AROMA: A Recursive Deep Learning Model for Opinion Mining in Arabic as a Low Resource Language

AHMAD AL-SALLAB, Cairo University
RAMY BALY, American University of Beirut
HAZEM HAJJ, American University of Beirut
KHALED BASHIR SHABAN, Qatar University
WASSIM EL-HAJJ, American University of Beirut
GILBERT BADARO, American University of Beirut

While research on English opinion mining has already achieved significant progress and success, work on Arabic opinion mining is still lagging. This is mainly due to the relative recency of research efforts in developing natural language processing (NLP) methods for Arabic, handling its morphological complexity, and the lack of large-scale opinion resources for Arabic. To close this gap, we examine the class of models used for English and that do not require extensive use of NLP or opinion resources. In particular, we consider the Recursive Auto Encoder (RAE). However, RAE models are not as successful in Arabic as they are in English, due to their limitations in handling the morphological complexity of Arabic, providing a more complete and comprehensive input features for the auto encoder, and performing semantic composition following the natural way constituents are combined to express the overall meaning. In this paper, we propose **A R**ecursive Deep Learning Model for **O**pinion **M**ining in **A**rabic (AROMA) that addresses these limitations. AROMA was evaluated on three Arabic corpora representing different genres and writing styles. Results show that AROMA achieved significant performance improvements compared to the baseline RAE. It also outperformed several well-known approaches in the literature.

CCS Concepts:•**Information systems** → **Sentiment analysis;**

ACM Reference Format:
Ahmad Al-Sallab, Ramy Baly, Hazem Hajj, Khaled Bashir Shaban, Wassim El-Hajj, and Gilbert Badaro, 2016. AROMA: A Recursive Deep Learning Model for Opinion Mining in Arabic as a Low Resource Language. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.* V, N, Article A (January YYYY), 20 pages.
DOI: http://dx.doi.org/10.1145/0000000.0000000

1. INTRODUCTION

Opinion mining, or sentiment analysis, refers to the task of automatically extracting measures of people's opinions from digital data. With the new wave in Web 2.0 technology, users became the data generators creating the so-called "Big Data" [Ravi and Ravi 2015; Agerri et al. 2015]. The abundance and diversity of users' opinions on the Web raised the need for automated systems to estimate the public opinion and track trends such as interests and stock market shifts. While many customized solutions already exist to address these needs, some languages have lagged behind in reaping

Authors' addresses: A. Al-Sallab, Computer Engineering Department, Cairo University, Cairo, Egypt; email: ahmad.elsallab@gmail.com; R. Baly, H. Hajj, Electrical and Computer Engineering Department, American University of Beirut, Beirut, Lebanon; emails: rgb15@mail.aub.edu, hh63@aub.edu.lb; K. B. Shaban, Computer Science and Engineering Department, Qatar University, Doha, Qatar; email: khaled.shaban@qu.edu.qa; W. El-Hajj, Computer Science Department, American University of Beirut, Beirut, Lebanon; email: we07@aub.edu.lb.
Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
© YYYY ACM. 2375-4699/YYYY/01-ARTA $15.00
DOI: http://dx.doi.org/10.1145/0000000.0000000

the benefits of such solutions, mainly due to the limited advances of natural language processing (NLP) research in these languages.

Research on opinion mining for English has already achieved considerable success with the abundance of NLP solutions and opinion lexical resources. However, significant efforts need to be made to achieve similar performances in other languages, including Arabic. In this paper, we focus on developing opinion mining models for Arabic, which has become the fifth most-spoken language in the world [UNESCO 2014]. Arabic language is known for its morphological complexity and the existence of a large number of dialectal variants, which adds to the complexity of the task.

Many methods have been proposed to perform automated opinion mining, most of which are based on training machine learning models using different choices of features, exploring multiple levels of text granularity (words, phrases, sentences and full documents), and producing output classes typically as binary (positive/negative) or integer values representing opinions ranging from very negative to very positive [Liu and Zhang 2012].

In general, training accurate opinion models requires the availability of opinion lexical resources including opinion corpora and lexicons, which are not readily available for all languages. For instance, creating sentence-level annotated opinion corpora requires time and effort from well-trained human annotators to correctly identify opinions while dealing with semantic issues such as indirect opinion inference, sarcasm and dynamically arising expressions. The complexity increases since annotation is often performed through crowd-sourcing, where guidelines must be developed and pilot experiments must be conducted to ensure quality annotation of large amounts of data.

State-of-the-art opinion models are based on deep learning (DL), which is known for its ability to learn embedded and abstract representations from raw data with minimal human intervention [Socher et al. 2011b; Socher et al. 2013; Tang et al. 2015; Tai et al. 2015]. DL models have produced state-of-the-art performances benefiting from opinion corpora with fine-grained annotations at different levels of constituency, such as the Stanford sentiment treebank [Socher et al. 2013]. Such annotation efforts are more expensive and difficult to create as they require more sophisticated guidelines and a larger number of well-trained human annotators. As a result, creating large-scale reliable opinion-annotated lexical resources is a challenging task that has gained significant research attention. To overcome this limitation, while making advances in opinion mining for languages that lack such lexical resources, we propose to explore models that perform well with smaller-scale resources.

In this paper, we consider recursive deep learning models that aim to model semantic interactions between words and constituents in sentences to infer sentiment from the derived context. These models have shown success at both aspects of modeling semantic composition and sentiment inference for English. In particular, the Recursive Auto Encoders (RAE) [Socher et al. 2011b] and the Recursive Neural Tensor Networks (RNTN) [Socher et al. 2013] are considered among state-of-the-art models for English opinion mining. However, these methods cannot be directly applied to Arabic. RNTN needs a sentiment treebank with sentiment annotations at every level of constituency, including words and their combinations in a syntactic parse tree, which is an expensive resource that is currently unavailable in Arabic. RAE addresses this problem by separately extracting semantic sentence representations in an unsupervised way, and training an opinion model using corpora annotated at the sentence-level only. RAE has been recently evaluated for Arabic opinion mining in [Al Sallab et al. 2015] and has outperformed several well-known machine learning models aimed at Arabic opinion mining. However, RAE models that are directly trained on Arabic still suffer limitations in their ability to model semantic interactions among morphemes (the smallest meaningful grammatical units that cannot be further divided) and to generalize semantics. Furthermore, RAE suffers from additional language-independent challenges including the incompleteness of the word representations at capturing sentiment-related aspects, and the need to improve the order of recursion for a better capture of compositionality.

To address the limitations in the RAE model [Socher et al. 2011b], which we refer to as baseline RAE in rest of the paper, we propose AROMA, where morphological tokenization is applied to the input text, in order to reduce the language sparsity and allow modeling composition at the morpheme-level. Then, both sentiment and semantic embedding models are used to derive better word vectors. Finally, the order of the model's recursion is determined by the structure of automatically generated syntactic parse trees for better modeling of composition.

AROMA is evaluated with three Arabic corpora that represent different genres, and that use different Arabic writing styles: 1) newswire data extracted from the Arabic Treebank (ATB) [Maamouri et al. 2004] and written in modern standard Arabic (MSA), 2) online comments extracted from the Qatar Arabic Language Bank (QALB) corpus [Mohit et al. 2014] and written in MSA with some misspellings and dialectal Arabic (DA), and 3) microblogs (tweets) [Refaee and Rieser 2014] written in MSA and non-standard and dialectal Arabic coming in short and grammatically erroneous text. We highlight the performance improvements achieved by the different components of the proposed method, separately and combined. Results indicate that, on all datasets, the proposed AROMA introduces significant performance improvements compared to the baseline RAE model.

The rest of the paper is organized as follows. Section 2 provides an overview of the related work. Section 3 briefly describes the baseline RAE model as proposed by [Socher et al. 2011b]. Section 4 describes the AROMA framework along with the approaches to achieve the proposed contributions. Section 5 presents the experiments and evaluation results, and Section 6 concludes the paper.

2. RELATED WORK

This section presents an overview of popular approaches proposed to perform opinion mining in Arabic, recent opinion models based on deep learning (DL) techniques, and opinion lexical resources developed for Arabic language.

Most opinion models in Arabic are based on training machine learning classifiers using different choices of feature engineering. For instance, word *n*-grams were proposed with different preprocessing and representation settings, with *bi*-grams and *tri*-grams achieving best performances when used to train Support Vector Machines (SVM) [Rushdi-Saleh et al. 2011; Aly and Atiya 2013; Al-Kabi et al. 2013; Shoukry and Rafea 2012]. Naïve Bayes also achieved competitive performances as reported by [Mountassir et al. 2012; Elawady et al. 2014]. Ensemble classifiers achieved further improvements [Omar et al. 2013]. Other than word *n*grams features, syntactic and stylistic features also achieved high performances when applied to web forum contents [Abbasi et al. 2008].

