Natural Language Engineering (2021), **27**, pp. 315–338
doi:10.1017/S135132492000008X

ARTICLE

Syntax-ignorant N-gram embeddings for dialectal Arabic sentiment analysis

Hala Mulki[1,*], Hatem Haddad[2], Mourad Gridach[3] and Ismail Babaoğlu[4]

[1]Department of Computer Engineering, Konya Technical University, Turkey, [2]RIADI Laboratory, National School of Computer Sciences, University of Manouba, Tunisia, [3]Department of Computer Science, University of Oxford, Oxfordshire, United Kingdom and [4]Department of Computer Engineering, Konya Technical Univeristy, Turkey
[*] Corresponding author. E-mail: hallamulki@gmail.com

(Received 5 November 2018; revised 23 June 2019; accepted 1 July 2019; first published online 16 March 2020)

Abstract
Arabic sentiment analysis models have recently employed compositional paragraph or sentence embedding features to represent the informal Arabic dialectal content. These embeddings are mostly composed via ordered, syntax-aware composition functions and learned within deep neural network architectures. With the differences in the syntactic structure and words' order among the Arabic dialects, a sentiment analysis system developed for one dialect might not be efficient for the others. Here we present syntax-ignorant, sentiment-specific n-gram embeddings for sentiment analysis of several Arabic dialects. The novelty of the proposed model is illustrated through its features and architecture. In the proposed model, the sentiment is expressed by embeddings, composed via the unordered additive composition function and learned within a shallow neural architecture. To evaluate the generated embeddings, they were compared with the state-of-the art word/paragraph embeddings. This involved investigating their efficiency, as expressive sentiment features, based on the visualisation maps constructed for our n-gram embeddings and word2vec/doc2vec. In addition, using several Eastern/Western Arabic datasets of single-dialect and multi-dialectal contents, the ability of our embeddings to recognise the sentiment was investigated against word/paragraph embeddings-based models. This comparison was performed within both shallow and deep neural network architectures and with two unordered composition functions employed. The results revealed that the introduced syntax-ignorant embeddings could represent single and combinations of different dialects efficiently, as our shallow sentiment analysis model, trained with the proposed n-gram embeddings, could outperform the word2vec/doc2vec models and rival deep neural architectures consuming, remarkably, less training time.

Keywords: n-gram embeddings; Unordered compositionality; Arabic dialects; Sentiment analysis

1. Introduction

With the recent rapid growth of the Arabic language across social media platforms along with the challenging morphological and high inflectional nature of Modern Standard Arabic (MSA) and Dialectal Arabic (DA), Arabic Sentiment Analysis (ASA) is gaining an increased interest from the Natural Language Processing (NLP) research community. According to the used features, existing ASA systems can be classified into either (a) hand-crafted-based systems where SA models employ linguistic and lexical features usually generated by morphological analysers and semantic resources (Abdulla *et al.* 2013; Sayadi *et al.* 2016; Alomari *et al.* 2017; El-Beltagy, Kalamawy, and Soliman 2017) or (b) text embeddings-based systems which adopt text distributed representations the so-called word/sentence embeddings, where the sentence embeddings are composed out of its

Table 1. Free word order of dialectal Arabic

هالفكرة	انا	حبيتا
O	S	V
هالفكرة	حبيتا	انا
O	V	S
حبيتا	انا	هالفكرة
V	S	O
انا	حبيتا	هالفكرة
S	V	O

constituent word embedding vectors using one of the composition models (Altowayan and Tao 2016; Dahou *et al.* 2016; Gridach, Haddad, and Mulki 2017; Medhaffar *et al.* 2017).

Composition models aim to construct phrase/sentence embeddings based on its constituent word embeddings and structural information (Gormley, Yu, and Dredze 2015). Two main types of these models can be recognised: (a) ordered or syntactic models where the order and linguistic/grammatical structure of the input words do count while constructing the phrase/sentence vector and (b) unordered models in which the word embeddings are combined irrespective of their order using algebraic operations. Sum of word embeddings (SOWE), average (Avg), minimum (Min), maximum (Max) and multiplication (Mul) functions are examples of such models (Mitchell and Lapata 2010).

Based on the theory 'you shall know a word by the company it keeps' (Firth 1957), context words along side their syntactic properties were considered essential to build effective word embeddings able to infer the semantic/syntactic similarities among words, phrases, or sentences. Consequently, most of the recently developed SA systems adopted recursive neural networks (RecNNs) or long, short-term memory neural models (LSTMs) in which ordered composition models are employed to grasp the syntactic and linguistic relations between the words (Socher *et al.* 2013; Al Sallab *et al.* 2015). These systems usually require more training time to learn words' order-aware embeddings due to the high computational complexity consumed at each layer of the model (Iyyer *et al.* 2015). However, the embeddings resulting from ordered compositionality might not be sufficient to handle the Arabic dialects which have a free word order and varying syntactic/grammatical nature (Brustad 2000; Chiang *et al.* 2006). While MSA can be described as a Verb-Subject-Object (VSO) or Subject-Verb-Object (SVO) language, Arabic dialects go beyond that; for instance, the dialectal (Levantine) sentence investigated in Table 1 refers to the meaning 'I liked this idea', can be represented by several word orders: VSO, SVO, OSV and OVS; yet, retains the meaning and the sentiment.

On the other hand, Arabic dialects show phonological, morphological, lexical and syntactic differences such that the same word might infer different syntactic information across several Arabic dialects (Brustad 2000). For example, Table 2 shows how the word 'ماشي' has several part-of-speech (POS) tags, multiple meanings and different sentiments across three Arabic dialects.

To handle the informality of DA, unordered composition models can form an efficient replacement of the syntactic ones, as they construct sentence/phrase embeddings, regardless of the order and the syntax of the contextual words. Nevertheless, it should be noted that for the sentiment analysis task, sentence/phrase embeddings, which are merely composed and learned based on the context words, do not always infer the accurate sentiment. This is usually encountered when words of opposite sentiments are mentioned within identical contexts. Hence, opposite words will be mapped close to each other in the embedding space. To clarify that, both sentences in Examples

Table 2. Syntactic differences across the Arabic dialects

Dialect	Sentence	Word	Meaning	POS	Polarity
Levantine	الوضع **ماشي** الحال	ماشي	Okay	Adjective	Positive
	The situation is okay				
Moroccan	نحن **ماشي** سعداء	ماشي	Not	Negation	Negative
	We are not happy				
Egyptian	كنت **ماشي** للبيت	ماشي	Walking	Verb	Neutral
	I was walking home.				

1 and 2 contain the same context words, organised in the same order; yet, the first sentence implies a positive polarity while the second has a negative sentiment; where the words 'ممتع' (*interesting*) and 'ممل' (*boring*) are antonyms.

Example 1.
هالفيلم كان ممتع بشكل ما بينوصف
This movie was incredibly interesting

Example 2.
هالفيلم كان ممل بشكل ما بينوصف
This movie was incredibly boring

One way to address this issue is to learn the embeddings from sentiment-annotated corpora such that the sentiment information is incorporated along with the contextual data within the composed embeddings during the training phase. This was examined with the English language in Tang *et al.* (2014), where the authors presented sentiment-specific word embeddings (SSWE) composed by Min, Max and Avg unordered composition functions and learned within a simple neural model. The learned embedding vectors were, then, fed to classical supervised classifiers, where a better sentiment classification could be achieved compared to word2vec (Mikolov *et al.* 2013) and hand-crafted features-based systems.

In Iyyer *et al.* (2015), an English SA model called deep averaging neural network (DAN) was presented. DAN was implemented as a neural network with two hidden layers. Based on the assumption that unordered composition functions can efficiently encode sentiment within the composed embeddings, DAN learned and produced n-gram embedding features for the SA task through pairing between the Avg unordered composition function and supervised learning. The experimental study indicated the ability of the embeddings learned by DAN to rival those resulting from ordered syntactic models. Where DAN achieved a comparable performance against RecNNs and CNNs-Multi Channel (CNN-MC) models in which both semantic and syntactic information were encoded using ordered composition functions.

Recently, some ASA systems considered the syntactic information while composing the embeddings for MSA (Al Sallab *et al.* 2015), while other SA models used pretrained and unsupervised word or document (sentence) embeddings to recognise the sentiment of MSA/DA contents (Altowayan and Tao 2016; Gridach *et al.* 2017). However, mining the sentiment of DA using syntax-aware, ordered embeddings might be ineffective especially with the drastic differences between Eastern and Western Arabic dialects (Zaidan and Callison-Burch 2014). In addition, for the SA task, the embeddings learned from unlabeled data are not as discriminating as those learned with the sentiment information integrated in the embedding vectors (Tang *et al.* 2014).

This evokes the need to provide a sentiment-specific, dialect-independent embeddings with which the gap resulting from the differences among Arabic dialects can be bridged. To achieve that, the embeddings, to be used as sentiment features, should be composed in a way that ignores the words' order and the syntactic/contextual regularities and focuses on the semantic and sentiment information.

