKDD Int. Conf. on Knowledge Discovery and Data Mining. pp. 168–177 (2004)
14. Joshi, A., Bhattacharyya, P., Carman, M.J.: Automatic Sarcasm Detection: A Survey. ACM Comput. Surv. **50**(5), 73:1–73:22 (Sep 2017)
15. Mitchell, T.M.: Machine Learning and Data Min. Com. ACM **42**(11), 30–36 (1999)
16. Mohammad, S.M., Turney, P.D.: Crowdsourcing a Word–Emotion Association Lexicon. Computational Intelligence **29**(3), 436–465 (2013)
17. Mohammad, S.M., Zhu, X., Kiritchenko, S., Martin, J.: Sentiment, Emotion, Purpose, and Style in Electoral Tweets. Inf. Process. Manag. **51**(4), 480 – 499 (2015)
18. Plutchik, R.: The Nature of Emotions. American Scientist **89**(4), 344–350 (2001)
19. Reyes, A., Rosso, P., Veale, T.: A Multidimensional Approach for Detecting Irony in Twitter. Language Resources and Evaluation **47**(1), 239–268 (2013)
20. Riloff, E., Qadir, A., Surve, P., Silva, L.D., Gilbert, N., Huang, R.: Sarcasm as Contrast between a Positive Sentiment and Negative Situation. In: Proc. of the Conf. on Empirical Methods in Natural Lang. Processing. pp. 704–714. ACL (2013)
21. Skalicky, S., Crossley, S.: A Statistical Analysis of Satirical Amazon.com Product Reviews. Eur. J. Humour Res. **2**, 66–85 (2015)
22. Van Hee, C., Lefever, E., Hoste, V.: SemEval-2018 Task 3: Irony Detection in English Tweets. In: Proc. of the 12th Int. Workshop on Semantic Evaluation. SemEval-2018, ACL (June 2018)