Deep learning (DL) models have recently gained popularity and were successfully used to learn embedded semantic representations of text that can be used for accurate opinion mining in English. Broadly, we can divide the most well-known representation DL models into two groups: 1) Convolutional Restricted Boltzmann Machines (CRBM), Deep Belief Networks (DBN) and Deep Auto Encoders (DAE) [Hinton et al. 2006; Bengio 2012] and 2) Recursive Auto Encoder (RAE) [Socher et al. 2011b], Recursive Neural Networks (RNN) [Socher et al. 2011a], Recursive Neural Tensor Networks (RNTN) [Socher et al. 2013] and Gated Recurrent Neural Networks (GRNN) [Tang et al. 2015]. The main difference between these groups is in the way of feeding the input sequence of words. In the first group, this feeding happens one shot, while in the second group of algorithms the feeding occurs in a recursive manner. The recursive DL models proved to be more efficient in terms of modeling the syntactic structure of the input, especially in problems of sequential nature like NLP. Moreover, one-hot models such as DBN or DAE require the input to be represented in a “bag-of-words” (BoW) fashion, which suffers two major issues: 1) the syntactic and sequential dependency information are lost and 2) the resulting input vector is sparse affecting the network’s ability to infer patterns from the input data, and hence degrades the quality of the learned feature representation. Another way of representing words vectors is the Bag-of-Concepts used in SenticNet [Cambria and Hussain 2015], which is a knowledge-based approach.

To the best of our knowledge, the only work that investigated DL models for opinion mining in Arabic was [Al Sallab et al. 2015], which evaluated different DL models including DNN, DBN, DAE and RAE. Results indicate that RAE was the best-performing model, emphasizing the advantage of recursive over one-shot models at learning accurate semantic representations. Furthermore, although RAE was applied without special consideration for the linguistic properties of Arabic language, it outperformed most feature engineering-based models. The authors of that paper introduced a separate word embedding block trained on unlabeled instances from the QALB corpus. They also reported the effect of sparsity on the learned representations and the overall performance when using the DBN and DAE models.

In terms of opinion lexical resources, several resources have been developed for Arabic language, mainly sentiment lexicons and opinion annotated corpora. Sentiment lexicons are used to train supervised [Abbasi et al. 2011] and unsupervised [Badaro et al. 2015; Nakov et al. 2016] opinion models, and helped improving classification performance. Examples of sentiment lexicons in Arabic include ArSenL [Badaro et al. 2014], SIFAAT [Abdul-Mageed et al. 2011], SANA [Abdul-Mageed and Diab 2014] and ArSeLEX [Ibrahim et al. 2015]. Most lexicons are lemma-centric and use lemmas to represent words that are morphologically related. SIFAAT contains 3,982 adjectives from the ATB Part 1 V3.0, that are manually labeled as positive, negative or neutral. ArSeLEX contains 5,244 adjectives that are automatically expanded from a gold set of 400 adjectives. On the other hand, ArSenL is a large-scale sentiment lexicon that contains 28,760 lemmas, each associated with three scores representing the positivity, negativity and neutrality of the given lemma. This list is compiled by mapping between different resources including English WordNet [Miller et al. 1990], Arabic WordNet [Black et al. 2006], English SentiWordNet [Esuli and Sebastiani 2006] and the Standard Arabic Morphological Analyzer (SAMA) [Maamouri et al. 2010b].

At last, a variety of corpora has been used to evaluate sentiment analysis models in Arabic. The Opinion Corpus for Arabic (OCA) [Rushdi-Saleh et al. 2011] is a relatively small corpus that contains 500 movie reviews that are split between positive and negative reviews. The Large-scale Arabic Book Reviews (LABR) [Mountassir et al. 2012] is among the largest published Arabic opinion corpora and contains over 63,000 books reviews. A sentence-level opinion corpus was compiled by sampling 2,855 sentences that make up 400 documents from Part 1 V3.0 of the ATB. Many Twitter datasets were annotated to evaluated opinion models on social media content. For instance, 8,868 tweets were extracted and annotated by [Refaee and Rieser 2014]. Another sentence-level opinion corpus was developed by [Farra et al. 2015] by performing topic modeling to extract a balanced set of online comments from the QALB dataset.

Overall, although several opinion lexical resources have been published for Arabic language, the number and diversity of these resources are still relatively small compared to those developed for English.

3. RAE FOR OPINION MINING

In this section, we describe the baseline RAE opinion model proposed for English [Socher et al. 2011b] and evaluated for Arabic [Al Sallab et al. 2015]. Figure 1 illustrates the framework of this model that is composed of two main stages. The first stage is unsupervised and derives a vector representation for each sentence by applying an auto-encoder (AE) to its word sequence in a recursive manner, combining two constituents at a time. The second stage is supervised and uses the derived sentence representations for opinion classification using logistic regression (softmax).

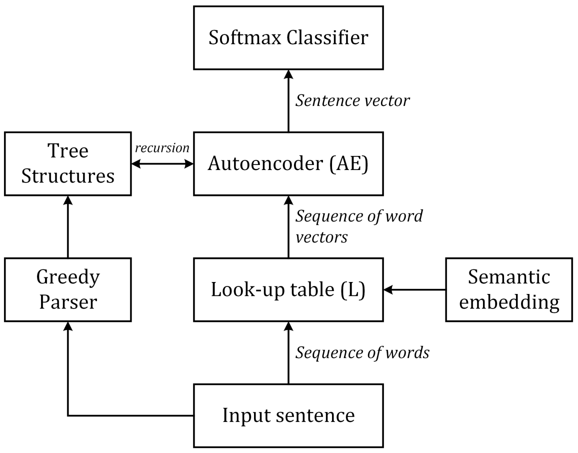

Fig. 1. The framework of the RAE model [Socher et al. 2011b], referred to as baseline RAE in this paper.

The AE block is trained one sentence at a time, and each word in the sentence is represented by a d-dimensional vector derived using a neural language model (NLM) that generates word vectors, where each element corresponds to a latent feature of the word [Collobert and Weston 2008; Bengio 2012; Mikolov and Dean 2013]. The word vectors are dense and low-dimensional, compared to the commonly used BoW and one-hot features, thus eliminating sparsity issues. The transformation from raw words to word vectors is done using a look-up table $L \in \mathbb{R}^{d \times V}$ that is generated by the “semantic embedding” block, and that contains d-dimensional vectors for each word in the language vocabulary of size V, derived using the NLM proposed by [Collobert and Weston 2008]. Briefly, this model generates word vectors through supervised training of word n-gram validity classifier as follows. Word n-grams are automatically labeled using a simple trick; any n-gram that already exists in the corpus is considered ‘valid’, and by randomly changing one of the n-gram’s words, the resulting n-gram is considered ‘invalid’. The labeled word n-grams are then transformed into vector n-grams, using a randomly initialized look-up table L, and are used to train a softmax classifier to discriminate between valid and invalid n-grams. Classification errors are back-propagated to the input layer to update the word vectors in L. It is worth mentioning that results reported by [Socher et al. 2011b] indicate that using randomly initialized word vectors achieved very similar performances to the case of using word vectors that are pre-trained using the above-mentioned NLM, which indicates the ability of RAE to tune the word vectors using small annotated datasets.

To derive the sentence embedded representation, a binary tree is used to outline the order in which words are combined together to form a sentence. In that tree, leaf nodes correspond to words, non-terminal nodes correspond to intermediate constituents, and the root node represents the full sentence. At each step of the model's recursion, the AE takes two input vectors $x_1, x_2 \in \mathbb{R}^{d\times 1}$, and produces an output 'parent' vector, c, that is fed again to the AE along with the vector of the next node in the tree. This process continues, for each sentence, until an output vector is produced for the root node. This vector, c^*, corresponds to the sentence representation that is used in the second stage of the RAE opinion model.

