Journal of Big Data

RESEARCH

Open Access

Emotion analysis of Arabic tweets using deep learning approach

Massa Baali and Nada Ghneim*

*Correspondence:
nada.ghneim@gmail.com
Department of Artificial Intelligence, Arab International University, Damascus, Syria

Abstract

Nowadays, sharing moments on social networks have become something widespread. Sharing ideas, thoughts, and good memories to express our emotions through text without using a lot of words. Twitter, for instance, is a rich source of data that is a target for organizations for which they can use to analyze people's opinions, sentiments and emotions. Emotion analysis normally gives a more profound overview of the feelings of an author. In Arabic Social Media analysis, nearly all projects have focused on analyzing the expressions as positive, negative or neutral. In this paper we intend to categorize the expressions on the basis of emotions, namely happiness, anger, fear, and sadness. Different approaches have been carried out in the area of automatic textual emotion recognition in the case of other languages, but only a limited number were based on deep learning. Thus, we present our approach used to classify emotions in Arabic tweets. Our model implements a deep Convolutional Neural Networks (CNN) trained on top of trained word vectors specifically on our dataset for sentence classification tasks. We compared the results of this approach with three other machine learning algorithms which are SVM, NB and MLP. The architecture of our deep learning approach is an end-to-end network with word, sentence, and document vectorization steps. The deep learning proposed approach was evaluated on the Arabic tweets dataset provided by SemiEval for the EI-oc task, and the results-compared to the traditional machine learning approaches-were excellent.

Keywords: Deep learning, Big Data—emotion recognition of Arabic texts, CNN sentence classification, Data mining, SVM, NB, MLP

Introduction

In our daily life, every one of us face different situations and the outcome of it is developing a feeling about it. Emotion is a strong feeling about human's situation or relation with others [1]. It has a big role in customer decision in many domains including e-commerce, restaurants, movies, interests, and satisfaction with a service or a product. Moreover, it affects our health! Lately, Facebook added some reactions including angry, happiness, love, and surprise to allow users to express their emotions toward a comment, picture or an event.

In research, emotional analysis is regarded as a sort of higher, evolved form of sentiment analysis. Sentiment analysis aims to classify texts (posts, sentences, or documents) into negative, positive or neutral. Emotional analysis, on the other hand, is a more elaborated, deeper analysis of users' emotions that tries to inspect the psychology of different

Table 1 Corresponding emotion reflection in tweets

Emotion	Tweets
😠	مبحبش التهديد حتى لو ف مصلحتى Mbhb$ Althdyd httY lw f mSlhty I hate threatening even when it's my benefit
😁	قد ايش حماسي دا الفيلم Qd Ay$ hmAsy dA Alfilm How much enthusiasm in this movie
😒	لا تدققيين والله ضغطي مرتفع وزاقة معي WzAqp mEy lA tdqqyyn wAllh DgTy mrtfE I am fed up don't probe I swear my pressure is high
😖	ضعيفه انا قدام تعب جدتي ودموعها والله ضعيفه وجدا DEyfh AnA qdAm tEb jdty w dmwEhA wAllh DEyfh w jdF I am so weak in front of my grandma's fatigue and her tears I swear I am so weak

user behaviors revealing deeper human emotional connotations such as, anger, disgust, trust, sadness, joy, surprise, etc.

The English language has been highly recognized in emotion detection domain including datasets and dictionaries availability [2–4], in contrast to the Arabic language, which has a very limited resources.

In this research, we explore the automatic emotion recognition for Arabic language with minimal input (sentence) using Convolutional Neural Networks (CNN) using four steps: word, sentence, document vectorization then classification. Moreover, we compared this approach with other machine learning algorithms to show the performance and accuracy the deep learning has reached so far. We applied our approaches to analyze emotions of user's tweets in SemiEval dataset. Tweets examples with emotions are shown in Table 1.

In the rest of this paper, a brief of related works are summarized in "Related works" section. In "Methods/experimental" section, we represent our methodologies that we have implemented to recognize the emotion of the Arabic tweets, with the dataset used. In "Results and discussion" section, a brief discussion of the results is addressed. At the end, insights for the future and a short summary are presented.

Related works

A lot of researches has been done lately in the domain of emotion recognition. In [5], Kim has trained a simple CNN with single layer of convolution built on top of word vectors obtained from an unsupervised neural language model. The word vectors were trained on the 100 billion words of Google News. Word vectors were kept static and only the other parameters of the model were learnt.

