International Journal of Informatics and Communication Technology (IJ-ICT)
Vol.9, No.2, August 2020, pp. 73~82
ISSN: 2252-8776, DOI: 10.11591/ijict.v9i2.pp73-82

Neural network dealing with Arabic language

Adnan Souri[1], Mohammed Al Achhab[2], Badr Eddine Elmohajir[3], Abdelali Zbakh[4]
[1,2,3]MPNT Center for Doctoral Studies, Abdelmalek Essaadi University, Morocco
[4]Faculty of Sciences, Mohammed V University, Morocco

Article Info

Article history:

Received Aug 14, 2019
Revised Dec 12, 2019
Accepted Jan 12, 2020

Keywords:

Arabic language
Natural language processing
Recurrent neural networks
Text generation
Text prediction

ABSTRACT

Artificial Neural Networks have proved their efficiency in a large number of research domains. In this paper, we have applied Artificial Neural Networks on Arabic text to prove correct language modeling, text generation, and missing text prediction. In one hand, we have adapted Recurrent Neural Networks architectures to model Arabic language in order to generate correct Arabic sequences. In the other hand, Convolutional Neural Networks have been parameterized, basing on some specific features of Arabic, to predict missing text in Arabic documents. We have demonstrated the power of our adapted models in generating and predicting correct Arabic text comparing to the standard model. The model had been trained and tested on known free Arabic datasets. Results have been promising with sufficient accuracy.

Corresponding Author:

Adnan Souri,
MPNT Center for Doctoral Studies,
Abdelmalek Essaadi University,
Avenue Khenifra, Tétouan 93000, Maroko.
Email: adnan.souri@gmail.com

1. INTRODUCTION

The need to Arabic language resources is increasing in several fields such as text summarization, information retrieval and machine translation [1, 2]. Moreover, Arabic information amount in the Web is growing daily. However, Arabic language still suffers from a lot of lack in the field of Natural Language Processing (NLP) [3, 4]. For these reasons, it can be said that Arabic language is in the top interests of NLP research domains [1]. One of the recent and promising research domains of NLP is the application of Artificial Neural Networks (ANN) on language models.

In this paper, we have used both Reccurrent Neural Networks (RNN) and Convolutional Neural Networks (CNN) to prove learning process and to measure text comprehension. RNN have been used to generate acceptable sequences of text, i.e: correct Arabic text, while CNN have been used to predict missing text from some Arabic documents. Our idea is based on child language learning process, especially learning words meanings and expression meanings. This process matches ideally with the RNN operating principle. We recall here the words of Ibn Taymiya in his book "Al Iman" (The Faith, page 76 as shown in Figure 1.): If the discrimination appears from the child, he heard his parents or his educators utter verbally, and refer to the meaning, he understand so that word is used in that meaning, i.e. : the speaker wanted that meaning [8].

By analogy to this, RNN models take a text dataset at their inputs and try to learn the meaning by training on. At the output, RNN models produce new sequences of text according to their learning process. The success of the learning process increases while increasing the quantity of input data and increasing the training operation, too. Otherwise, if given an incomplete text, it is more appropriate to apply CNN language models on the text. CNN models have the ability to predict missing sequences from text by training on some dataset.

In this paper, we used the Long-Short Term Memory (LSTM) model, as it is a more tool equipped neural network, to deal with Arabic text generation. The choice of LSTM model was motivated by its ability in steps memorization, which was a required task for our experiments while generating text at each step. In another side, given Arabic language features and specificities, the standard architecture of RNN was not suitable for our test requirements on Arabic text. Our model had been so built basing on standard LSTM definition as described in [9]. Moreover, we modified the model to support some Arabic language features such as word schemes and the non-adjacency of letters. We fed up our model by these features in its input. The main challenge of our contribution was to prove that our modification on the LSTM model dealing with Arabic text gives a satisfactory accuracy result.

Once done, the operation of text generation motivates us to contribute at missing text prediction, which is more interesting in order to complete text of some documents such as old manuscripts for example. For this, we have adapted a CNN model to deal with Arabic features and then we have applied it on Arabic language. Dealing with a notable number of documents, CNN model has showed a satisfactiry accuracy of prediction.