In addition to the output vector, the AE produces an internal reconstruction of the input word vectors $x'_1, x'_2, \in \mathbb{R}^{d\times 1}$. The AE model is parameterized by a set of weights $\psi = \{W_1, W_2\}$ that are used to generate both the output and the reconstruction vectors using an element-wise nonlinearity function f (usually 'tanh'), as shown in Equations (1) and (2), respectively.

$$c = f\left(W_1^T\,[x_1; x_2]\right) \tag{1}$$

$$[x'_1; x'_2] = f\left(W_2^T c\right) \tag{2}$$

Figure 2 illustrates the structure of the AE along with its parameters. The AE is evaluated by how well it is able to reconstruct its input word vectors. This measure is referred to as the reconstruction error E_{rec}, and is calculated as shown in Equation (3).

$$E_{rec} = \|\ [x_1; x_2] - [x'_1; x'_2]\ \|^2 \tag{3}$$

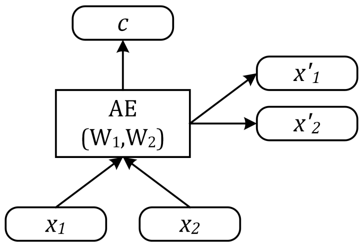

Fig. 2. The structure of the AE block including the model's parameters, inputs and outputs.

The objective of training the AE is to obtain the parameters ψ^* that minimize the reconstruction error for all sentences. The model parameters are updated after each sentence by traversing the sequence of its word vectors $X = \{x_1, \cdots, x_\ell\}$, two vectors at a time, until the whole sentence sequence is processed. The reconstruction error E_{rec} is calculated at every step in the sentence, and the model parameters are updated to minimize the overall reconstruction error, shown in Equation (4), where $(\ell - 1)$ is the number of parsing steps for a sentence of length ℓ. The solution is derived using stochastic gradient descent (SGD), and the overall E_{rec} is back-propagated down to the input layer in order to update and fine-tune the word (nodes) vectors.

$$\psi^* = \arg\min_{\psi} \sum_{i=1}^{\ell-1} E_{rec,\,i} \tag{4}$$

In [Socher et al. 2011b], the sentences' trees are assumed not to be available, and an unsupervised greedy approach is used to derive these trees while training the AE. For each sentence, given the sequence of its word vectors $X = \{x_1, \cdots, x_\ell\}$, the greedy approach starts by applying the AE to all adjacent word vector pairs $[x_i; x_{i+1}]$, $\forall x_i \in X$. The pair with the minimum E_{rec} is added to the parse tree. An internal sequence of word vectors η is initialized with the input sequence X, and is updated after each parsing step by replacing $[x_i; x_{i+1}]$ by their AE output c vector. The process repeats by attempting all adjacent word vector pairs in η, until the whole input sequence is traversed and the sentence tree structure is obtained. Then, similar to the case where the sentence trees are available, the AE parameters are updated to minimize the reconstruction errors for all parsing steps in the sentence, as shown in Equation (4). The final AE model is obtained after processing all sentences in the training set.

After obtaining the sentence (root node) representations, the second stage of the RAE opinion model uses these representations to train a supervised softmax layer to predict the class distribution for each sentence. Assuming there are K opinion classes, the softmax layer outputs, for each sentence, a K-dimensional multinomial distribution as follows: $y = 1/\left(1 + \exp\left(-\theta^\top c^*\right)\right)$, where $c^* \in \mathbb{R}^d$ is the sentence representation that was learned using RAE in the first stage, and $\theta \in \mathbb{R}^{d \times K}$ is the set of softmax parameters. The k^{th} element in y is interpreted as the conditional probability of the k^{th} class given the sentence representation; $y_k = p\left(k|c^*\right)$. The set of parameters θ is obtained by minimizing the cross-entropy error, shown in Equation (5), using SGD. This is the only supervised learning step in the RAE opinion model.

$$E_{ce}\left(c^*, y; \theta\right) = -\sum_{k=1}^{K} t_k \log y_k \tag{5}$$

where t_k is the gold conditional probability of the k^{th} class given sentence representation c^*. For binary classification, $t \in \mathbb{R}^2$ is $[1, 0]$ for class 1 and $[0, 1]$ for class 2.

Overall, the baseline RAE [Socher et al. 2011b] has the following advantages. It models the semantic interactions between the different constituents and embeds them into a sentence-level representation that reflects the overall semantics of the sentence, and that is useful for opinion classification. It also derives these sentence representations in an unsupervised way, eliminating the need for additional resources such as the sentiment treebank needed for RNTN [Socher et al. 2013].

4. PROPOSED AROMA

In this section, we present the challenges related to applying the baseline RAE opinion model to Arabic text, in addition to its limitations equally applicable to English. Then, we propose AROMA that augments the RAE model with the necessary components to address these challenges and limitations.

The baseline RAE suffers when trying to capture the complexity of Arabic language morphology, which mainly results in lexical ambiguity and limits the model's ability to generalize. Also, the input word vectors do not capture the full word-level semantics as they do not correctly embed its sentiment aspects. These "sentiment-inaccurate" word-level embeddings are used to derive sentence representations that also do not fully capture sentence-level sentiment, which affects opinion classification. At last, in the baseline approach, the sentence tree that is used for recursion is derived following a greedy approach and may not reflect the natural order in which words and constituents are combined to express the overall meaning of a sentence.

Figure 3 shows the proposed AROMA framework to address the above-mentioned challenges and limitations. The changes are highlighted in blue color. We perform morphological tokenization of the input text to overcome the issue of morphological complexity and over-fitting We propose a neural network architecture to derive embeddings that capture word-level sentiment information. We also propose an unsupervised pre-training block to improve the initialization of both semantic and sentiment embedding models. Finally, we use phrase structure parsers instead of the greedy algorithm to generate grammatically motivated parse trees that are used as a basis for AE recursion. These solutions are described with further details in the following subsections.

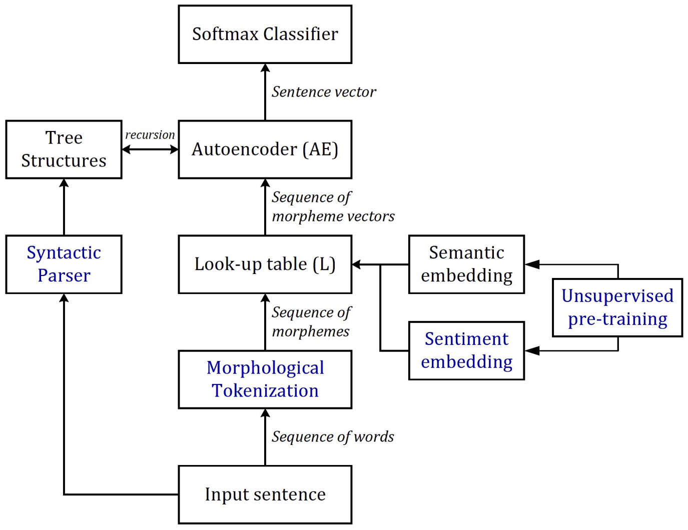

Fig. 3. The framework of the proposed AROMA, with contributions highlighted in blue.

4.1. Tokenization to overcome Arabic-related Challenges

Several issues and challenges arise when applying the baseline RAE model to Arabic text, mainly due to the complex morphology of the language. We focus on addressing challenges related to lexical sparsity and ambiguity, which affect opinion modeling.

4.1.1. Lexical Sparsity. Arabic is an agglutinative language, where complex words are formed by concatenating morphemes (stems and clitics) using derivational and inflectional morphology. As a result, a large vocabulary is generated from a smallish set of roots via morphology. We refer to this phenomena as '*lexical sparsity*'. For example, the root كتب *ktb* generates at least 30 different words, most of which share the same core meaning. Training opinion models using raw Arabic words will suffer from this high sparsity leading to poor generalizations. For instance, a model that has learned the semantics of the word ينتصر *yntSr* 'he wins' cannot use this knowledge to understand the new unseen word such as ينتصران *yntSrAn* 'they [dual] win', although both words carry the same semantics and sentiment information, but have different forms due to different morphological properties. Comparing to English, the words-to-morphemes ratio in Arabic is two folds greater than that in English when observed in very large corpora [Alotaiby et al. 2014]. Also, the number of unique Arabic word forms in an Arabic-English parallel corpus is two times greater than that in English, whereas the overall count of Arabic words is 20% less than that in English [El Kholy and Habash

2012]. These observations are indications to the sparsity of Arabic language, and also to the complexity of Arabic words that are often packed with many information corresponding to complex structures in English. For example, the word وسيكاتبونها *wa+sa+yu-kAtib-uwna+hA* translates to a phrase in English: ‘and they will correspond with her’.

