Combining Character and Word Embeddings for Affect in Arabic Informal Social Media Microblogs

Abdullah I. Alharbi^{1,2(✉)} and Mark Lee^{1(✉)}

¹ School of Computer Science, University of Birmingham, Birmingham, UK

{aia784,m.g.lee}@cs.bham.ac.uk

² Faculty of Computing and Information Technology, King Abdulaziz University,
Rabigh, Kingdom of Saudi Arabia

aamalharbe@kau.edu.sa

Abstract. Word representation models have been successfully applied in many natural language processing tasks, including sentiment analysis. However, these models do not always work effectively in some social media contexts. When considering the use of Arabic in microblogs like Twitter, it is important to note that a variety of different linguistic domains are involved. This is mainly because social media users employ various dialects in their communications. While training word-level models with such informal text can lead to words being captured that have the same meanings, these models cannot capture all words that can be encountered in the real world due to out-of-vocabulary (OOV) words. The inability to identify words is one of the main limitations of this word-level model. In contrast, character-level embeddings can work effectively with this problem through their ability to learn the vectors of character-ngrams or parts of words. We take advantage of both character- and word-level models to discover more effective methods to represent Arabic affect words in tweets. We evaluate our embeddings by incorporating them into a supervised learning framework for a range of affect tasks. Our models outperform the state-of-the-art Arabic pre-trained word embeddings in these tasks. Moreover, they offer improved state-of-the-art results for the task of Arabic emotion intensity, outperforming the top-performing systems that employ a combination of deep neural networks and several other features.

Keywords: Word-level embeddings · Character-level embeddings · Arabic affect tweets

1 Introduction

People use language not only to express their sentiments and emotions, but also to show how intense these feelings may be. We use the term ‘affect’ to refer to different emotion-related categories, ranging from the sentiment classification

(positive to negative) to finer grained sentiment strength (e.g. high positive to low positive) and emotional intensity (e.g. high anger to low anger). Detecting affect in text is challenging, especially in the context of social media, such as Twitter, owing to difficulties involving the limited number and informal nature of words used, with the latter including slang and symbols. However, this task becomes even more challenging when considering morphology-rich languages, such as Arabic [4]. Social media users employ various dialects and sub-dialects in their communications. In contrast to the use of Modern Standard Arabic (MSA), the form of dialectal Arabic words used varies widely, and there is a general lack of rules and standards. Therefore, the need for effective resources and tools to better understand and treat these various linguistic forms is important when targeting Arabic affect in tweets.

Word embedding is one of the most important methods that have been applied recently to many natural language processing tasks [9, 14, 19, 28]. Word embedding uses dense vectors to represent words projecting into a continuous vector space, thus reducing the number of dimensions [20]. However, these models do not always work effectively in Arabic tweet contexts. While training word-level models with such informal text can lead to the capture of words with the same meanings, it has been shown in testing that these models are unable to recognise other forms of the same words encountered in the real world. These unknown words are called out-of-vocabulary (OOV) words, and this is one of the main limitations of the word-level model. In contrast, character-level embeddings can work effectively in resolving the problem of OOV words through their ability to learn the vectors of character n-grams or parts of words. However, this sensitivity of character-level embedding leads the model to encode all variants of a word’s morphology that are closer to each other in the embedded space than those with semantic similarity. Table 1 shows two examples of dialectal affect words متنزف mtnrfz¹ (uptight) and مروق mrwq (relaxed), where the similarity of the words is mostly based on morphology at the character level and semantics at the word level.

In this paper, we take advantage of both character-level and word-level models to discover more effective means of representing Arabic affect in tweets, which we call affect Character and Word Embeddings (ACWE). We first trained both levels of models on a massive number of tweets, which were collected carefully to ensure that there was significant variation of affect and Arabic dialects in the words. We then employed a novel method that concatenates both levels of models to represent each word morphologically and semantically. We evaluate the effectiveness of our ACWE model by applying it only as a feature under a supervised learning, using the benchmark datasets of SemEval-2018 task 1 (Affect in Tweets) [21]. Our method advances a state-of-the-art approach to the task of discerning Arabic emotional intensity, outperforming the top-performing systems. In addition, our method achieves better results compared to other Arabic pre-trained word embeddings. ACWE has been released to be used in pre-trained

¹ We used Buckwalter transliteration [10].

