Arabic sentiment analysis using GCL-based architectures and a customized regularization function

Mustafa Mhamed a,c, Richard Sutcliffe b,a,*, Xia Sun a, Jun Feng a,*, Ephrem Afele Retta a

a School of Information Science and Technology, Northwest University, Xi’an 710127, China
b School of Computer Science and Electronic Engineering, University of Essex, Wivenhoe Park, Colchester CO4 3SQ, UK
c College of Information and Electrical Engineering, China Agricultural University, Beijing 100089, China

Article info
Article history:
Received 3 September 2022
Revised 18 April 2023
Accepted 22 April 2023
Available online 29 May 2023

Keywords:
Arabic sentiment analysis (ASA)
Natural language processing (NLP)
Custom regularization function (CRF)
Gated convolution long (GCL)

Abstract
Sentiment analysis aims to extract emotions from textual data; with the proliferation of various social media platforms and the flow of data, particularly in the Arabic language, significant challenges have arisen, necessitating the development of various frameworks to handle issues. In this paper, we firstly design an architecture called Gated Convolution Long (GCL) to perform Arabic Sentiment Analysis. GCL can overcome difficulties with lengthy sequence training samples, extracting the optimal features that help improve Arabic sentiment analysis performance for binary and multiple classifications. The proposed method trains and tests in various Arabic datasets; The results are better than the baselines in all cases. GCL includes a Custom Regularization Function (CRF), which improves the performance and optimizes the validation loss. We carry out an ablation study and investigate the effect of removing CRF. CRF is shown to make a difference of up to 5.10% (2C) and 4.12% (3C). Furthermore, we study the relationship between Modern Standard Arabic and five Arabic dialects via a cross-dialect training study. Finally, we apply GCL through standard regularization (GCL+L1, GCL+L2, and GCL+L1 elasticNet) and our L_new on two big Arabic sentiment datasets; GCL+L_new gave the highest results (92.53%) with less performance time.

© 2023 Karabuk University. Publishing services by Elsevier B.V. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

1. Introduction

Sentiment analysis is a form of natural language processing which detects the sentiment expressed in a text [1]. By 2025, it is estimated that sentiment analysis will be worth $3.8 billion [2], due to its many practical applications in business and politics. As a result, it has become a very active research field in recent years [3].

Initially, the majority of sentiment analysis research related to English text [4–9]. However, there has been a lot of interest in the Arabic language lately [10–14]. Furthermore, surveys have examined Arabic resources and strategies, in order to draw conclusions and identify difficulties associated with Arabic sentiment analysis [15–18]. This is not surprising, since Arabic is spoken by many people all over the world, is a significant language accepted by the United Nations [19], and is the fourth most popular language on the Internet [20].

The Arabic language comprises three classes, modern standard Arabic (MSA), dialect Arabic (DA), and classical Arabic (CA) [21]. MSA is used in official settings, including news reporting, educational institutions, and commercial forums. In contrast, Arabic dialects that vary from country to country are employed in casual writing, notably on social media. Classical Arabic is used in religious writings such as the Holy Qur’an and for prayer.

Deep Learning (DL) is an area of machine learning that deals with artificial neural networks, which are algorithms inspired by the structure and function of the brain [22]. Many DL techniques are now used in Arabic sentiment analysis systems. In particular, methods such as word embeddings, Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), Long short-term memory (LSTM), and Hierarchical Attention Networks (HAN) have been used with great success [23–25]. However, despite data indicating that increasingly hard tasks necessitate more complex structures [17], especially in Arabic sentiment analysis, more sophisticated approaches to address classification difficulties are few. The following are important aspects of this study:

https://doi.org/10.1016/j.jestch.2023.101433
2215-0986/© 2023 Karabuk University. Publishing services by Elsevier B.V.
This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).
1. We create an Arabic Sentiment Analysis (ASA) model which domain-independent. We test the proposed approach utilizing evaluations from various domains with a wide range of word relationships and assess the effectiveness of each dataset independently.

2. Utilizing the suggested strategy with thorough preprocessing techniques aids in the finest data cleaning, followed by the extraction of the best features that increase the effectiveness of the model.

3. Most earlier research lacked optimization of loss functions; our strategies, together with a customized loss function, concentrate on enhancing accuracy.

4. The proposed Custom Regularization Function (CRF) is more extensive than the standard, allowing us to optimize zone choices for our hyperparameter weights and so give the greatest features for future selection. Relative to current baselines, our technique offers the highest categorization performance and optimizes the performance through various loss functions.

5. A cross-dialect training study investigates the relationships between Modern Standard Arabic and five Arabic dialects, as follows:
   - An Arabic collection is chosen that only includes MSA vocabulary;
   - Dialect samples are selected that are readily accessible;
   - Baselines are generated via training and validation on MSA data;
   - Models are trained using the suggested approach on MSA data and validated through dialect datasets;
   - The results are analysed.

Regularization is a fundamental component of machine learning, especially deep learning, that allows for good generalization to unknown data even when trained on a small training set or with a poor optimization process. Loss functions are significant in every predictive method because they establish a goal to measure the approach's performance. There are several types that differ according to the tasks, whether in classification or regression. The parameters learned by the model are set by minimizing a given loss function.

Below are the main contributions:

- We propose a new architecture called Gated Convolution Long (GCL) for Arabic sentiment analysis. We address the issue of long training samples, extract the best features for binary and n-ary classification, and boost the effectiveness of ASA.
- We develop a custom regularization function (CRF), which helps to improve the performance of the proposed model.
- We perform an ablation study which demonstrates that the improved results are due to CRF.