4.1.2. Lexical Ambiguity. Refers to cases where words with identical forms express different meanings or sentiments. Lexical ambiguity was quantified in ATB, where, on average, every word form has more than two different morphological analyses, more than English [Habash and Rambow 2005]. Ambiguity is attributed to the fact that diacritization (marking short vowels, nunation and gemination) is optional in Arabic orthography, despite its key role at disambiguation [Shahrour et al. 2016]. For example, diacritics help distinguishing عَذَّبَ *Ea*∼aba* ‘tortured’ from عَذِب *Ea*ibo* ‘sweet’. Agglutination (adding clitics to stems) also contributes to ambiguity. For example, the word بشر *b$r* can be clitic-free بشر *b$r* ‘people’, or compound بـ+شر *b+$r* ‘by evil’. Finally, ambiguity may result from the idiosyncratic semantics of Arabic roots whose meanings are identified based on the context. Hence, words derived from the same roots can be used to express different meanings [Habash 2010]. For example, مصيبة *muSiybap*, which is derived from صيب *S-y-b* (‘target-related’) can be positive as in ‘she is right (on target)’ or negative as in ‘disaster (being a target of something bad)’.

Based on this discussion, the baseline RAE model will face the following issues when applied directly to Arabic text. The input word vectors will not be able to reflect the different meanings of each raw word, especially in case of extreme semantic variations that involve a change in sentiment polarity. Also, the model will not generalize well to new unseen words due to the lexical sparsity of the language. Finally, training the model using raw words will prevent it from modeling the semantic interactions among the morphemes, which often plays a key role at identifying the correct meaning and sentiment.

To overcome these challenges, we propose to perform morphological tokenization and train AROMA using the tokenized text. We perform morphological tokenization using MADAMIRA, a morphological analyzer and disambiguator [Pasha et al. 2014], by splitting each word into its morphemes (base word and clitics) according to the ATB scheme, where all clitics except for the definite article ال *Al* ‘the’ are separated from the base word [Habash and Sadat 2006]. Although several levels of tokenization exist due to Arabic’s complex morphology, we used the ATB scheme in order to remain consistent with the tokenization scheme in the Arabic Treebank (ATB) [Maamouri et al. 2004]; a linguistic resource that is used to train and develop many Arabic NLP tools, including syntactic parsers that will be described later in the paper. Also, diacritics are automatically predicted using MADAMIRA. Reducing words to their basic morphemes improves the ability of AROMA to generalize to new unseen morphological variations of training instances, because such variants tend to share the same stem that is identified by tokenization. For example, if AROMA was trained to learn the semantics of ينتصران *yntSrAn* ‘they [dual] win’, it can use this knowledge to understand a new word سينتصرون *syntSrwn* ‘they [plural] will win’ because both words share the same stem ينتصر *yntSr* ‘he wins’ that is already identified due to tokenization. Furthermore, tokenization allows AROMA to distinguish between words that share the same surface form but differ in their morphology, such as بـ+شر+ها *b+$r+hA* ‘by her evil’ and بشر+ها *b$r+hA* ‘told her good news’. Finally, the automatic prediction of diacritics allows further disambiguation by identifying morphological variations. For example, diacritization marks the change in the "voice" morphological feature from active as in هَزَمَ *hazama* 'he defeated' to passive as in هُزِمَ *huzima* 'he was defeated'.

4.2. Sentiment Embedding to overcome the Inaccurate Word Representations

The RAE model [Socher et al. 2011b] used shared-task learning to generate input word embeddings using the NLM proposed by [Collobert and Weston 2008]. These embeddings are then used to perform a different task, which is opinion classification. Results reported by [Socher et al. 2011b] indicate that using these word embeddings achieved a marginal improvement (less than 1%) compared to the case of using randomly initialized word vectors. Although shared-task learning proved successful for many NLP tasks including POS tagging, NER and chunking [Collobert and Weston 2008], the resulting representations should be improved for the task of opinion mining. The main reason is that most NLMs produce word vectors, where each dimension reflects some latent feature of the words, mostly related to syntax and semantics, but not to sentiment. Furthermore, most of these models depend on co-occurrence statistics to generate the embeddings, which may cause words of opposing sentiment polarities to have similar representations just because they appear in similar contexts. Examples of such words are 'good/bad' that appear in similar contexts such as 'the movie was good' and 'the movie was bad'. As a result, given word representations that are inaccurate at capturing sentiment, the RAE generates representations that are not able to accurately capture the overall sentiment of the sentence, thus affecting the classification performance.

To solve the problem of inaccurate word representation, we propose to generate embeddings that capture a broader range of the words' semantics, with sentiment included. We develop an approach, described below, that is inspired by [Collobert and Weston 2008] to derive word semantic embeddings, and that can benefit from existing sentiment lexicons to derive word sentiment embeddings. Both semantic and sentiment vectors are then used to derive more meaningful sentence representations that can train more accurate opinion classifiers.

4.2.1. Semantic Embedding. The input word vectors that are used to train RAE in [Socher et al. 2011b] are derived using the NLM that is based on n-gram validity classification [Collobert and Weston 2008] In this paper, we introduce an unsupervised stage to pre-train the look-up table L before using it for validity classification. This stage is shown as Stage 1 in Figure 4. All word n-grams in the corpus are transformed into vector n-grams using a randomly initialized look-up table L, and are used to train a Deep Belief Network (DBN) generative model by stacking Restricted Boltzmann Machines (RBM). During this stage of pre-training, the reconstruction error is back-propagated to the input layer in order to update and fine-tune the word vectors in L. At the end of pre-training, the updated L is used to initialize the look-up table for use in the validity supervised learning (Stage 2 in Figure 4) that is described in the previous paragraph. Such initialization should improve the validity classifier, compared to the random initialization in [Collobert and Weston 2008], as it provides word vectors that reflect their local context in the n-grams.

4.2.2. Sentiment Embedding. We use the same two-stage approach that we proposed for semantic embedding in order to embed sentiment-related information in the word vectors. The main difference is in the objective of the supervised learning (second stage). Instead of predicting the validity or invalidity of word n-grams, the objective is to predict the sentiment polarity scores of the individual words. In other words, each training input corresponds to a single word vector. Each word vector is updated, in the supervised stage, by back-propagating the sentiment classification error to the weights of

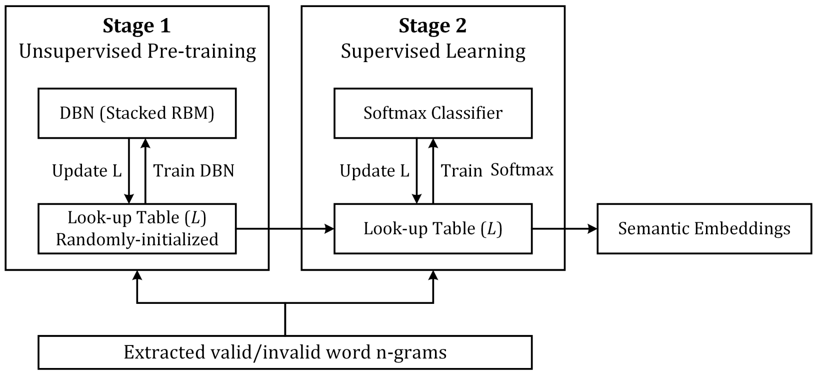

Fig. 4. The architecture of the two-stage NN approach proposed to perform semantic embedding.

the input layer. The error, for each word, reflects the difference between the output of the softmax classifier and the words correct sentiment label, according to a sentiment lexicon that is used as a reference. In this paper, we use the large-scale Arabic Sentiment Lexicon (ArSenL) [Badaro et al. 2014] to obtain the words' sentiment scores. The constructed look-up table corresponds to the word vector representations that reflect word-level sentiment aspects. For words that do not exist in the ArSenL lexicon, we generate sentiment scores by averaging the sentiment scores of all words that co-exist with them in the same sentences. For instance, if a word appears in sentences that contain more positive than negative words, it receives a higher positivity score, and vice versa. These scores are generated using the unlabeled corpus that is used to generate the semantic embeddings. Sentences from the evaluation corpora are not used, to eliminate any potential overfitting.