In [6], Nguyen et al. used deep Convolutional Neural Network and considered the information of words' characters to support word-level embedding. A Bi-LSTM produces a sentence-wide feature representation based on these embedding. The approach was applied on tweets sentiment analysis.

In [7], Majumder et al. exhibited a strategy to extract personality characteristics from different papers utilizing a convolutional neural system. They prepared five unique systems, every one of them with a similar architecture, on every one of the five personality traits. Each network was a binary classifier that anticipated if the attribute is positive or negative. The framework consolidated two distinct methodologies: The first is a deep learning approach called N-Stream ConvNets, and the second is XGboost regressor dependent on set of embedding and lexicons based features. The structure beats each and every unique approach for the Arabic interpretation of valence power backslide and valence ordinal request SemEval 2018 errands.

Authors proposed a graph-based system to separate rich-feeling bearing examples, which encourages a deeper examination of online emotional articulations, from a corpus. The examples were then improved with word embeddings and assessed through a few feeling acknowledgment assignments. Besides, they directed examination on the feeling focused examples to exhibit its pertinence and to investigate its properties. Their test results exhibit that the proposed methods outperform most cutting edge emotion recognition strategies.

In [8], Shaheen et al. proposed a system for emotion recognition classification in English sentences where feelings are treated as summed up ideas extricated from the sentences. Right off the bat, a moderate enthusiastic information portrayal of a given info sentence was created dependent on its syntactic and semantic structure, at that point it was summed up utilizing different ontologies so as to extricate a feeling seed [called an emotion recognition rule (ERR)]. At the point when connected on various datasets, the proposed methodology fundamentally beat the current state-of-the-art in machine learning and rule-based classifiers.

In [9], Tilakraj proposed emotion detector system that takes a text document or audio and the emotion word ontology as inputs and produces the scores of six emotion classes (i.e. happy, sad, fear, surprise, anger and disgust) as the output. Their model performs semantic analysis from semantic information, exclamatory keywords, and direct emotional keywords.

In [10] George et al. experimented the intensity prediction as a text classification problem that evaluates the distributed representation text using aggregated sum and dimensionality reduction of the glove vectors of the words present in the respective texts (English and Arabic language).

Du described in [11] the system MuTuX, which aims at exploring the potential of context information of terms for English emotion analysis. A recurrent neural network is adopted to capture the context information of terms in tweets. Only term features and the sequential relations are used in the system.

Abdullah described in [12] the approach used in the UNCC system to detect emotions in English and Arabic tweets. They present the same architecture for all the five subtasks in both English and Arabic. The main input to the system is a combination of word2vec and doc2vec embeddings and a set of psycholinguistic features (e.g. AffectiveTweets from Weka-package). They apply a fully connected neural network architecture and obtain performance results that show substantial improvements in Spearman correlation scores. They are detecting the intensity of an emotion which has a different output such as a real-valued score between 0 (least emotion intensity) and 1 (most emotion intensity). Also, they are using LSTM network.

In [13], Daood et al. focused on emotion detection of Arabic language. They constructed a corpus of Arabic Levantine tweets, and annotated it with their target emotions. They implemented different methods to classify text messages of users to discover their emotional states. They compared the results of different machine learning algorithms, including different features, to reach the best emotion recognition result.

Mohammad summarized in [14] the results of the SemEval-2018 Task 1: Affect in Tweets, which incorporates a variety of subtasks on inducing the affectual condition of an individual from their tweet. For each task, they made labeled information from English, Arabic, and Spanish tweets. The individual tasks were: Emotion Intensity Regression, Emotion Intensity Ordinal Classification, Valence (Sentiment) Regression, Valence Ordinal Classification, and Emotion Classification. They summarized the strategies, assets, and tools utilized by the taking part groups, with a focus on the methods and assets that are especially helpful. They additionally dissected frameworks for reliable predisposition towards a specific race or gender.

One of this work contributions is the use of an end-to-end network that comprises four main steps: *word vectorization*, *sentence vectorization*, *document vectorization*, then *classification*. We compare the results with other three machine learning approaches.

Methods/experimental

In this section, we will present the dataset used, and our methodologies to recognize emotions of Arabic tweets using deep learning approach in addition to other three machine learning algorithms, which are Naïve Bayes, Support Vector Machine and multilayer perceptron.