Table 1. Most similar words of different affect words using character and word level embeddings.

Example of a negative query term: متنرفز mtnrfz (uptight)		Example of a positive query term: مرووق mrwq (relaxed)	
Character-level model	Word-level model	Character-level model	Word-level model
متنرفزه mtnrfz (uptight-feminine)	معصب mESb (angry)	ومرووق wmrwq (and relaxed)	مصصص mSHSH (mindful)
متنرفزين mtnrfzyn (uptight-plural)	متوتر mtwtr (tense)	مرووقه mrwqh (relaxed-feminine)	ومرووق wmrwq (and relaxed)
تتتتفز ntnrfz (uptight-present verb)	متضايق mtDAYq (annoyed)	ومرووقه wmrwqh (and relaxed-feminine)	فايق fAyq (awake)
بيتتتفز bytnrfz (uptight-future verb)	منفس mnfs (furious)	رايق rAyq (relaxed)	مفلل mflf (restful)
تتتتفز ttnrfz (uptight- feminine verb)	مضغوط mDgwT (enraged)	رايقه rAyqh (relaxed-feminine)	مستانس mstAns (happy)

word embeddings for applications and research relying on Arabic sentiment and emotion analysis².

2 Related Works

Most work on Arabic word embeddings has relied on word-level models [3,6,27], and to a lesser degree, character-level models have been employed [5]. To our knowledge, there is no existing work that aims to combine both levels to generate word representations specifically for sentiment or emotion analysis. One of the largest open-source word embeddings is AraVec [27], which consists of six different word embedding models for the Arabic language. Here, the researchers derived the training data from three separate sources: Wikipedia, Twitter and Common Crawl webpages crawl data; they employed two word-level models to learn word representations for general NLP tasks.

More recently, [3] proposed the largest word-level embeddings by using 250M Arabic tweets. Although the models are trained on many words, they cannot realise other forms of the same words that can be seen in the real world due to the limitations of such word-level models. Furthermore, it has been observed that the effectiveness of word embeddings is more likely to be task-dependent [25], and it is highly influenced by the richness of related words to the target task [11].

Much research has been undertaken on Arabic sentiment analysis, but research has focussed on other affect aspects such as emotion analysis or intensity remains limited [21]. Most of the existing work on affect in Arabic is based on the SemEval-2018 competition, Affect in Tweet. Most of the top-performing

² <https://github.com/aialharbi/ACWE>.

systems proposed for this shared task employed deep learning approaches, such as CNN, LSTM and Bi-LSTM [1, 2, 16]. The majority of these systems employed AraVec as a feature besides other input features, Arabic sentiment and emotion lexicons [7, 22, 24].

3 Methodology

In this work, we aim to generate an effective distributed word representation model for Arabic affect in tweets. The data collection method and the different models of word embeddings used are detailed in the subsections below.

3.1 Data Collection

One of the main factors in improving the quality of word embeddings is associated with the training dataset size and its richness. We collected a large number of tweets (10M) containing various affect-associated words of different Arabic dialects. To ensure the tweets contained a variety of affect-associated words, we first used English NRC lexicons [23] to select a number of words (63 words)³ from different emotional expressions and intensity levels. Then, the selected words were translated into Arabic using the online translation application Reverso Context⁴. We also used Reverso to find synonyms of these translated words to extend our list of terms from 63 to 228 words. At this stage, our list of terms contained MSA affect words, which was an expected result of this means of English-Arabic translation.