- We conduct a comparison study with a standard loss function, and show that our custom regularization aids in optimizing the loss function's performance.
- We show that the proposed method offers the best classification performance relative to current baselines.
- We use the proposed method to investigate the link between sentiments in Modern Standard Arabic and those in five different Arabic dialects.
- Finally, we compare the proposed technique with the standard regularization function on very large Arabic datasets; our model incorporating CRF was more effective, performed better, and took less time.

The paper is organized as follows. Section 2 reviews previous work on sentiment analysis for Arabic. Section 3 outlines the proposed approach and model architecture. Section 4 presents our experiments, including preprocessing steps, experimental settings, baselines, results, and discussion. Finally, Section 5 is the conclusion and suggests future work.

2. Previous work

Arabic content has been significantly produced on websites and social media over the last ten years. On social media, opinions are freely expressed, making them an excellent source for trend analyses in various professional, commercial, and popular periodicals. See [54,55] for recent surveys of work in Arabic sentiment, addressing models, datasets, and results for significant modern research on the Arabic language. For contemporary deep learning methodologies and semi-supervision, refer to [56–58].

Current sentiment datasets for Arabic are shown in Table 1. As can be seen, five datasets are for Modern Standard Arabic (MSA) alone, seven are for Arabic dialects, Algerian (ALG), Jordanian (JOR), Lebanese (LEB), Levantine (LEV), Moroccan (MOR), Saudi (SAU), and Sudanese (SUD), while two combine MSA with Dialects (DIA). In this work, we will use AHSD, ArTwitter, BBN, and MASC, as will be described later.

Table 2 summarizes recent research on Arabic sentiment analysis, including the dataset used, the form of Arabic (MSA or dialect), the model, and the performance result. We will now review these works, starting with those using machine learning. After this we will discuss neural network approaches.

Tabii et al. [50] applied Naïve Bayes (NB) [59], Maximum Entropy [60], and Support Vector Machines (SVM) [61] on two datasets, the Moroccan Sentiment Analysis Corpus (MSAC) [30] and SemEval-2017 [62]. Among the individual classifiers the SVM was the best, and when they used ensemble classifiers they achieved the highest accuracy (83.45%).

Table 1
Arabic sentiment datasets.

<table>
<thead>
<tr>
<th>Datasets</th>
<th>Language</th>
<th>Source</th>
<th>Size</th>
<th>#Classes</th>
<th>Balanced</th>
</tr>
</thead>
<tbody>
<tr>
<td>AHSD [28]</td>
<td>MSA</td>
<td>Twitter</td>
<td>104 K</td>
<td>2</td>
<td>N</td>
</tr>
<tr>
<td>ArTwitter [29]</td>
<td>JOR</td>
<td>Twitter</td>
<td>179.55 K</td>
<td>2</td>
<td>Y</td>
</tr>
<tr>
<td>MASC [30]</td>
<td>MOR</td>
<td>Google Play, Twitter, Facebook</td>
<td>3.27 MB</td>
<td>2</td>
<td>N</td>
</tr>
<tr>
<td>BBN [31]</td>
<td>LEV</td>
<td>Website posts</td>
<td>207 K</td>
<td>3</td>
<td>N</td>
</tr>
<tr>
<td>SudSent2 [32]</td>
<td>SUD</td>
<td>Facebook, YouTube</td>
<td>344 K</td>
<td>2</td>
<td>Y</td>
</tr>
<tr>
<td>DzSenti2 [33]</td>
<td>ALG</td>
<td>Facebook</td>
<td>100 K</td>
<td>2</td>
<td>Y</td>
</tr>
<tr>
<td>AO [34]</td>
<td>MSA</td>
<td>Facebook</td>
<td>30.15 K</td>
<td>2</td>
<td>N</td>
</tr>
<tr>
<td>LD [35]</td>
<td>LEB</td>
<td>Google Maps, Zomato</td>
<td>31.32 K</td>
<td>2</td>
<td>N</td>
</tr>
<tr>
<td>ABD [29]</td>
<td>MSA</td>
<td>Twitter</td>
<td>20 K</td>
<td>2</td>
<td>Y</td>
</tr>
<tr>
<td>YT [36]</td>
<td>MSA</td>
<td>YouTube</td>
<td>70 K</td>
<td>2</td>
<td>N</td>
</tr>
<tr>
<td>HARD [37]</td>
<td>MSA</td>
<td>Book</td>
<td>54.36 MB</td>
<td>3</td>
<td>N</td>
</tr>
<tr>
<td>LABR [38]</td>
<td>MSA</td>
<td>Book</td>
<td>11.6 MB</td>
<td>3</td>
<td>N</td>
</tr>
<tr>
<td>ASTD [39]</td>
<td>MSA + DIA</td>
<td>Twitter</td>
<td>10 K</td>
<td>4</td>
<td>N</td>
</tr>
<tr>
<td>Shami-Senti [40]</td>
<td>DIA</td>
<td>Twitter</td>
<td>2.5 K</td>
<td>3</td>
<td>N</td>
</tr>
<tr>
<td>ArSentID-LE [41]</td>
<td>MSA + DIA</td>
<td>CrowdFlower platform</td>
<td>4 K</td>
<td>5</td>
<td>Y</td>
</tr>
</tbody>
</table>
Afooz et al. [36] compared their Ensemble model, which included XGBoost (XG), Gradient Boosting (GB), AdaBoost (ADA) and Random Forest (RF), with machine learning classifiers on Arabic text which was collected from YouTube comments. SVM, followed by Linear Regression (LR), had the best performance accuracy (77.00%).