4.2.3. Fusing Sentiment and Semantic Embeddings. To fuse both types of the acquired word embeddings, we independently generate two sentence representations, c^*_{sem} and c^*_{senti}, using the semantic and sentiment embeddings, respectively. Then, we form the complete sentence representation $c^*_{complete}$ by concatenating both representations. The softmax classifier that is trained on top of the unsupervised RAE produces a probability distribution $y \in \mathbb{R}^K$, where K is the number of opinion classes, and the k^{th} element in y corresponds to the conditional probability of the k^{th} class given $c^*_{complete}$ the complete sentence representation, i.e., $p(k|c^*_{complete})$. This probability can be modeled as a Bayes Network as shown in Equation (6).

$$p\left(k|c^*_{complete}\right) = p\left(k|c^*_{sem}, c^*_{senti}\right) = \frac{p\left(c^*_{sem}, c^*_{senti}|k\right) p\left(k\right)}{p\left(c^*_{sem}, c^*_{senti}\right)} \tag{6}$$

Since the sentence representations for each type of embedding were generated independently, then Equation (6) can be simplified as follows:

$$p\left(k|c^*_{complete}\right) = \frac{p\left(c^*_{sem}|k\right) p\left(c^*_{senti}|k\right) \cdot p\left(k\right)}{p\left(c^*_{sem}\right) p\left(c^*_{senti}\right)} = \frac{p\left(k|c^*_{sem}\right) p\left(k|c^*_{senti}\right)}{p\left(k\right)} \tag{7}$$

where the term $p\left(k\right)$ can be regarded as a normalization factor that can be easily obtained for the opinion distribution in the annotated training data. Each of the remaining terms can be obtained by training AROMA with the corresponding type of word embeddings.

The sentence-level fusion is described in Algorithm 1. First, each embedding block is trained to generate the corresponding look-up table L_{sem} and L_{senti} resulting from the sub-routines *train_semantic_embeddings* block and *train_sentiment_embeddings* block, respectively. Each output is used separately to generate the corresponding sentence representation c^*_{sem} and c^*_{senti}, which are then concatenated together and used as input to the supervised softmax classifier.

It is worth mentioning that the word embeddings are derived according to the tokenization scheme of the text. In other words, if AROMA is applied to raw text, then embeddings are learned for raw words, and if AROMA is applied to morphologically tokenized text, then embeddings are learned for tokenized words. Furthermore, to learn sentiment embeddings using ArSenL, which contains sentiment scores associated with lemmas, the full corpus is lemmatized using MADAMIRA. To learn sentiment embeddings for raw text, the words' lemmas are directly looked up in ArSenL. To learn embeddings for tokenized text, lemmas corresponding to base words (after tokenization) are looked-up in ArSenL, whereas the remaining clitics are directly assigned neutral scores.

Finally, it is worth mentioning that fusion could have been done at the word-level instead of the sentence-level. However, both sentiment and semantic embedding are assumed to be independent; a reasonable assumption based on the fact that semantic embedding takes into account the context of each word, whereas the sentiment embedding model that we proposed does not explore the context, and depends solely on the score or label provided for each word in the lexicon, which is usually out-of-context. Preliminary experiments comparing the two types of fusion support this assumption of independence.

ALGORITHM 1: Training AROMA with the fusion of semantic and sentiment embeddings.

```
Data: Unlabeled corpus $U$, sentiment lexicon $Lex$ and the tree structures $T$ for $N$ sentences
train_semantic_embeddings($U$)
train_sentiment_embeddings($Lex$)
for $i = 1 : N$ (for all training sentences) do
    extract word semantic vectors: $x_{sem,\,i}$
    extract word sentiment vectors: $x_{senti,\,i}$
    derive sentence semantic representation: $c^*_{sem,\,i} = train\_RAE\,(x_{sem,\,i}, T_i)$
    derive sentence sentiment representation: $c^*_{senti,\,i} = train\_RAE\,(x_{senti,\,i}, T_i)$
    obtain complete sentence representation $c^*_{complete,\,i} = concatenate\left(c^*_{sem,\,i}, c^*_{senti,\,i}\right)$
    predict sentence-level opinion: $y_i = softmax\left(c^*_{complete,\,i}\right)$
end
```

4.3. Syntactic Parsing to improve Composition

As described in Section 3, the Auto Encoder in the baseline RAE is recursively trained over a tree structure that is discovered using a greedy algorithm that combines two words (or constituents) at a time such that the reconstruction error of the RAE is minimized at every step of the recursion [Socher et al. 2011b]. However, this algorithm cannot capture information about the language syntax and grammatical structure, unless trained with tremendous corpora. Hence, the generated trees are not optimal in the context of combining semantically and syntactically related constituents, which affects the ability of RAE to accurately capture compositional semantics in text.

Alternatively, we use the Stanford lexicalized phrase structure parser [Green and Manning 2010] to automatically generate syntactic parse trees, over which the AE will be recursively trained. The parser is applied to input text that is morphologically tokenized according to the ATB scheme [Habash and Sadat 2006], and that is consistent with our choice of tokenization as described in Subsection 4.1. Using syntactic parse trees to guide the model's recursion should improve the derived sentence representations, since the path for AE recursion is consistent with the grammatical rules and reflects the natural order in which constituents are combined to express the overall meaning of the sentence.

It is worth mentioning that the Stanford parser produces parse trees that are not necessarily binary, and therefore cannot be used to train recursive models, which require inputs and outputs with consistent dimensionalities. Therefore, we transform the parse trees' grammar to the Chomsky Normal Form (CNF) [Chomsky 1959] using left-factoring, where the choice of left (vs. right) factoring was made such that the model's recursion is consistent with the direction readers follow to combine words while reading Arabic text. The CNF grammar contains only unary and binary production rules. By collapsing the unary productions, we obtain binary parse trees that can be used to train recursive models.

5. EXPERIMENTS AND RESULTS

In this section, we evaluate the different contributions of AROMA, separately and combined, against the baseline RAE opinion model [Socher et al. 2011b]. Then, we compare AROMA to several well-known Arabic opinion models from the literature. We focus on the common task of binary opinion classification, where the opinion class can be either positive or negative.

5.1. Datasets and Experimental setup

In this paper, we evaluate AROMA with three Arabic corpora that represent different genres and that use different Arabic writing styles. The first corpus was developed by [Abdul-Mageed et al. 2011] and consists of newswire articles written in MSA, extracted from the ATB Part 1 V 3.0, and annotated at the sentence-level. For our experiments, we select the subset of sentences with either positive or negative annotations (we exclude neutral and objective sentences). We refer to this corpus as *ATB*. The second corpus is a set of tweets collected by [Refaee and Rieser 2014]. Similar to *ATB*, we select the subset of tweets with either positive or negative annotations, and we refer to this corpus as *Tweets*. The third corpus was compiled by [Farra et al. 2015] by extracting online comments from QALB using topic modeling. This corpus was originally annotated and used for opinion target identification. For our experiments, we annotated this corpus at the comment-level using the CrowdFlower platform, where each comment was assigned to 3-5 independent annotators. The resulting annotations for each comment are then aggregated using majority voting. These comments are written in MSA, but also contain misspellings and dialectal Arabic (DA). We refer to this corpus as *QALB*. Table I illustrates the size and opinion distribution for these corpora.

Table I. Characteristics of the different evaluation corpora.

Dataset	**Corpus size**	**Positive opinions (%)**	**Negative opinions (%)**
ATB	1,180 sentences	68.7%	31.3%
Tweets	2,311 tweets	58.4%	41.6%
QALB	1,133 comments	34.8%	65.2%

Each corpus is preprocessed to clean up and improve the representation of the input data. Preprocessing included: (1) removing non-Arabic words, (2) segmentation by separating punctuation and digits from words using MADAMIRA [Pasha et al. 2014] and (3) normalization. The main purpose of normalization is to improve the quality and coverage of the word embeddings, and also to provide cleaner input to the parser to generate more accurate parse trees. We applied normalization to *characters* by normalizing repeated characters in elongated words, to *emoticons* by replacing emoticons with global 'happy/sad' tokens using a manually compiled list of emoticons' shortcuts, and to *parentheses* by normalizing the different forms of parentheses into one form (the square brackets). To evaluate the impact of normalization, we trained several baseline RAE models using different versions of the inputs, each reflecting a specific type of normalization. We used the *Tweets* dataset for this experiment because it contains significant amounts of elongated words and emoticons, compared to the other datasets. We observed that applying *character*, *emoticons* and *parenthesis* normalization improved the classification accuracy by 0.7%, 0.8% and 0.8%, respectively, compared to the case of "no normalization". Furthermore, applying all these normalization together improved accuracy by 2.2%.

Additional preprocessing was applied to the *Tweets* dataset including removing user mentions, re-tweet (RT) labels and URLs, and also preprocessing hashtag mentions by removing the hashtag symbol and the 'under-scores' connecting between multiple words in a single tag. Hence, these techniques are common practice in the literature [Khan et al. 2015; Go et al. 2009; Kouloumpis et al. 2011].