The organization of this document is as follows. In Section 2 (Related Work), we present some work dealing with both Reccurrent Neural Networks and Convolutional Neural Networks. In Section 3 (Neural Networks), we define RNN and CNN, and we present how the models are used in text processing. In Section 4 (Experiments), we present our experiments inluding preparing data, creating the models, generating Arabic text sequences using RNN, and predicting missing text using CNN. We present then our results. In Section 5 (Conclusion), we conclude our research works as well as we discuss some further application as perspectives.

فالمولود إذا ظهر منه التمييز، سمع أبويه أو من يربيه ينطق باللفظ، ويشير إلى المعنى، فصار يفهم أن ذلك اللفظ يستعمل في ذلك المعنى، أي: أراد المتكلم به ذلك المعنى،

Figure 1. Excerpt from Ibn Taymiya's book "Al Iman". Page 76.

2. RELATED WORK

From one side, several works have used RNN to deal with Arabic text. In [12], authors use a bidirectional LSTM model. The model is introduced as a character-to-word model that takes as input character-level representation of a word and generates vector representation of the word. Moreover, a word–character hybrid language model had been applied on Chinese using a neural network language model in [13]. A deep neural network produced high performance part-of-speech taggers in [14]. The network learns character-level representation of words and associates them with usual word representations. In [15], authors use RNN models to predict characters based on the character and word level inputs. In [16], authors present word–character hybrid neural machine translation systems that consult the character-level information for rare words.

From other side, a considerable amount of work has dealt with CNN application on Arabic language as well as on other languages. In [17], authors propose to auto-encode text a byte-level using CNN with recursive architecture. Experiments had been done on datasets in Arabic, Chinese and English. The motivation was to explore whether it is possible to have scalable and homogeneous text generation at byte-level in a non-sequential fashion through the simple task of autoencoding. The work showed that non-sequential text generation from a fixed-lengths representation is not only possible, but also acheived much better auto-encoding results than the use of RNN. Reference [18] has used CNN as a deep learning classifier for Arabic scene text recognition. Authors assume that such an approach is more appropriate in cursive scripts. Thus, their model had been applied to learn patterns of visual images in which Arabic text was written. The experimental results indicate that CNN can improve accuracy on large and variant datasets.

Moreover, the work presented in this paper involves as continuity to our previous work dealing with Recurrent Neural Networks application on Arabic language. In [3], we assumed that Arabic scripts have numerous challenges associated such as (1) the variant shape of letters depending on their position in a word, (2) the problem of writing units, (3) the length of sentences due to little use of punctuation marks, (4) the lack of space between sentence components in usual cases, and (5) the lack of vowelization. For this, it involves being harder to deal with Arabic script. Therefore, by keeping in view these features in Arabic script, we have used our segmentation method proposed in [3]. The segmentation method helps in partitioning text into units that will be converted in a next step into vectors (numerical representations) that will serve as input to the CNN model.

3. NEURAL NETWORKS

3.1. Recurrent neural networks (RNN)

Recurrent neural networks (RNN) are sets of nodes, with inputs and outputs, linked together for the purpose of communicating and extracting results that respond to specific problems such as sequences generation [19, 20]. RNNs highlight is the large number of hidden layers, between inputs and outputs, that exchange information from and towards inputs and outputs nodes each time step in order to give more performing results.

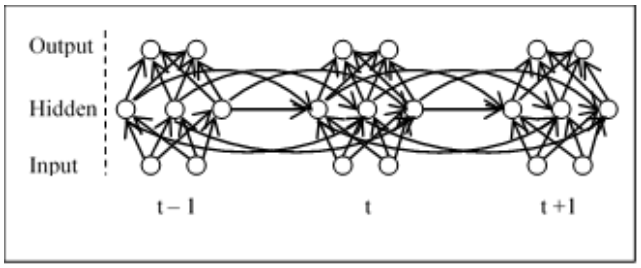

Figure 2. A Recurrent neural network is a very deep feedforward network whose weights are shared across time. Hidden nodes activate a non-linear function that is the source of the RNN's rich dynamics

In general, RNN are able to generate sequences of arbitrary complexity, but are unable to memorize information about past inputs for very long [8]. This memorization task helps to formulate better predictions and to recover from past mistakes. An effective solution will be then another kind of architecture designed to be better at storing and accessing information than standard RNN.

