Received December 19, 2019, accepted January 27, 2020, date of publication January 30, 2020, date of current version February 10, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2970504

A Superior Arabic Text Categorization Deep Model (SATCDM)

M. ALHAWARAT, (Member, IEEE), AND AHMAD O. ASEERI
Department of Computer Science, College of Computer Engineering and Sciences, Prince Sattam Bin Abdulaziz University, Al-Kharj 11942, Saudi Arabia
Corresponding author: M. Alhawarat (m.alhawarat@psau.edu.sa)

ABSTRACT Categorizing Arabic text documents is considered an important research topic in the field of Natural Language Processing (NLP) and Machine Learning (ML). The number of Arabic documents is tremendously increasing daily as new web pages, news articles, social media contents are added. Hence, classifying such documents in specific classes is of high importance to many people and applications. Convolutional Neural Network (CNN) is a class of deep learning that has been shown to be useful for many NLP tasks, including text translation and text categorization for the English language. Word embedding is a text representation currently used to represent text terms as real-valued vectors in vector space that represent both syntactic and semantic traits of text. Current research studies in classifying Arabic text documents use traditional text representation such as bag-of-words and TF-IDF weighting, but few use word embedding. Traditional ML algorithms have already been used in Arabic text categorization, and good results are achieved. In this study, we present a Multi-Kernel CNN model for classifying Arabic news documents enriched with n-gram word embedding, which we call A Superior Arabic Text Categorization Deep Model (SATCDM). The proposed solution achieves very high accuracy compared to current research in Arabic text categorization using 15 of freely available datasets. The model achieves an accuracy ranging from 97.58% to 99.90%, which is superior to similar studies on the Arabic document classification task.

INDEX TERMS Documents classification, deep learning, Arabic language, convolutional neural networks, word embedding, skip-gram, word2vec.

I. INTRODUCTION

Classification of text documents is of high importance for many NLP technologies. Document classification is the process of categorizing documents into classes based on their contents. Classifying Arabic documents has always been a challenge due to the nature of the language itself having rich dialects and enormous numbers of synonyms. It also reflects the lack of Arabic resources compared to other languages such as English, inaccurate stemming algorithms, the high-derivative nature of the Arabic language, and equivocalness inflicted by diacritic are reasons to make such a classification task so complex [1], [2]. Categorizing Arabic text documents is considered an important research topic in the field of Arabic Natural Language Processing (ANLP) and Machine Learning (ML). Classifying Arabic documents in specific classes is of high importance to many people and applications. In this study, we present an innovative deep learning methodology to classify Arabic text documents to achieve better results using the latest deep learning technology and algorithms, including CNN and word embedding.

Deep learning has achieved extraordinary advancements in the field of Computer Vision and Speech Recognition [3], [4], yet gradually improving for natural language processing, especially in the Arabic language. Many research studies have investigated the Arabic text classification problem in different domains, mainly for news, in which the majority of these studies apply traditional ML techniques [5]–[9]. In fact, the vast majority of these studies have profoundly investigated the English language. In contrast, few are concerned with the Arabic language. Deep learning is a class of neural network method that has been widely employed to solve complex problems in many fields of studies, including NLP and text classification. This study presents a deep learning model that is based on CNN and n-gram word embedding language models with sub-word information. The model is called Superior Arabic Text Categorization Deep Model (SATCDM), and it utilizes an efficient multi-kernel CNN

The associate editor coordinating the review of this manuscript and approving it for publication was Jenny Mahoney.

architecture inspired by [10] for text classification and uses a skip-gram word embedding language model enriched with sub-word information [11].

Free datasets found for Arabic text classification are used in this study; these are 15 datasets representing Arabic News text documents that comes in Modern Standard Arabic (MSA) format. In order to compare the results of the model, baseline models are defined to be the traditional ML techniques that are usually used in performing the Arabic text classification task. The results of this study would significantly contribute to helping researchers in the field of ANLP to classify Arabic text documents more correctly into predefined classes, increasing the accuracy of retrieving Arabic documents in search engines and other applications.

Our study, according to our knowledge, is the first that utilizes word embedding (word2vec) and CNN to classify Arabic **news** text documents in **MSA** format and apply it on almost all free available datasets. The manuscript is organized as the following: section II shows literature review, section III introduces CNN briefly, description of the data used in the study is given in section IV, section V introduces the SATCDM model, section VI describes the methodology and experiments setup, section VII shows and discusses the results of experiments, and finally section VIII concludes the paper.

II. RELATED WORK

There have been studies that applied traditional ML algorithms for document classification including Support Vector Machine (SVM), Naïve Bayes classifier (NB), K-Nearest Neighbor (KNN), Decision Trees (DT) and Rocchio classifier [7], [12]–[27]. The authors in [24] built and investigated the word embedding model for sentiment classification for the Arabic language. Their model slightly outperformed other existing word embedding models tested on the task of sentiment classification. In [25], the author proposed a three-stage algorithm to classify Arabic documents using deep belief networks and Markov clustering. Compared with traditional ML algorithms, his method outperforms NB, KNN, and SVM algorithms tested on two public datasets. Moreover, the authors in [26] classified Arabic documents Based on document embeddings (doc2vec) and reported better results compared to traditional ML algorithms. Furthermore, the authors in [27] provided pre-trained distributed word representation models for the Arabic language called AraVec. They used different domains, including tweets, web pages, and Wikipedia. The size of the model is 3.3 billion tokens, and the model has two versions; one in a continuous bag of words (CBOW), and the other is in Skip-gram. Lastly, one outstanding work is done by Kanan and Fox [20] who constructed a dataset of 237,000 Arabic news articles and then applied traditional ML algorithms such as SVM, NB, and Random Forest using different stemming approaches. The authors developed a new stemming algorithm called p-stemmer and applied the aforementioned ML algorithms with the proposed stemmer. The best result was using SVM via the p-stemmer algorithm.