Performance is quantified using accuracy and F1-score averaged over both opinion classes. To ensure statistical significance of results, the different models are evaluated using 10-fold cross-validation. The AE is formed of three layers, and both the size of the word embeddings and the number of hidden neurons in each layer are set to 50, which yield the best results in a preliminary experiment on a random fold of the *ATB* dataset.

For all experiments, we train the word semantic embeddings using an unlabeled dataset of 20,000 comments extracted from the full QALB, and that do not pertain to any evaluation corpus. The reason for using QALB is that it contains a collection of text written in both MSA and DA, and hence its vocabulary is more likely to cover the different evaluation corpora used in this paper. The unlabeled dataset is also preprocessed using the above-mentioned steps.

5.2. Ablation Analysis

In this subsection, we evaluate the improvements achieved by each of the proposed contributions. Table II illustrates the impact of applying (1) morphological tokenization to combat morphological complexity and improve model's generalization, (2) semantic embedding with and without the unsupervised pre-training, (3) sentiment embedding to provide better input word vectors and (4) syntactic parsing to allow better composition. We consider our baseline to be the baseline RAE model [Socher et al. 2011b] that uses randomly initialized input word vectors, and that iterates over trees derived using the greedy parsing algorithm.

5.2.1. Impact of Morphological Tokenization. According to Table II, tokenization yields significant improvements over the baseline RAE, for all datasets. These results highlight the importance of tokenization to perform composition at a finer-level of Arabic morphology instead of the raw words that are usually packed with many information. Results also show that reducing the language sparsity through tokenization, which renders raw words to their stems, allows the model to generalize to new unseen words.

5.2.2. Impact of Sentiment Embedding. We evaluated the impact of incorporating the unsupervised pre-training stage to the semantic embedding in [Collobert and Weston 2008]. We also evaluated the impact of using sentiment embeddings versus using semantic embeddings. Finally, we evaluated the impact of fusing both embeddings as

Table II. The impact of each of the proposed contributions compared to the baseline RAE, evaluated on the different corpora.

	ATB		QALB		Tweets	
	accuracy	F1-score	accuracy	F1-score	accuracy	F1-score
Baseline RAE	74.3	73.5	71.6	66.5	69.7	61.1
tokenization	77.3 *(+3.0)*	76.2 *(+2.7)*	75.4 *(+3.8)*	71.6 *(+5.1)*	70.9 *(+1.2)*	62.7 *(+1.6)*
sem. embed (no pretrain)	75.9 *(+1.6)*	74.9 *(+1.4)*	72.4 *(+0.8)*	68.1 *(+1.6)*	71.1 *(+1.4)*	63.1 *(+2.0)*
sem. embed (pretrain)	76.5 *(+2.2)*	75.3 *(+1.8)*	72.6 *(+1.0)*	68.4 *(+1.9)*	71.3 *(+1.6)*	64.4 *(+3.3)*
sentiment embed	79.4 *(+5.1)*	78.7 *(+5.2)*	73.8 *(+2.2)*	70.3 *(+3.8)*	71.6 *(+1.9)*	65.8 *(+4.7)*
both embeddings	81.9 *(+7.6)*	82.3 *(+8.8)*	74.3 *(+2.7)*	71.1 *(+4.6)*	73.8 *(+4.1)*	67.4 *(+6.3)*
both embed + tokenization	84.6 *(+10.3)*	84.2 *(+10.7)*	78.5 *(+6.9)*	74.6 *(+8.1)*	75.2 *(+5.5)*	68.1 *(+7.0)*
syntactic parsing	76.7 *(+2.4)*	76.1 *(+2.6)*	71.0 *(-0.6)*	65.6 *(-0.9)*	67.4 *(-2.3)*	58.1 *(-3.0)*
All (AROMA)	**86.5** *(+12.2)*	**84.9** *(+11.4)*	**79.2** *(+7.6)*	**75.5** *(+9.0)*	**76.9** *(+7.2)*	**68.9** *(+7.8)*

described in Subsection 4.2.3. Since we are comparing to the baseline RAE, experiments are carried on the raw words without tokenization. Results in Table II illustrate the importance of pre-training the word vectors as a better initialization for the supervised n-gram validity classification. They also show that, for all datasets, sentiment embedding outperformed semantic embedding, which indicates that incorporating information from ArSenL lexicon helps generating representations that capture sentiment-related information, which is clearly more relevant for opinion mining. Results also indicate that fusing both types of embedding introduces additional improvements since the classifier is now trained using sentence representations that are more complete as they capture semantic and sentiment information. At last, it can be observed that using semantic embeddings introduced significant improvements compared to using randomly initialized word vectors, which does not align with the findings in English [Socher et al. 2011b], where semantic embeddings only achieved marginal improvements (less than $> 1\%$). This is due to the unsupervised pre-training stage that we introduced to the semantic embedding model. It is also an indicator that Arabic words, as opposed to English, are packed with rich information that cannot be captured by only fine-tuning the randomly initialized vectors during RAE training, but require a better initialization using pre-trained embeddings.

5.2.3. Impact of Syntactic Parsing. We evaluated the impact of using syntactic parse trees versus using trees generated using the greedy parsing algorithm. Since we are comparing to the baseline RAE, experiments are carried on the raw words without tokenization. Since Stanford parse trees are generated for tokenized input text, we de-tokenize the resulting trees by merging nodes corresponding to morphemes of the same word into one node.

According to Table II, using the Stanford syntactic parse trees improves the performance only on the ATB dataset, while it hurts the performance on the other datasets. The main reason for this behavior is that ATB sentences are written in MSA and comply to the grammatical rules of the language, which allows the Stanford parser to generate parse trees that reflect the natural order in which words and constituents are combined in a sentence. On the other hand, the QALB and Tweets datasets contain significant amounts of noise represented by dialectal Arabic and misspellings, especially the Tweets dataset. This is reflected in the results, where automatic parsers failed to produce meaningful parse trees, causing performance degradation compared to the greedy discovery algorithm.

To confirm the importance of the syntactic trees, we evaluate the impact of using gold syntactic trees that are manually developed by expert linguists, and hence free of automatic parsing errors. Gold trees are only available for the ATB dataset since its sentences are extracted from the ATB Part 1 V 3.0. Having access to V 4.1 [Maamouri et al. 2010a], we map its treebanks to those in V 3.0, from which we obtain the gold trees for the ATB dataset. We evaluated three different types of parse trees: those discovered using the greedy algorithm, those generated by automatic parsers (Stanford parser) and the gold trees. Since the gold trees assume the input to be morphologically tokenized, we evaluated the different trees following the same assumption. Table III illustrates the performance improvement achieved by eliminating the automatic parsing errors. Results indicate that using the gold trees achieve the best performance, and that errors introduced by automatic parsing cause around 0.5% degradation on the overall performance, when used to train RAE models on MSA data.

Table III. Impact of different tree structures on the performance of AROMA.

	Accuracy	F1-score
Tokenization + greedy trees	77.2	76.2
Tokenization + automatic Stanford trees	78.5 *(+1.3)*	77.8 *(+1.6)*
Tokenization + gold PATB trees	78.9 *(+1.7)*	78.1 *(+1.9)*

5.2.4. Impact of all contributions. The AROMA framework is realized by applying all different approaches to address the identified challenges. Results in Table II show that AROMA achieves highest performance compared to its individual components. Compared to baseline RAE, AROMA achieves absolute accuracy improvement of 12.2%, 7.6% and 7.2% on the ATB, QALB and Tweets datasets, respectively. The highest improvement was achieved on the ATB dataset since it consists of sentences written in MSA, hence benefiting from all proposed contributions including the use of syntactic parse trees, as opposed to the other datasets where syntactic parsing did not achieve the hoped-for performances. Results also indicate that the different components of AROMA are synergistic and achieve state-of-the-art performance in Arabic opinion mining. For example, tokenization helps producing better embeddings as it reduces the language sparsity and resolves much of the morphological complexity that becomes explicitly highlighted at the surface (form)-level instead of being embedded into the learned representations. Also, although syntactic parsing had a negative impact on the QALB and Tweets datasets, it performs better than greedy parsing when applied to morphologically tokenized text, as shown in Table II. This suggests that these two factors work in harmony with each other.

We conducted the following analysis on results obtained with the QALB dataset. Among the 321 sentences that were misclassified by the baseline RAE, we compiled two sets of sentences. (S_1): 178 sentences that were correctly classified using AROMA, and (S_2): 143 sentences that remained misclassified despite using AROMA.

