A Sentiment Treebank and Morphologically Enriched Recursive Deep Models for Effective Sentiment Analysis in Arabic

RAMY BALY, American University of Beirut
HAZEM HAJJ, American University of Beirut
NIZAR HABASH, New York University Abu Dhabi
KHALED BASHIR SHABAN, Qatar University
WASSIM EL-HAJJ, American University of Beirut

Accurate sentiment analysis models encode the sentiment of words and their combinations to predict the overall sentiment of a sentence. This task becomes challenging when applied to morphologically-rich languages (MRL). In this paper, we evaluate the use of deep learning advances, namely the Recursive Neural Tensor Networks (RNTN), for sentiment analysis in Arabic as a case study of MRLs. While Arabic may not be considered the only representative of all MRLs, the challenges faced and proposed solutions in Arabic are common to many other MRLs. We identify, illustrate, and address MRL-related challenges, and show how RNTN is affected by the morphological richness and orthographic ambiguity of the Arabic language. To address the challenges with sentiment extraction from text in MRL, we propose to explore different orthographic features as well as different morphological features at multiple levels of abstraction ranging from raw words to roots. A key requirement for RNTN is the availability of a sentiment treebank; a collection of syntactic parse trees annotated for sentiment at all levels of constituency, and that currently only exists in English. Therefore, our contribution also includes the creation of the first Arabic Sentiment Treebank (ArSenTB) that is morphologically and orthographically enriched. Experimental results show that, compared to the basic RNTN proposed for English, our solution achieves significant improvements up to 8% absolute at the phrase-level and 10.8% absolute at the sentence-level, measured by average F1-score. It also outperforms well-known classifiers including SVM, Recursive Auto Encoders (RAE) and Long Short-Term Memory (LSTM) by 7.6%, 3.2% and 1.6% absolute respectively, all models being trained with similar morphological considerations.

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

ACM Reference Format:
Ramy Baly, Hazem Hajj, Nizar Habash, Khaled Bashir Shaban and Wassim El-Hajj, 2016. A Sentiment Treebank and Morphologically Enriched Recursive Deep Models for Effective Sentiment Analysis in Arabic. *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

Sentiment analysis aims to identify opinions or sentiments expressed towards entities such as organizations, products, and individuals. Nowadays, huge amounts of user-generated content are flooding the Internet as people tend to express their opinions through social networks, blogs and reviews websites. Therefore, sentiment analysis has become a very active area of research due to the diversity of its applications and the associated challenges that need to be addressed. Research on sentiment analysis first appeared in English [Pang et al. 2002; Turney 2002; Pang and Lee 2004], with state-of-the-art models using deep learning to model semantic compositionality to infer the overall sentiment [Socher et al. 2013; Tang et al. 2015; Tai et al. 2015].

Authors' addresses: R. Baly, H. Hajj, Electrical and Computer Engineering Department, American University of Beirut, Beirut, Lebanon; emails: rgb15@mail.aub.edu, hh63@aub.edu.lb; N. Habash, Computer Science Department, New York University Abu Dhabi, Abu Dhabi, United Arab Emirates; email: nizar.habash@nyu.edu; 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.

DOI: http://dx.doi.org/10.1145/0000000.0000000

Language-independent deep learning techniques in NLP have been successful in modeling the syntactic and semantic aspects of many languages. However, performing sentiment analysis in morphologically-rich languages (MRL) is challenging due to ambiguity and data sparsity issues. We consider Arabic language an example of MRLs, where Arabic ranks 5^{th} among most-spoken languages worldwide, and 4^{th} in the number of Internet users [IWS 2015; UNESCO 2014]. However, while MRLs differ from each other, they share problems such as lexical sparsity. Hence, we believe that other MRLs can benefit from our solution proposed for Arabic, which mainly explores morphological abstraction to reduce sparsity.

Table I shows how a single ambiguous Arabic word can be tokenized and diacritized differently producing very different sentiments in different contexts, and corresponding to different multi-word English phrases. Another challenge is the lack of sentiment resources supporting recursive deep models in MRLs, including Arabic. Despite the recent release of several resources such as sentiment lexicons [Badaro et al. 2014; Abdul-Mageed and Diab 2014] and annotated corpora [Abdul-Mageed and Diab 2012; Rushdi-Saleh et al. 2011], the number and diversity of these resources are still small compared to those developed in English, and they also lack support for recursive sentiment models.

Table I: Arabic morphology affecting sentiment.

Surface Form	وبشرها *wb$rhA*	
Tokenization	و+بشر+ها *w+b$r+hA*	و+ب+شر+ها *w+b+$r+hA*
Diacritics	وَبَشَّرَهَا *waba$~arahaA*	وَبِشَرِّهَا *wabi$ar~ihaA*
Gloss	‘and he told her good news’	‘and with her evil’
Sentiment	positive	negative

In this paper, we explore and evaluate the use of recursive deep models for sentiment analysis in Arabic. We believe that most of the ideas and solutions explored are applicable to other MRLs. Specifically, we apply the Recursive Neural Tensor Networks (RNTN) model that is considered among state-of-the-art models in English [Socher et al. 2013]. RNTN models require the presences of a sentiment treebank, a collection of phrase structure parse trees with sentiment annotations at all levels of constituency. Such a resource currently only exists in English. Hence, in this paper we present the Arabic sentiment treebank (ArSenTB), the first of its kind for Arabic, which will enable Arabic sentiment analysis at different levels of text granularity starting from the word-level. This resource will be publicly released.

Our results show that the basic RNTN model does not achieve outstanding performances in Arabic as it did in English due to the complex morphology of Arabic. We overcome this limitation by enriching our ArSenTB, and consequently the RNTN models, with orthographic features such as diacritization and elongation, as well as morphological features including several levels of morphological abstraction ranging from raw words up to roots.

Our proposed morphologically-enriched RNTN achieved significant accuracy improvements, up to 8.2% and 9.4% at the phrase and the sentence-level, respectively, compared to the baseline RNTN that does not account to neither morphology nor orthography of the language. It also outperforms well-known classifiers including SVM, Recursive Auto Encoders (RAE) and Long Short-Term Memory (LSTM) that are trained with similar morphological considerations.

The rest of paper is organized as follows. Section 2 presents an overview on the related work. Section 3 presents challenges that affect sentiment analysis in Arabic, and the proposed solution of integrating Arabic morphology with RNTN. Section 4 describes the approach for developing

ARSENTB and evaluating its quality. Section 5 presents the evaluation of the morphologically-enriched RNTN and shows its superior performance. Conclusion is provided in Section 6.

2. RELATED WORK

In this section, we provide an overview of previous sentiment analysis models proposed in English. Then, we describe methods developed for Arabic and highlight gaps to be addressed.

2.1. English Sentiment Analysis

Research on sentiment analysis has achieved its highest level of accuracy in English, mainly due to the availability of NLP tools and sentiment lexical resources. State-of-the-art models in English are based on sentence encoders; "parametric functions" that transform a sequence of word embedding vectors into a sentence vector. Recently-proposed sentence encoders include the sequence-based recurrent neural networks (RNN) [Tarasov 2015] and convolutional neural networks [Zhang et al. 2015]. Also, tree-structured models have been proposed to propagate information up a binary parse tree, such as the Recursive Auto Encoders (RAE) [Socher et al. 2011], Deep Recursive Neural Networks (DRNN) [Irsoy and Cardie 2014], Recursive Neural Tensor Networks (RNTN) [Socher et al. 2013], Deep Convolutional Neural Networks [Kalchbrenner et al. 2014] and Long Short-Term Memory [Tai et al. 2015]. Recursive models were also used to improve word embeddings for morphologically-complex and rare words [Luong et al. 2013]. The model that performed best on the task of three-way sentiment classification in tweets, in SemEval 2016, used distant supervision to train a two-layer convolutional neural networks with on a huge corpus of tweets [Deriu et al. 2016]. The model that performed best on the five-way sentiment classification task, also in SemEval 2016, used a set of hand-crafted features previously, proposed by [Kiritchenko et al. 2014], to train a logistic regression model by optimizing the evaluation measure, namely the macro F1-score [Balikas and Amini 2016]. The feature set included word and character *n*-grams, POS tags, stylistic features (count of exclamation and question marks, capitalized and elongated words, and negated contexts) and semantic features (count of positive/negative emoticons, positive/negative words, and the existence of emojis). Aside from deep learning, a framework that automates the human reading process improved the performance of several state-of-the-art models [Baly et al. 2016].