Atoum and Nouman [51] used SVM and NB with n-gram vector selection (bigrams, unigrams, and trigrams), on Jordanian dialect tweets (JDT). Results showed that the SVM gave the higher accuracy in all cases (bigrams 74.00%, stemmed unigrams 82.10%, trigrams 76.00%). Al-Salman [52] developed a Discriminative Multinomial Bayes (DMNB) approach, then compared it with NB, SVM, K-Nearest Neighbor (KNN) [63], and Decision Trees [64] on an Arabic dataset, which consists of 2,000 Arabic tweets with two classes. DMNB had the best accuracy with 87.50%, better than the baselines. Al Omar et al. [35] used LR [65] on the Lebanon dialect which they collected from restaurants, shops, hotels, google, and Zomato. Their results indicated that the binary rating of negative feelings (P = 0.80, R = 0.80) is less than the positive (P = 0.88, R = 1.00). Salameh et al. [47] created a dataset of Levantine Arabic sentiment which they called the BBN Dataset, then applied their Linear systems, with a performance of 65.31%. El Beltagy et al. [49] also used the BBN Dataset, this time applying a Complement Naïve Bayes (CBNB) classifier, then compared it with NB, SVM, K-Nearest Neighbor (KNN) [63], and Decision Trees [64] on an Arabic dataset; accuracy was 92.61%, for the Syrian Corpus [73] accuracy was 85.28%, and for ArTwitter, it was 85.00%. Alayba et al. [28] applied LR, SVM, CNN, and CNN to the AHSD dataset. CNN gave the highest performance with accuracy 90.00%. Recently, Elfaik et al. [44] used a Bidirectional LSTM [75] model on several datasets: ASTD [39], ArTwitter, LABR, MPQA [76], Multi-Domain [77], and AHSD. Accuracies were 79.25%, 91.82%, 80.70%, 75.85%, 89.70%, and 92.61%, respectively.

In summary, for Arabic sentiment models using ML, Tabii et al. [50], Afooz et al. [36], and Al-Kabi et al. [34] use SVM, Atoum and Nouman [51] combine SVM with LR, Al-Salman [52] use DMNB, Al Omar et al. [35] [47] use LR, and Salameh et al. [49] use CNB.

Concerning the machine learning classifiers, we note that SVM was the most used, while LR achieved the highest performance. Generally, ML models have three problems. First, they are susceptible to noise; a small amount of incorrectly labeled samples can have a significant impact on performance. Second, selecting the perfect kernel is a difficult undertaking. Third, when the dataset is large, training is slow.

Regarding DL, CNNs were more applied, but Bi-LSTM showed the best accuracy among all the algorithms. For the CNNs, obstacles were the standard selection of the optimal architecture with normal hyperparameters for training, and poor preprocessing of the Arabic text, causing the data held in adjacent words not to be learned effectively, hence reducing the CNN’s capability to select the best features for prediction. For Bi-LSTM, i.e. using two LSTM cells, one for each direction, it is costly, and it took a long time to train the Arabic context. Generally, we note that the DL models are better than the ML classifiers.

### Table 2

<table>
<thead>
<tr>
<th>Paper</th>
<th>Dataset</th>
<th>Language</th>
<th>Model</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>[42]</td>
<td>AHSD (2C)</td>
<td>SAU</td>
<td>CNN-LSTM</td>
<td>88.10%</td>
</tr>
<tr>
<td>[28]</td>
<td>AHSD (2C)</td>
<td>SAU</td>
<td>CNN</td>
<td>90.00%</td>
</tr>
<tr>
<td>[43]</td>
<td>AHSD (2C)</td>
<td>SAU</td>
<td>CNN + Word2Vec</td>
<td>92.00%</td>
</tr>
<tr>
<td>[44]</td>
<td>AHSD (2C)</td>
<td>SAU</td>
<td>BLSTM</td>
<td>92.61%</td>
</tr>
<tr>
<td>[45]</td>
<td>ArTwitter (2C)</td>
<td>JOR</td>
<td>RNN</td>
<td>85.00%</td>
</tr>
<tr>
<td>[46]</td>
<td>ArTwitter (2C)</td>
<td>JOR</td>
<td>LSTM</td>
<td>87.27%</td>
</tr>
<tr>
<td>[47]</td>
<td>BBN (3C)</td>
<td>LEV</td>
<td>Linear Classifier</td>
<td>65.31%</td>
</tr>
<tr>
<td>[48]</td>
<td>BBN (3C)</td>
<td>LEV</td>
<td>CNN</td>
<td>66.67%</td>
</tr>
<tr>
<td>[49]</td>
<td>BBN (3C)</td>
<td>LEV</td>
<td>CNB</td>
<td>71.06%</td>
</tr>
<tr>
<td>[46]</td>
<td>Web-crawled (2C)</td>
<td>MSA</td>
<td>CNN</td>
<td>85.01%</td>
</tr>
<tr>
<td>[50]</td>
<td>MSAC (2C)</td>
<td>MOR</td>
<td>SVM</td>
<td>83.45%</td>
</tr>
<tr>
<td>[36]</td>
<td>YouTube text (2C)</td>
<td>MSA</td>
<td>SVM</td>
<td>77.00%</td>
</tr>
<tr>
<td>[51]</td>
<td>JDT (2C)</td>
<td>JOR</td>
<td>SVM + LR</td>
<td>82.10%</td>
</tr>
<tr>
<td>[52]</td>
<td>ABD (2C)</td>
<td>MSA</td>
<td>DMNB</td>
<td>87.50%</td>
</tr>
<tr>
<td>[51]</td>
<td>SudSenti2, SudSenti3 (2C,3C)</td>
<td>SUB</td>
<td>SCM + MMA</td>
<td>92.75%, 84.39%</td>
</tr>
<tr>
<td>[35]</td>
<td>Lebanon dialect (2C)</td>
<td>LEB</td>
<td>LR</td>
<td>86.00%</td>
</tr>
<tr>
<td>[34]</td>
<td>Arabic opinions (2C)</td>
<td>MSA</td>
<td>SVM</td>
<td>76.33%</td>
</tr>
</tbody>
</table>
In this work, we will present an architecture called GCL, with unique regularization functions for 2C and 3C sentiment classification. Regularization (see next section) is an extra approach aimed at improving the model’s generalisation, producing better results on the test set [78].