To ensure the tweets reflected a variety of dialects, we used our MSA terms list to find synonyms in Arabic dialects from two online dictionaries (Atlas Allhajaat⁵ and Mo3jam⁶). This expanded our list of terms by 217 different dialectal affect words. In addition, it should be noted that emojis could be employed, given that, according to [17], they function as a universal language. Therefore, we selected the 30 most frequently used emojis from different sentiment scores obtained from [17] and added them to our list of terms. Finally, we assumed that tweets from specific Arabic-speaking countries would more likely be associated with the dialects of these locations. Therefore, we collected tweets that included all the identified terms (about 500 terms) using the Twitter Search API by specifying the geolocations of different Arab countries.

3.2 Data Preprocessing

The data that we extracted from Twitter typically contained a range of content that could be considered useless for our task, such as hashtags, website links and

³ These are words that directly convey meanings of sentiment or emotion, such as *anger* or *rage*. They are not words that indirectly convey sentiment, such as *dead* or *tears*.

⁴ <http://context.reverso.net>.

⁵ <http://atlasallhajaat.com>.

⁶ <http://en.mo3jam.com>.

mentions. It was important that such noisy content be removed before training our learning models to reduce both the noise and vector space size [18, 26]. We followed the procedure laid out by several research works [3, 15], which involved the following steps:

- **Normalisation of letters:** Letters that appeared in different forms in the original tweets were rendered into a single form. For example, the ‘hamza’ on characters {أ, إ} was replaced with {ا}, while the ‘t marbouta’ {ة} was replaced with {o}.
- **Hashtags:** Hashtags are used to draw attention to words or phrases that are trending. For example, #anger, #happy. While it is common to remove the hash symbols and words, we only removed the hash symbols and kept the words. Users sometimes express their emotions using these hashtags, so it was considered useful to retain them.
- **Cleaning:** All unknown symbols and other characters were eliminated. For example, other language letters, diacritics, punctuation and URLs were removed. However, emojis were not removed, and like the words, each emoji was represented by a vector.

3.3 Embedding Models

After retrieving and pre-processing a massive number of tweets that are rich in Arabic affect-related words, we used this to generate a language model. Word embeddings are learned representations of text, with words of similar meanings represented in similar ways. An essential element of this methodology is the concept of employing dense distributed representations for every word. Here, each word is encoded to a real-valued vector with a few hundred dimensions. Given a large corpus, there are different models and levels available for learning word embeddings. We first employed the Word2Vec model [20] for word-level embeddings and FastText model [8] for character-level embeddings. Hence, we leveraged these two pre-trained embeddings as an input feature after combining them with a novel concatenation approach. These main steps are detailed in the following subsections.

Word-Level Embeddings (WE). To learn individual words with their embeddings from our collected data, we used the Word2Vec algorithm [20]. Word2Vec is based on a pair of learning techniques: the Continuous Bag-of-Words (CBOW) and Skip-Gram (SG) models. The CBOW model effectively averages the vectors of all the words in a given context. The model is trained by predicting the current word based on the projected average of the surrounding context. The continuous SG model is similar, but instead of predicting the current word based on context, it predicts the surrounding words based on the current one. Words within a certain distance before and after the current word are predicted with the network optimised for these predictions. We used both

models (CBOW and SG) to generate affect word embeddings by training the models on a massive number of tweets that we retrieved. We used the Gensim Library⁷ to implement the Word2Vec models. We assumed that each tweet was a sentence, so the input of the word-level model was a list of pre-processed tweets that were tokenised into words on whitespace. The main parameters that we used were (200) for size, (5) for the window context and (3) to ignore words with a total frequency lower than three.

Character-Level Embeddings (CE). To learn morphological features found in each word, we used a character n-grams model (FastText) [8]. FastText differs from Word2Vec in its ability to learn the vectors of character n-grams or parts of words. This feature enables the model to capture words that have similar meanings but different morphological word formations. We used the Gensim Library⁸ to implement the FastText model. We assumed that each tweet was a sentence, so the input of the character-level model was a list of characters for each tweet. We used the same main parameters that we employed for Word2Vec. In addition, to control the lengths of character n-grams, we used 3 and 6 for parameters (min_n) and (max_n), respectively.