Inspired by Iyyer *et al.* (2015), we hypothesise that a DA sentence, with a free word order and various syntax, can be better represented if their constituent word embedding vectors are composed using an unordered composition function. On the other hand, motivated by Tang *et al.* (2014), we assume that these composed embedding features can be more expressive for the SA task if they are learned considering the sentiment labels of the input data. Therefore, this study aims to present a SA framework (Tw-StAR) whose features are n-gram embeddings learned from labeled data (sentiment-specific) and composed via the additive unordered composition function (syntax-ignorant) SOWE whose efficiency to capture the semantic information was proved in White *et al.* (2015). The embeddings composition and the sentiment learning processes were conducted within Tw-StAR which was constructed as a shallow feed-forward neural network of a single hidden layer. The contributions of this study can be described as follows:

1. According to Iyyer *et al.* (2015), Shen *et al.* (2015), for certain NLP tasks, achieving a good performance does not necessarily require using sophisticated models; as much simpler word embeddings-based architectures can provide a competent or even better performance compared to recurrence or convolutions-based models. Moreover, the powerful performance of deep neural models is not only attributed to the depth factor but also to other factors such as the recently developed training techniques that matches the properties of the building units used in deep models (Ba and Caruana 2014). Furthermore, in various scenarios where supervised learning is preferred to learn the features, shallow neural nets exhibit better memorisation relative to deep models (Krueger *et al.* 2017). Consequently, in contrast to previous studies, that composed unordered embeddings within deep neural models (Iyyer *et al.* 2015), the embeddings introduced here are generated and learned within a shallow feed-forward neural model; as we are seeking to investigate whether SA of DA can be performed using less training time through employing less complicated neural architectures.
2. Our Tw-StAR model and SSWE model, proposed in Tang *et al.* (2014), are both sentiment informed. This means that the polarity labels are involved in the training, yielding sentiment-specific embeddings. However, unlike SSWE which further seeks to incorporate syntactic alongside sentiment regularities within the learned embeddings, Tw-StAR assumes that the syntactic information cannot be relied on DA input data. Therefore, while SSWE was trained using the so-called *corrupted n-grams* (missing a word) (Tang *et al.* 2014) together with original ones to learn syntactic contextual embeddings, we fed non-corrupted, whole and original input n-grams to Tw-StAR, such that the syntactic contexts regularities were totally ignored. Hence, Tw-StAR learns and produces syntax-ignorant, semantic-aware and sentiment-specific n-gram embeddings, whereas SSWE deduces syntactic-aware and sentiment-specific embeddings.
3. Based on the outperformance of SOWE in sentence semantic similarity and considering its low computation complexity and ability to capture and encode semantic and synonymous information in the resulting composed embeddings (White *et al.* 2015), we assume that SOWE can be an effective replacement of Avg composition function used in Iyyer *et al.* (2015). Within this scope, we provided a comparison between the sentiment classification performances yielded from n-gram embeddings composed by SOWE and Avg functions, respectively.
4. Due to the limited work on ASA, especially for under-represented Arabic dialects, it was not always possible to compare our shallow model with deep neural SA baselines. Therefore, we developed our own deep neural models using CNNs and LSTMs as building units and applied

them to mine the sentiment in the tackled datasets. Consequently, Tw-StAR could be evaluated, in terms of the consumed training time and the achieved classification metrics against more complicated and deeper neural models.

2. Arabic sentiment analysis

ASA, in general, adopt one of two feature types: (a) hand-crafted features that are generated with the assistance of morphological analysers, NLP techniques and lexical resources or (b) text embedding features which are learned automatically from the text itself through unsupervised/supervised learning strategies, such that the usage of words besides the words' semantic, syntactic and synonymous relations can be incorporated within the produced features.

2.1 Hand-crafted features-based models

The most common features used in hand-crafted features-based models are bag-of-words and n-gram schemes in addition to various linguistic/stylistic features usually extracted by morphological analysers (Abdulla *et al.* 2013; Sayadi *et al.* 2016; Alomari, ElSherif, and Shaalan 2017; Elouardighi *et al.* 2017). In addition, for a better sentiment classification performance, some SA systems, the so-called hybrid, tend to use a combination of linguistic and lexicon-derived features (El-Beltagy *et al.* 2017).

N-grams features refer to a sequence of adjoining N items collected from a given text. Extracting n-grams can be thought of as exploring a large piece of text through a window of a fixed size (Piryani *et al.* 2017).

Several combinations of n-gram features were used by Sayadi *et al.* (2016) to perform SA on MSA/Tunisian tweets. The produced features were reduced using information gain (IG) method which can extract the most prominent features with respect to the class attribute. Later, six different classifiers such as support vector machines (SVMs), Naïve Bayes (NB), K-nearest neighbour (KNN), random fields (RF) and decision trees (DTs) were trained with the selected features. Two classification types were performed: binary classification for positive/negative tweets and multi-class classification with neutral tweets considered. The best performance for binary classification was achieved by SVM with an accuracy of 71.1% and an F-measure of 63%.

Aiming to focus on the most informative n-gram features, weighting schemes such as term frequency/term frequency-inverse document frequency (TF/TF-IDF) are usually used. This was examined in Elouardighi *et al.* (2017), where TF/TF-IDF was employed along with combinations of n-gram features to train SVM and NB supervised classifiers. Features reduction was conducted using the score 'Between Group to within Group Sum of Squares' (BSS/WSS) strategy. To evaluate the proposed model, it was applied on an MSA/Moroccan dataset of Facebook comments. The best performance was achieved when unigrams and bigrams were both adopted with TF-IDF weighting scheme, achieving an accuracy of 78%.

In Alomari *et al.* (2017), high-level n-grams were used to classify the sentiment of MSA/Jordanian tweets. With SVM and NB used as classifiers, the authors investigated the impact of stemming/light stemming on the classification performance. It was noted that for bigram training features, which were optimised and reduced by stemming and TF-IDF, SVM was the best performing classifier, as it outperformed NB with an accuracy of 88.7% compared to 81.4%.

Within a lexicon-based SA framework, Karmani (2017) generated linguistic features to mine the sentiment of Tunisian customers' reviews. To produce the linguistic features, the author developed a Tunisian Arabic morphological analyser along with a transliteration machine to handle Arabizi.[a] In addition, a Tunisian version of the SentiWordNet lexicon was used. The model was applied on a manually collected corpus (TAC) which combined positive, negative and neutral

[a]The Arabic chat alphabet written using English letters.

tweets. The binary classification accuracy of the used Tunisian lexicon was 72.1% considering positive/negative tweets.

A comparison between a supervised and a lexicon-based SA models was performed by Abdulla *et al.* (2013). Both models were applied on MSA/Jordanian tweets. Root stemming and light stemming were conducted to enrich the bag-of-words features in the supervised model and to increase the lexicon size in the lexicon-based model. The models were evaluated using a dataset called ArTwitter containing positive/negative tweets. The experimental study indicated the outperformance of the supervised model as it achieved an accuracy of 87.2% using SVM, compared to 59.6% scored by the lexicon-based model.

Despite the good performance achieved by these models, hand-crafted feature generation remains a labor-intensive task as it requires using several NLP techniques such as sentence parsing and POS tagging for which language-specific or dialect-specific morphological tools need to be developed (Piryani *et al.* 2017). On the other hand, lexicon-derived features generated using a certain dialectal lexicon might not be efficient for other datasets even if they are used for the same dialect. This is due to the fact that most lexicons are corpus-based which makes them domain-specific and corpus-specific while high coverage and large-scale dialectal lexicons are, relatively, difficult to build and compile (Abdulla *et al.* 2013). Therefore, many recent SA systems replaced the hand-crafted features with word/document embeddings to be used either to train ML classifiers or to enrich sentiment lexicons.

2.2 Text embedding feature-based models

The recently developed ASA systems have used text embeddings to represent MSA/DA textual content. Distributed text representations or text embeddings can be divided into two main types:

- Word embeddings: in which every word in the corpus is mapped to a real-valued low-dimensional vector in the embedding space using one of the word mapping algorithms such as word2vec (Mikolov *et al.* 2013; Pennington *et al.* 2014).
- Document embeddings: through which continuous representations of larger blocks of text such as sentences, paragraphs or whole documents are learned using a document mapping algorithm such as doc2vec (Le and Mikolov 2014).

In a successful attempt to replace the hand-crafted features with word embeddings, the authors in Altowayan and Tao (2016) generated their own Arabic word embeddings through training Continuous Bag of Words (CBOW) algorithm from Mikolov *et al.* (2013) using an Arabic corpus of 190 million words. The authors indicated that the produced embeddings could handle dialects efficiently as it mapped different writing shapes of dialectal words. To evaluate the generated embeddings, they were used to train several binary classifiers to perform subjectivity and sentiment classification for a combination of twitter datasets: ASTD (Nabil, Aly, and Atiya 2015), ArTwitter (Abdulla *et al.* 2013) and QCRI (Mourad and Darwish 2013); in addition to other two datasets representing book reviews: LABR (Aly and Atiya *et al.* 2013) and MSA news articles obtained from Banea, Mihalcea, and Wiebe (2010). The model's performance was slightly better than Mourad and Darwish (2013) in subjectivity classification, while for the polarity classification of the twitter datasets, the best metric values were scored using Nu-SVM classifier with an accuracy of 80.2% and an F-measure of 79.6%.

To improve the compositionality of the constructed embeddings, a study by Dahou *et al.* (2016) introduced a CNN-based deep learning SA model. The model was trained with word embeddings learned from a corpus of 3.4 billion Arabic words using CBOW and Skip-Gram (SG) (Mikolov *et al.* 2013). Using CNN as a building unit, a neural model with one non-static channel and one convolutional layer was developed. Multiple filter window sizes were adopted to perform the convolutional operation while a max-overtime pooling layer was utilised to capture the most relevant global features (Collobert *et al.* 2011). The model was applied on several datasets such as ASTD (Nabil *et al.* 2015) and ArTwitter (Abdulla *et al.* 2013). The results revealed that the performance of the presented model mostly outperformed all the state-of-the-art systems where for ArTwitter, the achieved accuracy was 85.0%.

The idea of including Arabic pre-trained word embeddings in a deep neural SA model was introduced by Gridach *et al.* (2017). The authors used word embeddings from Zahran *et al.* (2015) which were previously trained with MSA/Egyptian corpora using Glove, SG and CBOW algorithms. These embeddings were employed to initialise the input word embeddings training features of the proposed CNN-ASAWR model. CNN-ASAWR was developed as a variant of Collobert *et al.* (2011) system and customised to conduct SA on two MSA/dialectal datasets: ASTD (Nabil *et al.* 2015) and SemEval-2017 (Rosenthal *et al.* 2017). The results showed that using pre-trained word embeddings led to better evaluation measures compared to the baseline systems. In ASTD dataset, for instance, the best F-measure scored by CNN-ASAWR was 72.14% compared to 62.60% achieved by Nabil *et al.* (2015), while for SemEval, an F-measure of 63% is achieved against 61% scored by the system of El-Beltagy *et al.* (2017).