Used dataset

We used the Arabic tweets dataset provided by SemiEval for the EI-oc task [14], which is an emotion intensity ordinal classification task: Given a tweet and an emotion E, classify the tweet into one of four ordinal classes of intensity of E that best represents the mental state of the tweeter.

The dataset consists of 5600 tweets, 1400 for each of the provided emotions: anger, fear, joy, and sadness.

We split our dataset into 5064 tweets in the training set and 561 tweets in the testing set. We used 90% of the dataset for training (1266 tweets for each emotion) and 10% for testing (561 tweets for each emotion). The training datasets were used to train the classifier and to optimize the parameters, while the test dataset (unseen to the model) was reserved to test the built model, to provide an indication of how good the trained model is. Also, we tried to split the data 70% for training and 30% for testing it gave us the same results which is 99.82%.

Data preprocessing

Since our dataset was in Arabic language, we did a specific preprocessing to find out the best pattern. The steps we followed are:

Table 2 Normalizing characters

Character	Normalized
إأآا	ا
ى	ي
ؤ	ء
ئ	ء
ة	ه
گ	ك

Table 3 Diacritics

Character	Diacritics
◌ّ	Tashdid
◌َ	Fatha
◌ً	Tanwin Fath
◌ُ	Damma
◌ٌ	Tanwin Damm
◌ِ	Kasra
◌ٍ	Tanwin Kasr
◌ْ	Sukun

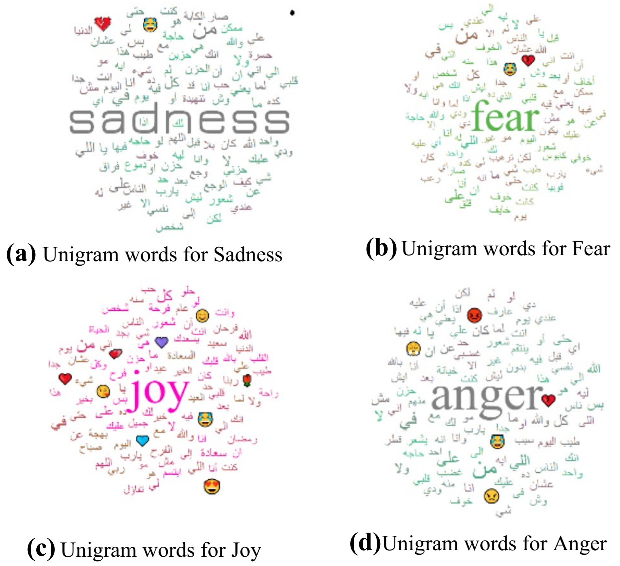
a. Normalization of some characters that can be written in different ways, such as writing “أ، إ، آ، ا” in the normal form "أ", as shown in Table 2.
b. Remove all the diacritics as shown in Table 3.
c. Remove punctuation marks.
d. Remove repeated characters: twitter users usually intentionally repeat a character in a word to emphasize and to exaggerate in describing something like laughing“ههههههههه” Hahahaha, magnification“واااااااااااااااو” Wooooow, indignation“لاااااااااا” Noooooo, etc. We considered that a word cannot have more than two repeated characters, hence, every other repeated character was eliminated.

In addition, we can add a step that removes the stop words from the input text, where stop words include prepositions, conjunctions, etc.

The correlation between tweets’ words and each of the studied emotions is shown in Fig. 1, based on the frequency of the word n-grams for each emotion.

Deep learning approach

In this section, we will show our deep learning approach and we will go through each step we followed.



(a) Unigram words for Sadness **(b)** Unigram words for Fear

(c) Unigram words for Joy **(d)**Unigram words for Anger

Fig. 1 Correlations between Arabic words and emotions: based on the frequency of the word n-grams for each emotion

Training word vectors

Initializing word vectors obtained from an unsupervised neural language model is a popular method to improve performance in the absence of a large supervised training set [15–17].

In our research, we created a word2vec model using Genism implementation trained on our dataset in order to make it more specific to our case (Arabic tweets written in Dialects) so we won't suffer in finding many unknown words. If a word was not found in the list, we assigned it with a uniform distribution in [− 0.25, 0.25] as suggested by [7] since uniform distribution's outcomes are equally likely; each variable has the same probability that it will be the outcome. With this type of distribution, every variable has an equal opportunity of appearing, yet there are a continuous number of points that can exist.