2.2. Arabic Sentiment Analysis

Previous research on Arabic sentiment analysis used word *n*grams to train several classifiers, mainly the Support Vector Machines (SVM) [Rushdi-Saleh et al. 2011; Aly and Atiya 2013; Al-Kabi et al. 2013; Shoukry and Rafea 2012], Naïve Bayes models [Mountassir et al. 2012; Elawady et al. 2014] and ensemble classifiers [Omar et al. 2013]. A rich feature set that contains surface (word *n*grams), syntactic (root and part-of-speech *n*grams) and stylistic (letter and digit *n*grams, word length, etc.) features was proposed by [Abbasi et al. 2008] along with the Entropy-Weighted Genetic Algorithm (EWGA); an effective and efficient feature reduction approach. Sentiment lexicons provided an additional source of sentiment features that were used to train sentiment classifiers [Badaro et al. 2014; Badaro et al. 2015]. Sentence and document-level sentiment analysis in Arabic was explored by [Farra et al. 2010]. They proposed two approaches, the first is based on the grammatical structure of Arabic sentences and the second is based on lexicon-based semantic orientation. Sentiment analysis in Arabic tweets was also explored in [Baly et al. 2017], where a SVM model was trained with a variety of features including character, word and lemma *n*-grams, counts of positive and negative words, part-of-speech tags, and stylistic features such as counts of punctuation. To overcome the language complexity, word morphological features (voice, lemma, gender, etc.) were used along different feature sets, but could not improve performance [Abdul-Mageed et al. 2011; Refaee and Rieser 2014], whereas lexemes and lemmas were found useful in [Abdul-Mageed et al. 2014]. Machine translation was used to translate text to English and then apply state-of-the-art English models, namely the NRC system [Kiritchenko et al. 2014] and the RNTN [Socher et al. 2013], which both exhibited slight accuracy drop due to loss of information in translation [Refaee and Rieser 2015; Salameh et al. 2015]. However, this approach is considered an efficient alternative to building sentiment models in complex low-resource languages. Finally, the RAE model [Socher et al. 2011] was applied to Arabic, and outperformed several machine learning and neural network models that are trained with bag-of-words [Al Sallab et al. 2015].

Previous sentiment models in Arabic did not fully tackle the morphological complexity of the language. One effort that comes close to this scope evaluated lemma versus lexeme abstraction, but did not cover the full space of Arabic morphology [Abdul-Mageed et al. 2014]. Also, due to the lack of lexical resources with fine-grained sentiment annotations, previous Arabic sentiment models were only evaluated at the sentence and document-level, although sentiment is expressed in subtle manners and should be modeled at lower levels including words and phrases.

3. ARABIC PROCESSING CHALLENGES AND SOLUTIONS

In this section, we describe the challenges and solutions associated with natural language processing of Arabic. While Arabic has many challenges for NLP in general, as discussed in [Habash 2010], we present and highlight here the challenges specific to the problem of sentiment extraction from sentences.

3.1. Arabic Processing Challenges

Morphological Richness. Arabic is a morphologically rich and complex language where multiple affixes and clitics can be attached in different ways to the words they modify. This leads to a higher degree of sparsity than other non-MRLs such as English: e.g., given an Arabic-English parallel corpus, the number ‘white-space-separated’ tokens in Arabic is 20% less than that in English, whereas the number of unique Arabic word forms is two times greater [El Kholy and Habash 2012]. For example, the word وسيكاتبونها _wa+sa+yu-kAtib-uwna+hA_ corresponds to the phrase: ‘and they will correspond with her’. This example includes two proclitics and one enclitic as well as a prefix and a suffix, and the word is an example of the stem _kAtib_ and the lemma _kAtab_ ‘to correspond’. The word is said to be an inflected form of the lemma, which itself is derived from the root _k-t-b_ ‘writing-related’. Arabic has a relatively-small number of roots that derive a large number of lemmas, which inflect into a larger number of words. We combine in the sense of _word inflection_, both orthographic cliticization and inflectional morphology. In the context of sentiment analysis, we expect that inflectional variants sharing the same lemma or stem will maintain the same core meaning and sentiment: كتب _ktb_ ‘he wrote’, يكتب _yktb_ ‘he writes’, سيكتب _syktb_ ‘he will write’, وسيكتب _wsyktb_ ‘and he will write’, and so on. It is not necessary apparent that derivational variants sharing the same root will carry the same sentiment, however. Regardless, we expect that models trained using raw Arabic text will suffer from high sparsity.

Orthographic Ambiguity. In Arabic orthography, diacritization, which primarily marks short vowels and consonantal doubling, is optional. This is the primary cause of Arabic’s notoriously high ambiguity rate: according to [Shahrour et al. 2016], the Standard Arabic Morphological Analyzer (SAMA) [Maamouri et al. 2010] produces an average of 12.8 analyzes and 2.7 lemmas per word, out of context. In many cases, words with identical forms can have different meanings with even different sentiment polarities, e.g., the undiacritized word عذب _E*b_ can be interpreted as عَذَّبَ _Ea*~aba_ ‘he tortured’ and عَذب _Ea*obo_ ‘sweet’. Another example that involves different interpretations of word tokenization complicated by dropped diacritics is the word بشر _b$r_, which can be interpreted as بَشَر _ba$ar_ ‘human’ (neutral), بَشَّرَ _ba$~ara_ ‘he delivered good news’ (positive), and بِشَرٍّ _bi+$ar~K_ ‘with evil’ (negative).

Without contextually-correct tokenization, machine learning models will be unable to model morpheme-level semantic and sentiment interactions.

3.2. Arabic Processing Solutions

Morphological Analysis and Disambiguation. To overcome the above-mentioned challenges of Arabic richness and ambiguity, the common solution in Arabic NLP is to perform automatic in-context morphological analysis and disambiguation. The goal of this process is to identify for each word in-context the exact analysis, which is represented as a set of inflectional features (voice, gender, part-of-speech, etc.), different levels of morphological tokenization and abstraction (lemmas, morphemes, stems, roots, etc.), as well as diacritics.

Morphological Feature Abstraction. Of course, disambiguation, while reducing ambiguity does not address the question of sparsity. This is why it is common to see efforts exploring the use of different morphological features in machine learning models [Habash 2010; Abdul-Mageed et al. 2014; Abdul-Mageed et al. 2011]. These features include:

(1) The **stem** is defined as the part of the word that remains after deleting all clitics and affixes.
(2) The **lemma** is a conventionalized choice of one word to represent the set of all words related by inflectional (and not derivational) morphology.
(3) The **root**, in Arabic and other Semitic languages, is typically a sequence of three (sometimes four or five) consonantal radicals that abstracts away from all inflectional and derivational morphology [Habash 2010]. Roots are often associated with highly abstract - and sometimes idiosyncratic - meanings that are shared in specific ways by specific lemmas, which may realize to the surface in different stems.
(4) **ATB-tokens** refer to the result of performing morphological tokenization based on the Arabic Treebank (ATB) scheme [Habash and Sadat 2006]. The ATB-tokens differ from stems by the fact that they split-off from the base word all clitics (except the definite article), whereas stemming removes all clitics, suffixes and affixes. For instance, the word وبحسناتهم *wbHsnAthm* reduces to the following stem: حسن *Hsn*, whereas its morphologically segmentation reduces to the 'base' word حسنات *HsnAt* and the free morphemes و, ب and هم. Therefore, ATB-tokens are slightly less abstract than stems given that not all free morphemes are chopped-off from the base word. Finally, although ATB-tokens produce base words that are close to stems, and to a less extent to lemmas, we evaluate the ATB-token features to identify the impact of the free morphemes (clitics) on the composition model for sentiment prediction.