Long-Short Term Memory (LSTM) is a RNN architecture, equipped with memory cells, that has recently given state-of-the-art results in a variety of sequence processing. It is both used as a predictive and a generative model; it can learn the sequences of a given text, in its input, and then generate new possible sequences by making predictions. In principle, to predict the next element, RNN use the hidden layer function; an element wise application of a sigmoid function. LSTM do, too. Moreover, LSTM are better at finding and exploiting long-range dependencies in the data [20]. The LSTM model definition had been inspired from [9] judged as a basic reference. It is based on equations below:

$$\sigma_t = (W_o[h_{t-1}, x_t]) + b_o \quad (1)$$

$$f_t = \sigma\big(W_f[h_{t-1}, x_t]\big) + b_f \quad (2)$$

$$i_t = \sigma(W_i[h_{t-1}, x_t]) + b_i \quad (3)$$

$$\check{C} = tanh(W_c[h_{t-1}, x_t]) + b_c \quad (4)$$

$$C_t = f_t * C_{t-1} + i_t * \check{C}t \quad (5)$$

$$h_t = o_t * tanh(C_t) \quad (6)$$

$$y_t = soft\,max\big(W_{hy}h_t\big) \quad (7)$$

Figure 3 illustrates the representation of one LSTM cell. It shows how the prediction process is turning on.

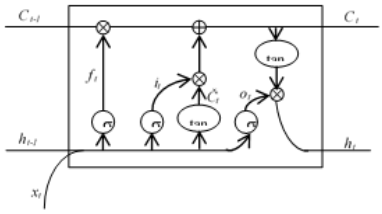

Figure 3. A LSTM cell modelisation showing the prediction process architecture using equations presented above

Briefly, previous equations assume the LSTM model is required to compute the hidden state at a time step (t). It is also able to decide whether to forget (ft) the previous hidden state and to update the current state using the previous state. Moreover, LSTM is able to compute the temporal cell state (Čt) for the current time step using the tanh activation function as well as to compute the *actual* cell state (Ct) for current time step, using the forget gate and input gate. Intuitively, doing so makes LSTM be able to keep only the necessary information and forget the unnecessary one.

The computation of the current cell state is then used to compute the current hidden state. Consequently, comes the computation of the *actual* output (yt).

3.2. Convolutional neural networks (CNNs)

CNN are feedforward networks based on combination of input layers, hidden layers (convolution layers, max pooling, drop out) and output layers. Deep CNN-models are powerful in terms of learning hierarchical representation of the data, and extracting the features from raw data. Their powerful is illustrated by the manner on which the layers are connected. The connection between layers requires fewer parameters and efficiency of CNN-models requires fewer operations. A commonly used CNN-model architecture is presented in Figure 4. The architecture of a CNN can be detailed as follows:

- Input layer: It serves for feeding the model by raw data to process the network.
- Convolution layers: Layers within the attributes: Input and Number of filters. The input is a tensor determining quantity of data (number, width, height, and depth). While the number of filters is caracterized by width, height and depth, a convolutional operation is carried out by the layer and its result is passed to the next layer.
- Convolution filter: Small spatial window with width and height smaller than the width and height of the input data.
- Rectified linear unit (ReLU): An element-wise activation function g(z) = max(0; z) which is applied on the input data. It changes its value while preserving its spatial dimensions in output.
- Pooling layers: They combine the ouputs if neuron clusters at a layer into a single neuron in the next layer in order to reduce data dimensions. In general, pooling computes a max or an average. Max pooling conserves the maximum value from each cluster of a neuron while average pooling conserves the average value.
- Fully connected layers: They connect all-to-all, i.e. every neuron in a layer is connected to every neuron in another layer.