Network architecture

We trained our CNN classifier, with unique architecture, for the emotion detection of Arabic tweets. The processing flow in our end-to-end network comprises four main steps derived from the architecture used by Majumder et al. in [7]:

- Word vectorization, in which we use our trained word2vec word embedding as input data;
- Sentence vectorization, from sequences of words in each sentence to fixed-length sentence vectors;
- Document vectorization, from the sequence of sentence vectors to the document vector; and

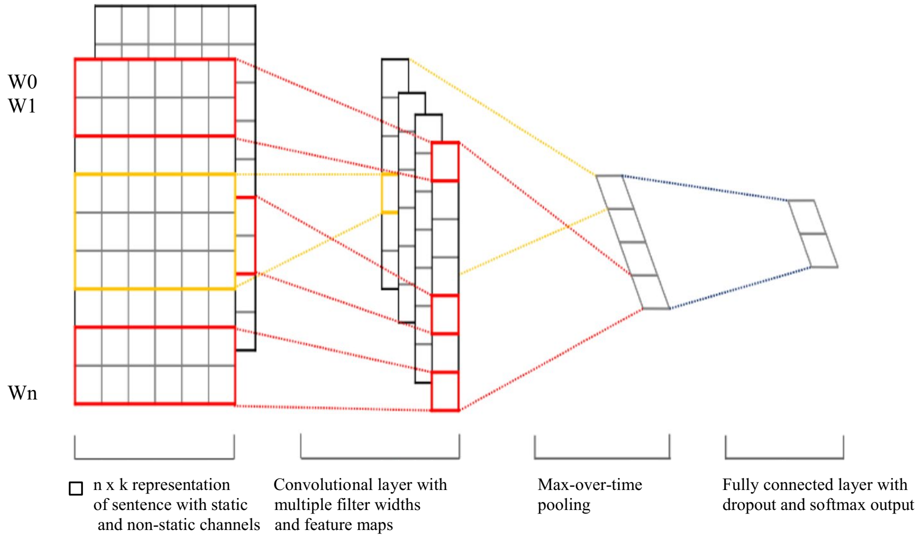

W0
W1
Wn
□ n x k representation of sentence with static and non-static channels
Convolutional layer with multiple filter widths and feature maps
Max-over-time pooling
Fully connected layer with dropout and softmax output

Fig. 2 Model architecture with two channels for an example sentence

- Classification, from the document vector to the classification result (anger, joy, sad, fear).

Our network comprises seven layers: input (word vectorization), convolution (sentence vectorization), max pooling, Flatten, concatenation, Dense with ReLu activation (classification), and 4-neuron softmax output (classification). Figure 2 presents the model architecture with two channels for an example sentence [5].

1. Input
   Our input layer is a four-dimensional real-valued array from $\mathrm{R}^{\mathrm{D} \times \mathrm{S} \times \mathrm{W} \times \mathrm{E}}$, in which D is the number of documents in the dataset, S is the maximum number of sentences in a document across all documents, W is the maximum number of words in a sentence across all documents, and E is the length of word.
   In our implementation, in order to force all documents to have the same number of sentences, shorter documents were padded with dummy sentences. Likewise, shorter sentences were padded with dummy words [7].
2. Convolution layer
   To extract the n-gram features, we apply a convolutional filter of size $\mathrm{n} \times \mathrm{E}$ on each sentence $\mathrm{s} \in \mathrm{R}^{\mathrm{W} \times \mathrm{E}}$. We use 300 n-gram feature maps for each $\mathrm{n} = 1, 2, 3$. So, for each n, our convolutional filter applied on the matrix s is $\mathrm{F}_{\mathrm{n}}^{\mathrm{conv}} \in \mathrm{R}^{300 \times \mathrm{n} \times \mathrm{E}}$.
   We apply the Rectified Linear Unit (ReLU) function to the feature maps FM_{n}, where ReLu function is stated as [18] (Fig. 3):