On the other hand, few studies have used deep learning techniques for Arabic document classifications, including social media text [25], [28]–[30]. Dahou *et al.* [28] addressed the problems of word embedding and sentiment classification for Arabic text, specifically Arabic reviews, and social media contents, for sentiment classification. Their proposed solution uses Convolutional Neural Network (CNN) with word embedding. They built a word embedding model consisted of 3.4 billion words from a 10 billion web-crawled corpus. In the benchmark datasets, their proposed methodology outperforms other existing methods that rely on linear SVM algorithm and SVM-BOW (Bag of Words). Sayed *et al.* [29] investigated and compared the performance of three traditional ML algorithms: SVM, NB, and KNN, against deep learning algorithm on a new dataset in the context of n-gram and similarity variables. The best two reported algorithms are deep neural networks and NB classifier. Recently, Biniz *et al.* [30] used Convolutional Neural Networks to classify Arabic documents. First, they use stemming algorithms to minimize the number of features. Then, they used Term Frequency-Inverse Document Frequency (TD-IDF) to weigh and select important features as input to the CNN. They report better results compared to ML algorithms. Recently, [31], [32] used different deep learning models, including CNN, to classify Arabic text documents using a new large dataset called SANAD. In the CNN model, they used three convolutional layers and archived outstanding accuracy results.

For more information about deep leaning and Arabic NLP, please refer to [33], which is a survey paper that contains the advancement of research related to the Arabic language in different fields of studies. It is worth noting that very few studies concerns with Arabic text categorization using Deep learning techniques.

III. CONVOLUTIONAL NEURAL NETWORKS (CNNs)

Convolutional neural networks have become one of the most successful foundation in the fields of computer vision, natural language processing and pattern recognition. It was introduced first time by LeCun *et al.* [34] as a means to replace traditional recognition systems with new paradigm that operates directly on pixel level. The architecture of CNN is in fact developed to address the issue of handling 2D data structure of images (or even other 2D inputs such as voice signals). CNN is very similar to the multilayer perceptron (MLP) in which they consist of neurons connected by learnable weights and biases. Technically speaking, CNNs are mainly recognized to identify and extract features of images followed by a fully-connected multilayered perceptron as a classifier. CNN outperforms the traditional image classification algorithms for learning the feature representations of input images very efficiently.

The core building block of CNNs is the mathematical operation called "convolution" which serves as a feature extractor from the images. The 2D convolution (formally, cross-correlation) is given by:

$$S(i,j) = (I \cdot F)(i,j) = \sum_{m} \sum_{n} I(i+m, j+n)F(m,n)$$

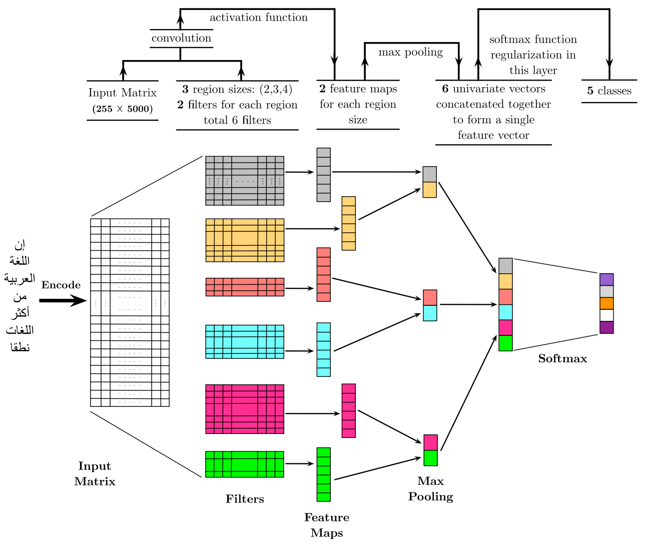
activation function
convolution
max pooling
softmax function regularization in this layer
Input Matrix (255 × 5000)
3 region sizes: (2,3,4) 2 filters for each region total 6 filters
2 feature maps for each region size
6 univariate vectors concatenated together to form a single feature vector
5 classes
إن اللغة العربية من أكثر اللغات نطقا
Encode
Input Matrix
Filters
Feature Maps
Max Pooling
Softmax

FIGURE 1. A generic CNN architecture for Text Categorization.

where $S(i, j)$ is the output feature map, I is the input image, F is the filter/kernel, and $\cdot$ is the convolution operation. A deep convolutional neural network generally refers to an architecture composed of convolutional layers, pooling (subsampling) layers, and finally a fully-connected network, grouped in modules [35], [36]. There are two learning stages, the feature learning followed by classification where the former incurs the majority of computation costs. That is, an input image is run through several modules of convolution and pooling. Subsequently, the final representations are fed to fully-connected layers for classification.

CNN can be applied for the purpose of text classification. As illustrated in Figure 1, the process of applying CNN in texts starts with encoding the documents into a real-valued matrix. By choosing appropriate filters, the convolution process takes place to generate the feature maps followed by pooling layers to reduce the number of parameters and computation cost, making it ideal for controlling overfitting. The output is then passed through an activation nonlinearity layer, such as ReLU, to speed up training. Eventually, and possibly after several convolutional modules, the final feature map is then passed to a fully-connected multi-layer dense layer, having a Softmax activation layer at the end for the classification task.

IV. DATA PREPARATION

To assure the reliability of this study, 15 of the most freely accessible Arabic news documents datasets are used. These can be classified into three categories:

- Small-Medium size datasets with original text.
- Small-Medium size datasets with stemmed text.
- Large-size datasets with original text.

More information about these datasets is given in the following subsections. After that, a summary of data preprocessing is given, and finally, the word embedding model used is explained.

TABLE 1. Details of the small-medium size datasets with original text.

Dataset	No Classes-Docs.	No Terms-Unique Terms	Doc. Avg. Length
Abuaiadah(V1) [12]	9 - 2,700	878,726 - 96,859	325
Aljazeera [37]	5 - 1,500	388,653 - 50,099	259
Alwatan [38]	6 - 20,291	9,876,786 - 261,909	487
Alkhaleej [39]	4 - 5,690	2,472,763 - 122,162	435
OSAC [40]	10 - 22,429	18,183,511 - 677,972	811
BBC [40]	7 - 4,763	1,794,123 - 88,953	377
CNN [40]	6 - 5070	2,166,109 - 105,047	427

TABLE 2. Details of the small-medium size datasets with stemmed text.