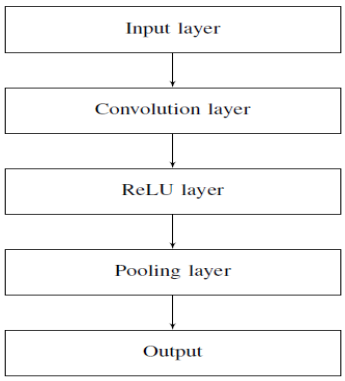

Figure 4. Typical CNN architecture

4. EXPERIMENTS

4.1. Arabic features

The creation of the LSTM model is based on its definition as cited in paragraph III (Recurrent Neural Networks). Moreover, as additional inputs, we added two gates respecting some Arabic language criteria. It is a kind of rule-based method. Our idea is to feed the model by (A) schemes meaning and (B) letters non-adjacency principle. The application of this idea gave more performance to text generation process. We explain below the advantages we can draw from (A) and (B).

- Schemes meaning is one of the highlights of the Arabic language. We can get the meaning of such a word for example just by interpreting its scheme meaning and without having known the word before. The word < "كاتب": author > has the scheme "فاعل", which means that the word refers to someone who is responsible of the writing act. In like manner, the word < "جالس": sitting > has also the scheme "فاعل", which means that it refers to someone who is responsible of the sitting act and so on. Table 1 below shows some of schemes meaning we used in our LSTM model implementation.

Table 1. The association scheme-meanings

Schemes	Translitteration	The associate meaning
فاعل	fAîl	The subject, the responsible of such an action
مفعول	mafôl	The effect of an action
مِفعَلة	mifâala	A noun of an instrument, a machine
فَعلة	faâla	Something done for once

- The principle of letters non-adjacency indicates what letter cannot be adjacent (before or after) to another letter. It is due to pronunciation criteria in Arabic. We mention here the couple (ع, خ). These two letters cannot be adjacent (ع before خ by respecting this order for next couples, too) in a word or a writing unit. Our idea was then proposed to reduce the tuning of prediction proceeding by elimination. So, once the model in front of the ع letter, it cannot predict the خ letter. Couples like (غ, ع), (د, ض) and (ص, س) respect the same rule.

4.2. Data preparation

At the starting of every work dealing with a huge quantity of data, the operation of data preparation remains to be a necessary task. We have first freely downloaded several text documents in portable document format (pdf) from three sources on the web: Arab World Book (AWB) [21], Shamela Library (ShL) [22], and Hindawi organisation (HND) [23]. We have collected several text novels and poems of some Arab authors. The global size of our dataset was about 130 MB of text, distributed over 144 text files with more than four million ns of words. Table 2 gives an overview about our dataset.

Table 2. An overview of some documents used in this research

Dataset	Number of documents	Number of words
AWB	38	1009365
ShL	67	2133442
HND	39	1082727
TOTAL	144	4225534

- Arab World Books is a cultural club and Arabic bookstore that aims to promote Arab thought, provide a public service for writers and intellectuals (http://www.arabworldbooks.com/index.html.).
- Hindawi Foundation is a non-profit organization that seeks to make a significant impact on the world of knowledge. The Foundation is also working to create the largest Arabic library containing the most important books of modern Arab heritage.
- Shamela is a huge free program that aimed, to be comprehensive for all what the researcher needs of books and research. The library is currently working on a suitable system for receiving files of various texts and arranging them in one frame with the possiblity of searching.

All novels and poems have been in a PDF file format with a global size of 127MB. We proceeded by converting these files to a text format using "Free PDF to Text Converter" tool available at: http://www.01net.com/telecharger/windows/Multimedia/scanner_ocr/fiches/115026.html.

Table 3 lists some of documents we have used in our experiments and their authors respectively. We have assigned an ID for each document to designate it during the Python implementation phase.