Additionally, while evaluating, we consider the loss function’s accuracy and performance compared to other loss functions such as Binary-Cross-Entropy, Hinge, Poisson, and KL-divergence.

3. Proposed method

3.1. Outline

We designed a new architecture for ASA called GCL, based on various deep neural architectures with novel CRF. We also developed our previous preprocessing approaches [79], with different cleaning of the Arabic context (CP). We start by choosing the Arabic sentence inputs $X_n = (x_1, x_2, \ldots, x_n)$ and figuring out the context length, which varies from corpus to corpus. The data cleaning process $Y_m = (y_1, y_2, \ldots, y_m)$, begins from the input by removing special characters, punctuation marks, and all diacritics (see next subsection). The proposed method works with both 2C and 3C classification. We trained and tested on the AHSD (2C, SAU), ArTwitter (2C, JOR), MASC (2C, MOR), and BBN (3C, LEV) Arabic datasets.

3.2. Text preprocessing and normalization steps

Text data created from natural language is noisy and unstructured. Text preprocessing entails putting text into a neat, standardized structure to convert it into a form suitable for further analysis and training [80]. Text preprocessing methods may be broad so that they can be used in various applications, or they can be tailored for a particular goal. For instance, the techniques used to analyze scientific articles, including equations and other mathematical symbols, may differ greatly from those used to analyze user feedback on social networking sites [81]. The preprocessing steps used (see Fig. 2) were similar to those we developed previously [79]:

- We removed all digits, including dates.
- We removed repeated characters, keeping only one or two repeated characters.
- We removed any non-Arabic characters.
- We applied the tokenizer from the Keras package [82].
- We used the standard Arabic stopwords from NLTK [83].
- We carried out text normalization [79].

3.3. Input layer

All the features in a sample are represented by the initial vector sequence $M \, (m_1, \ldots, m_n, \ldots, m_n)$ where n is the number of features.

We use AraVec [85] to convert Arabic words into vectors. This process is based on Word2vec [74] which is an open-source tool that pre-trains word embeddings on a large data set, and which was originally used for English.

As is well known, the key idea behind word embeddings is that words with similar meanings are converted to similar vectors, which enables that similarity to be determined by vector comparison methods such as dot product or angle. They are also ideal for input to neural network models, as has been shown for a large number of NLP tasks in many different languages, including Arabic.

In the context of word embeddings, there are three interesting questions to consider, (1) how to handle word polysemy, i.e. words with many meanings, (2) how to handle metaphorical or hidden meanings, and (3) how to handle words whose meanings differ from country to country.

Concerning polysemy, a good example in Arabic is ‘hib’ which can mean ‘love’ or ‘seed’. Word embeddings such as AraVec are the result of training on datasets. In cases where a word can have many meanings within the training data used, the resulting vector will reflect aspects of all these, i.e. it will maintain and reflect the ambiguity. Then, later stages of the neural network model, which uses the embeddings as inputs, will tend to select the appropriate senses through the domain specific learning process.

Turning to hidden meanings, an example is ‘Asad’ which literally means ‘he is a lion’, but which actually means ‘he is a hero’. The datasets used in our experiments comprise social media exchanges, which are informal and contain many indirect uses of words and phrases, such as the description of a hero as a lion. A neural network such as the proposed model cannot solve this prob-
lem directly, but it can do so implicitly. For example, if a tweet refers to someone as a lion and the sentiment associated with the training data instance is positive, the model can learn that being described as a lion is a positive attribution, similar to being described as a hero. In such a way, the sentiment assigned to an unseen input can still be correct, even in cases of metaphorical language use.

Thirdly, we can find words whose meaning varies radically from country to country, depending on the Arabic dialect spoken. For example, ‘lbin’ can mean raw milk in one country and a product called ‘Laban Rayeb’ in others. In a system based on vector embeddings from AraVec, the meaning(s) associated with a word will depend on the dialects spoken in the training data used to create the embeddings. A typical dataset may indeed contain instances of different Arabic dialects. However, the proposed model does not rely on the interpretation of any single word in the input text; instead, the meanings of all words are converted to vector form and then input to the CNN model. In this way, the model can learn to overcome contradictions resulting from the incorrect interpretation of words. Thus, overall, we can see that the use of word embeddings in a neural network model can alleviate these three problems, still resulting in a sentiment analysis tool of high accuracy, while it cannot completely solve them.

After the embedding layer to vectorize the Arabic context in Fig. (3), we then applied the GCL architecture, which is a GRU with CNN through LSTM. We trained the model to perform sentiment analysis on various dialects. In addition, the relationship between MSA and other Arabic dialects was explored via a cross-dialect training study.

3.4. GRU layer

As is well-known [86], a GRU has gating units that modulate information flow within the unit without providing separate memory cells. It calculates two gates, called update and reset, that control information flow through each hidden unit. It is shown in Fig. 4 and defined by the following equations:

\[ r_s = \sigma(W_r x_s + U_r h_{s-1} + b_r) \]  
\[ z_s = \sigma(W_z x_s + U_z h_{s-1} + b_z) \]  
\[ h_s = \tanh(W_x x_s + U_r h_{s-1} + b_h) \]  
\[ h_s = z_s \odot h_{s-1} + (1 - z_s) \odot \tilde{h}_s \]  

where \( r_s \) represents the update gate, \( W_r, U_r \) are weight matrices, and \( r_s \) is a reset gate. The input vector \( M_s \), of all of these components is set to produce the now concealed state \( h_s \) and the previously hidden state \( h_{s-1} \). The logistic sigmoid function is denoted by \( \sigma \), and \( \odot \) is the multiplication of elements.