Affect Character and Word Embeddings (ACWE). As explained in the introduction, while *CE* seems to encode all variants of a word's morphology closer in the embedded space, *WE* seems to give more importance to semantic similarity. To take advantage of both models, we propose ACWE, a novel approach that aims to concatenate these two pre-trained embeddings; hence, it can be used as an input feature for a range of sentiment and emotion tasks.

Given a tweet t_i that has a sequence of words $\{w_1, w_2, \dots, w_n\}$, our goal is to morphologically and semantically represent each word in each tweet $w_i \in t_i$ as an n -dimensional continuous vector. To achieve this goal, we assumed that each word $w_i \in t_i$ is represented semantically by $WE(w_i)$ and morphologically by $CE(w_i)$, where $WE(w_i)$ is the word embedding of w_i , while $CE(w_i)$ is the character embedding of w_i . The $ACWE(w_i)$ method is used to concatenate both embeddings, and it can be obtained in three different cases. The first case is a direct concatenation of $CE(w_i)$ and $WE(w_i)$, and it arises if w_i can be found in both embeddings. However, if w_i cannot be found in WE ⁹, we assume this is due to variants in the given word's morphology. Consequently, instead of using a vector of zeros for unseen w_i , it will be replaced by another word's morphology that can be realised by WE . Alternative words can be obtained using $most\_similar(w_i)$, which aims to find the most similar word based on the cosine similarity of the w_i vector and the vectors for each word in CE . Finally, if w_i cannot be determined using CE and WE , it will be represented by a vector of zeros.

⁷ <http://radimrehurek.com/gensim/models/word2vec.html>.

⁸ <http://radimrehurek.com/gensim/models/fasttext.html>.

⁹ As explained in the introduction, WE cannot process OOV words.

$$ACWE(w_i) = \begin{cases} CE(w_i) \oplus WE(w_i), & \text{if } w_i \in (CE|V|, WE|V|) \\ CE(w_i) \oplus WE(\text{most\_similar}(w_i)), & \text{if } w_i \notin (WE|V|) \\ \text{zeros of } (CE + WE) \text{ dimensions} & \text{otherwise} \end{cases} \quad (1)$$

4 Experiments

To validate the effectiveness of our embeddings, we incorporated them into a supervised learning framework for a range of affect-sensitive tasks. We compared our models against available state-of-the-art pre-trained Arabic word embeddings. We also compared our method with top systems targeting these different tasks.

4.1 Datasets

We evaluated our model using different affect tasks in the SemEval 2018 task 1 (Affect in Tweets) datasets [21]. We selected these datasets because of the variety of affect tasks and Arabic dialects. These tasks can be categorised as follows:

- **Emotion Intensity Task:** When given an emotion and a tweet, compute the emotional intensity (EI) that most accurately represents the emotion experienced by the publisher using a real-value score as follows: 1) The EI-regression (EI-reg) task scores range from 0 to 1, from least to most emotion; and 2) the EI-ordinal classification (EI-oc) Task scores range from 0 to 3, where 0 refers to an unrelated emotion.
- **Sentiment or Valence Intensity Task:** When given a tweet, predict the valence (V) that most effectively represents the tweeter’s mental state using a real-value score as follows: 1) the V-reg task scores range from 0 to 1, from most negative to most positive; and 2) The V-oc task scores range from -3 (very negative) to $+3$ (very positive).

Table 2. Number of tweets in the SemEval 2018 Task 1 (Affect in Tweets) datasets.