   $$A(x) = max(0, x)$$

   It gives x as an output if x is positive and 0 otherwise.
3. Max pooling

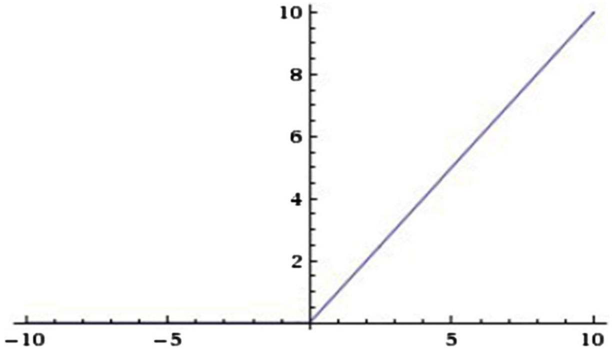


Fig. 3 ReLu function

Next, we apply max pooling to each feature map $\mathrm{FM_n}$ to further down-sample it to a feature map $\mathrm{DFM_n} \in \mathrm{R}^{300\times1\times1}$, which we flatten to obtain a feature-vector of size 300. In max pooling, the pooling operator maps a subregion to its maximum value [19]

$$\max : y_i^{l+1},_j^{l+1},_d = \max_{0\le i<H, 0\le W} x_i^{l\,l+1} \times H + i, j^{l+1} \times W + j, d$$

4. Classification
   For final classification, we use a four-layer perceptron consisting of a fully connected layer of size 300 and a final softmax layer of size four, representing the anger, sadness, joy and fear classes.
   The softmax activation function is:
   $$Y = softmax(X.W + b)$$
5. Training the system using CNN
   We experiment our system with several variants of the CNN model that have been introduced by Kim in [5].

- CNN-rand: randomly initialized word vectors.
- CNN-static: our word2vec model and all unknown words are kept static, and they are not updated during the CNN training.
- CNN-non-static: The input word vectors are pre-trained by Word2Vec first, and they are fine-tuned during training CNN model.

Other machine learning approach

In order to classify the tweet into each class (happiness, anger, fear, and sadness) we applied three different machine-learning approaches: Support Vector Machines (SVM), Naïve Bayes (NB), and multi-layer perceptron (MLP).

In this implementation, we used the stem of the words as the main feature, and we compared the results of three Arabic stemmers: Light stemmer, ISRI stemmer and snowball stemmer where we removed the stop words and the punctuations.

We also implemented the Count Vectorizer and the TF-IDF to calculate the values of each feature.

Table 4 Training parameters

Exper. ID	Embedding	Filter size	Filters nb.	Dropout	Batch size	Hidden dims	Remove stop words	Pre-processing	Acc. (%)
Ex1	300	3, 8	30	0.5, 0.8	64	50	No	No	47.59
Ex2	300	3, 8	30	0.5, 0.8	128	50	No	Yes	54.55
Ex3	300	3, 8	30	0.5, 0.8	128	100	No	Yes	52.05
Ex4	300	3, 8	40	0.5, 0.8	128	100	No	Yes	53.48
Ex5	300	3, 8	50	0.5, 0.8	128	100	No	Yes	51.69
Ex6	300	3, 4, 5	40	0.5, 0.8	128	100	No	Yes	55.26
Ex7	512	3, 4, 5	40	0.5, 0.8	128	100	No	Yes	56.51
Ex8	300	3, 4, 5	40	0.5, 0.8	128	100	Yes	Yes	99.82

where, *Embedding* is the first layer in a model. It requires that the input data be integer encoded, so that each word is represented by a unique integer, *Filter size* is the size of the filter used in the experiment, *Filter nb* is an integer which represents the dimensionality of the output space, *Dropout* represents applying a technique where randomly selected neurons are ignored during training, *Batch size* is the number of training examples in one forward/backward pass, *Hidden Dims* is the number of neurons in this hidden layer, *Remove stop words* is the elimination of the stop words from the text in this experiment, and *Pre-processing* is the implementation of the preprocessing steps mentioned in "Data preprocessing" section excluding the step of the stop words elimination

Results and discussion

Deep learning approach

During our first experiments with the different, models (CNN-rand, CNN-static, CNN-non-static), CNN-non-static model showed the best results. We can see that the CNN-non-static model uses pre-trained word vectors compared to CNN-rand mode, so it has more prior knowledge. On the other hand, when compared with CNN-static model, CNN-non-static model fine-tunes the word vector during training, so that the word vector can learn more meaningful task-related vector expression.

Therefore, in the following experiments, the CNN-non-static mode will be used for experiments. We worked more on the model and implemented multiple experiments to reach the best parameters, as shown in Table 4. The last experiment, written in bold, has the highest accuracy.