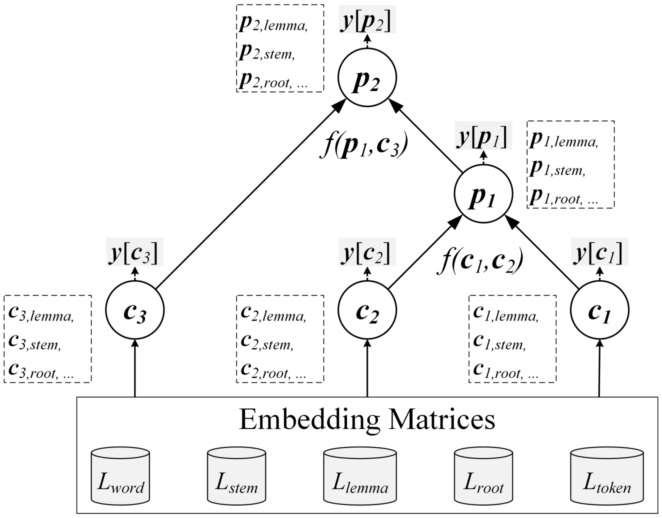
Naturally, while these abstract forms reduce sparsity, they also introduce ambiguity. Roots are more ambiguous than lemmas and stems; and lemmas are more general than stems. For example, while the lemma ملحمة *mlHmp* means 'epic', 'butchery' or 'soldering shop', its root لحم *lHm* includes additional meanings such as لحم*laHm* 'meat', لِحَام*liHAm* 'soldering', لَحَّام*laH~Am* 'butcher', إِلتحَام*AiltiHAm* 'cohesion', لَحمِيَّة*laHmiy~ap* 'adenoids', etc. Some stems may be similar for different lemmas, which cause added noise. For example, قَائِم *qA}im* is the stem from a number of words that belong to two lemmas: قَائِم *qA}m* 'existing' and قَائِمة *qA}mp* 'list'.

In the context of our work, it is important to determine the level of morphology that achieves the best trade-off between model sparsity and ambiguity for the task of Arabic sentiment analysis.

3.3. RNTN Integration with Arabic Processing for Sentiment Analysis

The Recursive Neural Tensor Networks (RNTN) model has proven successful at sentiment analysis in English [Socher et al. 2013]. However, we argue that it will not yield similar performances when applied to Arabic, mainly due to the morphological richness, ambiguity and sparsity issues associated with Arabic language, as discussed in Subsection 3.1. To overcome these challenges, we perform morphological disambiguation to represent the text with multiple levels of morphological abstraction including words, ATB tokens, stems, lemmas and roots, as discussed in Subsection 3.2. We also incorporate orthographic features including diacritization to reduce ambiguity and marking letter repetitions to use them as sentiment indicators. While handling Arabic morphology in the context of sentiment analysis is not new, we present the first attempt to incorporate morphology features into recursive deep models by developing the first morphologically enriched sentiment treebank (ARSENTB), which will be discussed later in Section 4.

Overall, RNTN aims to encode the sentiment and semantics of variable-length sentences into fixed-size vectors that can be used for sentiment classification. Figure 1 illustrates how RNTN is used to predict the sentiment of a three-word sentence $\{c_1, c_2, c_3\}$, while being integrated with Arabic processing to address the Arabic-specific challenges and improve sentiment analysis in Arabic.



p2,lemma, p2,stem, p2,root, ...
y[p2]
p2
f(p1,c3)
y[p1]
p1,lemma, p1,stem, p1,root, ...
p1
y[c3]
y[c2]
f(c1,c2)
y[c1]
c3,lemma, c3,stem, c3,root, ...
c3
c2,lemma, c2,stem, c2,root, ...
c2
c1,lemma, c1,stem, c1,root, ...
c1
Embedding Matrices
Lword
Lstem
Llemma
Lroot
Ltoken

Fig. 1: Integration of RNTN with Arabic Processing for improved sentiment analysis in Arabic.

Each sentence is represented with a binary parse tree whose nodes correspond to constituents ranging from words (leaf nodes) to full sentences (the root node). Each word c_i is represented with a d-dimensional vector $\boldsymbol{c}_i$ that can be either randomly-initialized or pre-trained using embedding models capturing distributional syntactic and semantic aspects of each word [Collobert and Weston 2008; Mikolov et al. 2013; Pennington et al. 2014]. All word vectors are stacked in a word embedding matrix $L \in \mathbb{R}^{d \times |V|}$, where d is the embedding size and $|V|$ is the vocabulary size. For Arabic, the nodes in the parse trees are enriched with different variations of the raw words they originally contain, each form reflecting particular morphological and orthographic features that we use to improve sentiment analysis in Arabic. An embedding matrix $L_{\text{feature}} \in \mathbb{R}^{d \times |V|_{\text{feature}}}$ can also be either randomly initialized or pre-trained for each of the different features, where $|V|_{\text{feature}}$ is the vocabulary size assuming a particular feature. For instance, increasing the level of morphological abstraction reduces the size of the corresponding vocabulary, i.e., $|V|_{\text{word}} < |V|_{\text{stem}} < |V|_{\text{root}}$. Consequently, each node of the tree can be represented with multiple vectors, each corresponding to the feature being used to train and evaluate RNTN. For example, $\boldsymbol{c}_{1,\text{lemma}}$ is the embedding of the first word's (c_1) lemma, and $\boldsymbol{c}_{3,\text{stem}}$ is the embedding of the third word's (c_3) stem. To train RNTN for sentiment analysis in Arabic, we first define the combination of orthographic and morphological features that will be used to represent the input text, and then select the corresponding embeddings for each node to perform composition. For each tree, a tensor-based composition function is applied recursively, in a bottom-up fashion. At each recursion, the composition function takes two input vectors (child nodes) and produces an output vector (parent node). This process continues until we obtain the vector representation of the root node corresponding to the full sentence. Equation (1) shows how the first composition in Figure 1 is performed at the stem-level.

$$\boldsymbol{p}_{1,\text{stem}} = f\left(\boldsymbol{c}_{1,\text{stem}}, \boldsymbol{c}_{2,\text{stem}}\right) = f\left(\begin{bmatrix}\boldsymbol{c}_{1,\text{stem}}\\ \boldsymbol{c}_{2,\text{stem}}\end{bmatrix}^T V \begin{bmatrix}\boldsymbol{c}_{1,\text{stem}}\\ \boldsymbol{c}_{2,\text{stem}}\end{bmatrix} + W \begin{bmatrix}\boldsymbol{c}_{1,\text{stem}}\\ \boldsymbol{c}_{2,\text{stem}}\end{bmatrix}\right) \tag{1}$$

where $\boldsymbol{c}_{1,\text{stem}}, \boldsymbol{c}_{2,\text{stem}} \in \mathbb{R}^d$ are the input vectors corresponding to the stems of the words c_1 and c_2, $\boldsymbol{p}_{1,\text{stem}} \in \mathbb{R}^d$ is the output vector corresponding to the stem of the union of (c_1, c_2), f is the an element-wise nonlinearity activation function (usually sigmoid or tanh), and $W \in \mathbb{R}^{d\times 2d}$ and $V \in \mathbb{R}^{2d\times 2d\times d}$ are the composition parameters that are learned during training, where V is a tensor matrix that is composed of d slices, where each slice $V^{[i]} \in \mathbb{R}2d \times 2d$ captures a specific type of composition. The same function is then used to derive the output vector $\boldsymbol{p}_{2,\text{stem}}$ given input vectors $\boldsymbol{c}_{3,\text{stem}}$ and $\boldsymbol{p}_{1,\text{stem}}$, as shown in Figure 1.