We ran the sentences from S_1 through different models, each incorporating a particular contribution from AROMA. We found that 73% were correctly classified by the model that uses both semantic and sentiment embeddings, 34% were correctly classified by the model that applies morphological tokenization, and 23% were correctly classified by the model that uses the syntactic parser. These observations align with the results in Table II, and suggest that producing word representations that capture sentiment-related information is more critical than having correct tokenization or order of parsing. We further inspected the sentences that were correctly classified by the model that uses the syntactic parser. These sentences tend to be longer and contain interrupting phrases, which requires grammatically inspired parsing in order to combine the constituents properly.

Finally, after inspecting the 143 sentences from (S_2), we observed that many of these sentences are inherently difficult and challenging and require human-level knowledge and intelligence to capture the correct sentiment, which in many cases is affected by a small part of the sentence. This raises the need for approaches to properly model the relations and contributions of the different constituents to the overall sentiment of the sentence.

5.3. Results Benchmarking

In this section, we evaluate AROMA against several Arabic opinion models proposed in literature. We compare to SVM and NB models trained using BoW features with different choices of preprocessing and feature representation [Rushdi-Saleh et al. 2011; Elawady et al. 2014]. We train using word, stem and lemma *n*grams ($n = 1, 2, 3$), represented using presence, term frequency (TF) and term frequency inverse document frequency (TFiDF) scores. We report the best results achieved using the TFiDF scores. We also train SVM using aggregated sentence-level sentiment scores based on the ArSenL sentiment lexicon as proposed by [Badaro et al. 2014]. This model achieved better results than [Abdul-Mageed et al. 2011] on the ATB dataset. At last, we compare to several deep learning models that were evaluated for Arabic opinion mining [Al Sallab et al. 2015]. These models are: Deep Belief Networks (DBN), Deep Neural Networks (DNN), and Deep Auto Encoders (DAE) combined with DBN. These models are trained using two types of features: the BoW and the sentence-level aggregated sentiment scores. All the above-mentioned models are also trained using 10-fold cross-validation to ensure statistical significance of the results. Table IV illustrates the performance improvement achieved by AROMA compared to the aforementioned opinion models. Comparing to the second-best approach, AROMA introduces accuracy improvements of 7.3%, 1.7% and 7.6% on the *ATB*, *QALB* and *Tweets* datasets, respectively.

6. CONCLUSION

In this paper, we have presented AROMA: a recursive deep learning model for opinion mining in Arabic. AROMA was proposed by addressing several challenges and limitations that arise when applying the RAE model to perform opinion mining in Arabic. Arabic-specific challenges including the morphological complexity and language sparsity were addressed by modeling semantic composition at the Arabic morpheme-level after performing morphological tokenization. We also proposed to perform word sentiment embedding in order to provide a broader set of features that cover syntactic, semantic and sentiment information. At last, we used phrase structure parser to generate syntactic parse trees that are used as a reference for AROMA's recursion. This allowed modeling semantic and sentiment composition following the natural order in which words and constituents are combined together in a sentence.

The proposed model was evaluated on three Arabic corpora that correspond to different genres (newswire, online comments and tweets) and different writing styles (MSA and dialectal Arabic). Experiments showed that each of the proposed contributions in AROMA was able to achieve significant improvement. In particular, the combination of all contributions, which makes up for the complete AROMA model, was able to improve the classification accuracy by 12.2%, 8.4% and 7.2% on the *ATB*, *QALB* and *Tweets* datasets, respectively, compared to the baseline RAE model. Furthermore, AROMA outperformed several well-known approaches from the literature by 7.3%, 1.7% and 7.6% on the same datasets. These results indicate the ability of AROMA to perform accurate opinion classification when applied to a complex language such as Arabic that lacks large-scale opinion lexical resources.

Future work will include the exploration of the full orthographic and morphological space in Arabic, including different levels of surface representation and morphological abstraction, in order to boost the performance of deep learning models for opinion mining.

Table IV. Results of benchmarking the performance of AROMA against opinion models from the literature.

Model	Features	ATB		QALB		Tweets	
		accuracy	F1-score	accuracy	F1-score	accuracy	F1-score
DNN	ArSenL scores	54.7	43.9	52.3	48.9	58.3	50.7
	BoW	39.3	38.8	43.6	40.1	54.6	38.9
DBN	ArSenL scores	56.9	46.2	55.4	47.5	61.2	54.5
	BoW	40.9	39.7	45.0	42.3	57.6	43.2
DAE-DBN	ArSenL scores	59.7	59.9	59.2	54.2	63.7	57.8
	BoW	42.9	43.3	47.5	44.6	59.3	44.6
linear SVM	ArSenL scores	62.8	56.7	71.0	62.8	68.7	40.7
	word 1-grams	75.3	73.9	76.1	71.3	62.1	54.7
	stem 1-grams	77.5	76.6	77.5	74.7	62.4	55.9
	lemma 1-grams	77.5	76.5	77.1	74.7	62.9	56.7
	word 1-2-grams	76.2	73.9	73.3	62.3	68.5	56.6
	stem 1-2-grams	79.2	77.7	77.4	70.3	68.4	57.4
	lemma 1-2-grams	78.7	77.2	76.9	69.9	68.7	57.8
	word 1-3-grams	75.3	71.8	69.9	54.0	68.5	54.5
	stem 1-3-grams	77.5	75.2	74.4	63.9	69.3	56.7
	lemma 1-3-grams	79.1	77.1	74.5	64.4	68.7	55.7
NB	word 1-grams	69.8	69.4	69.5	65.7	54.7	53.5
	stem 1-grams	74.4	73.9	70.6	66.1	56.3	54.3
	lemma 1-grams	73.6	73.2	68.9	65.1	55.2	53.5
	word 1-2-grams	70.1	69.3	72.4	67.9	56.7	55.0
	stem 1-2-grams	73.8	73.0	73.3	67.8	57.9	55.3
	lemma 1-2-grams	74.2	73.4	71.5	65.7	56.0	53.8
	word 1-3-grams	70.3	69.5	72.6	68.2	56.8	55.2
	stem 1-3-grams	73.6	72.8	73.4	67.9	58.3	55.6
	lemma 1-3-grams	73.1	72.1	72.2	66.4	56.0	53.8
RAE	raw words	74.3	73.5	71.6	66.5	69.7	61.1
AROMA	tokenized words	**86.5**	**84.9**	**79.2**	**75.5**	**76.9**	**68.9**

REFERENCES

Ahmed Abbasi, Hsinchun Chen, and Arab Salem. 2008. Sentiment analysis in multiple languages: Feature selection for opinion classification in Web forums. *ACM Transactions on Information Systems (TOIS)* 26, 3 (2008), 12.

Ahmed Abbasi, Stephen France, Zhu Zhang, and Hsinchun Chen. 2011. Selecting attributes for sentiment classification using feature relation networks. *IEEE Transactions on Knowledge and Data Engineering* 23, 3 (2011), 447–462.

Muhammad Abdul-Mageed and Mona T Diab. 2014. SANA: A Large Scale Multi-Genre, Multi-Dialect Lexicon for Arabic Subjectivity and Sentiment Analysis.. In *LREC*. 1162–1169.

Muhammad Abdul-Mageed, Mona T Diab, and Mohammed Korayem. 2011. Subjectivity and sentiment analysis of modern standard Arabic. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2*. Association for Computational Linguistics, 587–591.

Rodrigo Agerri, Xabier Artola, Zuhaitz Beloki, German Rigau, and Aitor Soroa. 2015. Big data for Natural Language Processing: A streaming approach. *Knowledge-Based Systems* 79 (2015), 36–42.

Mohammed N Al-Kabi, Nawaf A Abdulla, and Mahmoud Al-Ayyoub. 2013. An analytical study of Arabic sentiments: Maktoob case study. In *Internet Technology and Secured Transactions (ICITST), 2013 8th International Conference for*. IEEE, 89–94.

Ahmad A Al Sallab, Ramy Baly, Gilbert Badaro, Hazem Hajj, Wassim El Hajj, and Khaled B Shaban. 2015. Deep Learning Models for Sentiment Analysis in Arabic. In *ANLP Workshop 2015*. 9.

Fahad Alotaiby, Salah Foda, and Ibrahim Alkharashi. 2014. Arabic vs. English: Comparative Statistical Study. *Arabian Journal for Science and Engineering* 39, 2 (2014), 809–820.