As we can see in experiment number 4 and experiment number 5 we increased the number of filters but the accuracy decreased. In experiment 4 and experiment 6 changing the size of the filter increased the accuracy. In experiment 6 and experiment 7 increasing the size of the embedding layer led to increasing the accuracy. In experiment 6 and experiment 8 we can notice that eliminating the stop words in the last experiment has increased the accuracy, as each sentence contains a lot of stop words that makes the categories of the emotions too similar and hard to classify, that is because the stop words would be counted too in the feature. And that it why by removing the stop words, the accuracy escalated.

We validated our system on 561 samples, where our dataset was 90% training and 10% testing. Table 5 presents a summary of the results of the last epoch (20th). The table is divided into both training phase results and validation results (Accuracy, Loss, Precision, Recall, F1).

Table 5 Summary of the results (last epoch)

Epoch	**20/20**
Training phase	
Accuracy	0.9990
Loss	0.0023
Precision	0.9994
Recall	0.9990
F1	0.9992
Validation phase	
Accuracy	0.9982
Loss	0.0031
Precision	1.0000
Recall	0.9982
F1	0.9991

Table 6 Results using Count Vectorizer

Count Vectorizer	**Light Stemming**	**ISRI Stemming**	**Snowball Stemming**
NB	0.43	0.44	*0.45*
SVM	*0.24*	0.23	0.23
MLP	*0.24*	0.23	0.23

The italic values indicate the highest accuracy (best one)

Table 7 Results using TF-IDF

TF-IDF	**Light Stemming**	**ISRI Stemming**	**Snowball Stemming**
NB	0.45	0.45	*0.46*
SVM	*0.24*	0.23	0.23
MLP	0.31	0.35	*0.36*

The italic values indicate the highest accuracy (best one)

Classical machine learning approaches

We compared the results of using three different machine-learning approaches: SVM, NB, and MLP, with three different Arabic stemmers (Light stemmer, ISRI, and Snowball) [20–22]. Table 6 presents the results using *Count* as feature values, whereas Table 7 presents the results using *TF-IDF* as feature values.

Using Count Vectorizer, we found that Naïve Bayes with Snowball Stemming has achieved the highest accuracy which is 45%. Using TF-IDF, we found also that Naïve Bayes with Snowball Stemming has achieved the highest accuracy which is 46%.

We notice the big difference between the two approaches. In the classical ML methods, we need to work more on the feature engineering step to find the best features (such as the number of exclamation marks, question marks, emoticons, number of joy/anger/sadness/fear words in the tweet), and not just taking the stem of the words, whereas in the deep learning approach the algorithm chooses the best features and does the feature extraction on its own which makes it more efficient than classical machine learning algorithms.

Conclusions

In this work, we introduced our approach for classifying the emotions of Arabic tweets. We implemented deep learning approach using Convolutional Neural Network. The architecture is an end-to-end network with word, sentence, and document vectorization steps. We used the Arabic tweets dataset provided by SemiEval for the EI-oc task, with four emotion categories: joy, anger, sadness, and fear. The approach achieved remarkable results with 99.90% as training accuracy, and 99.82% as validation accuracy.

We compared this result with three other machine-learning approaches: SVM, NB, and MLP, implemented using three different Arabic stemmers (Light stemmer, ISRI, and Snowball), and two basic feature values (Count and TF-IDF).

As a future work, we aim to run this model on a much bigger dataset and evaluate the results, and also to work on Long Short-Term Memory (LSTM) approach to detect emotions in order to compare between multiple methodologies. Moreover, we aim to work on sarcasm detection problem and evaluate its impact on emotion detection results.

Abbreviations

CNN: Convolutional Neural Network; NB: Naïve Bayes; SVM: Support Vector Machine; MLP: multi-layer perceptron; LSTM: Long Short-Term Memory; Bi-LSTM: Bidirectional Long Short-Term Memory; ERR: emotion recognition rule; RELU: Rectified Linear Unit.

Acknowledgements

There are no acknowledgements.

Authors' contributions

MB designed and developed the system, interpreted the results and wrote the manuscript under the supervision of NG as an academic supervisor. NG also made contribution to the conception and analysis of the work. Both authors read and approved the final manuscript.

Funding

There is no funding.

Availability of data and materials

Dataset from SemiEval.

Competing interests

The authors declare that they have no competing interests.