The update gate is determined from the current input and the preceding time phase hidden state. This gate determines how much new memory and old memory parts in the final memory can be mixed. The reset gate is measured similarly but with different sets of weights. It manages the balance between previous and new memory input. Here we applied our GRU layer with 128 filters and the custom regularization function. Our GRU layer has shown superior ability to handle lengthy Arabic context data Fig. (5) during the training, and to be faster than other approaches.

3.5. Convolutional layer

In this layer [87] we extract the local and multiple features, by using the following equation which illustrates how a filter \( F_i \) learns feature Map \( M^i_j \):

\[ M^i_j = f(V_{ij \cdot W \cdot 1}) \odot W^i + b^i \]  

where \( W \) represents the matrix weight bias, \( V_{ij \cdot W \cdot 1} \) is a token vector, \( \odot \) is the convolution operation, with max pooling, or average

---

**Fig. 2.** Steps for Arabic corpus data preparation.

**Fig. 3.** Arabic context visualization.

**Fig. 4.** GRU architecture (derived from Le [84]).
pooling to pick various features from the $M_i^{(j)}$. Here we add our Custom Regularization Function (see Section 3.8), and dropout to avoid overfitting and optimize the performance.

### 3.6. LSTM layer

This solves the problem of disappearing error gradients and captures long-term dependencies [89]. Three internal gates govern the flow of data to and from the memory blocks, as shown in Fig. 6 and defined as follows:

$$ h_i = f(W_i m_i + U_i h_{i-1}) $$ (6)

$$ f_s = \sigma(W_f X_i + U_f h_{i-1} + b_f) $$ (7)

$$ i_s = \sigma(W_i X_i + U_i h_{i-1} + b_i) $$ (8)

$$ o_i = \sigma(W_o X_i + U_o h_{i-1} + b_o) $$ (9)

$$ c_i = f_s \odot c_{i-1} + i_s \odot \tanh(W_c X_i + U_c h_{i-1} + b_c) $$ (10)

$$ h_i = o_i \odot \tanh(c_i) $$ (11)

where $h_i$ is a regular hidden state, $W_i, U_i$ are weight matrices, $m_i$ is the input vector, $f(\cdot)$ is a non-linear function, $f_s$ is the forget layer, $\sigma$ is a sigmoid function, $X_i$ is a cell parameter, $b$ is the Bias, $i_s$ is the input layer, chosen to be tanh, and $O_i$ is an output gate.

Our LSTM layer, with various outputs, processes the data and deals effectively with data noise as well as continuous values from the preceding layer.

### 3.7. Regularization

Prior to the development of deep learning, regularization was utilised for decades. Simple functions have usually been used with machine learning models and statistical approaches. The regularization did not need to be as sophisticated since the functions were less capable [90]. Classical regularizations are divided into two categories:

$$ L_2 = \text{loss} + \lambda/2M + \sum \|w^2\| $$ (12)

$$ L_1 = \text{loss} + \lambda/2M + \sum \|w\| $$ (13)

where $L_2, L_1$ are the Regularization functions, loss is the loss function, $\lambda$ is the regularization parameter, $M$ is the number of the layer, and $w$ is the weight for the layer.

### 3.8. CRF

The proposed Custom Regularization Function is illustrated in Fig. 7. We start with standard Regularization. $L_2$ on the weight side soon forces all the values from zero. It is very good. $L_1$ presses directly the weight to zero, and it is weak compared to $L_2$ [91].

When we customize our $L_{\text{new}}$ (Fig. 7) by calculating the absolute value among the values to be zero, the average of the values will tend to zero.

To optimize zone selections for our hyperparameter weights, our proposed extension is wider than $L_1$ and $L_2$, which helps to provide the best features for future selection. The Custom Regularization Function $L_{\text{new}}$ is defined as:

$$ L_{\text{new}} = \text{loss} + \lambda/2M + \sum \|w \ast w - w/2\| $$ (14)

$L_{\text{new}}$ helps improve the mean cost, which makes the overall errors small. In summary, the contribution of $L_{\text{new}}$ to the proposed method is:

- It is more sensitive to the quality of the output.
- It lessens the complexity of the model.

---

3.9. GCL model architecture

The proposed architecture was shown earlier in Fig. 1. This includes an embedding layer which can turn each word into a fixed-length, predetermined-size vector using embeddings; max-features represents the number of unique words, embedding-size equals 128 or 300 with a max-len of [30, 50, or 150]; after that a gated recurrent unit with 128 filters, which can solve long sequence training issues and improve efficiency and accuracy. After that, the convolutional neural network layers with 64 filters; they are capable of using different lengths and weights of windows for the number of feature maps to be created, and can be used for both dual and multiple classifications.

Kernel size is equal to three – this is the width and height of the filter mask for the CNN layer. Padding is set to 'valid', activation is equal to ReLU. This provides nonlinearity to a system that has essentially only been doing linear computations throughout the Conv layers.

Strides is equal to one, followed by [global average pooling 1D, global max pooling 1D]. Pool size equals two, then the customized regularization function for both previous layers, which helps us to improve the performance and optimize the validation loss when we compare to the classification loss functions – see Section (4.4). After that, Dropout (0.25) and an LSTM with output [90, 80, or 50], then Flatten, then batch normalization which lessens the gradient’s reliance on the parameters’ original values or scales and decreases the inner variational shifting, and finally, a dense layer with a softmax or Sigmoid layer i.e. a fully connected layer to predict the output of the class from either three sentiment classes (Positive, Negative, Neutral), or two classes (Positive and Negative).