Given K sentiment classes, a logistic regression “softmax” classifier is trained using the vector of each node $\boldsymbol{c}_{i,feature}$ to produce a K-dimensional sentiment distribution $\boldsymbol{y}[\boldsymbol{c}_{i,\text{feature}}]$, as shown in Equation (2), where the k^{th} element in $\boldsymbol{y}[\boldsymbol{c}_{i,\text{feature}}]$ corresponds to the probability of the k^{th} sentiment class given $\boldsymbol{c}_{i,\text{feature}}$.

$$\boldsymbol{y}[\boldsymbol{c}_{i,\text{feature}}] = \text{softmax}\left(W_s \cdot \boldsymbol{c}_{i,\text{feature}}\right) \tag{2}$$

where $W_s \in \mathbb{R}^{K\times d}$ is the sentiment classification matrix. RNTN is trained to minimize the cross-entropy error between the predicted sentiment distribution $\boldsymbol{y}[\cdot]$ and the target sentiment distribution $\boldsymbol{t}[\cdot]$ for all nodes of the sentence tree. Equation (3) shows the objective function in terms of the RNTN parameters $\theta = (L, V, W, W_s)$.

$$J(\theta) = \min_{\theta} \sum_{i=1}^{N}\sum_{j=1}^{K} \boldsymbol{t}[\boldsymbol{c}_{i,\text{feature}}]_j \log \boldsymbol{y}[\boldsymbol{c}_{i,\text{feature}}]_j + \lambda \parallel \theta \parallel^2 \tag{3}$$

where $\boldsymbol{c}_{i,\text{feature}}$ is the vector of the i^{th} node in the tree corresponding to a particular morphological and orthographic feature, N is the number of nodes in the tree, K is the number of sentiment classes and λ is a regularization parameter. This function is minimized using the AdaGrad algorithm [Duchi et al. 2011].

Training RNTN requires a sentiment treebank, i.e., a collection of parse trees with sentiment annotations at all levels of constituency. RNTN was initially trained and evaluated for sentiment analysis in English using the Stanford sentiment treebank [Socher et al. 2013]. Further details on the model and the sentiment treebank are available in [Socher et al. 2013]. For sentiment analysis in Arabic using RNTN, we developed ARSENTB; the first sentiment treebank in Arabic with morphological and orthographic enrichment, which will be described in details in the next section.

4. THE ARABIC SENTIMENT TREEBANK

To address the Arabic-specific challenges described in Section 3, we developed ARSENTB; the first Arabic sentiment treebank that is enriched with sentiment, orthographic and morphological features to effectively model sentiment compositionality for accurate sentiment modeling in Arabic, taking into consideration the complexity of the language.

Although many Arabic opinion corpora have been published recently, there were no attempts to create extensive corpora with fine-grained sentiment annotations that support recursive deep sentiment models. In this section, we describe the system architecture to create ARSENTB and to enrich it with morphological and orthographic features. We also provide an intrinsic evaluation to the quality of the treebank.

4.1. Treebank System Architecture

Creating sentiment treebanks in MRLs is not straightforward. For instance, generating the parse trees requires automatic preprocessing such as normalization, tokenization and parsing, which are prone to errors. Therefore, annotators may end up annotating "corrupted" text that do not reflect the intended sentiment. Also, given our goal of annotating all phrases in the treebank, we need to provide annotators with very clear guidelines and instructions to ensure a proper handling of special cases such as phrases that lack context or weird phrases that are obtained because of parsing errors, and so on.

Figure 2 describes the architecture of our proposed system to accurately and effectively create ArSenTB, while addressing the above-mentioned treebank-related challenges. This system is composed of the following components: morphological disambiguation, syntactic parsing, tree detokenization, sentiment annotation and morphological enrichment.

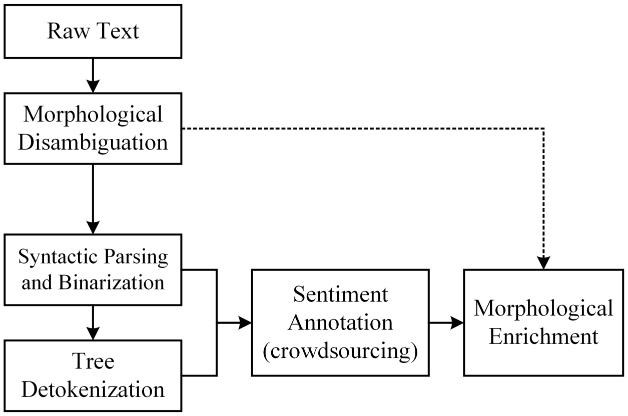

Raw Text
Morphological Disambiguation
Syntactic Parsing and Binarization
Tree Detokenization
Sentiment Annotation (crowdsourcing)
Morphological Enrichment

Fig. 2: Steps to develop the ArSenTB.

Morphological Disambiguation. We perform morphological analysis and disambiguation to extract in-context morphological features of the words. In particular, we focus on extracting the stem and lemma features, as well as the predicted diacritics. These features will be used later on to enrich the resulting treebank. We also perform Alef/Ya normalization and morphological tokenization, based on the ATB scheme, which is the required input format to typical phrase structure parsers. It is worth noting that character repetitions (elongations) were normalized as part of the text preprocessing prior to morphological disambiguation. However, since elongations are useful for sentiment analysis [Mourad and Darwish 2013], we preserve this information by extracting marking words that originally-contained elongation.

Parsing and Binarization. The ATB-tokenized text is fed to the Stanford parser [Green and Manning 2010] to generate phrase structure parse trees. These trees are not necessarily binary, and cannot be used to train recursive models that require inputs and outputs with consistent dimensionalities. Therefore, we use left-factoring to transform the trees to the Chomsky Normal Form (CNF) grammar that only contains unary and binary production rules. The choice of left (vs. right) was made such that sentiment composition follows the same direction readers follow to combine words when reading. After collapsing unary productions, we obtain binary trees that can be used to train recursive models.

Tree De-Tokenization. As mentioned earlier, tokenization and parsing performed so far may produce errors that affect the quality and readability of phrases to be annotated. In order to provide annotators with phrases that look as much as possible similar to the original surface, we perform tree 'de-tokenization' to obtain trees that represent raw untokenized text. De-tokenization is done by merging leaf nodes corresponding to tokens of the same word, while preserving the trees' binary structure. As a result, annotators will only see the raw text in phrases that are determined by a parser that ran on tokenized form.

Sentiment Annotation. We assign a sentiment label to each node in the treebank to train sentiment models at all constituency levels. We designed a crowd-sourcing sentiment annotation task using CrowdFlower. Before running the full task, we conducted a pilot study to tune the quality settings and get feedback on the clarity of the guidelines. Annotators were asked to assign each text one of five labels {*very negative, negative, neutral, positive, very positive*}. They were instructed not to be biased by their personal opinions, but rather reflect the authors' opinions, similar to [Mohammad 2016]. They were also asked to pay attention to linguistic phenomena such as elongation, emoticons and use of dialects. Annotators were provided with phrases randomly sampled from the treebank, so they are not affected by additional context. The model should be able to capture how different contexts affect sentiment.

Morphological Enrichment. Our solution to improve recursive deep sentiment models is to incorporate morphological and orthographic information. This is achieved by enriching ARSENTB with the in-context morphological features that were extracted at an earlier stage. These features include the ATB-tokens, stems and lemmas (each with and without diacritics). We also extract roots using lemma-root lookup tables for verbs and nouns [Habash and Rambow 2006; Altantawy et al. 2010]. Also, elongation markers that were extracted during preprocessing are also used to enrich the treebank. Each of these features is mapped into its appropriate node in the de-tokenized trees. Figure 3 illustrates a sample of an enriched sentiment tree.

4.2. Description and Evaluation of ARSENTB

The corpus we used to generate ARSENTB consists of 1,177 comments sampled by [Farra et al. 2015] from the Qatar Arabic Language Bank (QALB); a corpus of online comments on Al-Jazeera articles [Mohit et al. 2014].

Parsing the 1,177 comments produced a total of 123,242 nodes that correspond to phrases ranging from words (leaves) to full comments (roots). Since we want annotation to be performed out-of-context annotation, we do not need multiple annotations of the same phrase, even if it appears in different contexts. Therefore, we compiled a list of 78,879 unique phrases to be randomly distributed to annotators. Each phrase is independently annotated by 3 to 5 annotators, and annotations are aggregated based on majority. To resolve cases with tied votes, we follow a two-step approach to resolve cases with tied votes. First, we back-off from five to three sentiment classes {*negative, neutral, positive*} to match annotations with same polarity and regardless of their intensities. Then, if votes remain tied, we assign the text to an additional annotator to break the tie.