Received: 14 March 2019 Accepted: 21 September 2019
Published online: 16 October 2019

References

1. Sylvester K. Emotion detection and analysis using machine learning and deep learning. 2018. https://sylvesterkaczmarek.com/blog/emotion-detection-analysis-using-machine-learning-deep-learning/. Accessed 11 May 2019.
2. Agarwal A, Brijraj S, Jatin B, Durga T. A Datamining approach for emotions extraction and discovering Cricketers performance from Stadium to Sensex. arXiv preprint arXiv:1809.00310. 2018.
3. Agarwal A, Singh R, Toshniwal D. Geospatial sentiment analysis using twitter data for UK-EU referendum. J Inf Optim Sci. 2018;39(1):303–17.
4. Agarwal A, Durga T. Application of lexicon based approach in sentiment analysis for short Tweets. In: 2018 international conference on advances in computing and communication engineering (ICACCE). IEEE, 2018. p. 189–93.
5. Kim Y. Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882. 2014.
6. Nguyen H, Nguyen ML. A Deep Neural Architecture for sentence-level sentiment classification in Twitter Social Networking. In: International conference of the pacific association for computational linguistics. Springer, Singapore. 2017. p. 15–27.
7. Majumder N, Poria S, Gelbukh A, Cambria E. Deep learning-based document modeling for personality detection from text. IEEE Intell Syst. 2017;32(2):74–9.

8. Shaheen S, El-Hajj W, Hajj H, Elbassuoni S. Emotion recognition from text based on automatically generated rules. In: 2014 IEEE international conference on data mining, workshop. IEEE. 2014. p. 383–92.
9. Tilakraj MM, Shetty DD, Nagarathna M, Shruthi K, Narayan S. Emotion finder: detecting emotions from text, tweets and audio. Int J Sci Eng Appl Sci. 2016;2(5):71–4.
10. George A, HB, BG, Soman KP. TeamCEN at SemEval-2018 Task 1: global vectors representation in emotion detection. In: Proceedings of the 12th international workshop on semantic evaluation. 2018. p. 334–8.
11. Du P, Nie JY. Mutux at SemEval-2018 Task 1: exploring impacts of context information on emotion detection. In: Proceedings of the 12th international workshop on semantic evaluation. 2018. p. 345–9.
12. Abdullah M, Shaikh S. TeamUNCC at SemEval-2018 Task 1: emotion detection in English and Arabic Tweets using deep learning. In: Proceedings of the 12th international workshop on semantic evaluation. 2018. p. 350–7.
13. Daood A, Salman I, Ghneim N. Comparison study of automatic classifiers performance in emotion recognition of Arabic social media users. J Theor Appl Inf Technol. 2017;95(19):5172–83.
14. Mohammad S, Bravo-Marquez F, Salameh M, Kiritchenko S. Semeval-2018 task 1: affect in tweets. In: Proceedings of the 12th international workshop on semantic evaluation. 2018. p. 1–17.
15. Collobert R, Weston J, Bottou L, Karlen M, Kavukcuoglu K, Kuksa P. Natural language processing (almost) from scratch. J Mach Learn Res. 2011;12:2493–537.
16. Pennington RSJ, Ng EHHAY, Manning CD. Semi-supervised recursive autoencoders for predicting sentiment distributions. In: Proceedings of the conference on empirical methods in natural language processing (EMNLP), Edinburgh, Scotland, UK. 2011.
17. Iyyer M, Enns P, Boyd-Graber J, Resnik P. Political ideology detection using recursive neural networks. In: Proceedings of the 52nd annual meeting of the association for computational linguistics (volume 1: long papers), p. 1113–22. 2014.
18. Avinash SV. 2017. Understanding activation functions in neural network. https://medium.com/the-theory-of-everything/understanding-activation-functions-in-neural-networks-9491262884e0. Accessed 11 May 2019.
19. Wu J. Introduction to convolutional neural networks. Nanjing: National Key Lab for Novel Software Technology. Nanjing University; 2017.
20. NLTK. https://www.nltk.org/_modules/nltk/stem/isri.html. Accessed 11 May 2019.
21. NLTK. https://www.nltk.org/_modules/nltk/stem/snowball.html. Accessed 11 May 2019.
22. Python Package Index. https://pypi.org/project/Tashaphyne/. Accessed 11 May 2019.

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.