4. Experiments

4.1. Datasets

Our model is trained on the AHSD, ArTwitter, MASC, and BBN datasets (see Table 1 earlier). Table 3 shows the outline statistics. The datasets can be described as follows:
The Arabic Health Services Dataset, AHSD[^7], is for Saudi Arabic (SAU). It was collected from Twitter by Aly et al. [28] and contains 2,026 tweets, with two unbalanced classes, 628 positive tweets, and 1,298 negative tweets.

ArTwitter[^1] is for Jordanian Arabic (JOR). It was created manually from Twitter [29] and consists of two balanced classes, 1,000 positive and 1,000 negative.

The Multi-domain Arabic Sentiment Corpus, MASC[^4] is for Moroccan Arabic (MOR). It contains 8,860 ratings from various websites, Google Play, Twitter, and Facebook. There are 4,476 positive tweets and 2,257 negative.

The BBN Dataset, BBN[^5] is for Levantine Arabic (LEV) and consists of 1,200 Levantine dialect phrases taken from the BBN Arabic-Dialect-English Parallel Text which itself consists of Levantine-English and Egyptian-English parallel texts [92].

The DzSenti corpus[^3] consists of 49,864 items, 24,932 negatives, and 24,932 positives, including MSA with Algerian dialect. It is publicly available.[^6]

LABR, the Large-Scale Arabic Book Review dataset,[^7] was developed by Aly et al. [38] and encompasses 63,000 items in MSA, with three classes, 42,724 positives, 8,174 negatives, and 12,168 neutral.

4.2. Experimental settings

We train the model and evaluate using the following metrics. True Positives (TP) is the number of correctly classified positive Tweets, True Negatives (TN) is the number of correctly classified negative Tweets, False Positives (FP) is the number of tweets incorrectly classified as positive, and False Negatives (FN) is the number of tweets incorrectly classified as negative. After that, the following performance measures are computed [93]:

\[
\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \quad (15)
\]

\[
\text{Precision} = \frac{TP}{TP + FP} \quad (16)
\]

\[
\text{Recall} = \frac{TP}{TP + FN} \quad (17)
\]

\[
F1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \quad (18)
\]

These measures are widely used in related work [94–96]. The following tuning and hyperparameter settings were used: Embedding size [128, 300], Pooling [2, 4, 6], Batch-size [64, 128, 164].

### Table 3: Datasets for our experiments.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Dialect</th>
<th>POS</th>
<th>NEG</th>
<th>NEU</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>AHSD (2C)</td>
<td>SAU</td>
<td>628</td>
<td>1398</td>
<td>-</td>
<td>2,026</td>
</tr>
<tr>
<td>ArTwitter (2C)</td>
<td>JOR</td>
<td>1000</td>
<td>1000</td>
<td>-</td>
<td>2,000</td>
</tr>
<tr>
<td>MASC (2C)</td>
<td>MOR</td>
<td>4,476</td>
<td>2,257</td>
<td>-</td>
<td>6,733</td>
</tr>
<tr>
<td>BBN (3C)</td>
<td>LEV</td>
<td>498</td>
<td>575</td>
<td>126</td>
<td>1,119</td>
</tr>
<tr>
<td>DzSenti (2C)</td>
<td>ALG</td>
<td>24,932</td>
<td>24,932</td>
<td>-</td>
<td>49,864</td>
</tr>
<tr>
<td>LABR (3C)</td>
<td>MSA</td>
<td>42,724</td>
<td>8,174</td>
<td>12,168</td>
<td>63,066</td>
</tr>
</tbody>
</table>

[^1]: https://bitbucket.org/a_alayba/arabic-health-services-ahs-dataset/src/master/
[^5]: https://github.com/adelabdelli/DzSentiA.
[^7]: http://github.com/Alayba/arabic-health-services-ahs-dataset/src/master/.
Kernel-size [3, 5], Number-classes [2, 3], Epoch [10, 50, 100], with Adam optimizer and 0.001 Learning Rate. For the implementation, we used the Tensorflow framework.8

4.3. Experiment 1: 2C and 3C sentiment classification

We applied the proposed method (GCL) to the four datasets, AHSD (2C), ArTwitter (2C), MASC (2C), and BBN (3C). Ten-fold cross-validation was used for all models, using a random 80% for each training, and the remaining 20% for testing. Results are in Table 4. The accuracy of the proposed method, GCL, was 95.50% for AHSD, 93.88% for ArTwitter, 86.64% for MASC, and 74.92% for BBN. In all cases these are higher than the previous baselines.

4.4. Experiment 2: comparison of loss functions

The loss function is the function that determines the distance between the algorithm’s current outcome and the desired output. It provides a means of assessing how well the prediction mimics the data.9 Loss functions are used to calculate the amount a model should try to reduce its error throughout learning. There are various types.10

We utilized the proposed method (GCL) with various loss functions: Binary-Cross-Entropy, Hinge, Poisson, and KL-divergence. We also included our customized regularization plus binary-cross-entropy (CRF + binary-cross-entropy). The four datasets from the previous experiment were used, with the same ten-fold cross-validation for all approaches. Results are in Table 5.

For AHSD, validation losses with GCL + BC, GCL + Hinge, GCL + Poisson, GCL + KL-divergence, and GCL + (CR + Binary-Cross-Entropy) were 0.437, 0.5571, 0.6934, 0.4316, and 0.3822, respectively. GCL+ (CR + Binary-Cross-Entropy) had the lowest validation loss (0.382), and the lowest time except for GCL + Hinge.

For ArTwitter, MASC and BBN, GCL + (CR + Binary-Cross-Entropy) also had the lowest validation loss (0.3685, 0.5449, 0.5442). We therefore conclude that the proposed method with customized regularization plus binary-cross-entropy was the best performing model on the four datasets.