Mohamed A Aly and Amir F Atiya. 2013. LABR: A Large Scale Arabic Book Reviews Dataset.. In *ACL (2)*. 494–498.

Gilbert Badaro, Ramy Baly, Rana Akel, Linda Fayad, Jeffrey Khairallah, Hazem Hajj, Wassim El-Hajj, and Khaled Bashir Shaban. 2015. A Light Lexicon-based Mobile Application for Sentiment Mining of Arabic Tweets. In *ANLP Workshop 2015*. 18.

Gilbert Badaro, Ramy Baly, Hazem Hajj, Nizar Habash, and Wassim El-Hajj. 2014. A large scale Arabic sentiment lexicon for Arabic opinion mining. *ANLP 2014* 165 (2014).

Yoshua Bengio. 2012. Practical recommendations for gradient-based training of deep architectures. In *Neural Networks: Tricks of the Trade*. Springer, 437–478.

William Black, Sabri Elkateb, Horacio Rodriguez, Musa Alkhalifa, Piek Vossen, Adam Pease, and Christiane Fellbaum. 2006. Introducing the Arabic wordnet project. In *Proceedings of the third international WordNet conference*. Citeseer, 295–300.

Erik Cambria and Amir Hussain. 2015. *Sentic computing: a common-sense-based framework for concept-level sentiment analysis*. Vol. 1. Springer.

Noam Chomsky. 1959. On certain formal properties of grammars. *Information and control* 2, 2 (1959), 137–167.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*. ACM, 160–167.

Ahmed El Kholy and Nizar Habash. 2012. Orthographic and morphological processing for English–Arabic statistical machine translation. *Machine Translation* 26, 1-2 (2012), 25–45.

Rasheed M Elawady, Sherif Barakat, and M Elrashidy Nora. 2014. Sentiment Analyzer for Arabic Comments. *International Journal of Information Science and Intelligent System* 3, 4 (2014), 73–86.

Andrea Esuli and Fabrizio Sebastiani. 2006. Sentiwordnet: A publicly available lexical resource for opinion mining. In *Proceedings of LREC*, Vol. 6. Citeseer, 417–422.

Noura Farra, Kathleen McKeown, and Nizar Habash. 2015. Annotating Targets of Opinions in Arabic using Crowdsourcing. In *ANLP Workshop 2015*. 89.

Alec Go, Richa Bhayani, and Lei Huang. 2009. Twitter sentiment classification using distant supervision. *CS224N Project Report, Stanford* 1 (2009), 12.

Spence Green and Christopher D Manning. 2010. Better Arabic parsing: Baselines, evaluations, and analysis. In *Proceedings of the 23rd International Conference on Computational Linguistics*. Association for Computational Linguistics, 394–402.

Nizar Habash and Owen Rambow. 2005. Arabic tokenization, part-of-speech tagging and morphological disambiguation in one fell swoop. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics, 573–580.

Nizar Habash and Fatiha Sadat. 2006. Arabic preprocessing schemes for statistical machine translation. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*. Association for Computational Linguistics, 49–52.

Nizar Y Habash. 2010. Introduction to Arabic natural language processing. *Synthesis Lectures on Human Language Technologies* 3, 1 (2010), 1–187.

Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. 2006. A fast learning algorithm for deep belief nets. *Neural computation* 18, 7 (2006), 1527–1554.

Hossam S Ibrahim, Sherif M Abdou, and Mervat Gheith. 2015. Sentiment analysis for modern standard arabic and colloquial. *arXiv preprint arXiv:1505.03105* (2015).

Aamera ZH Khan, Mohammad Atique, and VM Thakare. 2015. Combining lexicon-based and learning-based methods for Twitter sentiment analysis. *International Journal of Electronics, Communication and Soft Computing Science & Engineering (IJECSCSE)* (2015), 89.

Efthymios Kouloumpis, Theresa Wilson, and Johanna D Moore. 2011. Twitter sentiment analysis: The good the bad and the omg! *Icwsm* 11 (2011), 538–541.

Bing Liu and Lei Zhang. 2012. A survey of opinion mining and sentiment analysis. In *Mining text data*. Springer, 415–463.

Mohamed Maamouri, Ann Bies, Tim Buckwalter, and Wigdan Mekki. 2004. The penn arabic treebank: Building a large-scale annotated arabic corpus. In *NEMLAR conference on Arabic language resources and tools*, Vol. 27. 466–467.

Mohamed Maamouri, Ann Bies, Seth Kulick, Fatma Gaddeche, Wigdan Mekki, Sondos Krouna, Basma Bouziri, and Zaghouani Wajdi. 2010a. Arabic Treebank: Part 1 v 4.1. *LDC Catalog No. LDC2010T13. ISBN* (2010).

Mohamed Maamouri, Dave Graff, Basma Bouziri, Sondos Krouna, and Seth Kulick. 2010b. LDC Standard Arabic morphological analyzer (SAMA) v. 3.1. *LDC Catalog No. LDC2010L01. ISBN* (2010), 1–58563.

T Mikolov and J Dean. 2013. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems* (2013).

George A Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine J Miller. 1990. Introduction to WordNet: An on-line lexical database. *International journal of lexicography* 3, 4 (1990), 235–244.

Behrang Mohit, Alla Rozovskaya, Nizar Habash, Wajdi Zaghouani, and Ossama Obeid. 2014. The first QALB shared task on automatic text correction for Arabic. In *Proceedings of the EMNLP 2014 Workshop on Arabic Natural Language Processing (ANLP)*. 39–47.

Asmaa Mountassir, Houda Benbrahim, and Ilham Berrada. 2012. A cross-study of Sentiment Classification on Arabic corpora. In *Research and Development in Intelligent Systems XXIX*. Springer, 259–272.

Preslav Nakov, Sara Rosenthal, Svetlana Kiritchenko, Saif M Mohammad, Zornitsa Kozareva, Alan Ritter, Veselin Stoyanov, and Xiaodan Zhu. 2016. Developing a successful SemEval task in sentiment analysis of Twitter and other social media texts. *Language Resources and Evaluation* 50, 1 (2016), 35–65.

Nazlia Omar, Mohammed Albared, Adel Qasem Al-Shabi, and Tareq Al-Moslmi. 2013. Ensemble of Classification Algorithms for Subjectivity and Sentiment Analysis of Arabic Customers' Reviews. *International Journal of Advancements in Computing Technology* 5, 14 (2013), 77.

Arfath Pasha, Mohamed Al-Badrashiny, Mona T Diab, Ahmed El Kholy, Ramy Eskander, Nizar Habash, Manoj Pooleery, Owen Rambow, and Ryan Roth. 2014. MADAMIRA: A Fast, Comprehensive Tool for Morphological Analysis and Disambiguation of Arabic.. In *LREC*, Vol. 14. 1094–1101.

Kumar Ravi and Vadlamani Ravi. 2015. A survey on opinion mining and sentiment analysis: Tasks, approaches and applications. *Knowledge-Based Systems* 89 (2015), 14–46.

Eshrag Refaee and Verena Rieser. 2014. An Arabic Twitter Corpus for Subjectivity and Sentiment Analysis.. In *LREC*. 2268–2273.

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

Anas Shahrour, Salam Khalifa, and Nizar Habash. 2016. Improving Arabic Diacritization through Syntactic Analysis. In *LREC*.

Amira Shoukry and Ahmed Rafea. 2012. Sentence-level Arabic sentiment analysis. In *Collaboration Technologies and Systems (CTS), 2012 International Conference on*. IEEE, 546–550.

Richard Socher, Cliff C Lin, Chris Manning, and Andrew Y Ng. 2011a. Parsing natural scenes and natural language with recursive neural networks. In *Proceedings of the 28th international conference on machine learning (ICML-11)*. 129–136.

Richard Socher, Jeffrey Pennington, Eric H Huang, Andrew Y Ng, and Christopher D Manning. 2011b. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 151–161.

Richard Socher, Alex Perelygin, Jean Y Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the conference on empirical methods in natural language processing (EMNLP)*, Vol. 1631. Citeseer, 1642.

Kai Sheng Tai, Richard Socher, and Christopher D Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. *arXiv preprint arXiv:1503.00075* (2015).

Duyu Tang, Bing Qin, and Ting Liu. 2015. Document modeling with gated recurrent neural network for sentiment classification. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. 1422–1432.

UNESCO. 2014. World Arabic Language Day. http://english.alarabiya.net/articles/2012/12/18/255853.html. (2014).