Task	Emotion	Train	Dev	Test	Total
EI-reg/EI-oc	Anger	877	150	373	1,400
	Fear	882	146	372	1,400
	Joy	728	224	448	1,400
	Sadness	889	141	370	1,400
V-reg/V-oc	Valence intensity	932	730	1,800	1,800

4.2 Pre-trained Word Embeddings

To evaluate the effectiveness of our models, we compared them with Arabic pre-trained word embeddings in the following models: Ara2Vec [27], Mazajak [3] and Altwyan [6]. To the best of our knowledge, these embeddings are the most commonly available resources released to the research community as free to use. Table 3 presents a summary of important information about each of these models with their sizes and pre-trained corpora (Table 2).

Table 3. Different pre-trained Arabic word embeddings used for experimental evaluation.

Model	No. of words	Corpus	Size
Ara2Vec	4,347,845	General - Twitter	77M Tweets
Mazajak	1,476,715	Sentiment - Twitter	250M Tweets
Altwaian	159,175	Sentiment - Twitter	190M words
Our WE	626,212	Affect - Twitter	100B tokens
Our CE	441,025	Affect - Twitter	3B tweets

4.3 Model Training

We pre-processed the datasets using the pre-processing techniques described in Sect. 3.2. To predict a real-value score for each task, we employed the XGBoost learning model [12] and used one of the aforementioned pre-trained word embeddings as an input feature. The XGBoost learning model is frequently employed for different problems because it performs extremely well on a wide range of significant challenges. The tool is both extremely versatile and flexible, and it can address different classification and regression problems [12]. This is an algorithm of decision trees in which new trees correct errors of those trees which are already part of the model. Trees are added to the model until no further changes can be made. We input tweet vector representations obtained from an average of real-value word vectors for every word with matching vector representations derived from pre-trained embeddings.

4.4 Results

The results of our experiments were evaluated using Pearson’s correlation coefficient, which calculates a bivariate linear correlation between two given variables. In our experiments, this comprised the correlation between the score predicted by our systems and the score given by the test data. We used this evaluation metric because it is the official metric for all the relevant tasks. Our results and findings are discussed in the following subsections.

Comparison with State-of-the-Art Pre-trained Arabic Word Embeddings: We compared five pre-trained word embeddings (see Table 3), including three open-source models and both of our generated models. In addition, we compared these models with the ACWE method. The information presented in Table 4 shows the effectiveness of each model in the supervised framework of performing affect-sensitive tasks. The Pearson correlation coefficient for our *CE* significantly outperformed the other models. We consider that the main reason for this was associated with OOV problems. Although these models were trained using a massive corpus, we found that word-level embeddings could not realise more than 700 words from each dataset. Moreover, the ACWE method improved the results by 1.3% to 5% across all datasets. This indicates the effectiveness of the proposed method and the importance of leveraging character-level and word-level embeddings in Arabic words in the context of social networks and microblogs.

Table 4. Pearson correlation coefficient results for our models and State-of-the-art pre-trained Arabic Word Embeddings

Model	EI-reg					EI-oc					V-reg	V-oc
	Anger	Fear	Joy	Sad	Avg.	Anger	Fear	Joy	Sad	Avg.		
Ara2Vec	55.6	53.6	68.8	64.1	60.5	47.2	52.6	60.4	59.4	54.9	77.3	72.3
Mazajak	55.5	57.6	68.3	62.3	60.9	45.0	51.2	64.6	53.0	53.4	72.0	68.0
Altwyman	29.7	33.3	44.9	49.7	41.5	27.2	31.2	42.5	48.9	37.5	51.5	53.5
Our generated Arabic word Embeddings												
WE	53.9	52.9	65.3	60.7	58.7	47.9	51.1	62.8	55.6	54.4	75.6	70.2
CE	<u>60.1</u>	<u>59.5</u>	<u>70.4</u>	<u>65.8</u>	<u>64.3</u>	<u>51.1</u>	<u>53.1</u>	<u>64.7</u>	<u>60.6</u>	<u>57.6</u>	<u>78.3</u>	<u>73.1</u>
ACWE	63.8	62.2	75.8	68.6	67.6	54.3	57.2	67.5	60.9	60.0	81.8	76.7