One of the pioneering studies towards leveraging document or sentence embeddings in ASA was conducted by Medhaffar *et al.* (2017), where doc2vec algorithm was used to provide sentence feature vectors representing a Tunisian corpus. The proposed model was evaluated using a combination of publicly available MSA/DA datasets: OCA (Rushdi Saleh *et al.* 2011), LABR (Aly and Atiya *et al.* 2013) and a manually annotated Tunisian Sentiment Analysis Corpus (TSAC) obtained from Facebook comments about popular TV shows. Each comment was represented by a document vector which was used later to train SVM, Bernoulli NB (BNB) and multilayer perceptron (MLP) classifiers. The best results were scored by MLP classifier when TSAC corpus was solely used as a training set where it achieved an accuracy equals to 78% and an F-measure of 78%.

As each deep neural model has specific merits, usually related to its building unit, the authors in Al-Azani and El-Alfy (2017) investigated using various configurations of deep learning models to perform SA of Arabic tweets. The examined models were built either using a separate/stacked units of CNN, LSTM or by cascading CNN and LSTM within a single model. All the models were trained with word embedding features provided by word2vec techniques: CBOW and skip-gram with static and non-static word initialisation enabled. To evaluate the different model variants, ASTD dataset from Nabil *et al.* (2015) and ArTwitter by Abdulla *et al.* (2013) were used. The results indicated that updating the word embeddings during learning achieved the best results in most model configurations. In addition, while LSTM outperformed CNN in general, combined LSTMs architecture was the best-performing model achieving an accuracy of 87.2% for ArTwitter dataset.

In the above-mentioned studies, it could be observed that the used word/document embeddings were mostly context-aware as they have been generated via context-based algorithms such as SG and CBOW (Mikolov *et al.* 2013). This implicitly involved relying on the word order and context/syntactic nature (Mikolov *et al.* 2013), which might not always produce expressive features especially for DA where neither fixed word order nor syntactic rules do exist. Moreover, with the proposed systems being evaluated using mixed MSA-DA datasets, their ability to handle the DA-dominated or pure dialectal contents remains inconclusive. Therefore, we believe that our syntax-ignorant embeddings are crucial to handle the dialectal content as they cope with DA specifications, while their efficiency for SA against context-based techniques could be assessed through the exploration of the spatial relations among DA words in the embeddings space.

On the other hand, while the state-of-the-art employed different deep learning architectures, none of which provided all the needed metrics. This makes it a bit difficult to compare these systems with each other, let alone the evaluation of our shallow model against them. Especially that, a sentiment classifier performance cannot be only evaluated according to the achieved evaluation measures but with regard to other implementation details such as the architecture complexity, computation overhead and the consumed training time. For this purpose, we had to develop deep architectures to be applied on the tackled datasets and thus, we could obtain a comprehensive evaluation of the proposed shallow model Tw-StAR against more complicated deeper models in terms of the achieved evaluation measures and the consumed training time.

3. Tw-StAR model description

As seeking to answer the question: Can a shallow neural model, trained with embeddings specifically formulated to target the dialectal content, rival more complicated neural architectures? Tw-StAR model was implemented as a feed-forward neural network in which sentiment-specific, syntax-ignorant n-grams embeddings are composed via SOWE function and learned in a supervised manner. The generated n-gram embeddings were, then, employed as discriminating features to predict the positive/negative sentiment of DA contents.

As it is shown in Figure 1, given the input negative training tweet “نصيحة لا حدا ينزل الابديت ويندوز الجديد لما بيعمل ابديت بيخر ب ما بيحسن” (*upon update, the new version of Windows makes things worse rather than better; as an advice, do not update your system*), a sequence of six n-grams is generated by going through the tweet using a fixed-size sliding window. Having the n-grams generated, each of which is associated with the polarity [0, 1] which represents the negative polarity of the previous tweet. Later, n-grams are fed to Tw-StAR model where the correspondent embeddings for their constituent words are constructed at the embeddings layer, composed and formulated as n-gram embeddings at Lambda layer and learned as sentiment features while being forwarded through Tw-StAR layers. Finally, n-gram embeddings are exploited, at the output layer, to recognise the sentiment of the input tweet. Below is a detailed description of each layer, where the model's notations are listed in Table 3.

The embedding layer is responsible for projecting words in the input into their corresponding dense vector representations. Given the input sentences, in order to handle their varying lengths, each sentence S of l words was formulated as a sequence of fixed-length n-grams generated using a sliding window of a specific size C. Whole n-grams were fed to the embedding layer of our model such that each n-gram is accompanied with the sentiment label of the sentence from which it was derived. Having the n-grams prepared and accompanied with the vector representing their relevant sentiment label ([1, 0] for positive and [0, 1] for negative), their constituent words were mapped into the corresponding embeddings based on the embeddings weight matrix $M \in R^{|V| \times d}$ where $|V|$ is the vocabulary size and d denotes the embeddings dimension.

The weights of the embedding matrix M were initialised using Glorot uniform initialisation (Glorot and Bengio 2010), then optimised while training the model. It should be noted that we could not use pretrained word embeddings for initialisation, as the available Arabic pretrained word embeddings from Zahran *et al.* (2015) and Al-Rfou, Perozzi, and Skiena (2013) were generated using MSA/Egyptian corpora, which can lead to out-of-vocabulary (OOV) issues, especially for the Tunisian and Moroccan contents, where less common words with MSA/Egyptian do exist. Thus, for a single fixed-length n-gram containing a sequence of words $\{w_i, w_{i+1}, w_{i+2}, ..., w_{i+C-1}\}$, each word w_i is represented by a unique integer index $i \in [0,\text{V}]$ and stored as a one-hot vector vec_i whose values are zero in all positions except at the i-th index. To obtain the embedding vector v_i of a word w_i, its one-hot vector vec_i is multiplied by the embedding matrix M as in Equation (1)

$$v_i = \text{vec}_i * M \in R^{1 \times d} \tag{1}$$

As each row of the embedding matrix M denotes the dense embeddings of a specific word in the vocabulary, multiplying the one-hot vector of each word in the input by the embedding matrix M will essentially select one of M rows that corresponds to the embeddings of that word.

At the next linear layer Lambda, the resulting word embeddings generated for each word of the input n-gram were combined using the compositional model SOWE. Hence, the output of lambda layer is a single embeddings vector $O_{\text{lambda}} \in R^{1 \times |d|}$, resulting from the element-wise sum

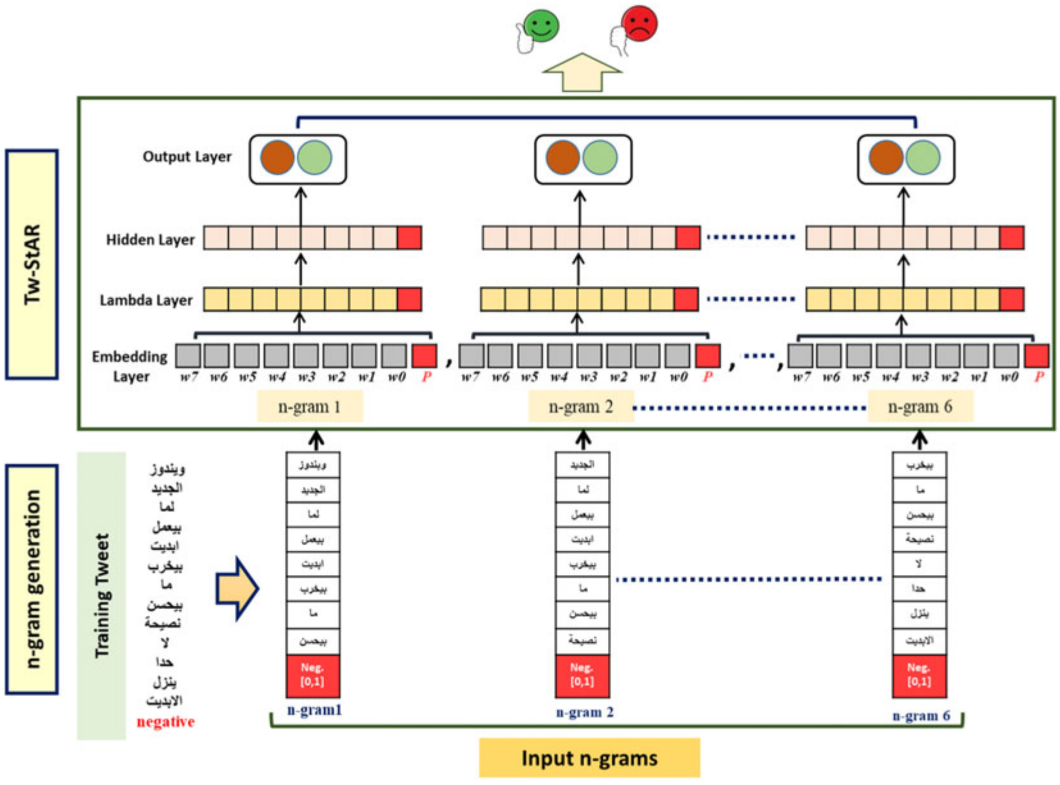


Figure 1. Tw-StAR neural sentiment analysis model.