Dataset	No Classes-Docs.	No Terms-Unique Terms	Doc. Avg. Length
Abuaiadah(V2) [12]	9 - 2,700	600,627 - 89,757	325
Abuaiadah(V3) [12]	9 - 2,700	600,552 - 42,571	222
Abuaiadah(V4) [12]	9 - 2,700	600,477 - 30,488	222
Abuaiadah(V5) [12]	9 - 2,700	600,602 - 13,803	222
NADA [44]	10 - 7,310	3,248,653 - 152,050	444

A. SMALL-MEDIUM SIZE DATASETS WITH ORIGINAL TEXT

Table 1 lists information about 7 small-medium size datasets where stopwords are not removed, and no stemming algorithms are applied. For more details about the categories and other information, please refer to the references that appear beside each in the table.

B. SMALL-MEDIUM SIZE DATASETS WITH STEMMED TEXT

Table 2 shows the basic information about the second type which is small-medium size datasets with stemmed text. The dataset taken from [12] has five versions:

1) V1: Text as is (This used in the previous dataset type).
2) V2: Text with no stop words.
3) V3: V2 where Light10 stemming algorithm [41] is applied.
4) V4: V2 where Chen's stemming algorithm [42] is applied.
5) V5: V2 where Khoja's stemming algorithm [43] is applied.

NADA dataset is composed of two known datasets: Abuaiadah and OSAC. OSAC dataset has been cleaned, normalized and stemmed using light10 stemmer. Moreover, a feature selection algorithm is used to reduce the dimensionality of the dataset from 22k text documents to around 7k.

C. LARGE-SIZE DATASETS WITH ORIGINAL TEXT

The third type represents large-size datasets with the original text. This is a recent dataset comprises of a large number of news documents wit ha total of around 195 thousand. Basic information is shown in table 3, and more information can be found in [45].

D. DATA PREPROCESSING

One of the advantages of the methodology of this study is requiring no preprocessing for the input text documents; any

TABLE 3. Details of the large-size datasets for SANAD [45].

Dataset	Nº Classes- Docs.	Nº Terms- Unique Terms	Doc. Avg. Length
Arabiya	6 - 71,246	16,817,633 - 429,597	236
Khaleej	7 - 45,500	16,801,740 - 689,155	369
Akhbarona	7 - 78,428	19,965,764 - 772,872	255
All Combined	7 - 195,174	53,585,137 - 1,295,948	275

form of Arabic text is valid, including original or stemmed words. Although there are different normalization and text preprocessing methods for Arabic text, all the experiments in this study take the input text documents as is without any preprocessing. Notwithstanding, some of the aforementioned datasets have already been preprocessed by their creators, but some are not.

E. WORD EMBEDDING

Although bag-of-words representation is still used in some NLP tasks, recently word embedding models provide better performance as they encapsulate semantic as well syntax of words. In contrast, bag-of-words models represent tokens as counts in the text where the position of the word in the context of other words is neglected. The most famous models for word embedding are Word2vec [46] and GloVe [47]. Many word embedding language models have been developed for the English language, whereas few exist for the Arabic language, such as [11], [27], [48]. In this study, the Arabic Wikipedia fast-text model [1] is used [11] which extends the original continuous skip-gram model of [49] to include word vector as the sum of corresponding character n-gram vectors as a representation for each word. The model outperformed baseline models as it enriched with such sub-word information. It consists of around 610k unique tokens and is available for many languages, including Arabic. This is beneficial to this study for three reasons: first, it includes vectors for variance forms of a specific word by averaging the n-grams representing that word. Second, stemmed words can have their corresponding vectors referring to similar character n-grams tokens, especially because this language model accounts for affixes and suffixes. Third, no preprocessing is required since the character n-grams tokens are included in the model. The fast-text model comes in 300-vector size.

V. SATCDM MODEL

SATCDM is a multi-kernel CNN model for classifying Arabic news documents. The model is enriched with n-gram word embedding that is based on the aforementioned skip-gram language model [11]. The best kernel sizes are 2,3 and 5, according to [10]. This setting is very effective yet simple in extracting multiple n-gram features, including bi, tri, and five-gram tokens. One significant benefit of the proposed model is its suitability for any Arabic text documents regardless of normalization, preprocessing, stemming algorithms, or methods usually used in preparing text datasets.

[1] https://fasttext.cc/docs/en/pretrained-vectors.html

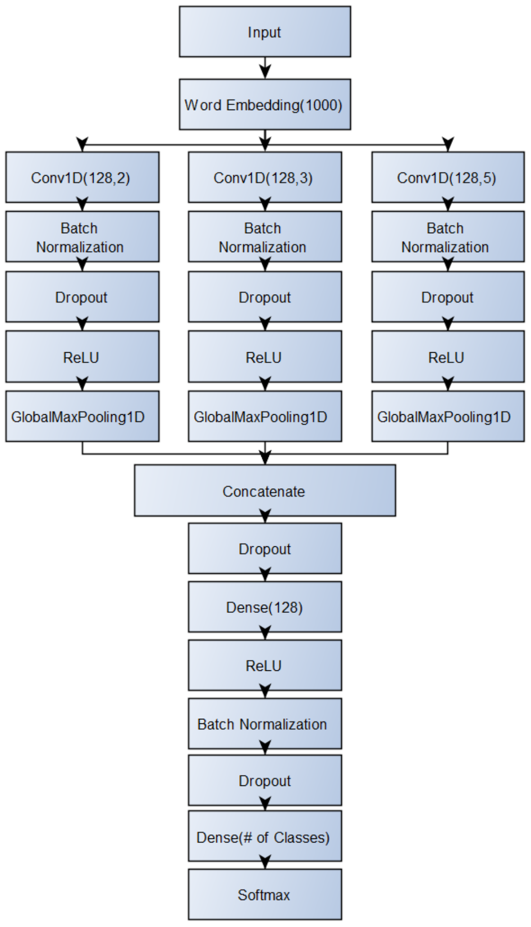

FIGURE 2. SATCDM architecture.