Comparison Against Top Systems Analysing Affect in Tweets: Most of the top-performing systems proposed for this shared task employed deep learning approaches, such as CNN, LSTM and Bi-LSTM. The majority of these systems used AraVec as a feature alongside other input features, such as the sentiment and emotional lexicons found in the Arabic language. We used our embeddings as the input feature for XGBoost, a machine learning classifier/regressor. As shown in Table 5, we achieved competitive results: We outperformed the top system in the EI-oc task by 1.3% and ranked second in the remaining tasks. Our goal was not to fully address affect tasks but rather to demonstrate that, by using a well-generated word embedding model, we could obtain competitive results. We will investigate other features and employ deep learning methods to improve the results in future works.

Table 5. Pearson correlation coefficient results for our ACWE and top systems across all tasks.

Task	1st best	2nd best	Our ACWE
Ei-reg	68.5	66.7	<u>67.6</u>
EI-oc	<u>58.7</u>	57.4	60.0
V-reg	82.8	81.6	<u>81.8</u>
V-oc	80.9	75.2	<u>76.7</u>

5 Conclusion

In this paper, we generated word and character embeddings to analyse affect in Arabic social media networks and microblogs. We also proposed a novel method that combines different levels of word embeddings to represent the morphology and semantics for each word in a given task. We evaluated the models by incorporating them into a supervised learning framework for a range of affect-sensitive tasks. Our models outperformed state-of-the-art pre-trained Arabic word embeddings on these tasks.

In future works, we will apply more sophisticated algorithms to improve the quality of our embeddings. Especially, we would like to employ contextualised word embeddings, such as BERT [13]. We would also like to investigate more deep learning algorithms to fully target affect tasks.

References

1. Abdou, M., Kulmizev, A., Ginés i Ametllé, J.: AffecThor at SemEval-2018 task 1: a cross-linguistic approach to sentiment intensity quantification in tweets. In: Proceedings of The 12th International Workshop on Semantic Evaluation, pp. 210–217. Association for Computational Linguistics, New Orleans (2018)
2. 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, pp. 350–357. Association for Computational Linguistics, New Orleans (2018)
3. Abu Farha, I., Magdy, W.: Mazajak: An online Arabic sentiment analyser. In: Proceedings of the Fourth Arabic Natural Language Processing Workshop, pp. 192–198. Association for Computational Linguistics, Florence (2019)
4. Al-Ayyoub, M., Khamaiseh, A.A., Jararweh, Y., Al-Kabi, M.N.: A comprehensive survey of Arabic sentiment analysis. *Inf. Process. Manage.* **56**(2), 320–342 (2019)
5. Altowayan, A.A., Elnagar, A.: Improving Arabic sentiment analysis with sentiment-specific embeddings. In: 2017 IEEE International Conference on Big Data (Big Data), pp. 4314–4320. IEEE, Boston (2017)
6. Altowayan, A.A., Tao, L.: Word embeddings for Arabic sentiment analysis. In: 2016 IEEE International Conference on Big Data (Big Data), pp. 3820–3825. IEEE, Washington (2016)