of the embeddings vectors produced by the embedding layer and correspond to the input words that belong to a single n-gram and contained in a window of a fixed-size C:

$$O_{\text{lambda}} = \sum_{i=1}^{C} v_i \in R^{1\times d} \tag{2}$$

In the subsequent hidden layer (hl), the output from the previous layer O_{lambda} is subjected to a linear transformation using the weights matrix $W_{hl} \in R^{d\times 2}$ and biases $b_{hl} \in R^{1\times 2}$:

$$O_{hl} = f(O_{\text{lambda}} * W_{hl} + b_{hl}) \in R^{1\times 2} \tag{3}$$

Where W_{hl} and b_{hl} form the model's parameters that are learned and optimised during the training process, and f refers to the activation function that introduces non-linear discriminating features to our model. Here we used Hard sigmoid activation function h_σ (Courbariaux *et al.* 2016), which is identified in Equation (4). Hard sigmoid is an approximation of the standard sigmoid activation function; it is defined as a piece-wise function whose output is very similar to the traditional sigmoid; however, it is computationally cheaper. This leads to a smarter model with the learning process accelerated in each iteration (Gulcehre *et al.* 2016).

$$h\_\sigma(x) = \text{clip}((x+1)/2, 0, 1) = \max(0, \min(1, (x+1)/2)) \tag{4}$$

where clipping bounds are −2.5 and 2.5 and its derivative is given as below:

$$h\hat{\_}\sigma(x) = \begin{cases} 0 & \text{if} \quad x < -2.5 \quad \text{or} \quad x > 2.5 \\ 0.2 & \text{otherwise} \end{cases} \tag{5}$$

Table 3. Notations used of Tw-StAR model

Symbol	Description
C	Sliding window's size
w	Input word
i	Integer index of a word
M	Weight embedding matrix
$\lvert V \rvert$	Vocabulary size
d	Embedding dimension
vec_i	One-hot vector of w_i
v_i	Embedding vector of w_i
O_{lambda}	Output of lambda layer
W_{hl}	Hidden layer's weights
b_{hl}	Hidden layer's biases
O_{hl}	Output of the hidden layer
h_σ	Hard sigmoid activation function
$\hat{y}$	Predicted sentiment label
y	The gold sentiment label
k	Number of the classes
θ	Model's weights and biases

Finally, the output O_{hl} resulting from the hidden layer is forwarded into the output layer (Ol), where a softmax function is applied to induce the estimated probabilities for each output label (positive/negative) of a specific n-gram. Where each n-gram is accompanied with the predicted two-dimensional label [1, 0] denoting positive or [0, 1] indicating negative.

$$\hat{y} = \mathrm{softmax}(O_{hl}) \in R^{1\times 2} \tag{6}$$

Softmax selects the maximum score among the two predicted conditional probabilities to denote positive or negative polarity of an input n-gram, where the distribution of the form [1, 0] was assigned for positive, while [0, 1] distribution form was adopted for negative. Thus, if the gold sentiment polarity of an n-gram is positive, the predicted positive score should be higher than the negative score, while if the gold sentiment polarity of a word sequence is negative, its positive score should be smaller than the negative score. Then, to decide the polarity of the whole sentence, the predicted positive scores and negative scores of n-grams are summed, then each of which is divided by the number of the n-grams contained in this sentence resulting in two values representing the potential positive and negative scores of the input sentence. The final sentence polarity is, thus, decided according to the greater among these two values. Cross-entropy loss between gold sentiment distribution and predicted distribution was adopted such that the loss function of the model is:

$$J(\theta) = -\sum_{k=\{0,1\}} y_k \log \hat{y}_k \tag{7}$$

where $y \in R^2$ is the gold sentiment value represented by a one-hot vector, $\hat{y}$ is the sentiment predicted by the model, while θ refers to the parameters (weights and biases) of the model to be learned and optimised during the training process.

4. Training details and model's parameters

The key hyperparameters of the proposed model are the sliding window size C which defines the n-gram scheme to be adopted and the embeddings dimension d. We have selected both parameter values empirically during the model tuning period. According to Socher *et al.* (2013), narrower windows lead to better performance in syntactic tests, while wider ones score a better performance in semantic tests. Therefore, we have tested different quite large values of the sliding window size C to select the size that achieves the best evaluation measures. Similarly, the emdeddings dimension size was selected empirically among several examined values.

For efficient training, Glorot uniform initialisation (Glorot and Bengio 2010) was used to set the weights of the embeddings layer. Glorot initialisation makes sure the weights are 'just right' across the model's layers, keeping the signal in a reasonable range of values, through avoiding too massive and too tiny weight values. In addition, back-propagation algorithm with Adaptive Moment estimation (Adam) and stochastic optimisation method (Kingma and Ba 2014) were used for tuning the model during the training phase. Adam optimiser combines the early optimisation speed of Adagrad (Duchi, Hazan, and Singer 2011) with the better later convergence of various other optimisation methods such as Adadelta (Zeiler 2012) and RMSprop (Tieleman and Hinton 2012).

The over-fitting issue was handled using the dropout regularization mechanism, where the value of the dropout parameter was selected empirically during the model's tuning period. This is done through calculating learning rates and storing momentum changes for each model's parameter separately.

5. Datasets

Tw-StAR was evaluated using MSA/DA datasets harvested from social media platforms and annotated for positive/negative polarities. We can distinguish between Western (Tunisian, Moroccan) and Eastern (Jordanian, Gulf) dialects as follows:

- Arabic Jordanian General Tweets (AJGT): A small-sized dataset composed of 1,800 positive/negative social Jordanian tweets obtained by Alomari *et al.* (2017).
- Arabic Twitter Dataset (ArTwitter): A small-sized political/social dataset collected and made publicly available by Abdulla *et al.* (2013), it combines 2000 positive/negative tweets mostly written in the Jordanian dialect. After reducing blank entries, a dataset of 1979 tweets was used.
- Tunisian Election Corpus (TEC): A medium-sized political dataset of 5521 positive, negative and neutral tweets collected by Sayadi *et al.* (2016) during the Tunisian elections period. It combines MSA/Tunisian dialect where Tunisian tweets form the majority of the data. Here, we used the reduced version of this dataset having 3043 positive/negative tweets.
- TSAC: A large-sized media/social dataset of 9976 positive/negative Facebook comments provided by Medhaffar *et al.* (2017). In this study, we filtered the Arabizi instances out of this dataset such that 7366 comments were used.
- Moroccan Election Corpus (MEC): Refers to a large-sized social/political dataset of 10,253 positive/negative tweets collected by Elouardighi *et al.* (2017) during the Moroccan elections in 2016.

Table 4. Statistics of the used datasets

Dialect	Dataset	Size	#words	Avg.S.L.	Train	Dev	Test
Jordanian	ArTwitter	1,979	6,083	9	1,266	317	396
	AJGT	1,800	5,933	9	1,152	288	360
Tunisian	TEC	3,043	9,457	11	1,947	487	609
	TSAC	7,366	15,005	10	4,680	1,170	1,516
Moroccan	MEC	10,253	31,546	15	6,561	1,641	2,051
Multi-dialects	JEG	4,294	16,445	12	2,747	687	860
Multi-dialects	TEAD	555,924	372,403	13	355,792	88,948	111,184

Table 5. F-measure values (%) obtained with dev sets for different window sizes

Dataset	$C=4$	$C=5$	$C=6$	$C=7$	$C=8$	$C=9$	$C=10$
AJGT	80.0	79.1	79.1	79.9	**82.0**	80.8	76.7
ArTwitter	81.4	82.7	82.7	83.0	**83.3**	82.3	81.5
TEC	84.7	85.1	87.6	**87.9**	**87.9**	83.6	81.2
TSAC	71.1	81.9	86.1	85.9	**86.6**	86.5	86.3
MEC	67.5	66.3	63.9	**68.6**	**68.6**	67.1	66.5
JEG	71.3	72.4	73.4	73.4	**73.8**	73.3	72.5

Boldface refers to the best F-measure values achieved within every dataset for each value of the window size C. For $C=8$, F-measure were the best (highest) in all datasets. Therefore, $C=8$ was adopted in the experiments.

- Jordanina Egyptian Gulf (JEG): A medium-sized multiple domains dataset investigated in Altowayan and Tao (2016). It combines 4294 positive/negative tweets from three datasets of MSA and dialectal content including: (a) Jordanian: Artwitter (Abdulla *et al.* 2013), (b) Egyptian: ASTD (Nabil *et al.* 2015) and (c) Gulf: QCRI (Mourad and Darwish 2013).
- Tweets Emoji Arabic Dataset (TEAD): A large scale dataset combines tweets from multiple domains.[b] It is composed of 555,924 positive/negative tweets written in several Eastern and Western Arabic dialects.

Each dataset was divided into a training set to train the model, dev set for tuning the parameters and a test set for evaluation. The detailed statistics of these sets are reviewed in Table 4, where Train, Dev, Avg.S.L. refer to the training, developing and average lengths of sentences, respectively.

6. Results and discussion

6.1 Hyper parameters adjustment of Tw-StAR model

The hyper parameters (C, d) of Tw-StAR were assigned empirically. Among several window sizes ranging from 4 to 10, a window size value of 8 was adopted as it produced the best F-measure in all datasets using the validation Dev dataset (see Table 5). Consequently, each input sentence is represented by a set of 8-grams to be fed to the model. Similarly, upon examining three embedding dimension sizes equal to 50, 100 and 150, and several dropout rates ranging from 0.2 to 0.5, $d=100$ and dropout=0.2 were adopted, since these values scored the best F-measure during the model's tuning period.

[b]https://github.com/HSMAabdellaoui/TEAD.