Table 3. Some documents and authors used in this research

Document title	Author name	ID
The days	TahaHussein	HND_TH_1
The misers	Al Jahid	ShL_JHD_1
Homeland	Mahmoud Darweesh	HND_MD_1
Diwan	Maarof Rosafi	AWB_MR_2

4.3. RNN application

First, our model reads the text file and then split the content into characters. The characters then are stored in a vector v_char, which represents data. In a next step, we store unique values of data in another vector v_data. Information about features gates is stored in associative tables scheme_meaning and nadj_letters. The two tables fed up the model by schemes word meanings and by non-adjacency letter specifications. As learning algorithm deals with numeric training data, we choose to assign an index (numerical value) to each data character. Once done, variables v_char, v_data, scheme_meaning and nadj_letters form the input of the LSTM model. To complete the model, we created the model with three LSTM layers; each layer has 700 hidden states, with Dropout ratio 0.3 at the first LSTM layer.

Under those circumstances, we have implemented our model under Python programming language using Keras API with TensorFlow library as backend. We present briefly Keras and TensorFlow.

Written in Python, Keras is a high-level neural networks API. It can be running whether on top of TensorFlow, Theano or CNTK. Implementation with Keras leads to results from idea with the least possible delay, which enables fast experimentation comparing to other tools (http://www.keras.io).

Using data flow graphs, TensorFlow is an open source software library dedicated to numerical computation (http://www.tensorflow.org.). Mathematical operations are represented by graph nodes while multidimensional data arrays (tensors) are represented by edges communicating between them (http://www.tensorflow.org.). This flexible architecture allows deploying computation to one or more CPUs or GPUs in a device with a single API. TensorFlow had been developed for the purposes of conducting machine learning and deep neural networks research. Thus, the system is general enough to be applicable in a wide variety of other domains as well (http://www.tensorflow.org.).

In our case, we deployed computation to one CPU machine. We discuss next material criteria and performance concerning time execution. Three cases had been evaluated to validate our approach and to calculate the accuracy given by our proposed method:

- Case 1: LSTM applied on Arabic text: We applied the standard LSTM architecture on Arabic text and tested it on our dataset.
- Case 2: Gated LSTM applied on Arabic text: As originality of this paper, we added two gates to the LSTM model dealing with two Arabic features in order to give more performance to text generation process and to compare with the case (1) above.
- Case 3: LSTM applied on English text and on Chinese text: Moreover, we applied the standard LSTM architecture on our dataset translated to English and Chinese in order to realise a kind of accuracy comparison.

Experiments have been performed on a PC using a single core i5 3.6 GHz CPU either for case 1, 2 and 3 above. We have encountered some encoding problems due to Arabic. We have used both utf-8 encoding to encode and decode “Hindawi” texts and Windows-1256 encoding for “Arab World Book” texts.

We trained our model using the data we prepared above. We launched training about a hundred times during 2 weeks. The model is slow to train (about 600 seconds per epoch on our CPU PC) because of data size and because of materiel performance. In addition to this slowness, we require more optimization, so we have used model check pointing to record model weights after each 10 epochs. Likewise, we observed the loss at the end of the epoch. The best set of weights (lowest loss) is used to instantiate our generative model.

After running the training algorithm, we gather 500 epochs each in a HDF5 file. We keep the one of the smallest loss value. We used it then to generate Arabic text. First, we define the model in the same way as in paragraph C (Creating model), except model weights are loading from the checkpoint file. The lowest loss encountered was 1.43 at the last epoch. We have used then the file to generate text after training.

4.3.1. Results

Our purpose in this work is to prove the ability of recurrent networks to generate Arabic text. But, generation itself is not the main goal of our research domain; it is just a first step to make prediction of some missing text from several documents. That said, we have proposed a gated LSTM model respecting some Arabic features we judged important at this level of research. Below, we present some results from our three cases of experiments: Figure 5 illustrates the loss function behavior after each 10 epochs of applying the

model on Arabic text dataset. We show values between epoch 140 and 240. The curve keep the same shape (tending to zero) while applying the model on English and Chinese. Surely, the standard model gives more accuracy for English than Arabic, because of the model, in his standard architecture, is more suited to Latin languages than other languages. Thus, we attend a notable accuracy concerning loss function which we present in Table 4. To attend more accuracy applying our model on Arabic text, we have built our gated model that gave a lower loss function value (0.73) after 500 epochs. We show in Table 5 the comparison between both standard model application and gated model application on Arabic text.