Table II shows the sentiment distribution in ARSENTB across the sets of unique phrases, all phrases and all comments (roots). We also observed that short phrases tend to be neutral as they mostly consist of words that need more context to become expressive, whereas stronger sentiment, particularly negative, builds up in longer phrases.

Figure 4 illustrates the normalized sentiment distribution at different levels of *n*-grams in ARSENTB. It can be observed that shorter phrases tend to be neutral as they mostly consist of words that need further context to become expressive, whereas stronger sentiment, particularly negative, builds up in longer phrases.

A sentiment composition takes sentiments of two phrases and produces the sentiment of their union. Table III illustrates the different types of sentiment compositions that are observed in ARSENTB. For each type, it shows the sentiment distribution of its output.

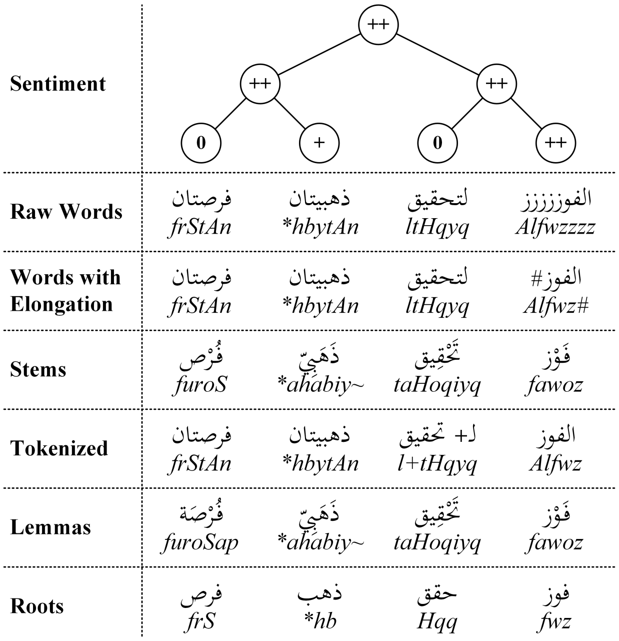


Fig. 3: Sentiment tree for the phrase: *frStAn ∗hbytAn ltHqyq Alfwzzzz* ‘two golden chances to winnnn’, enriched with morphological and orthographic features.

Table II: Sentiment distribution in ARSENTB

Sentiment	unique phrases	all phrases	comments
very negative	4.2%	3%	13.4%
negative	27.4%	17.7%	49.3%
neutral	47.8%	61.3%	3.7%
positive	19.5%	16%	25.5%
very positive	1.1%	2%	8.1%

Table III: Different types of sentiment compositions observed in ARSENTB

Type of composition	Sentiment distribution of Output			count
	positive	negative	neutral	
positive + positive	90.3%	3.8%	5.9%	2,874
positive + negative	26.4%	68.0%	5.6%	2,894
positive + neutral	68.9%	7.6%	23.5%	13,385
negative + negative	4.8%	93.0%	2.2%	3,753
negative + neutral	5.6%	84.3%	10.1%	14,460
neutral + neutral	9.7%	7.7%	82.7%	23,488

We can observe from Table III that the output of combining neutral with subjective phrases tends to be the same as that of the subjective phrase with a probability of 84% for negative and 69% for positive. Also, in more than 90% of the cases, the sentiment remains intact when combining subjective phrases of identical sentiment polarity. However, patterns become less obvious when combining phrases with different sentiments, where the output tends to be negative in 68% of the

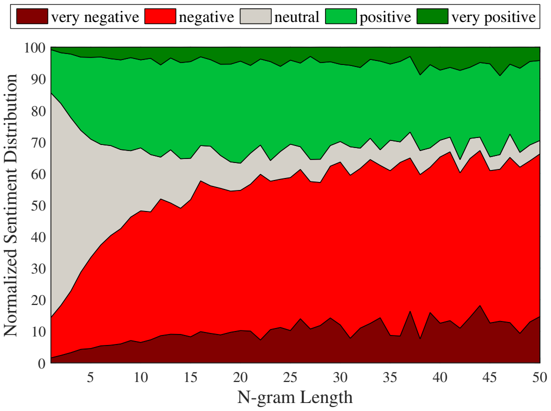


Fig. 4: Normalized distribution of aggregated sentiment labels at each level of *n*-gram in ArSenTB.

cases. Table IV illustrates some examples, from ArSenTB, of interesting sentiment compositions that may not be apparent at the first glance.

Table IV: Samples from ArSenTB that correspond to rare sentiment compositions.

composition	example
positive + positive → negative	[(الشباب المخلصين)$^+$ (يتم الاستغناء عنهم للصالح العام)$^+$]$^-$ [(*the loyal young people*)$^+$ (*have been laid off for the public interest*)$^+$]$^-$
positive + neutral → negative	[(ويلزم ٤ - ٦ اشهر)^0 (لاستخراج لقاح لهذا الداء)$^+$]$^-$ [(*developing a cure to this disease*)$^+$ (*requires 4-6 months*)^0]$^-$
neutral + neutral → negative	[(عدت الي ويندوز ٧)^0 (بعد ان قمت بالترقية الي ويندوز ٨)^0]$^-$ [(*I downgraded to windows 7*)^0 (*after upgrading to windows 8*)^0]$^-$

We used two methods to evaluate annotation quality. The first method analyzes ‘per-phrase’ agreement by measuring, for each phrase, how well the annotators agreed on its sentiment. We observed that 47.9% of phrases had full agreement among their annotators, and that 88% had a (>50%) agreement, which allows proper majority aggregation. For the remaining 12% with tied votes, backing-off from five to three classes resolved most of these cases, and the remaining cases were assigned to an independent annotator to break the tie. These statistics reflect the clarity of the guidelines and resulting annotations.

The second method is based on calculating inter-annotator agreement (IAA) statistics on a set of 600 phrases uniformly sampled from the treebank. This set was annotated by one of the authors and was compared to the outcome of aggregating the labels produced by CrowdFlower annotators. 87% of the phrases had the identical sentiment labels, and 90% of the phrases agreed in their sentiment polarities regardless of its intensity. We calculated the kappa statistics to measure the proportion of agreement above what would be expected by chance [Cohen 1960]. We obtained a linear κ equals to 77%, where all labels were assumed equally-important. We also calculated the weighted κ to account for the degree of disagreement among annotators [Cohen 1968]. For instance, a disagreement between “very positive” and “very negative” is stronger than that between “neutral” and “negative”. We predefined a $[5 \times 5]$ table of weights W that measures the degree of disagreement between both annotators (the output of CrowdFlower and the author). Weights ranged between $[0, 1]$, where 0 corresponds to an exact match and 1 corresponds to the most extreme disagreement, where both annotators assign “very negative” and “very positive” labels to the same text. The weighted κ is calculated using the formula shown in Equation (4).

$$\kappa_w = 1 - \frac{\sum_{i,j} w_{i,j} p_{i,j}}{\sum_{i,j} w_{i,j} e_{i,j}} \tag{4}$$

where $w_{i,j}$ is the weight of disagreement between class i and class j. $p_{i,j}$ and $e_{i,j}$ are the actual and expected probabilities that annotator A assigns class i while annotator B assigns class j, respectively. We obtained a weighted κ equals to 83%, and the increase compared to the 77%, in linear κ, indicates that most disagreements were not severe.

The κ agreement numbers indicate excellent levels of agreement according to [Fleiss et al. 2013]. Consequently, the sentiment labels in ArSenTB are robust in terms of 1) agreement among CrowdFlower annotators, and 2) agreement between their aggregated labels and the authors’ expectations.

5. EXPERIMENTS AND RESULTS

In this section, we evaluate the performance of RNTN for Arabic sentiment analysis using ArSenTB. We highlight the improvements achieved by the morphological and orthographic features. We also compare to similar improvements achieved on well-known classifiers.