The architecture of the CNN model used in this study is depicted in figure 2. The model is composed of three concatenated single one-dimensional convolutional layer with the ReLU activation function, followed by a Batch-Normalization layer and a dropout layer with a weight of 0.2 succeeded by a one-dimensional Global-Max-Pooling layer. All three convolution layers use 128 filters, and kernel sizes 2, 3, and 5, respectively. These accounts for 2-gram, 3-gram, and 5-gram features. After that, the concatenated sub-models are followed by a dropout layer with a weight of 0.2. This is followed by a dense layer with the size of 128 with a ReLU activation function followed by a Batch-Normalization layer and a dropout layer with a weight of 0.2 succeeded by another dense layer -with the size of the number of classes of the dataset under consideration- with a softmax activation function. Two optimizers are examined in setting up experiments: RMSProp and Adam. The later gives better performance and hence used in all experiments. One reason that makes Adam better than RMSProp is its ability to solve the sparse gradient problem that is found in RMSProp [50]. The loss function used is binary-crossentropy.

VI. METHODOLOGY AND EXPERIMENTS SETUP

This research studies the performance of a deep neural model (SATCDM) in classifying Arabic text documents.

The results of SATCDM model will be compared to a baseline model. In the last few decades many studies have applied traditional ML algorithms including SVM, Naive Bias, Logistic Regression and other techniques to the field of Arabic Text Categorization, where SVM, SGD, NB models were superior in this task [5]–[9]. Its worth mentioning here that other models and algorithms are used in the experiments and reported in this study because they give inferior results; these are random forests, boosting and bagging. Recently deep learning become the state-of-the-art for many NLP tasks. This study investigates how well can deep learning perform in Arabic text categorization. This is to be compared with the best traditional models that excelled in that task; SVM, SGD, and NB models. Before explaining the experimental setup for both traditional ML and SATCDM models, first, the methodology used to avoid overfitting and selection bias is introduced next.

A. AVOIDING OVERFITTING AND SELECTION BIAS

Deep learning models are naturally prone to Overfitting. Therefore two techniques are used in this study to avoid overfitting: Dropout [51] and Batch Normalization [52] as shown in figure 2. Also, cross-validation is used to avoid both overfitting and data selection bias.

In ML, it is imperative to make sure that the model generalizes well after training, and at the same time, the resulted performance of the trained model is not biased to the test set. Cross-validation overcomes these problems by dividing the dataset into three parts: train, validation, and test sets. In this study, the 5-fold cross-validation technique is used in both traditional and deep learning methods where data is divided as follows:

- Traditional ML: 20% for testing, 80% for training divided into: 64% for training and 16% for validation for each fold.
- Deep Learning: 10% for testing, 90% for training divided into: 72% for training and 18% for validation for each fold.

Here more training data is used in deep learning models as they are eager for learning as the result of the large number of parameters they have. Each model is trained and validated on training/validation sets for five-folds. After that, the resulted model is tested on the testing set. Not only cross-validation is important for avoiding selection bias, but it also helps in avoiding overfitting because the model is tested on seen and unseen data, and hence will perform and generalize well on future unseen data. With the K-fold CV technique, the model is trained and validated K times on a unique, different part of the data each time. Stratified sampling is used to assure that samples are picked evenly from all classes of the dataset. To make it more robust, data is shuffled each time before splitting into batches. The average of the validation accuracy is then computed along with the standard deviation to obtain accurate performance results. The only disadvantage of this method that it takes more time for the gain of a model with better performance and generalization.

B. EXPERIMENTAL SETUP

In all experiments, the python programming language is used where Keras and TensorFlow are utilized for both traditional and deep learning models. For the hardware, we use two platforms: for traditional machine learning a machine with i7 core and 32GB RAM equipped with a GTX-1070 GPU with 8GB RAM is used. Whereas for deep learning model we use the free K80 online GPU with 24GB RAM that is available through Google Colab environment.

1) BASE LINE

All traditional ML models are trained using stratified 5-fold cross-validation where data is shuffled each time splits occur. Training and validation splits are set to 80% and 20%, respectively.

Term Frequency-Inverse Document Frequency (TF-IDF) weighting is used for SVM, SGD, and NB models. This is better than Term Frequency because it counts for both very frequent and rare terms across all text documents. Common terms across all documents usually represent non-important terms for text classification, whereas rare terms across all text documents constitute highly important discriminative features.

The settings for SVM is the standard SVM with the linear kernel because text categorization is a linear separable ML problem [53]. The tolerance used for stopping criteria is 1e-5. LinearSVC is used, it is similar to SVC with linear kernel, but is implemented using liblinear instead of libsvm. The most advantage of this choice is performance scalability to large dataset size, in addition to supporting sparse data, which is the case in the bag-of-words model.

For SGD, the model uses mini-batch learning with a decreasing learning rate, which works well for sparse and dense data; by default, it fits a linear SVM. It also uses L2-norm regularizer, and the number of iterations is 100. For NB, the default settings are used.

2) SATCDM

The SATCDM model is trained also using stratified 5-fold cross-validation where data is shuffled each time splits occur. During training, the number of epochs is 20 although early stopping is implemented to increase the performance of the model if no improvement is made on validation loss for 4 successive epochs, the size of the batch used is 50, training validation split is set to 80% and 20% respectively.

Not all features are used during training the model to have better performance in terms of time and accuracy. The SATCDM uses a specific percentage of the most frequent terms according to the following criterion: for datasets with less than 400,000 unique tokens, then 40% of the most frequent tokens are used, else if the dataset contains more than 400,000 unique tokens then only 25% is used. These percentage rates are deduced experimentally. Finally, out of vocabulary (OOV), tokens are set to zeros. Similarly, not all features in each text document are used, the most 1000 frequent terms are used; this is also decided experimentally.

TABLE 4. Accuracy for small-medium size datasets with original text.

Dataset	SATCDM	SVM	SGD	NB
Abuaiadah(V1)	98.93%	97.15%	96.81%	95.93%
Aljazeera	98.77%	96.33%	96.60%	96.07%
Alwatan	98.11%	94.89%	92.01%	87.15%
Alkhaleej	97.58%	95.45%	94.06%	83.41%
OSAC	99.90%	99.32%	96.18%	92.68%
BBC	99.76%	90.30%	85.95%	62.69%
CNN	97.63%	94.38%	92.44%	73.39%

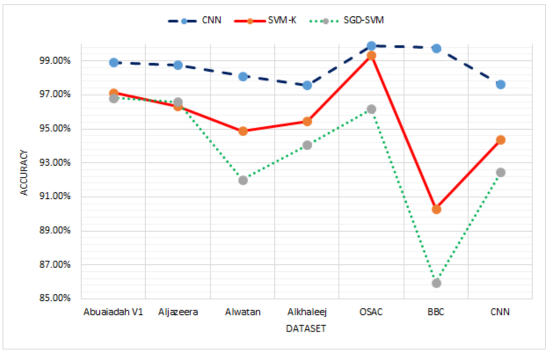

FIGURE 3. Accuracy for datasets with original text.