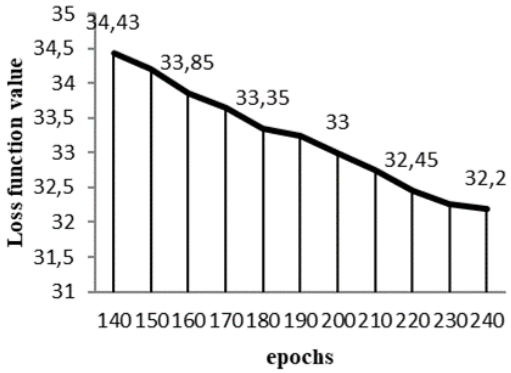

Figure 5. The shape of loss function curve for some arbitrarily chosen epochs

Table 4. Minimal loss function value while applying standard LSTM on different languages.

Standard LSTM	Languages		
	Arabic	Chinese	English
Loss function value	1.43	2.13	1.2

Table 5. Minimal loss function value while applying both standard and gated LSTM models on arabic

RNN model	Loss value	epoch
Standard LSTM	1.43	500
Gated LSTM	0.73	500

4.4. CNN application

Once data are cleaned up, we proceed by the operation of dividing it into three subsets: Training set (TrD), Validation set (VD), and Test set (TsD). The training process is an operation that consists of teaching the CNN model how to write the Arabic text, the categories of words in Arabic, the particularities of Arabic (especially those taken into consideration by our research), morphology, grammar, conjugation, and semantics. That being said, the model learns by browsing a multitude of documents written in Arabic while remembering, among other things, the order of words and the composition of the pahrases. At the end of the training operation, the model as enough luggage to be able to generate Arabic text or to predict it.We then proceed with the validation operation, which consists in evaluating the learning of the model by giving it documents already processed but this time with missing text.

The model, therefore, must predict the missing text and we compare with the original text and calculate the accuracy of the results. The step of test then comes up by feeding the model by new documents with missing text. These documents have not been treated by the model at all. The CNN-model try to predict text basing on its learning. AS in the most state of the art of data preparation, TrD took about 70% of data, i.e. 94 document files of the 144. VD and TsD took each one around 15% of data, i.e. 25 document files each. Table 6 shows the distribution of documents and words per dataset for each source of the three.

Table 6. Distribution of number of documents and words per each dataset

Dataset	TrD		VD		TsD	
AWB	24	762910	7	178235	7	158220
ShL	45	1544197	11	289077	11	300168
HND	25	810873	7	163448	7	108406
TOTAL	94	3117980	25	630760	25	566794

In the following, we provide details about our preprocessing strategy for text datasets, prior to feeding information to the CNN-model. Text already prepared to be as input to the CNN-model had been transformed into numerical codes since CNN architecture needs numerical data at inputs. The transformation had been carried out according to the following steps:

a. Dividing text into writing units (WU) separated by space.
b. Attributing a unique numerical ID to each WU.
c. For each ID, we have calculated its binary equivalent code. We have called it Binary Writing Unit Code (BWUC).
d. Creating a dictionary associating BWUC, which are unique, to their respective WU.

Another parametrizing step consists in representing each BWUC in a fixed dimensional vector (v) of size k, where (2k = vocabulary size). The elements of input feature vector (iv) are the associated BWUC of successive WU (wui) in a text document (D), such that: iv = (wu1, wu2, wu3, wu4, ..., wuN). The succession of N wui in a text necessarily generates a unique WU (or with higher accuracy at least, with other WU with lower accuracy) which will improve the prediction process. The iv is fed into the CNN-model and, at the output, the next WU is given, i.e. a vector (v) of N elements (wu1, wu2, wu3, wu4, ..., wuN) leads to the prediction of the next WU which will be wuN+1.

To reduce document browzing and to gain in terms of execution time, we no longer limited to the use of vectors, but rather we opted for matrices (rows and columns). We have created a matrix M containing a number N of BWUCs in its columns. N is determined depending the performance of the prediction results (we have evaluated N = 3, N = 4, and N = 5). The order of elements of a row in M is the same as the appearance of WU in text. The matrix M associated to the excerpt from text <alwatan> of the document HND MD 1 is illustrated by the table VI The number of rows (nr) of M is determined by : nr = nWU=N. where nWU is the total number of WU in the whole text. Each row in M contains N columns, and each column is an element (Mij, where i refers to the (i + 1)th line and j refers to the (j + 1)th column in M) referring to a WU. We have adjusted lenghts of all WU by completing these lengths by the empty character to have the same lenght for all WU.