4.5. Experiment 3: ablation study

We carried out an ablation study on the proposed method using the four datasets. Training of the proposed GCL model was done both with and without the proposed custom regulation function (CRF). We used ten-fold cross-validation and report the average results in Table 6. Accuracies of GCL + CRF using AHSD, ArTwitter, MASC, and BBN were 95.50%, 93.88%, 86.64%, and 74.92%. For GCL without CRF, these reduced to 90.40%, 91.77%, 84.50%, and 70.80% respectively, changes of −5.10%, −2.11%, −2.14%, and −4.12%. The

10 https://keras.io/api/losses/.

4.6. Experiment 4: MSA-dialect association study

We used our model to study the Arabic sentiment association between MSA and Arabic dialects. We chose 5,000 MSA texts from HARD11 [37] with 2,500 positive examples and 2,500 negative. For AHSD (SAU), ArTwitter (JOR), MASC (MOR), SudSenti212 (SUD) [53], and BBN (LEV) we took 2,000 tweets from each one to represent text samples in these Arabic dialects.

In the first part, we trained our model on HARD and then evaluated on AHSD, ArTwitter, MASC, SudSenti2, and BBN. Results are in Tables 7 and 8. As the results show, the best result is obtained by training and testing on MSA (94.27%). We can consider this our ‘baseline’ in comparing dialects with MSA. The highest accuracy after that is for SAU (89.80%, −8.98%), followed by MOR (84.20%, −10.07%), and LEV (83.24%, −11.03%). There is then a gap of 2.67% before we reach SUD (80.57%, −13.70%) and JOR (78.98%, −15.29%). So we can conclude that the most similar dialect to MSA is SAU and that the least similar dialects are SUD and JOR.

In the second part, we combined the five previous dialect data-sets and named it the Main Arabic Multi Binary Sets (MAMBS) corpus. We then trained GCL on HARD and tested on MAMBS. Results are in Tables 9 and 10. Accuracy was 76.70%, compared to our ‘baseline’ figure of 94.27% from Table 7. Naturally this is lower, and indeed it is behind the lowest figure in Table 7 (JOR, 78.98%), exactly as we would expect. What this result suggests is that we can achieve a useful performance figure on different dialects when training on MSA, but to achieve high accuracy, we need to use specialized training data.

The Saudi dialect comprises seven local dialects derived from ancient Arabic, while the Moroccan dialect has words from the Spanish and French dictionaries. Certain words from Turkish and English appear in the Sudanese dialect. Also, the Lebanese dialect contains influences from Aramaic and Syriac. Since the essence of all dialects is in the MSA, some differences in vocabulary resulted in various associations with MSA in the results when using the proposed methods in the classification tasks. Please refer back to Section 3.3 for further dis-

---

Table 8  
Experiment 4(a): Results in terms of P, R, F, Macro Average, broken down by Positive and Negative sentiment.

<table>
<thead>
<tr>
<th>Train vs. Test</th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
<th>Macro Avg</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Positive</td>
<td>Negative</td>
<td>Positive</td>
<td>Negative</td>
</tr>
<tr>
<td>HARD vs. HARD</td>
<td>0.93</td>
<td>0.93</td>
<td>0.95</td>
<td>0.95</td>
</tr>
<tr>
<td>HARD vs. AHSD</td>
<td>0.80</td>
<td>0.90</td>
<td>0.89</td>
<td>0.83</td>
</tr>
<tr>
<td>HARD vs. MASC</td>
<td>0.85</td>
<td>0.84</td>
<td>0.85</td>
<td>0.84</td>
</tr>
<tr>
<td>HARD vs. BBN</td>
<td>0.79</td>
<td>0.89</td>
<td>0.90</td>
<td>0.76</td>
</tr>
<tr>
<td>HARD vs. SudSenti2</td>
<td>0.78</td>
<td>0.83</td>
<td>0.84</td>
<td>0.77</td>
</tr>
<tr>
<td>HARD vs. ArTwitter</td>
<td>0.75</td>
<td>0.85</td>
<td>0.87</td>
<td>0.72</td>
</tr>
</tbody>
</table>

Table 9  
Experiment 4(b): Cross-dialect training between MSA and all dialects combined, using proposed GCL model.

<table>
<thead>
<tr>
<th>Train</th>
<th>Test</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>HARD (MSA)</td>
<td>MAMBS (SAU, JOR, MOR, LEV, SUD)</td>
<td>76.70%</td>
</tr>
</tbody>
</table>
Table 10
Experiment 4(b): Results in terms of P, R, F, Macro Average, broken down by Positive and Negative sentiment.

<table>
<thead>
<tr>
<th>Model</th>
<th>LABR Precision Positive</th>
<th>LABR Precision Negative</th>
<th>DzSenti Recall Positive</th>
<th>DzSenti Recall Negative</th>
<th>F1 Positive</th>
<th>F1 Negative</th>
<th>Macro Avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>HARD vs. MAMBS</td>
<td>0.73</td>
<td>0.81</td>
<td>0.81</td>
<td>0.72</td>
<td>0.77</td>
<td>0.76</td>
<td>0.77</td>
</tr>
</tbody>
</table>

Table 11
Experiment 5: GCL with several regularizations on the huge LABR and DzSenti datasets.