After training on the stratified 5-fold manner, then the following quantities are computed: Test Accuracy, Test Loss, Cross-Validation Average, Cross-Validation Standard Deviation for all folds.

VII. RESULTS AND DISCUSSION

This section lists and discusses the results of all experiments on all of the 15 datasets. There are mainly two experiments, as discussed previously in section VI; the first concerns traditional ML algorithms, including SVM, SGD, and NB, the other concerns the SATCDM model. Each of these experiments is applied to all 15 datasets. The results are presented in the three following subsections. After that, the results are compared to similar studies.

A. SMALL-MEDIUM SIZE DATASETS WITH ORIGINAL TEXT

The size of small-medium datasets with original text ranges from 1.5k to 22k text documents with a unique number of terms in the range 50k-678k. The results of the experiments on these datasets for all models are shown in table 4. The SATCDM dramatically outperforms the other models with accuracy ranging from 97.58% to 99.90%. Where SVM comes in second place, followed by SGD, and finally, the worst model was NB. This is clearly shown in figure 3.

More Information about the accuracy results is shown in table 5. The table assures that the results are stable since the test accuracy and validation accuracy are close to each other with reasonable standard deviation.

B. SMALL-MEDIUM SIZE DATASETS WITH STEMMED TEXT

The size of small-medium datasets with stemmed text ranges from 2.7k to 7.3k text documents with a unique number of terms in the range 14k-152k. The results of the experiments on these datasets for all models are shown in table 6. Again, the SATCDM dramatically outperforms the other models with accuracy ranging from 98.52% to 99.75%. Where SVM comes in the second place, followed by SGD and finally NB. This is clearly shown in figure 4. It is vital here to stress that SATCDM excels in classifying with very high accuracy in different types of text. For example, it performs nicely in text documents where stop-words are removed (Abuaiadah V2). Also, the accuracy of SATCDM was superior for text documents where light and heavy stemming algorithms are applied (Abuaiadah V3, V4 and V5, and NADA datasets). This clearly implies that SATCDM works perfectly with all shapes of text documents, including original, light-stemmed, heavy-stemmed with or without stop-words.

Again, the detailed results of the experiments shown in table 7 assures the stability of results since the test and validation accuracies are close to each other with reasonable standard deviation.

TABLE 5. More accuracy information for small-medium size datasets with original text for SATCDM.

Dataset	Test Acc.	Test Loss	Val. Acc. Avg.	Val. Acc. STD. DEV.
Abuaiadah(V1)	98.93%	0.030	98.97%	±0.155
Aljazeera	98.77%	0.031	99.09%	±0.061
Alwatan	98.11%	0.052	98.10%	±0.063
Alkhaleej	97.58%	0.091	96.85%	±0.625
OSAC	99.90%	0.003	99.92%	±0.008
BBC	99.76%	0.014	99.61%	±0.051
CNN	97.63%	0.065	97.68%	±0.251

TABLE 6. Accuracy for small-medium size datasets with stemmed text.

Dataset	SATCDM	SVM	SGD	NB
Abuaiadah(V2)	98.89%	97.15%	97.04%	96.44%
Abuaiadah(V3)	98.52%	97.11%	96.85%	96.30%
Abuaiadah(V4)	98.64%	97.07%	96.89%	96.04%
Abuaiadah(V5)	98.68%	96.67%	96.44%	95.22%
NADA	99.75%	99.40%	98.85%	82.43%

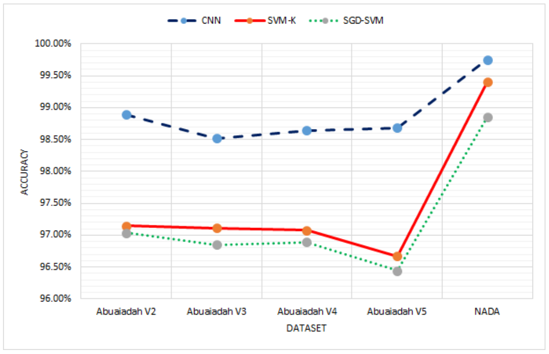

FIGURE 4. Accuracy for small-medium size with stemmed texts datasets.

TABLE 7. More accuracy information for small-medium size datasets with stemmed text for SATCDM.

Dataset	Test Acc.	Test Loss	Val. Acc. Avg.	Val. Acc. STD. DEV.
Abuaiadah(V2)	98.89%	0.029	98.82%	±0.265
Abuaiadah(V3)	98.52%	0.034	98.76%	±0.157
Abuaiadah(V4)	98.64%	0.036	98.60%	±0.309
Abuaiadah(V5)	98.68%	0.037	98.41%	±0.251
NADA	99.75%	0.006	99.81%	±0.036

TABLE 8. Accuracy for large-size datasets with original text.

Dataset	SATCDM	SVM	SGD	NB
Arabiya	99.49%	98.36%	95.62%	85.79%
Khaleej	99.57%	98.16%	96.87%	95.80%
Akhbarona	98.44%	94.88%	91.27%	87.45%
All-Combined	98.80%	96.02%	93.30%	90.75%

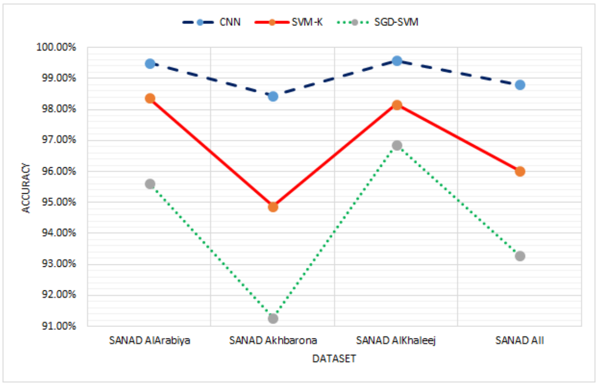

FIGURE 5. Accuracy for SANAD datasets.