6.2 Syntax-ignorant n-gram embeddings evaluation

The efficiency of our n-gram embeddings, composed by SOWE, was evaluated against word embeddings (word2vec) and document embeddings (doc2vec). To conduct a fair comparison, word2vec (Mikolov *et al.* 2013) and doc2vec (PV-DBoW/PV-DM) (Le and Mikolov 2014) algorithms were trained on each of the studied datasets with sentiment labels included in the training process. In addition, the training parameters such as the window size and embedding dimensions were unified across the evaluated embedding methods: word2vec, doc2vec (PV-DBoW/PV-DM) and Tw-StAR. It should be noted that, within doc2vec we can recognise two mapping methods: (a) distributed bag of words (DBoW) that learns and composes the sentence embeddings regardless of the order of words and (b) distributed memory (DM) that follows the CBOW mechanism, as it considers the words' order while learning the sentence embeddings vector (Le and Mikolov 2014).

Having the word embeddings, document embeddings and n-gram embeddings generated for each of the studied corpora, they were used one by one as features to train Tw-StAR on recognising the sentiment embedded in the datasets in Table 4. To train our model with word2vec and doc2vec embeddings, the embedding layer in Tw-StAR was replaced with the embeddings produced by word2vec and both variants of doc2vec, respectively. These embeddings were then passed through the shallow architecture of Tw-StAR. Table 6 lists the the sentiment classification performances achieved using n-grams by SOWE, word vectors by word2vec and sentence vectors by doc2vec (PV-DBoW/PV-DM) for all datasets.

The results in Table 6 suggest the outperformance of the proposed embeddings over those generated by word2vec and doc2vec models with a significant margin in F-measure values for most datasets. The best F-measure value was achieved for TEC dataset with a value of 87.8% compared to 58.4%, 56.4% and 56.7% scored by word2vec, doc2vec (PV-DM) and doc2vec (PV-DBoW), respectively. This could be explained by the ability of SOWE to accurately capture the semantic information along with the synonymous relations among words coping with the claim stated in White *et al.* (2015). This was further emphasised through examples from the visualisation maps provided in Section 6.3. In addition, for JEG dataset that combines three different dialects, the F-measure obtained using n-gram embeddings increased by 15.1%, 16.9% and 16.1% compared to word2vec, doc2vec (PV-DM) and doc2vec (PV-DBoW), respectively. This indicates how the proposed embeddings can address the differences among various dialects through ignoring the syntactic structure and word order, and focusing on the semantic relations based on the ability of SOWE to efficiently enrich the composed n-gram embeddings with semantic/synonymous regularities.

On the other hand, it can be seen from Table 6 that for datasets, having an MSA-dominated content such as MEC, doc2vec (PV-DM), which takes words' order into account, performed significantly better than word2vec and doc2vec (PV-DBoW). Indeed, the achieved accuracy for MEC dataset with the embeddings learned by doc2vec (PV-DM) was 76.6% compared to 69.1% and 69.3% scored by word2vec and doc2vec (PV-DBoW), respectively.

6.3 Syntax-ignorant n-gram embeddings visualisation

Aiming to inspect the performance of the n-gram embeddings thoroughly, we visualised text vectors learned by Tw-StAR and compared them towards word vectors generated by word2vec and paragraph vectors learned via doc2vec (PV-DBoW). This was done by projecting the embedding vectors into a two-dimensional space using the t-distributed stochastic neighbour embedding (t-SNE) technique (van der Maaten and Hinton 2008).

From the figures reviewed in Table 7, a clustering behaviour of the words that compose n-grams or document embeddings could be observed in both Tw-StAR and doc2vec (PV-DBoW) models. In word2vec model, however, word vectors tend to spread sparsely in the embeddings space. This was reflected on the performance of the embeddings as being discriminating features for the SA task. To clarify that, considering TSAC dataset, we have noticed that pure Tunisian dialectal words

Table 6. Tw-StAR performances for n-gram, word2vec and doc2vec embeddings

Dataset	Embeddings	P. (%)	R. (%)	F1 (%)	Acc. (%)
AJGT	word2vec	72.1	73.1	71.2	71.4
	doc2vec (DM)	54.4	54.4	51.9	51.9
	doc2vec (DBoW)	58.0	57.7	54.4	54.4
	n-gram (Tw-StAR)	**82.5**	**83.2**	**82.8**	**83.3**
ArTwitter	word2vec	72.0	71.9	71.9	72.0
	doc2vec (DM)	61.2	60.7	60.1	60.4
	doc2vec (DBoW)	63.1	60.6	58.2	59.9
	n-gram (Tw-StAR)	**85.4**	**84.9**	**84.8**	**84.9**
TEC	word2vec	62.6	59.7	58.4	61.9
	doc2vec (DM)	65.6	59.3	56.4	62.2
	doc2vec (DBoW)	62.9	58.9	56.7	61.4
	n-gram (Tw-StAR)	**87.4**	**88.4**	**87.8**	**88.2**
TSAC	word2vec	78.0	77.2	77.4	78.2
	doc2vec (DM)	61.0	58.3	57.2	61.7
	doc2vec (DBoW)	55.9	54.1	52.1	58.0
	n-gram (Tw-StAR)	**86.2**	**86.3**	**86.2**	**86.5**
MEC	word2vec	63.6	64.0	63.8	69.1
	doc2vec (DM)	74.7	65.0	66.4	76.6
	doc2vec (DBoW)	60.4	56.6	56.4	69.3
	n-gram (Tw-StAR)	**76.2**	**71.2**	**72.8**	**79.2**
JEG	word2vec	59.3	59.2	59.2	59.4
	doc2vec (DM)	58.5	57.9	57.4	58.4
	doc2vec (DBoW)	61.2	59.4	58.2	60.2
	n-gram (Tw-StAR)	**75.8**	**74.3**	**74.3**	**74.8**
TEAD	word2vec	70.3	60.8	61.4	74.3
	doc2vec (DM)	69.5	60.3	60.8	74.0
	doc2vec (DBoW)	**73.5**	61.1	61.7	**75.3**
	n-gram (Tw-StAR)	67.2	**61.9**	**62.7**	73.3

Boldface indicates the best performance (highest values of evaluation measures) and the best performing embeddings among word2vec, doc2vec (DM), doc2vec (DBoW) and n-gram (Tw-StAR).

such as "إنحبوك" (*we love you*), "يعجبنا" (*we like it*), "باهي" (*good*), "نحييك" (*we praise you*), which bear positive sentiments, were mapped by Tw-StAR model close to each other in the embeddings space. However, when looking to the representations created for the same dataset by doc2vec (PV-DBoW), we have come through the words "إنحبوك" (*we love you*), "مذهلة" (*magnificent*) and "هايلة" (*excellent*), which all refer to a positive sentiment, yet, they are mapped close to the negative words "ممسطها" (*dull*), "ماسطين" (*how boring*) and "خامج" (*a dirty man*) in the embeddings space.

6.4 SOWE composition function evaluation

According to White *et al.* (2015), in the context of sentence semantic similarity, SOWE was proved to be the best-performing composition function in expressing and encoding the semantic

Table 7. t-SNE visualisation of word vectors learned by word2vec/doc2vec against word vectors learned by Tw-StAR

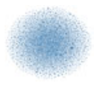
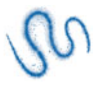
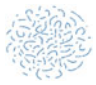
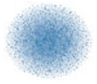
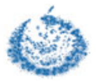
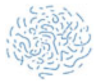
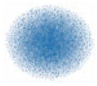
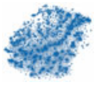
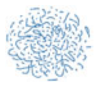
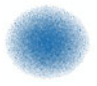
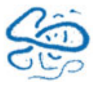
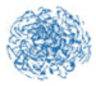
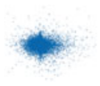
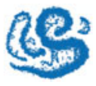
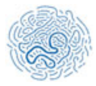
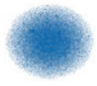
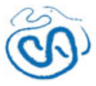
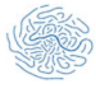
Dataset	word2vec	doc2vec	Tw-Star
AJGT			
ArTwitter			
TEC			
TSAC			
MEC			
JEG			

Table 8. AVG, SOWE impact on SA of the dialectal datasets

		SOWE		Avg	
Dialect	Dataset	F (%)	Acc. (%)	F (%)	Acc. (%)
Jordanian (Eastern)	AJGT	**82.8**	**83.3**	82.2	83.1
	ArTwitter	**84.8**	**84.9**	83.8	83.9
Tunisian (Western)	TEC	87.9	88.3	**89.0**	**89.5**
	TSAC	86.2	86.5	**87.1**	**87.7**
Moroccan (Western)	MEC	72.8	79.2	**74.0**	**80.6**
Jordanian-Egyptian-Gulf (Eastern)	JEG	**74.3**	**74.8**	73.5	74.2
Eastern+Western	TEAD	62.7	73.3	**68.1**	**75.7**

Boldface indicates the best performance (F-measure, Accuracy) achieved by either SOWE or AVG functions for each dataset.

information within the phrase/paragraph embedding vectors. Based on that, we investigated how SOWE would perform for the sentiment analysis task compared to another unordered composition function: Avg, which was used in Iyyer *et al.* (2015). For this purpose, we composed our n-gram embeddings first by SOWE then via Avg function such that both embedding variants were used to train Tw-StAR model.

Table 8 reviews the performances resulting from training Tw-StAR with embeddings composed by SOWE and Avg functions for the studied datasets.

Considering the results shown in Table 8, we can observe that for the Eastern dialect datasets: AJGT, ArTwitter and JEG, SOWE produced more expressive features than the Avg function. However, for the Western dialect datasets, Avg could slightly outperform SOWE as the achieved

F-measure values were 89.0%, 87.1% and 74.0% compared to 87.9%, 86.2% and 72.8% for TEC, TSAC and MEC datasets, respectively. This could be attributed to the nature of the Western dialects where many transliterated words derived from French, Spanish and Tamazight languages do exist (Zaidan and Callison-Burch 2014). Being transliterated, these words are usually written in different writing styles which makes SOWE missing the synonymous relations among such words bearing same or close sentiments, and hence leads to less expressive embedding features compared to Avg.