<table>
<thead>
<tr>
<th>Model</th>
<th>LABR</th>
<th>Time</th>
<th>DzSenti</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>GCL+L_1</td>
<td>91.36</td>
<td>43 m 30s</td>
<td>86.55</td>
<td>1 h 2 m 18s</td>
</tr>
<tr>
<td>GCL+L_2</td>
<td>91.71</td>
<td>1 h 24 m 18s</td>
<td>86.92</td>
<td>27 m 3s</td>
</tr>
<tr>
<td>GCL+ElasticNet</td>
<td>92.35</td>
<td>38 m 55s</td>
<td>86.97</td>
<td>21 m 41s</td>
</tr>
<tr>
<td>GCL+Lnew</td>
<td><strong>92.53</strong></td>
<td><strong>33 m 49s</strong></td>
<td><strong>87.26</strong></td>
<td><strong>20 m 27s</strong></td>
</tr>
<tr>
<td>Baselines</td>
<td>91.9% [97]</td>
<td>-</td>
<td>86.00% [17]</td>
<td>-</td>
</tr>
</tbody>
</table>

Fig. 8. Models applied to 2C Arabic sentiment datasets (GCL is proposed model).

Fig. 9. Models applied to 3C Arabic sentiment datasets (GCL is proposed model).
Fig. 10. Validation performance of proposed GCL model on HARD vs. Main-AHS, ArTwitter, MASC, SudSenti2, and BBN.

Fig. 11. Accuracy and validation accuracy on HARD vs. MAMB datasets.

Fig. 12. Loss and validation loss for proposed GCL method on AHSD dataset (2C).
Fig. 13. Loss and validation loss for proposed GCL method on ArTwitter dataset (2C).

Fig. 14. Loss and validation loss for proposed GCL method on MASC dataset (2C).

Fig. 15. Loss and validation loss for proposed GCL method on BBN dataset (2C).
discussion on NN models and the effects of polysemy, metaphor and dialects.

4.7. Experiment 5: evaluation of the proposed approach (GCL) with several regularization functions on massive Arabic corpora

We used GCL with different regularizations on LABR, the Large-Scale Arabic Book Review, containing 63,000 MSA items, and the DzSenti dataset, which comprises 49,864 items from social media, including both MSA and the ALG dialect (see Table 3).

When we applied GCL+L1, GCL+L2, GCL+LElasticNet, and GCL+Lnew on LABR, the accuracy and the times were 91.36% (43 m 30s), 91.71% (1 h 24 m 18s), 92.35% (38 m 55s), and 92.53% (33 m 49s), respectively, as shown in Table 11 and Fig. 16.

On the DzSenti dataset, the performance was 86.55% (1 h 2 m 18s) with GCL+L1, 86.92% (27 m 3s) with GCL+L2, 86.97% (21 m 41s) with GCL+LElasticNet, and 87.26% (20 m 27s) with GCL+Lnew, as shown in Table 11 and Fig. 17.

First, we note that GCL+Lnew had the highest performance and used less time with both datasets and exceeded the baseline; for LABR, accuracy was 92.53% compared to 91.9% [97], and for DzSenti it was 87.26% compared to 86.00% [17].

Second, the results show that our models can efficiently handle large Arabic datasets. Also, throughout the training, the accuracy remained consistent.

4.8. Validation loss training

Figs. 8 and 9 show the performance accuracy of the proposed method and baselines on the 2C and 3C datasets. Figs. 10 and 11 show the validation performance of HARD on the individual, and grouped training. Finally, Figs. 12–15 show the loss and validation loss of the GCL model with five standard loss functions, after 100 epochs, on the AHSD, ArTwitter, MASC, and BBN datasets respec-

---

15 Elastic Net ($L_1 + L_2$).
16 (h: hours, m: minutes, and s: seconds).
tively. GCL + (CRF + BC) gives us the best validation loss and the least execution time, when applied to the four datasets.

For the training and validation epochs, the suggested technique was stable. On different datasets, the loss and validation loss were also stable. This demonstrates that the proposed method can be used effectively within training regimes.

5. Conclusion and future work

In this study, we developed an Arabic Sentiment Analysis model called Gated Convolution Long (GCL), based on GRU, CNN, and LSTM. The model incorporates a Customized Regularization Function (CRF).

We then carried out five experiments. First, GCL was independently trained and tested on four different datasets, AHSD (2C), ArTwitter (2C), MASC (2C), and BBN (3C). The proposed model outperformed the baselines for all datasets.

Second, we created versions of GCL with five different loss functions, Binary-Cross-Entropy, Hinge, Poisson, KL-divergence, and CRF + Binary-Cross-Entropy. These were trained against the same four datasets. CRF + Binary-Cross-Entropy had the lowest validation loss in all cases.

Third, we conducted an ablation investigation using GCL and the same four datasets. We trained both with CRF and without CRF, and tested the resulting model. The results showed that CRF improved the performance of GCL for all datasets.

Fourth, we used the proposed model to compare the influence between emotion in Modern Standard Arabic and those in five distinct Arabic dialects. First, we trained on HARD (MSA) and evaluated on AHSD (SAU), ArTwitter (JOR), MASC (MOR), SudSent12 (SUD), and BBN (LEV). We found that the most similar dialect to MSA is SAU, and that the least similar dialects are SUD and JOR. Second, we trained on HARD and tested on all dialects together. This showed that a useful level of performance could be obtained in this way, but lower than when training and testing on a specific dialect.

Fifth, we applied GCL using standard regularizations (GCL+1, GCL+2, and GCL+Gated) and our $L_{new}$ on two big Arabic sentiment datasets, LABR (MSA) and DeSenti (MSA, ALG); GCL+$L_{new}$ gave the highest results with less training time.

Future research will examine the effectiveness of the proposed approaches using a variety of datasets, such as reviews of eateries, technology, the news, and different language-specific archives.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

This research was supported by the National Natural Science Foundation of China [grant no. 61877050] and Open Project Fund of Shaanxi Province Key Lab of Satellite and Terrestrial Network Technology, Shaanxi Province Financed Projects for Scientific and Technological Activities of Overseas Students (grant no. 202160002).

References