TABLE 9. More accuracy information for large size datasets with original text for CNN model.

Dataset	Test Acc.	Test Loss	Val. Acc. Avg.	Val. Acc. STD. DEV.
Arabiya	99.49%	0.016	99.44%	±0.060
Khaleej	98.44%	0.046	98.25%	±0.130
Akhbarona	99.57%	0.014	99.41%	±0.029
All-Combined	98.80%	0.035	98.75%	±0.031

C. LARGE-SIZE DATASETS WITH ORIGINAL TEXT

The size of large-size datasets with original text ranges from 45.5k to 78.5k with a total of 195K text documents. The number of unique terms is in the range 429k-772k, with a non-redundant total of 1,295k text documents. The results of the experiments on these datasets for all models are shown in table 6. Once again, the SATCDM dramatically outperforms the other models with accuracy ranging from 98.44% to 99.57%. Where SVM comes in the second place, followed by SGD and finally NB. This is clearly shown in figure 5.

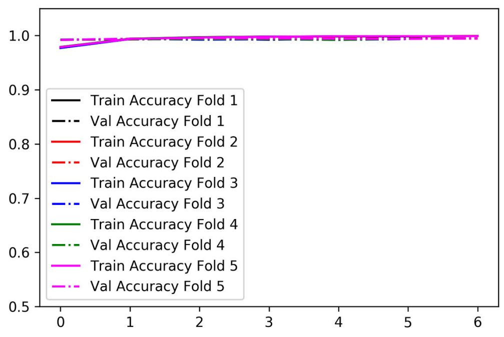

FIGURE 6. Training vs. validation accuracy for SANAD-AlKhaleej dataset.

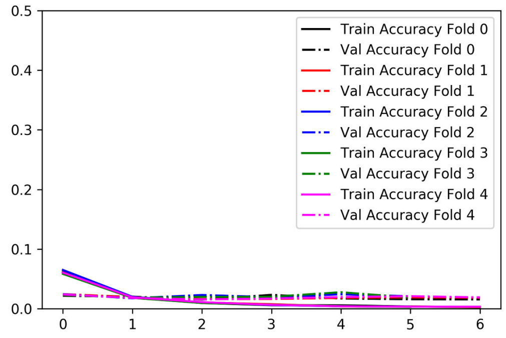

FIGURE 7. Training vs. validation loss for SANAD-AlKhaleej dataset.

TABLE 10. Accuracy comparison with NADA.

Dataset	SVM [44]	Our SVM	SATCDM
NADA	93.88%	99.40%	99.75%

D. SAMPLE OF ACCURACY AND LOSS

This subsection shows a sample of the training vs. validation accuracy and training vs. validation loss graphs. This is done for the five folds while training and validating the SATCDM model. The graph shows clearly the convergence between training and validation for both accuracy and loss. This implies that the model is robust to overfitting and hence will generalize well to unseen text documents.

E. NADA

This subsection compares the results of this study on NADA dataset with the results obtained by [44]. The authors argue that the low accuracy they obtained for NADA dataset (93.88%) is due to Abuaiadah dataset, where its classification accuracy was around 80%. However, our linear-kernel SVM classifier achieves accuracy between 96% and 97% for all versions of Abuaiadah dataset. Moreover, our CNN model achieved even better accuracy results around 98.50% for all versions of Abuaiadah dataset. For Nada, the accuracy results of our SVM and CNN models are shown in table 10. It is clear that the accuracy of our models with accuracy near 100% is superior to those obtained in [44].

TABLE 11. Accuracy for SANAD datasets.

Dataset	CNN [31], [32]	SATCDM
AlArabiya	96.05%	99.49%
AlKhaleej	95.89%	99.57%
Akhbarona	93.94%	98.44%

TABLE 12. Training time analysis in minutes.

Dataset	Nº Docs.	CNN	SVM	SGD	NB
Abuaiadah V1	2700	8	0.25	0.33	0.23
Aljazeera	1,500	4	0.11	0.12	0.10
Alwatan	20,291	28	3.95	4.49	3.78
Alkhaleej	5,690	10	1.63	1.75	1.63
OSAC	22,429	44	9.75	11.04	9.00
BBC	4,736	5	1.47	1.54	1.39
CNN	5070	12	1.48	1.54	1.37
Abuaiadah V2	2,700	7	0.23	0.27	0.19
Abuaiadah V3	2,700	8	0.19	0.24	0.16
Abuaiadah V4	2,700	8	0.20	0.24	0.17
Abuaiadah V5	2,700	8	0.14	0.18	0.14
NADA	7310	6	2.15	2.47	2.11
SANAD AlArabiya	71,246	43	22.83	24.19	22.62
SANAD Akhbarona	78,428	93	25.04	27.52	25.05
SANAD AlKhaleej	45,500	54	15.30	16.53	15.35
SANAD All	195,174	282	70.35	77.28	69.12

F. SANAD

In this section, the results of the experiments on SANAD datasets are discussed and compared to a similar study. The authors in [31], [32] achieved a maximum of 96% on their three datasets with a CNN model consisting of a dropout layer followed by three convolution layers each with a kernel size of size 5 and with 128 filters, followed by a global max-pooling layer and another dropout layer. The results of our model are superior to their results with a minimum of 98.44% and a maximum of 99.49%. Table 11 shows a comparison between the performance of both models.

G. TIME ANALYSIS

In this subsection the training time is reported in minutes for all experiments and for all models using 5-fold cross-validation method. This is shown in table 12. Please note that testing time was in seconds and some times fraction of a second. It's obvious from the results that traditional machine learning algorithms takes very short time. Also, notice that SVM, SGD and NB modeles take similar time for model building. In contrast, CNN model takes long time to train compared to traditional machine learning algorithms.

VIII. CONCLUSION

Deep learning became the sate-of-art for many research fields and applications. One such field is Text Categorization (TC), which is very important for many applications and studies. This study presents a new model for Arabic TC using CNN and word embedding that is called SATCDM. The model utilizes an efficient multi-kernel CNN architecture for TC and uses a skip-gram word embedding model enriched with sub-word information. The study uses 15 freely available Arabic datasets usually used for TC; these are different in sizes and use different preprocessing and normalization techniques.