6.5 Tw-StAR shallow architecture evaluation

Through the proposed model, we are seeking to obtain an efficient sentiment classification performance using a less complicated architecture and with the least time overhead. Therefore, Tw-StAR was designed as a shallow feed-forward neural network with one hidden layer. The ability of Tw-StAR to rival deep neural models was examined by feeding our n-gram embeddings to train two deep neural models having the building units: CNN and LSTM, in addition to the DAN model developed in Iyyer *et al.* (2015) and described in Section 1. The CNN-based model was cloned from Kim (2014), while LSTM-based model was developed with several depths (2,3,4,5) formulated by stacking LSTM layers. Within these architectures, two type of experiments were conducted:

- The first experiment involved using training embeddings composed by SOWE composition function.
- The second experiment, however, employed n-gram embeddings composed via Avg composition function.

Thus, the efficiency of the shallow Tw-StAR model was determined through conducting a comparison between the sentiment classification performances yielding from the previous shallow/deep SA models for SOWE and AVG compositionalities. Tables 9 and 10 summarise the obtained results, where (Arch.) and (Time) refer to the adopted model architecture and the consumed training time, respectively. It should be noted that, for LSTM-based model, we selected the best performances achieved by various depths. In addition, DAN system in Table 9 was trained with embeddings composed by SOWE and not by AVG as in Iyyer *et al.* (2015), while its 2-hidden layers deep architecture was retained.

As it can be seen from Table 9, in AJGT, ArTwitter and JEG datasets, Tw-StAR outperformed LSTM-based, CNN-based and DAN deep models, whereas it achieved a slightly better F-measure for MEC and TEAD datasets. A comparable performance, however, was observed for TEC and TSAC, where LSTM-based model was the best-performing system for these datasets. This could be attributed to the efficiency of SOWE in producing more discriminating features for Eastern dialect datasets such as AJGT, ArTwitter and JEG compared to Western dialect collections. However, in general, we cannot ignore the ability of Tw-StAR to be an efficient replacement of more complicated deep models. Especially that, for most datasets, Tw-StAR managed to provide a quite good sentiment classification performance with fewer parameters and much faster training time compared to LSTM-based, CNN-based and DAN deep models.

Similarly, the results listed in Table 10 show the competent performance achieved by Tw-StAR compared to the other models where it scored the best F-measure values for TEC, MEC and JEG datasets, whereas comparable performances were obtained for AJGT, ArTwitter, TSAC and TEAD datasets.

6.6 Tw-StAR training time evaluation

Besides the competent performance of the our shallow model Tw-StAR against more complicated deep architectures, Tw-StAR could accomplish the training phase consuming less time compared

Table 9. Tw-StAR versus CNN, LSTM and DAN deep models with SOWE function

Dataset	Arch.	Depth #	F1 (%)	Acc. (%)	Time
AJGT	**Tw-StAR**	1	**82.8**	**83.3**	23 seconds
	LSTM	4	81.6	82.8	77 seconds
	CNN	3	80.1	81.4	34 seconds
	DAN	2	79.0	79.7	**20 seconds**
ArTwitter	**Tw-StAR**	1	**84.1**	**84.1**	26 seconds
	LSTM	2	82.2	82.0	102 seconds
	CNN	3	80.0	80.2	41 seconds
	DAN	2	80.7	81.0	**24 seconds**
TEC	Tw-StAR	1	87.8	88.2	52 seconds
	LSTM	5	88.0	88.3	188 seconds
	CNN	3	87.1	88.0	75 seconds
	DAN	2	**88.6**	**89.2**	**47 seconds**
TSAC	Tw-StAR	1	86.2	86.5	121 seconds
	LSTM	2	**89.1**	**89.4**	412 seconds
	CNN	3	84.9	85.8	168 seconds
	DAN	2	87.3	87.7	**107 seconds**
MEC	**Tw-StAR**	1	**72.8**	**79.2**	466 seconds
	LSTM	5	72.1	77.2	1292 seconds
	CNN	3	70.1	76.3	637 seconds
	DAN	2	71.1	76.9	**388 seconds**
JEG	**Tw-StAR**	1	**74.3**	**74.8**	91 seconds
	LSTM	5	72.3	72.7	290 seconds
	CNN	3	73.9	74.1	130 seconds
	DAN	2	72.0	72.7	**80 seconds**
TEAD	**Tw-StAR**	1	**62.7**	73.3	20 hours
	LSTM	4	62.7	73.4	31 hours 30 minutes
	CNN	3	62.0	74.6	24 hours
	DAN	2	62.2	**74.1**	**19 hours 50 minutes**

Boldface refers to the best performance (highest values of F1 and Accuracy) and the best training time (least values of Time) achieved by the different architectures (LSTM, CNN, DAN and Tw-StAR).

to LSTM-based and CNN-based models. This is reviewed in Tables 9 and 10 as we can see that, in AJGT dataset, it took 23 seconds to train the features composed by SOWE (Table 9), while LSTM and CNN models consumed 77 seconds and 34 seconds, respectively. Similar behaviour could be observed in Table 10.

This could be explained by the high computational complexity consumed at each layer of the LSTM model, where at every time step, in addition to the recurrent input, if the input is already yielded from an LSTM layer (in the case of stacked LSTMs), the current LSTM, then, can create

Table 10. Tw-StAR versus CNN, LSTM and DAN deep models with Avg function

Dataset	Arch.	Depth #	F1 (%)	Acc. (%)	Time
AJGT	Tw-StAR	1	82.2	83.1	23 seconds
	LSTM	5	**82.4**	**83.3**	78 seconds
	CNN	3	80.4	81.9	34 seconds
	DAN	2	79.6	81.1	**21 seconds**
ArTwitter	TW-StAR	1	83.3	83.3	26 seconds
	LSTM	1	**84.4**	**84.4**	99 seconds
	CNN	3	82.5	82.6	41 seconds
	DAN	2	76.5	76.8	**25 seconds**
TEC	**Tw-StAR**	1	**89.0**	**89.5**	52 seconds
	LSTM	4	87.5	87.8	194 seconds
	CNN	3	86.3	86.7	76 seconds
	DAN	2	87.9	88.3	**47 seconds**
TSAC	Tw-StAR	1	87.1	87.7	123 seconds
	LSTM	3	**88.9**	**89.2**	411 seconds
	CNN	3	87.3	87.9	169 seconds
	DAN	2	81.7	83.2	**108 seconds**
MEC	**Tw-StAR**	1	**74.0**	**80.6**	478 seconds
	LSTM	5	72.9	79.0	1316 seconds
	CNN	3	71.0	79.3	641 seconds
	DAN	2	73.5	79.3	**400 seconds**
JEG	**Tw-StAR**	1	**73.1**	**73.5**	91 seconds
	LSTM	4	71.4	72.0	286 seconds
	CNN	3	72.8	73.3	129 seconds
	DAN	2	72.4	73.0	**81 seconds**
TEAD	Tw-StAR	1	68.1	75.7	**18 hours 20 minutes**
	LSTM	4	66.2	75.0	31 hours 10 minutes
	CNN	3	62.3	75.1	25 hours 30 minutes
	DAN	2	**69.5**	**77.4**	18 hours 40 minutes

Boldface refers to the best performance (highest values of F1 and Accuracy) and the best training time (least values of Time) achieved by the different architectures (LSTM, CNN, DAN and Tw-StAR).

a more complex feature representation of the current input (Al-Azani and El-Alfy 2017); which, in turn, raises the time overhead. In addition, the learning mechanism adopted by the CNN-based model involves detecting multiple feature patterns through using various kernel sizes then concatenating their outputs at each convolution (Kim 2014); consequently, more time is required for training compared to that needed by Tw-StAR. The training time consumed across the studied models: CNN, LSTM, DAN and Tw-StAR is illustrated in Figure 2 for SOWE (Figure 2(a)) and Avg (Figure 2(b)) composition functions, respectively.

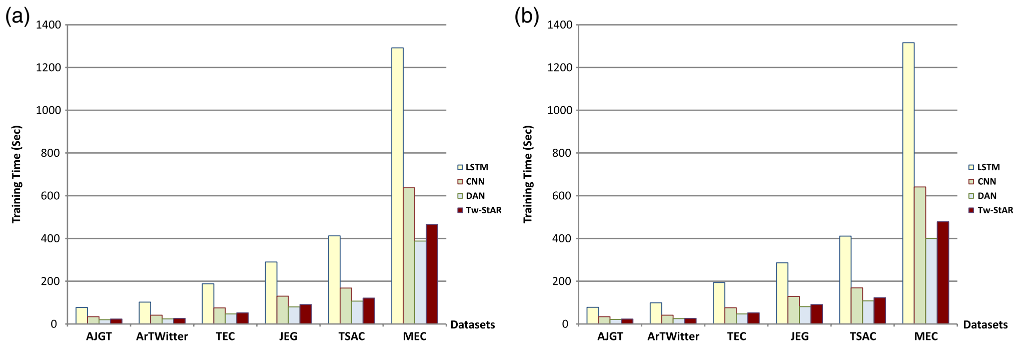


Figure 2. Training time across all architectures for SOWE and Avg functions. (a) Training time with SOWE. (b) Training time with Avg.