The next step involves creating a column vector Y containing one element per each row. Y has the same nr as M. The correspondance between M and Y is that after each group of elemnets in a row of M, we find necessarily the element in the same row of Y, i.e: after each group of N WU in the text, we have the WU which is referred by the elemnt in Y. Table VII shows the vector Y correponding to matrix M presented in Table VI. We created then M codes, an equivalent matrix to M, and Y codes an equivalent vector to Y. M codes and Y codes serve as inputs for the CNN-model. M codes and Y codes contain BWUC of WU in text.

4.4.1. Implementation

We have implemented our model under Python programming language using Keras API with TensorFlow library as backend. Keras is a high-level neural networks API written with Python. It provides functionalities to deal with deep learning models. TensorFlow is a Python library used for symbolic numrical computations. Symbolic expressions in TnsorFlow define computational architectures such as CNN models.

The structure of our CNN model consists of an input layer, two convolution layers with ReLU as nonlinear activation function, two max pooling layers, and a fully connected layer. And last, the output layer.

4.4.2. Results

Our experiments are based on three essential processes; training, validation and test. The training, validation and test operations were carried out in two stages according to three different processes in order to analyze and interpret the results of the missing text prediction, and also aiming at improving the overall accuracy of this prediction. First, we started the training on the documents of a single author, we repeat the process for each author, we validate on documents already treated in training and then we test our results on documents that will be provided to CNN-model for the first time. The second stage is to provide documents from different authors but from the same source, since each of the three sources (AWB, ShL and HND) had its own priorities of classifying documents, choosing documents, choosing topics, putting up topics, and choosing authors obviously. We carry out the same process concerning training, validation and test.

Stage 1: We have proceeded by training the CNN model on the documents of the same author. Generally, each author is characterized by his writing style, his method of describing facts or expressing ideas and more other features. These features have been taken into consideration by the model when it predicts the missing text in a document. When the model has to predict a missing text from a document of Mahmoud Darweesh for example, the text is often in relation with the notions of the homeland, the earth, and love, so text of these domains has a haigher rate to be predicted rather than text of other domain. However, if dealing with Nabil Farouq’s documents, we find that his writings, as a whole, are in touch with the world of crime and the security of states, so the higher probability of prediciton corresponds to text of these domains. The prediction of

missing texts on the processed documents had been carried out on the basis of both a statistical and a probabilistic approach. The model calculates the rate of a WU appearance after N other WU in a text. In the same text, even of the same autor, we can find after 5 WU for example the appearance of multiple WU depending on the context. The model calculates then the probability of each WU appearance (PWUA) and predicts the missing text according to the higher probability. At this experimental stage, results have shown a high level of overall accuracy attending a hundred percent in several cases. Given these results, our interpretation is that the model did learn the author's writing style to the point where he can correctly predict the missing text of his writings with a widely acceptable error rate. We noticed another aspect of prediction that was not part of our research is that the model can predict even formatting (text in bold, underlined text, text in color).

This can be the subject of a future work. For the test, we propose to our CNN-model documents with missing text, always of the same author, but this time these documents have never been treated by the CNN-model neither at the training stage nor at the validation stage. The model responded satisfactorily as the predicted results reached a maximum of 92.8% as overall accuracy. Results are represented in Table 7.