The presented model achieves superior accuracy results compared to similar studies on the Arabic language and is suitable for any Arabic text documents regardless of normalization, preprocessing, stemming algorithms, or methods. Therefore, we argue that SATCDM will be of great benefit for many researchers in the field of ANLP and Information retrieval.

Although results are extremely outstanding, more research opportunities are available for more accuracy gain using other methods and models, including recurrent models such as Long Short-Term Memory (LSTM) and Gated recurrent units (GRUs) in addition to attention models.

REFERENCES

[1] A. Farghaly and K. Shaalan, "Arabic natural language processing: Challenges and solutions," *ACM Trans. Asian Lang. Inf. Process.*, vol. 8, no. 4, pp. 14:1–14:22, Dec. 2009.

[2] M. Saad and W. Ashour, "Arabic morphological tools for text mining," in *Proc. 6th Int. Symp. Electr. Electron. Eng. Comput. Sci.* Lefke, Cyprus: European University Lefke, 2010, pp. 112–117.

[3] A. Schmidt and M. Wiegand, "A survey on hate speech detection using natural language processing," in *Proc. 5th Int. Workshop Natural Lang. Process. Social Media*, 2017, pp. 1–10.

[4] X. Xue-Feng and Z. Guo-Dong, "A survey on deep learning for natural language processing," *Acta Autom. Sinica*, vol. 42, no. 10, pp. 1445–1465, 2016.

[5] I. Hmeidi, M. Al-Ayyoub, N. A. Abdulla, A. A. Almodawar, R. Abooraig, and N. A. Mahyoub, "Automatic Arabic text categorization: A comprehensive comparative study," *J. Inf. Sci.*, vol. 41, no. 1, pp. 114–124, Feb. 2015.

[6] A. H. Mohammad, T. Alwada'n, and O. Al-Momani, "Arabic text categorization using support vector machine, Naïve Bayes and neural network," *GSTF J. Comput.*, vol. 5, no. 1, p. 108, 2016.

[7] I. Hmeidi, B. Hawashin, and E. El-Qawasmeh, "Performance of KNN and SVM classifiers on full word Arabic articles," *Adv. Eng. Inform.*, vol. 22, no. 1, pp. 106–111, Jan. 2008.

[8] M. A. H. Madhfar and M. A. H. Al-Hagery, "Arabic text classification: A comparative approach using a big dataset," in *Proc. Int. Conf. Comput. Inf. Sci. (ICCIS)*, Apr. 2019, pp. 1–5.

[9] M. S. Khorsheed and A. O. Al-Thubaity, "Comparative evaluation of text classification techniques using a large diverse Arabic dataset," *Lang Resour. Eval.*, vol. 47, no. 2, pp. 513–538, Jun. 2013.

[10] Y. Kim, "Convolutional neural networks for sentence classification," 2014, *arXiv:1408.5882*. [Online]. Available: https://arxiv.org/abs/1408.5882

[11] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *Trans. Assoc. Comput. Linguistics*, vol. 5, pp. 135–146, Dec. 2017.

[12] D. Abuaiadah, J. E. Sana, and W. Abusalah, "Article: On the impact of dataset characteristics on Arabic document classification," *Int. J. Comput. Appl.*, vol. 101, no. 7, pp. 31–38, Sep. 2014.

[13] H. Sawaf, J. Zaplo, and H. Ney, "Statistical classification methods for Arabic news articles," in *Proc. Natural Lang. Process. (ACL)*. Toulouse, France: Citeseer, 2010.

[14] A. Abu-Errub, "Arabic text classification algorithm using TFIDF and chi square measurements," *Int. J. Comput. Appl.*, vol. 93, no. 6, pp. 40–45, May 2014.

[15] S. Al-Harbi, A. Almuhareb, A. Al-Thubaity, M. S. Khorsheed, and A. Al-Rajeh, "Automatic arabic text classification," in *Proc. 9th Journees Internationales d'Analys e statistique des Donnees Textuelles (JADT)*, vol. 9, 2008.