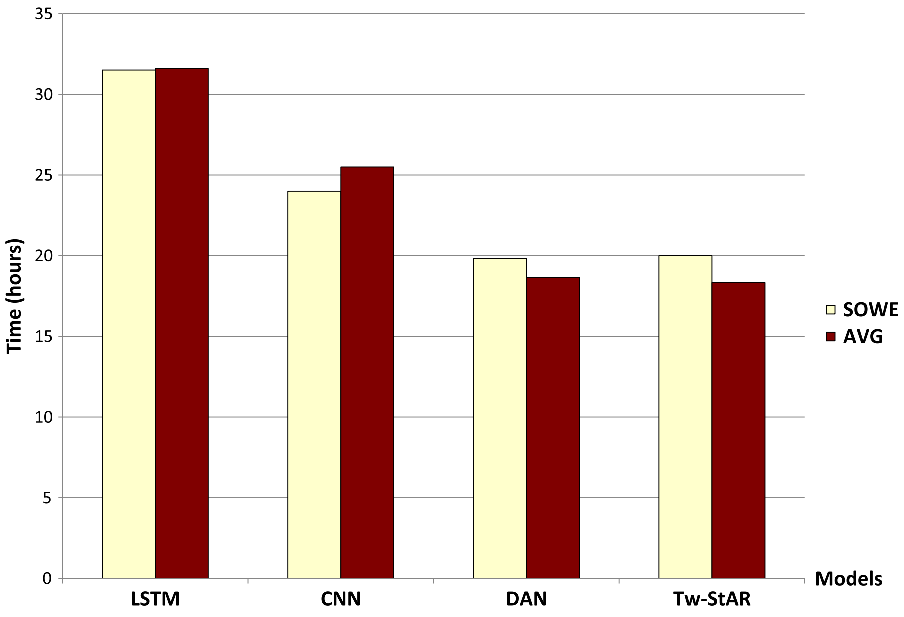


Figure 3. Training time consumed by all models for SOWE, Avg in TEAD.

On the other hand, considering Figure 2(a) and (b), it could be noted that the 2-hidden layer deep model (DAN) proposed by Iyyer *et al.* (2015) achieved the best training time among all the model architectures for both SOWE and Avg composition functions. However, when exploring the training time values recorded for Tw-StAR model and those of DAN's, we can see that the latter is consuming slightly less time compared to Tw-STAR as DAN needed a training time less by 3, 2, 5 and 11 seconds for AJGT, ArTwitter, TEC and JEG datasets, respectively. In addition, for large-sized datasets such as MEC and TSAC, although this time difference increases, yet, it does not exceeds 78 seconds and it is compensated by the better sentiment classification performance achieved by Tw-StAR for these datasets (see Tables 9 and 10). Similarly, for the large-scaled dataset, TEAD, DAN and Tw-StAR consumed quite same training time and were of the least time overhead among the studied architectures, as it can be seen from Figure 3, where the comparison was done for both SOWE and Avg composition functions. This reflects the ability of less complicated models such as Tw-StAR and DAN to maintain a robust performance overhead while handling different sizes of datasets. In the same context, for Tw-StAR model, although learning

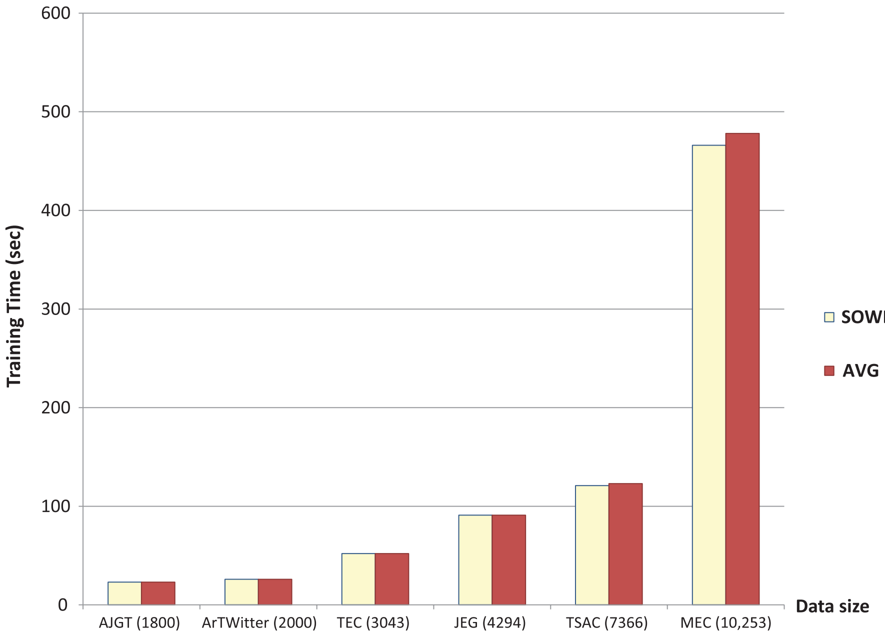


Figure 4. Training time consumed by Tw-StAR for SOWE, Avg due to data size.

features composed by Avg require a little bit more time than those composed by SOWE for large-sized datasets, the time needed to train the model with both feature variants increases according to the size of the input data as it can be seen from Figure 4.

6.7 Tw-StAR model versus baseline systems

The performances obtained by Tw-StAR using SOWE composition function were further compared against the baseline systems that tackled the same datasets (see Table 11).

Due to the lack of embeddings-based Arabic SA systems, we had to compare with the available hand-crafted baseline models: Alomari *et al.* (2017), Sayadi *et al.* (2016), Elouardighi *et al.* (2017) for AJGT, TEC and MEC, while for the datasets: ArTwitter, JEG and TSAC embedding-based baseline models were provided by Al-Azani and El-Alfy (2017), Altowayan and Tao (2016), Medhaffar *et al.* (2017). Compared to the state-of-the-art applied on the investigated datasets (see Table 4), our results showed that Tw-StAR trained with syntax-ignorant n-gram embeddings could improve the classification performance over the baseline systems in most of the datasets.

As we can see in Table 11, with Tw-StAR applied, the accuracy values increased by 17.1%, 8.3% and 1.2% for TEC, TSAC and MEC datasets, respectively. Here, we can notice that the less accuracy increment was reported in MSA/Moroccan MEC dataset; this defines the proposed embeddings as expressive features of pure dialectal content more than they are for MSA ones; as the free word order and varying syntactic structure of dialects can be better handled by SOWE. Moreover, for ArTwitter dataset, a competent performance was achieved by Tw-StAR against complicated neural architectures such as CNNs adopted by Dahou *et al.* (2016) and combined LSTMs used in Al-Azani and El-Alfy (2017), where the accuracy decreased by 0.1% and 2.3% compared to Dahou *et al.* (2016) and Al-Azani and El-Alfy (2017), respectively. Consequently, a shallow neural model such as Tw-StAR trained with embeddings specifically composed to target the Arabic dialectal content can rival much more complicated neural architectures. In addition, for JEG dataset that contains three different dialects, although Tw-StAR could not outperform

Table 11. Tw-StAR versus baseline models

Dataset	Model	F1 (%)	Acc. (%)
AJGT	Hand-crafted (Alomari *et al.* 2017)	**88.3**	**88.7**
	Tw-StAR	82.8	83.3
ArTwitter	Combined LSTM (Al-Azani and El-Alfy 2017)	**87.2**	**87.2**
	CNN (Dahou *et al.* 2016)	–	85.0
	Tw-StAR	84.1	84.9
TEC	Hand-crafted (Sayadi *et al.* 2016)	63.0	71.1
	Tw-StAR	**87.8**	**88.2**
TSAC	MLP/doc2vec (Medhaffar *et al.* 2017)	78.0	78.0
	Tw-StAR	**86.2**	**86.5**
MEC	Hand-crafted (Elouardighi *et al.* 2017)	–	78.0
	Tw-StAR	**72.8**	**79.2**
JEG	Word embeddings (Altowayan and Tao 2016)	**79.6**	**80.2**
	Tw-StAR	74.3	74.8

Boldface highlights the best performance in terms of F1 and Accuracy achieved by the baseline systems orTw-StAR for each dataset.

the baseline system, a satisfying performance was achieved without the need for a huge external knowledge resources such as the training corpus used by Altowayan and Tao (2016) to provide their own word embeddings.

7. Conclusion

In this paper, we introduced syntax-ignorant, n-gram embeddings to be used as discriminating features in the context of sentiment analysis of Arabic dialects. The presented model TwStAR trained with these embeddings could classify the sentiment of several dialects better than most baseline systems and deep complicated architectures. Being composed via SOWE function, our embeddings emphasized the efficiency of using unordered additive composition model in the SA task as the performances produced by n-gram embeddings were better than those learned via word2vec and doc2vec (PV-DM/PV-DBoW) models. Based on the visualisation of the word embeddings learned by Tw-StAR, word2vec and doc2vec (PV-DBoW) models, it was possible to deduce that several words of close sentiments were better mapped using Tw-StAR model. Moreover, another comparison between SOWE and Avg for the SA task of Eastern and Western Arabic dialectal content showed that the sentiment of Eastern dialects was better expressed by SOWE-composed features, while the features formulated by Avg led to a slightly better sentiment classification performance for Western dialect datasets. At the implementation level, it was revealed that, a shallow neural model such as Tw-StAR, trained with unordered embeddings, can address the varying syntax structure and free word order issues of DA yielding a competent performance with much more complicated deep learning architectures. This was emphasised by evaluating Tw-StAR performance against deep SA models having the building units: LSTM and CNN, where Tw-StAR rivalled and sometimes overcome these models. In addition, compared to LSTM/CNN-based models, Tw-StAR consumed less training time in all datasets. In contrast, compared to DAN system (Iyyer *et al.* 2015), our shallow model consumed quite similar training time for the large-scaled dataset TEAD indicating that less complicated architectures with unordered composed embedding features can handle the increased size of training data exhibiting a reduced time overhead. A natural future step would involve examining whether the proposed embeddings can be employed in SA of informal social media posts written in other languages. Furthermore, a multi-dialectal lexicon would be constructed based on the distances among the presented word embedding vectors learned via Tw-StAR and visualised by t-SNE tool.

References

Abdulla N.A., Ahmed N.A., Shehab M.A. and Al-Ayyoub M. (2013). Arabic sentiment analysis: Lexicon-based and corpus-based. In 2013 *IEEE Jordan Conference on Applied Electrical Engineering and Computing Technologies (AEECT)*, pp. 1–6.