7. Badaro, G., Baly, R., Hajj, H., Habash, N., El-Hajj, W.: A large scale Arabic sentiment lexicon for Arabic opinion mining. In: *Proceedings of the EMNLP 2014 Workshop on Arabic Natural Language Processing (ANLP)*, pp. 165–173. Association for Computational Linguistics, Doha, October 2014
8. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching word vectors with subword information. *Trans. Assoc. Comput. Linguist.* **5**, 135–146 (2017)
9. Bordes, A., Chopra, S., Weston, J.: Question answering with subgraph embeddings. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 615–620. Association for Computational Linguistics, Doha (2014)
10. Buckwalter, T.: Buckwalter Arabic morphological analyzer version 2.0. LDC catalog number LDC2004l02, ISBN 1-58563-324-0 (2004)
11. Çano, E., Morisio, M.: Quality of word embeddings on sentiment analysis tasks. In: Frasincar, F., Ittoo, A., Nguyen, L.M., Métails, E. (eds.) *NLDB 2017. LNCS*, vol. 10260, pp. 332–338. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-59569-6_42
12. Chen, T., Guestrin, C.: XGBoost: a scalable tree boosting system. In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2016*, pp. 785–794. Association for Computing Machinery, New York (2016)
13. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4171–4186. Association for Computational Linguistics, Minneapolis (2019)
14. Devlin, J., Zbib, R., Huang, Z., Lamar, T., Schwartz, R., Makhoul, J.: Fast and robust neural network joint models for statistical machine translation. In: *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Baltimore, Maryland, pp. 1370–1380 (2014)
15. Duwairi, R., El-Orfali, M.: A study of the effects of preprocessing strategies on sentiment analysis for Arabic text. *J. Inf. Sci.* **40**(4), 501–513 (2014)
16. Jabreel, M., Moreno, A.: EiTAKA at SemEval-2018 task 1: an ensemble of n-channels ConvNet and XGboost regressors for emotion analysis of tweets. In: *Proceedings of The 12th International Workshop on Semantic Evaluation*, pp. 193–199. Association for Computational Linguistics, New Orleans (2018)
17. Kralj Novak, P., Smailović, J., Šluban, B., Mozetič, I.: Sentiment of emojis. *PLoS ONE* **10**(12), 1–22 (2015)
18. Li, Q., Shah, S., Liu, X., Nourbakhsh, A.: Data sets: word embeddings learned from tweets and general data. In: *Proceedings of the Eleventh International Conference on Web and Social Media (ICWSM-17)*, pp. 428–436. AAAI Press, Montréal (2017)
19. Lin, C.C., Ammar, W., Dyer, C., Levin, L.: Unsupervised POS induction with word embeddings. In: *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1311–1316. Association for Computational Linguistics, Denver (2015)
20. Mikolov, T., Sutskever, I., Chen, K., Corrado, G., Dean, J.: Distributed representations of words and phrases and their compositionality. In: *Proceedings of the 26th International Conference on Neural Information Processing Systems, NIPS 2013*, vol. 2, pp. 3111–3119. Curran Associates Inc., Red Hook (2013)

21. 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*, pp. 1–17. Association for Computational Linguistics, New Orleans (2018)
22. Mohammad, S., Salameh, M., Kiritchenko, S.: Sentiment lexicons for Arabic social media. In: *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, pp. 33–37. Portorož (2016)
23. Mohammad, S.M.: Word affect intensities. In: *Proceedings of the 11th Edition of the Language Resources and Evaluation Conference (LREC-2018)*, Miyazaki, Japan, pp. 174–183 (2018)
24. Mohammad, S.M., Turney, P.D.: Crowdsourcing a word-emotion association lexicon. *Comput. Intell.* **29**(3), 436–465 (2013)
25. Qu, L., Ferraro, G., Zhou, L., Hou, W., Schneider, N., Baldwin, T.: Big data small data. domain out-of domain, known word unknown word: the impact of word representations on sequence labelling tasks. In: *Proceedings of the Nineteenth Conference on Computational Natural Language Learning*, pp. 83–93. Association for Computational Linguistics, Beijing, July 2015
26. Singh, T., Kumari, M.: Role of text pre-processing in Twitter sentiment analysis. *Proc. Comput. Sci.* **89**, 549–554 (2016)
27. Soliman, A.B., Eissa, K., El-Beltagy, S.R.: AraVec: a set of Arabic word embedding models for use in Arabic NLP. *Proc. Comput. Sci.* **117**, 256–265 (2017)
28. Zhang, J., Liu, S., Li, M., Zhou, M., Zong, C.: Bilingually-constrained phrase embeddings for machine translation. In: *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 111–121. Association for Computational Linguistics, Baltimore (2014)