[16] A. M. El-Halees, "Arabic text classification using maximum entropy," *Arabic Text Classification Using Maximum Entropy*, vol. 15, no. 1, pp. 157–167, 2007.
[17] M. El Kourdi, A. Bensaid, and T.-E. Rachidi, "Automatic Arabic document categorization based on the Naïve Bayes algorithm," in *Proc. Workshop Comput. Approaches Arabic Script-Based Lang.*, 2004, pp. 51–58.
[18] T. F. Gharib, M. B. Habib, and Z. T. Fayed, "Arabic text classification using support vector machines," *IJ Comput. Appl.*, vol. 16, no. 4, pp. 192–199, 2009.
[19] G. Kanaan, R. Al-Shalabi, S. Ghwanmeh, and H. Al-Ma'adeed, "A comparison of text-classification techniques applied to Arabic text," *J. Amer. Soc. Inf. Sci. Technol.*, vol. 60, no. 9, pp. 1836–1844, Sep. 2009.
[20] T. Kanan and E. A. Fox, "Automated Arabic text classification with P-Stemmer, machine learning, and a tailored news article taxonomy," *J. Assoc. Inf. Sci. Technol.*, vol. 67, no. 11, pp. 2667–2683, Nov. 2016.
[21] A. H. Mohammad, O. Al-Momani, and T. Alwada'n, "Arabic text categorization using k-nearest neighbour, decision trees (C4. 5) and Rocchio classifier: A comparative study," *Int. J. Current Eng. Technol.*, vol. 6, no. 2, pp. 477–482, 2016.
[22] R. Alshammari, "Arabic text categorization using machine learning approaches," *Int. J. Adv. Comput. Sci. Appl.*, vol. 9, no. 3, pp. 1–5, 2018.
[23] M. K. Saad, "The impact of text preprocessing and term weighting on Arabic text classification," M.S. thesis, Islamic Univ.-Gaza, Gaza, Palestine, 2010.
[24] A. A. Altowayan and L. Tao, "Word embeddings for Arabic sentiment analysis," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2016, pp. 3820–3825.
[25] V. Jindal, "A personalized Markov clustering and deep learning approach for Arabic text categorization," in *Proc. ACL Student Res. Workshop*, 2016, pp. 145–151.
[26] A. El Mahdaouy, E. Gaussier, and S. O. El Alaoui, "Arabic text classification based on word and document embeddings," in *Proc. Int. Conf. Adv. Intell. Syst. Inform.* Cham, Switzerland: Springer, 2016, pp. 32–41.
[27] A. B. Soliman, K. Eissa, and S. R. El-Beltagy, "AraVec: A set of Arabic word embedding models for use in Arabic NLP," *Procedia Comput. Sci.*, vol. 117, pp. 256–265, Jan. 2017.
[28] A. Dahou, S. Xiong, J. Zhou, M. H. Haddoud, and P. Duan, "Word embeddings and convolutional neural network for Arabic sentiment classification," in *Proc. 26th Int. Conf. Comput. Linguistics (COLING)*, 2016, pp. 2418–2427.
[29] M. Sayed, R. Salem, and A. E. Khedr, "Accuracy evaluation of Arabic text classification," in *Proc. 12th Int. Conf. Comput. Eng. Syst. (ICCES)*, Dec. 2017, pp. 365–370.
[30] M. Biniz, S. Boukil, F. El Adnani, L. Cherrat, and A. E. El Moutaouakkil, "Arabic text classification using deep learning techniques," *Int. J. Grid Distrib. Comput.*, vol. 11, pp. 103–114, 09 2018.
[31] A. Elnagar, O. Einea, and R. Al-Debsi, "Automatic text tagging of Arabic news articles using ensemble deep learning models," in *Proc. 3rd Int. Conf. Natural Lang. Speech Process.*, 2019, pp. 59–66.
[32] A. Elnagar, R. Al-Debsi, and O. Einea, "Arabic text classification using deep learning models," *Inf. Process. Manage.*, vol. 57, no. 1, 2020, Art. no. 102121.
[33] M. Al-Ayyoub, A. Nuseir, K. Alsmearat, Y. Jararweh, and B. Gupta, "Deep learning for Arabic NLP: A survey," *J. Comput. Sci.*, vol. 26, pp. 522–531, May 2018.
[34] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
[35] M. A. Nielsen, *Neural Networks and Deep Learning*, vol. 25. San Francisco, CA, USA: Determination Press, 2015.
[36] M. Sarigül, B. Ozyildirim, and M. Avci, "Differential convolutional neural network," *Neural Netw.*, vol. 116, pp. 279–287, Aug. 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0893608019301315
[37] D. Said, N. M. Wanas, N. M. Darwish, and N. Hegazy, "A study of text preprocessing tools for Arabic text categorization," in *Proc. 2nd Int. Conf. Arabic Lang.*, 2009, pp. 230–236.
[38] M. Abbas, K. Smaïli, and D. Berkani, "Evaluation of topic identification methods on Arabic corpora," *J. Digit. Inf. Manage.*, vol. 9, no. 5, pp. 185–192, 2011.
[39] M. Abbas and K. Smaili, "Comparison of topic identification methods for Arabic language," in *Proc. Int. Conf. Recent Adv. Natural Lang. Process. (RANLP)*, 2005, pp. 14–17.
[40] M. K. Saad and W. Ashour, "OSAC: Open source Arabic corpora," in *Proc. 6th ArchEng Int. Symp., 6th Int. Symp. Elect. Electron. Eng. Comput. Sci. (EEECS)*, vol. 10, 2010, pp. 118–123.
[41] L. S. Larkey, L. Ballesteros, and M. E. Connell, *Light Stemming for Arabic Information Retrieval*. Dordrecht, The Netherlands: Springer, 2007, pp. 221–243.
[42] A. Chen and F. C. Gey, "Building an Arabic stemmer for information retrieval," in *Proc. TREC*, 2002, pp. 631–639.
[43] S. Khoja and R. Garside, "Stemming Arabic text," M.S. thesis, Dept. Comput., Lancaster Univ., Lancaster, U.K., 1999.
[44] N. Alalyani and S. L. Marie-Sainte, "NADA: New Arabic dataset for text classification," *Int. J. Adv. Comput. Sci. Appl.*, vol. 9, no. 9, pp. 1–7, 2018.
[45] O. Einea, A. Elnagar, and R. Al Debsi, "SANAD: Single-label Arabic news articles dataset for automatic text categorization," *Data Brief*, vol. 25, Aug. 2019, Art. no. 104076.
[46] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," 2013, *arXiv:1301.3781*. [Online]. Available: https://arxiv.org/abs/1301.3781
[47] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2014, pp. 1532–1543.
[48] G. Khazal *et al.*, "Arabic word embedding method for NLP," in *Proc. Inf. Technol. Math. Modeling (ITMM)*, vol. 6, 2019, pp. 99–105.
[49] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 3111–3119.
[50] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*. [Online]. Available: https://arxiv.org/abs/1412.6980
[51] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, 2014.
[52] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," 2015, *arXiv:1502.03167*. [Online]. Available: https://arxiv.org/abs/1502.03167
[53] T. Joachims, "Text categorization with support vector machines: Learning with many relevant features," in *Proc. Eur. Conf. Mach. Learn.* Berlin, Germany: Springer, 1998, pp. 137–142.

M. ALHAWARAT (Member, IEEE) received the Ph.D. degree in chaotic spiking neural networks from Oxford Brookes University, in June 2007. He is currently working as an Associate Professor of computer science with the Department of Computer Science, Prince Sattam Bin Abdulaziz University, Saudi Arabia. His research interests include chaotic neural networks, machine learning, Arabic NLP, and text mining. He is a Professional Member of ACM.

AHMAD O. ASEERI is currently an Assistant Professor with the Department of Computer Science, College of Computer Engineering and Sciences, Prince Sattam Bin Abdulaziz University, Saudi Arabia. His main researches focus are in the area of deep learning with application to neural network-based risk analysis and deep learning for computer vision, data mining with application to clustering techniques, including bisecting K-means clustering (BKM), limited-iteration bisecting K-means (LIBKM), and memory-aware clustering algorithms. His research interests include artificial intelligence with application to natural language processing, knowledge representation and reasoning, and deep learning for medical applications.