5.1. Experimental Setup

The input to RNTN is a table of word embeddings derived using word2vec by training a CBOW model [Mikolov et al. 2013] on the full corpus of QALB that contains 550,273 comments on Al-Jazeera articles. We used MADAMIRA, the state-of-the-art morphological analyzer and disambiguator in Arabic [Pasha et al. 2014] that extracts the required orthographic and morphological features at a high degree of accuracy. Previous efforts used elongation as an explicit feature to train machine learning models [Mourad and Darwish 2013; Kiritchenko et al. 2014], whereas in this paper we incorporate the impact of elongation by learning word embeddings from a version of QALB in which words with elongation (e.g., هدفـفـف *hadafff* ‘goalll’) are marked to be distinguished from same word-forms with non elongation (e.g., هدف *hadaf* ‘goal’). Finally, different embeddings were learned for each combination of morphological abstraction (word, ATB token, lemma, stem, root) and orthographic representation (diacritized/undiacritized, elongation normalized/marked).

Table V: Evaluation splits of ArSenTB.

	Size	Comments #	Phrases #
Train	70%	823	85,622
Dev	10%	118	12,518
Test	20%	235	24,991

We created the {*train, dev, test*} splits by uniformly sampling from ArSenTB as illustrated in Table V. We used the *train* and the *dev* sets to tune the word embedding size and the learning rate. The model was then evaluated on the *test* set using the parameters that achieved the best performance in the tuning stage. Performance was quantified using accuracy and weighted F1-score, where the weights are determined by the percentage of each class in the *test* set. The model was evaluated at both phrase and comment-level, where phrases correspond to all nodes in the treebank and comments correspond to root nodes only.

5.2. Results

We compare our solution to the majority baseline, which automatically assigns the most frequent class in the *train* set to all instances in the *test* set. We also compare against other well-known sentiment classifiers from the literature including SVM, the Recursive Auto Encoders (RAE) and the Long Short-Term Memory (LSTM). The SVM classifier that is trained with word *n*grams has been successful at sentiment analysis in both Arabic and English languages. We trained SVM using word *n*-grams, with different lengths (*n*). Preliminary experiments showed that word *bi*-grams are better than *uni*-grams and *tri*-grams, and hence we report only results for *bi*-grams.

RAE differs from RNTN in the following aspects. RNTN performs composition using a tensor-based function, whereas RAE is based on minimizing the encoder/decoder reconstruction error [Socher et al. 2011]. Second, the RNTN model trains a softmax classifier on top of each node of the parse tree, whereas the RAE model trains a softmax layer on top of the root node only, hence performs sentiment classification at the sentence-level only. In our experiments, we used the RAE model with the setup that achieved best performance for sentiment analysis in Arabic, according to [Al Sallab et al. 2017]. The input text is morphologically-tokenized, and the word embeddings are formed by concatenating embeddings learned using the C&W model [Collobert and Weston 2008] with embeddings learned using a sentiment embedding model [Al Sallab et al. 2017].

LSTM belongs to the class of recurrent neural networks, where at each time step the current word is combined with the output representation of the preceding words using a structured cell of input, output, forget and memory gates. These gates help capturing long and short-term dependencies between the current word and previous information. To train LSTM for sentiment classification in sentences with variable lengths, all sentences are padded so they all become of a length equal to *max length*. Then, the LSTM cell is recurrently applied to the padded sentences, and the output at the step where padding starts is used to train the softmax classifier.

To ensure a fair comparison with RNTN, we also trained the SVM, RAE and LSTM models with the same levels of morphological abstraction (word, stem, lemma and root). Table VI illustrates the performances of the different models for five-way and three-way sentiment analysis.

5.2.1. Baselines. First, we compare the basic RNTN model (Table VI, row 5) that is trained using raw words to the majority baseline (row 1), the basic SVM (row 2), the basic RAE (row 3) and the basic LSTM (row 4). Results indicate that the basic RNTN outperforms the majority classifier. It also outperforms the basic SVM at the phrase-level, but achieves lower results at the comment-level. While [Al Sallab et al. 2015] showed that RAE outperform SVM when applied to raw MSA sentences, we can notice a slight advantage of SVM over the deep learning models. This can be explained by the fact that the sentiment treebank contains mixtures of MSA, dialects and spelling errors, and hence does not necessarily comply with the grammatical rules that are used by the deep learning models to infer sentiment. At the phrase-level, the SVM models are mainly affected by the smaller number of words in a phrase, which increases the sparsity of the training features and hence affects the model's ability of learning. These results confirm the fact the different models do not achieve outstanding performances in Arabic as they did in English, which is mainly attributed to the complex morphology of Arabic. Next, we explore the value of enriching the RNTN with the orthographic and morphological features that are provided as part of ARSENTB.

5.2.2. Impact of Orthographic Features. Experimental results shown in Table VII illustrate the performance improvement introduced to the baseline RNTN by either marking or normalizing character repetition (elongation) and by adding diacritics. The baseline RNTN model is trained with raw unprocessed text as they originally appeared in the corpus.

It can be observed that marking elongation did not have much impact at the phrase-level, mainly due to its scarce occurrence, as it can be observed in only 0.15% of the words and 0.4% of the phrases in the treebank. However, elongations become more evident at the comment-level, and could be identified in 8% of the comments, leading to significant performance improvement. These observations confirm the importance of elongations as sentiment indicators. It can also be observed that

Table VI: Performance of the different RNTN models, and a comparison to SVM, RAE and LSTM classifiers. *Numbers in bold indicate best performance under each section: (1-5), (6-8), (9-13), (14-18) and (19-23). Numbers with underline indicate best performance across all classifiers.*

		Five Classes				Three Classes			
		Phrases		Comments		Phrases		Comments	
		Acc.	F1	Acc.	F1	Acc.	F1	Acc.	F1
1	Majority	60.0	45.8	47.2	31.6	60.0	45.8	63.0	48.5
2	basic SVM (raw words)	66.3	59.6	**52.8**	**46.3**	68.8	63.4	**71.9**	**69.7**
3	basic RAE (raw words)	–	–	50.2	42.8	–	–	69.8	67.5
4	basic LSTM (raw words)	–	–	51.4	43.1	–	–	70.4	67.7
5	basic RNTN (raw words)	**72.4**	**72.0**	52.3	43.4	**75.2**	**75.1**	70.6	68.2
6	SVM (stems)	72.7	68.2	55.3	48.2	75.6	72.7	72.7	71.3
7	SVM (lemmas)	74.2	68.8	56.2	48.6	76.3	73.5	**72.7**	70.9
8	SVM (roots)	**75.0**	**71.3**	**56.6**	**49.0**	**78.3**	**76.3**	72.6	**71.4**
9	RAE (words)	–	–	51.0	45.0	–	–	73.7	70.9
10	RAE (ATB tokens)	–	–	51.7	46.2	–	–	75.2	73.0
11	RAE (stems)	–	–	**58.0**	**48.8**	–	–	**78.0**	**75.8**
12	RAE (lemmas)	–	–	53.6	46.1	–	–	76.9	74.5
13	RAE (roots)	–	–	53.4	45.7	–	–	70.3	69.2
14	LSTM (words)	–	–	53.1	47.0	–	–	74.7	73.9
15	LSTM (ATB tokens)	–	–	53.7	47.2	–	–	76.2	76.0
16	LSTM (stems)	–	–	**59.2**	**49.9**	–	–	**78.4**	**77.4**
17	LSTM (lemmas)	–	–	55.4	48.1	–	–	77.5	74.9
18	LSTM (roots)	–	–	55.8	47.9	–	–	72.4	70.2
19	RNTN (words)	75.0	74.6	55.0	48.0	79.2	78.2	76.7	74.9
20	RNTN (ATB tokens)	77.2	74.2	55.7	48.2	80.1	78.5	79.2	77.8
21	RNTN (stems)	**79.6**	78.2	**60.0**	**51.8**	**83.4**	**83.1**	**80.0**	**79.0**
22	RNTN (lemmas)	79.4	78.3	56.6	49.1	83.3	82.9	77.9	75.5
23	RNTN (roots)	77.5	76.1	57.4	48.7	81.0	80.7	72.3	71.2

Table VII: The impact of adding orthographic features on the performance of baseline RNTN for five-way classification.