Al-Azani S. and El-Alfy E.-S.M. (2017). Hybrid deep learning for sentiment polarity determination of Arabic microblogs. In *International Conference on Neural Information Processing*, pp. 491–500.

Alomari K.M., ElSherif H.M. and Shaalan K. (2017). Arabic tweets sentimental analysis using machine learning. In *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems*, pp. 602–610.

Al-Rfou R., Perozzi B. and Skiena S. (2013) Polyglot: Distributed word representations for multilingual NLP. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pp. 183–192.

Al Sallab A., Hajj H., Badaro G., Baly R., El Hajj W. and Shaban K.B. (2015). Deep learning models for sentiment analysis in Arabic. In *Proceedings of the Second Workshop on Arabic Natural Language Processing*, pp. 9–17.

Al-Sallab A., Baly R., Hajj H., Shaban K.B., El-Hajj W. and Badaro G. (2017). AROMA: A recursive deep learning model for opinion mining in Arabic as a low resource language. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)* **16**(4), 25.

Altowayan A.A. and Tao L. (2016). Word embeddings for Arabic sentiment analysis. In *2016 IEEE International Conference on Big Data (Big Data)*, pp. 3820–3825.

Aly M. and Atiya A. (2013). LABR: A large scale Arabic book reviews dataset. In *ACL (2)*, pp. 494–498.

Ba J. and Caruana R. (2014). Do deep nets really need to be deep? In *Advances in Neural Information Processing Systems*, pp. 2654–2662.

Banea C., Mihalcea R. and Wiebe J. (2010). Multilingual subjectivity: Are more languages better? In *Proceedings of the 23rd International Conference on Computational Linguistics, Association for Computational Linguistics (ACL)*, pp. 28–36.

Baniata L.H. and Park S.-B. (2016). Sentence representation network for Arabic sentiment analysis. In *Proceedings of the Korean Information Science Society*, pp. 470–472.

Brustad K.E. (2000). The Syntax of Spoken Arabic. A Comparative Study of Moroccan, Egyptian, Syrian, and Kuwaiti Dialects. Washington, DC: Georgetown University Press.

Chiang D., Diab M., Habash N., Rambow O. and Shareef S. (2006). Parsing arabic dialects. In *11th Conference of the European Chapter of the Association for Computational Linguistics*.

Collobert R., Weston J., Bottou L., Karlen M., Kavukcuoglu K. and Kuksa P. (2011). Natural language processing (almost) from scratch. *Journal of Machine Learning Research* **12**, 2493–2537.

Courbariaux M., Hubara I., Soudry D., El-Yaniv R. and Bengio Y. (2016). Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1. arXiv preprint arXiv:1602.02830

Dahou A., Xiong S., Zhou J., Haddoud M.H. and Duan P. (2016). Word embeddings and convolutional neural network for Arabic sentiment classification. In *COLING 2016*, pp. 2418–2427.

Duchi J., Hazan E. and Singer Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research* **12**, 2121–2159.

El-Beltagy S.R., Kalamawy M.E. and Soliman A.B. (2017). NileTMRG at SemEval-2017 task 4: Arabic sentiment analysis. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pp. 790–795.

Elouardighi A., Maghfour M., Hammia H. and Aazi F.-z. (2017). A machine Learning approach for sentiment analysis in the standard or dialectal Arabic Facebook comments. In *3rd International Conference of Cloud Computing Technologies and Applications (CloudTech-2017)*, pp. 1–8.

ElSahar H. and El-Beltagy S.R. (2015). Building large Arabic multi-domain resources for sentiment analysis. In *CICLing 2015*, pp. 23–34.

Firth J.R. (1957). A synopsis of linguistic theory 1930–1955. In Studies in linguistic analysis (pp. 1–32). Oxford: Philological Society. [Reprinted in F. R. Palmer (Ed.) (1968). Selected papers of J. R. Firth 1952–1959. London: Longman.]

Glorot X. and Bengio Y. (2010). Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pp. 249–256.

Gormley M.R., Yu M. and Dredze M. (2015). Improved relation extraction with feature-rich compositional embedding models. arXiv preprint arXiv:1505.02419.

Gridach M., Haddad H. and Mulki H. (2017). Empirical evaluation of word representations on Arabic sentiment analysis. In *International Conference on Arabic Language Processing (ICALP)*, pp. 147–158.

Gulcehre C., Moczulski M., Denil M. and Bengio Y. (2016) Noisy activation functions. In *International Conference on Machine Learning*, pp. 3059–3068.

Iyyer M., Manjunatha V., Boyd-Graber J. and Daumé H., III. (2015) Deep unordered composition rivals syntactic methods for text classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, pp. 1681–1691.

Karmani N. (2017). *Tunisian Arabic Customer's Reviews Processing and Analysis for an Internet Supervision System*. PhD Dissertation, National Engineering School of Sfax, Tunisia.

Kim Y. (2014). Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882, pp. 120–150.

Kingma D.P. and Ba J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

Kiritchenko S., Zhu X. and Mohammad S.M. (2014). Sentiment analysis of short informal texts. *Journal of Artificial Intelligence Research* **50**, 723–762.

Krueger D., Ballas N., Jastrzebski S., Arpit D., Kanwal M.S., Maharaj T., Bengio E., Fischer A. and Courville A. (2017). Deep nets don't learn via memorization. In *Workshop track- ICLR 2017.*

Le Q. and Mikolov T. (2014). Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pp. 1188–1196.

Medhaffar S., Bougares F., Esteve Y. and Hadrich-Belguith L. (2017) Sentiment analysis of tunisian dialect: Linguistic resources and experiments. In *Proceedings of The Third Arabic Natural Language Processing Workshop (WANLP)*, pp. 55–61.

Mikolov T., Sutskever I., Chen K., Corrado G.S. and Dean J. (2013) Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pp. 3111–3119.

Mitchell J. and Lapata M. (2010). Recursive deep models for semantic compositionality over a sentiment treebank. In *Composition in Distributional Models of Semantics*, 1388–1429.

Mohammad S.M., Kiritchenko S. and Zhu X. (2013). NRC-Canada: Building the state-of-the-art in sentiment analysis of tweets. arXiv preprint arXiv:1308.6242.

Mourad A. and Darwish K. (2013). Subjectivity and sentiment analysis of modern standard Arabic and Arabic microblogs. In *Proceedings of the 4th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pp. 55–64.

Nabil M., Aly M. and Atiya A. (2015). ASTD: Arabic sentiment tweets dataset. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 2515–2519.

Pennington, J., Socher R. and Manning C. (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543.

Piryani R., Madhavi D. and Singh V.K. (2017). Analytical mapping of opinion mining and sentiment analysis research during 2000–2015. *Information Processing & Management* **53**, 122–150.

Refaee E. and Rieser V. (2014) An Arabic twitter corpus for subjectivity and sentiment analysis. In *LREC*, pp. 2268–2273.

Rosenthal S., Farra N. and Nakov P. (2017). SemEval-2017 task 4: Sentiment analysis in twitter. In *Proceedings of the 11th International Workshop on Semantic Evaluations (SemEval-2017)*, pp. 502–518.

Rushdi Saleh M., Teresa Martin Valdivia M., Alfonso Urena-Lopez L. and Perea Ortega J.M. (2011). OCA: Opinion corpus for Arabic. *Journal of the American Society for Information Science and Technology* **62**(10), 2045–2054.

Sayadi K., Liwicki M., Ingold R. and Bui M. (2016). Tunisian dialect and modern standard Arabic dataset for sentiment analysis : Tunisian election context. In *2nd International Conference on Arabic Computational Linguistics (acling)*, pp. 120–150.

Shen D., Wang G., Wang W., Min M.R., Su Q., Zhang Y., Li C., Henao R. and Carin L. (2018). Baseline needs more love: On simple word-embedding-based models and associated pooling mechanisms. arXiv preprint arXiv:1805.09843.

Socher R., Bengio Y. and Manning C. (2013). Deep learning for NLP. In *Tutorial at Association of Computational Logistics (ACL) and North American Chapter of the Association of Computational Linguistics (NAACL).*

Socher R., Perelygin A., Wu J., Chuang J., Manning C.D., Ng A. and Potts C. (2013). Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1631–1642.

Tang D., Wei F., Yang N., Zhou M., Liu T. and Qin B. (2014). Learning sentiment-specific word embedding for twitter sentiment classification. In *Acossation of Comptational Linguistics ACL* (1), pp. 1555–1565.

Tieleman T. and Hinton G. (2012). Lecture 6.5–RmsProp: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural Networks for Machine Learning*.

van der Maaten L. and Hinton G. (2008). Visualizing data using t-SNE. *Journal of Machine Learning Research* **9**, 2579–2605.

White L., Togneri R., Liu W. and Bennamoun M. (2015). How well sentence embeddings capture meaning. In *Proceedings of the 20th Australasian Document Computing Symposium*, p. 9.

Zahran, M.A., Magooda A., Mahgoub A.Y., Raafat H., Rashwan M. and Atyia A. (2015). Word representations in vector space and their applications for Arabic. In *Computational Linguistics and Intelligent Text Processing (CICLing)*, pp. 430–443.

Zaidan O.F. and Callison-Burch C. (2014). Arabic dialect identification. *Computational Linguistics* **40**(1), 171–202.

Zbib R., Malchiodi E., Devlin J., Stallard D., Matsoukas S., Schwartz R., Makhoul J., Zaidan O.F. and Callison-Burch C. (2012). Machine translation of Arabic dialects. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 49–59.

Zeiler M.D. (2012). ADADELTA: An adaptive learning rate method. arXiv preprint arXiv:1212.5701.

Cite this article: Mulki H, Haddad H, Gridach M and Babaoğlu I (2021). Syntax-ignorant N-gram embeddings for dialectal Arabic sentiment analysis. *Natural Language Engineering* **27**, 315–338. https://doi.org/10.1017/S135132492000008X