	Five Classes				Three Classes			
	Phrases		Comments		Phrases		Comments	
	Acc.	F1-score	Acc.	F1-score	Acc.	F1-score	Acc.	F1-score
Baseline RNTN (raw words)	72.4	72.0	52.3	43.4	75.2	75.1	70.6	68.2
Normalizing Elongation	73.4	73.2	53.0	46.9	77.9	77.9	71.5	70.2
Marking Elongation	73.5	73.4	54.5	48.0	78.2	77.8	74.7	73.9
Adding Diacritics	74.5	73.9	54.2	47.5	78.8	78.0	73.4	73.6
Marking Elongation & Diacritics	**75.0**	**74.6**	**55.0**	**48.0**	**79.2**	**78.2**	**76.7**	**74.9**

adding diacritics to the words consistently improved the performance at both phrase and comment levels. This observation confirms the value of diacritization at reducing ambiguity. Based on these observations, all results included in Table VI assume that diacritics are known and elongation is marked, except for rows (2-5) that correspond to baseline models trained with raw text.

5.2.3. Impact of Morphological Abstraction. We evaluated RNTN under several morphological abstractions: {*words, tokens, stems, lemmas, roots*} from most specific to most abstract. ArSenTB contains 20,459 unique words that correspond to 14,262 tokens, 9,842 stems, 8,836 lemmas and 4,669 roots. These numbers reflect the concept hierarchy in Arabic morphology, leading to better generalization as we climb through this hierarchy. However, as mentioned in Section 3, increasing the level of abstraction introduces lexical ambiguity. For instance, 20% of the unique roots in

ArSenTB represent words with different sentiments. This percentage drops to 12% and 11% for lemmas and stems, respectively.

Rows 19-23 show that abstracting away from raw words improves performance, with best results achieved with stems. However, going beyond that level of abstraction and operating at the root-level leads to a performance degradation, reflecting loss of semantic information due to over-generalized representations, especially with roots. This behavior is similar to the classical bias-variance dilemma, where we are trying to simultaneously minimize two sources of errors; lexical variety and ambiguity. The best trade-off was achieved at the stem level, achieving absolute average F1-score improvements (over the baseline RNTN) of 8% and 10.8% at the phrase and the comment level, respectively, on three-way sentiment classification. Similar improvements are also observed on the five-way classification task. It can also be observed that the performances are comparable between stems and lemmas at the phrase-level, but definitely better for stems at the comment-level. Finally, it is worth mentioning that the reported improvements were obtained through one iteration (or cycle) of training/testing. To calculate statistical significance of results, we repeated the experiments two more times on different training/testing splits. We obtained similar improvements that are statistically significant with 95% confidence at 2 degrees of freedom.

We also evaluated the impact of morphology on the SVM model. Rows 6-8 indicate that, similar to RNTN, increasing the level of abstraction improves performance compared to the baseline SVM. However, in this case, performance keeps improving and reaches its highest with roots. This performance behavior indicates that SVM benefits from morphological abstraction to mitigate the effects of *curse of dimensionality* and *sparsity* associated with the bag-of-words, whereas the semantics of these abstractions are better captured by RNTN. For instance, although roots are better than stems and lemmas at reducing lexical sparsity, they produce worse results because they over-generalize the words' semantics. The best RNTN model (stem-based) outperforms the best SVM model (root-based) by 6.8% and 7.6% absolute average F1-score at the phrase and the comment-level, respectively, on three-way sentiment classification. Similar improvements are also observed on the five-way classification task.

It can also be observed that, at the comment level, improvements due to morphology are more observed in RNTN than in SVM, suggesting that recursive deep models are able to better explore the morphology space to improve sentiment prediction.

Rows 9-13 and Rows 14-18 illustrate the performance of RAE and LSTM at the comment-level, because these models were initially proposed for sentence-level sentiment classification. Results indicate that the impact of morphology on RAE and LSTM is similar to that on RNTN; both models achieve best performance at the stem-level, while accuracy decreases at the root-level due to over-generalization. RNTN performs better than RAE, which is mainly due to the phrase-level sentiment prediction that helps modeling sentiment composition and its intricacies. RNTN also performs better than LSTM, although the latter has better properties in terms of modeling long and short-distance dependencies due to the "keep" and "forget" gates. This advantage is attributed to the order of composition, where tree-based recursive models are better at inferring semantics than sequential models [Mitchell and Lapata 2010; Tai et al. 2015]. It is worth mentioning that the advantage of RNTN comes at the expense of building a treebank with expensive fine-grained sentiment annotations, whereas RAE and LSTM can be trained on corpora with simple sentence-level annotations.

Results Analysis. Out of the 235 comments (trees) in the test set, the predictions of 37 comments was corrected when using the stem-based RNTN model instead of the basic (word-based) model. The main advantage and reason for abstracting away from raw words is to reduce lexical sparsity, and hence improve generalization. By inspecting the 37 comments at both the word and the stem levels, we observed that 70% of all unigrams and bigrams in the word-level comments already exist in the train split, i.e., are used to train the sentiment model. This coverage increases to 88% in the stem-level comments, indicating better generalization capabilities. Since abstraction comes with the risk of ambiguity, we also evaluated the levels of ambiguity associated with stems. We observed that in the 37 comments whose predictions were corrected by training RNTN with the stem features, only ∼4% of the stems correspond to words with different sentiments. However, in the 45 comments whose predictions remained incorrect even with the use of stems, ∼13% of stems correspond to words with different sentiments. Based on these observations, we can infer that morphological abstraction (stemming in particular) improves the model's generalization, especially when the abstract representation (e.g., the stem) shares similar meanings across the different words it represents.

On the other hand, out of the 235 comments in the test set, 23 comments were misclassified regardless of the level of morphological abstraction; 59% of these comments were positive and 41% were neutral. Table VIII illustrates two positive comments that were misclassified as negative. It can be observed that these examples contain a significant number of words with intensive negative semantics, which made it challenging for the models to classify them as positive.

Table VIII: Examples of positive comments misclassified by all RNTN models as negative.

علي الرغم من بطش النظام السادي، بابواقه وشبيحته إلا أن الثورة لمنتصرة وسيذهب هذا النظام إلي مزبلة التاريخ _ElY Alrgm mn bT$ AlnZAm AlfA$y AlsAdy, b>bwAqh w$byHth AlA >n Alvwrp lmntSrp wsy*hb h*A AlnZAm AlY mzblp AltAryx_
الذي لايساعد مصر في هذه المرحلة هو الخاسر لأن مصر ستّخطى هذه المحنه فمصر في تاريخها البعيد والقريب لم تمر بمحنة كهذه حتي عنما حاربت في أعوام ٥٦، ٧٣ _Al*Y lAysAEd mSr fY h*h AlmrHlp hw AlxAsr lAn mSr sttxTY h*h AlmHnh fmSr fY tAryxhA AlbEyd wAlqryb lm tmr bmHnp kh*h HtY EnmA HArbt fY AEwAm 56, 73_

5.2.4. Impact of morphology in English. We evaluated the impact of extending the use of morphological features to recursive deep models in English, which is a less complex language. For instance, on average, a lemma in the English Stanford sentiment treebank represents 1.25 words, whereas in ArSenTB it represents 2.3 words. To prove this evaluation, we trained RNTN on the Stanford sentiment treebank [Socher et al. 2013] that is enriched with stems and lemmas extracted using the Stanford CoreNLP toolkit [Manning et al. 2014]. Results in Table IX indicate that adding morphological features improved RNTN by only 1%, much less than what has been achieved in Arabic. Therefore, morphologically-complex languages would benefit more from incorporating features that reduce their complexity.

Table IX: Impact of morphology in English.

	Phrases		Comments	
	Accuracy	F1-score	Accuracy	F1-score
RNTN (raw words)	80.7	78.1	45.7	39.7
RNTN (stems)	**81.0**	**78.9**	**47.1**	**40.8**
RNTN (lemmas)	80.8	78.8	46.1	40.0

6. CONCLUSION

In this paper, we have presented several contributions to sentiment analysis in MRLs, considering Arabic as an example. We highlighted challenges that exist in Arabic, namely morphological richness, ambiguity and lexical sparsity, which affect the performance of sentiment analysis in Arabic. Then, we addressed these challenges by evaluating RNTN under several morphological abstractions of raw words to combat sparsity, by adding diacritics to reduce ambiguity and by marking letter repetition to serve as sentiment indicators. While morphological abstraction reduces the impact of lexical sparsity, it introduces semantic ambiguity. Therefore, we conducted a comparative analysis to identify which level of abstraction achieves the best results.

To conduct the required experiments, we introduced ARSENTB, the first Arabic sentiment treebank with morphological and orthographic enrichment that help addressing the morphology-related challenges in Arabic. We presented the system architecture to develop ARSENTB including morphological tokenization and disambiguation, syntactic parsing and binarization, and sentiment annotation through crowdsourcing along with the design of the annotation task. This resource should help expanding research on sentiment analysis in Arabic, since it represents the first Arabic corpus with both word-level and phrase-level sentiment annotations. Furthermore, ARSENTB is developed using online comments that can be short or long and can be either written in MSA or contain significant amount of noise, depending on the comment's author. Such data behaves similar to what would exist on the Internet, and hence the reported results can serve as indicators to the level of performance that should be expected in real sentiment analysis systems.

Experimental results showed that operating at the stem-level achieves the best sparsity/ambiguity trade-off, and that marking elongation and including diacritics further improved the performance. The best RNTN performance was achieved using stems with improvement of over 10% absolute average F1-score on three-way sentiment classification compared to basic RNTN. We also evaluated the impact of enriching RNTN with morphology features in English, and results showed only 1% improvement compared to Scoher's RNTN. This relatively-small improvement reflects the importance of morphological abstraction to improve sentiment classification in MRLs.

Future work will include using ARSENTB to augment and validate existing sentiment lexicons, and also to propose approaches to use multiple levels of abstraction together. Additionally, the approach can be adopted for dialectal Arabic, as well as other MRLs.

REFERENCES

2015. Internet World Stats: Internet World Users by Language. (2015). http://www.internetworldstats.com/.

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.

Muhammad Abdul-Mageed, Mona Diab, and Sandra Kübler. 2014. SAMAR: Subjectivity and sentiment analysis for Arabic social media. *Computer Speech & Language* 28, 1 (2014), 20–37.

Muhammad Abdul-Mageed and Mona T Diab. 2012. AWATIF: A Multi-Genre Corpus for Modern Standard Arabic Subjectivity and Sentiment Analysis.. In *LREC*. Citeseer, 3907–3914.

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.

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.

Ahmad A Al Sallab, Ramy Baly, Gilbert Badaro, Hazem Hajj, Wassim El Hajj, and Khaled B Shaban. forthcoming 2017. AROMA: A Recursive Deep Learning Model for Opinion Mining in Arabic as a Low Resource Language. *ACM Transactions on Asian and Low-Resource Language Information Processing* (forthcoming 2017).

Mohamed Altantawy, Nizar Habash, Owen Rambow, and Ibrahim Saleh. 2010. Morphological Analysis and Generation of Arabic Nouns: A Morphemic Functional Approach. In *Proceedings of the International Conference on Language Resources and Evaluation, LREC. Valletta, Malta*.

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).

Georgios Balikas and Massih-Reza Amini. 2016. TwiSE at SemEval-2016 Task 4: Twitter Sentiment Classification. *arXiv preprint arXiv:1606.04351* (2016).

Ramy Baly, Gilbert Badaro, Georges El-Khoury, Rawan Moukalled, Rita Aoun, Hazem Hajj, Wassim El-Hajj, Nizar Habash, and Khaled Bashir Shaban. 2017. A Characterization Study of Arabic Twitter Data with a Benchmarking for State-of-the-Art Opinion Mining Models. *WANLP 2017 (co-located with EACL 2017)* (2017), 110.

Ramy Baly, Roula Hobeica, Hazem Hajj, Wassim El-Hajj, Khaled Shaban, and Ahmad El-Sallab. 2016. A Meta-Framework for Modeling the Human Reading Process in Sentiment Analysis. *ACM Transactions on Information Systems (TOIS)* (2016).

J Cohen. 1960. A coefficient of agreement for nominal scales. Educational and Psychosocial Measurement, 20, 37-46. (1960).

Jacob Cohen. 1968. Weighted kappa: Nominal scale agreement provision for scaled disagreement or partial credit. *Psychological bulletin* 70, 4 (1968), 213.

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.

Jan Deriu, Maurice Gonzenbach, Fatih Uzdilli, Aurelien Lucchi, Valeria De Luca, and Martin Jaggi. 2016. SwissCheese at SemEval-2016 Task 4: Sentiment classification using an ensemble of convolutional neural networks with distant supervision. *Proceedings of SemEval* (2016), 1124–1128.

John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research* 12, Jul (2011), 2121–2159.

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.

Noura Farra, Elie Challita, Rawad Abou Assi, and Hazem Hajj. 2010. Sentence-level and document-level sentiment mining for arabic texts. In *Data Mining Workshops (ICDMW), 2010 IEEE International Conference on*. IEEE, 1114–1119.

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

Joseph L Fleiss, Bruce Levin, and Myunghee Cho Paik. 2013. *Statistical methods for rates and proportions*. John Wiley & Sons.

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. 2006. MAGEAD: a morphological analyzer and generator for the Arabic dialects. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 681–688.

Nizar Habash and Fatiha Sadat. 2006. Arabic preprocessing schemes for statistical machine translation. (2006).

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

Ozan Irsoy and Claire Cardie. 2014. Deep recursive neural networks for compositionality in language. In *Advances in Neural Information Processing Systems*. 2096–2104.

Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. 2014. A convolutional neural network for modelling sentences. *arXiv preprint arXiv:1404.2188* (2014).

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

Thang Luong, Richard Socher, and Christopher D Manning. 2013. Better Word Representations with Recursive Neural Networks for Morphology.. In *CoNLL*. 104–113.

Mohamed Maamouri, Dave Graff, Basma Bouziri, Sondos Krouna, Ann Bies, and Seth Kulick. 2010. Standard Arabic morphological analyzer (SAMA) version 3.1. *Linguistic Data Consortium, Catalog No.: LDC2010L01* (2010).

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP Natural Language Processing Toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*. 55–60. http://www.aclweb.org/anthology/P/P14/P14-5010

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

Jeff Mitchell and Mirella Lapata. 2010. Composition in distributional models of semantics. *Cognitive science* 34, 8 (2010), 1388–1429.

Saif M Mohammad. 2016. A practical guide to sentiment annotation: Challenges and solutions. In *Proceedings of the Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*.

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.

Ahmed Mourad and Kareem Darwish. 2013. Subjectivity and sentiment analysis of modern standard Arabic and Arabic microblogs. In *Proceedings of the 4th workshop on computational approaches to subjectivity, sentiment and social media analysis*. 55–64.

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.

Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd annual meeting on Association for Computational Linguistics*. Association for Computational Linguistics, 271.

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up?: sentiment classification using machine learning techniques. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*. Association for Computational Linguistics, 79–86.

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*. 1094–1101.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global Vectors for Word Representation.. In *EMNLP*, Vol. 14. 1532–43.

Eshrag Refaee and Verena Rieser. 2014. Subjectivity and sentiment analysis of Arabic Twitter feeds with limited resources. In *Proceedings of the Workshop on Free/Open-Source Arabic Corpora and Corpora Processing Tools*. 16.

Eshrag Refaee and Verena Rieser. 2015. Benchmarking Machine Translated Sentiment Analysis for Arabic Tweets. In *NAACL-HLT 2015 Student Research Workshop (SRW)*. 71.

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.

Mohammad Salameh, Saif M Mohammad, and Svetlana Kiritchenko. 2015. Sentiment after translation: A case-study on arabic social media posts. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 767–777.

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, Jeffrey Pennington, Eric H Huang, Andrew Y Ng, and Christopher D Manning. 2011. 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.

DS Tarasov. 2015. Deep recurrent neural networks for multiple language aspect-based sentiment analysis of user reviews. In *Proceedings of International Conference of Computational Linguistics and Intellectual Technologies Dialog-2015*, Vol. 2. 53–64.

Peter D Turney. 2002. Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews. In *Proceedings of the 40th annual meeting on association for computational linguistics*. Association for Computational Linguistics, 417–424.

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

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*. 649–657.