Table 7. Prediction overall accuracy per each author

Author	TrD	VD		TsD	
		ND	POA (%)	ND	POA (%)
Taha Hussein	22	5	93.0	5	92.8
Jabran Khalil Jabran	14	4	86.9	4	83.7
Ghassan Kanafani	8	2	80.0	2	78.4
May Ziyada	9	3	80.1	3	80.0
Mahmoud Darweesh	19	4	90.3	4	89.4
Najeeb Mahfod	17	4	88.1	4	87.9
Maarfo Rosafi	3	2	65.4	2	60.3
Al Sulayti	2	1	77.2	1	62.8
TOTAL	94	25	82.62	25	79.41

As fisrt observation, prediction accuracy seems to have proportional relation with the amount of both trained and tested text. TableIX shows that more text amount is higher (in both training step or validation step) more prediction overall accuracy (POA) is higher (in both validation step or test step). This proves that the CNN-model learning is more interesting when the amount of data is more important. So we try next to feed up the CNN-model by a larger quantity of data and calculate the accuracy.

2) Stage 2: The second stage took place while keeping in mind that each data source is different from the other. We tried then, in this step, to evaluate the prediction of the missing text belonging to the same data source, without taking into consideration the author of the text, with the purpose of assuming if the prediction is realized by following the same criteria as in the first stage (prediction by domain, writing style, and conservation of formatting). The model had been fed up by texts from the same source of any confused author. We did carry out the training, the validation and the test in the same way as for the first stage. The model had been trained on AWB TrD set, provides AWB VD set texts for validation, then had been tested with AWB TsD set texts. The same operations are repeated for texts of ShL apart and texts of HND apart. The prediction results of this stage are described in the Table 8 where the number of documents used is presented by ND.

Table 8. Prediction overall accuracy per each data source

Author	TrD	VD		TsD	
		ND	POA (%)	ND	POA (%)
AWB	24	7	97.3	7	94.8
ShL	45	11	98.1	11	96.9
HND	25	7	98.4	7	95.6
TOTAL	94	25	97.93	25	95.77

Certainly, at this experimental stage we have achieved satisfactory results in terms of POA. It is clear and obvious that POA, at the validation step (98.4 as maximum), is higher than POA calculated at the test step (96.9), since the VD texts have already been processed by the model while the TsD texts are treated by the model for the first time. The POA at this stage is even higher compared to the first stage, knowing that the variant that is discussed at this level is the amount of text provided in each of the two stages. We partially conclude that the amount of text provided during the training and test steps is proportional with the POA calculated at the validation step.

5. CONCLUSION

In this paper, Neural Networks language-models have been applied on Arabic text. To give satisfactory results, models have been slightly modified to respect some Arabic language features. Experiments, in one hand, have applied RNN-model (LSTM precisely) on Arabic text in order to generate correct sequences in Arabic. In the other hand, we applied CNN-model to predict missing text from some documents. The text generation had been promising at the level of generating words then sentences. It had the impact to push this research to predict text in some sequences applying CNN-model, which is one of best solutions for automatic learning using the raw text data directly.

In this work, our CNN-model depends on both words and characters of Arabic language to encode text data, which involves in dealing with large input vectors while using word encoding. The CNN-model initialization requires then adequate weight parametrizing, but in our experiments, we have opted for low level representation of text data to reduce the input feature vector. We have observed that the performance and the accuracy of the model had been influenced by the way we feed up the CNN at inputs. Our model had achieved a competitive accuracy and our results have shown the powerful use of CNN and its ability to predict Arabic missing text using a statistical and probabilistic approach.

Our work has shown limitations on the amount of input data. Indeed, at a number of fifty MB the machine crashes, knowing that we have worked on a CPU. We will adapt our algorithm and our model to work on a GPU and therefore we can discuss more efficient results.

REFERENCES

[1] Khurana D., Koli A., Khatter K., and Singh S., "Natural language processing: State of the art, current trends and challenges," 2017. arXiv preprint arXiv:1708.05148, 2017.
[2] Souri A., Alachhab M. and El mohajir B.E., "A study towards a building an Arabic corpus (ArbCo)," *The Second National Symposium on Arabic Language Engineering (JDILA 2015),* 2015.
[3] Souri A., Alachhab M. and El mohajir B.E., "A proposed approach for Arabic language segmentation," *First International Conference on Arabic Computational Linguistics*. Cairo, Egypt, pp. 43-50, 2015.
[4] Elarnaoty M., AbdelRahman S. and Fahmy A., "A machine learning approach for opinion holder extraction in Arabic language," *International Journal of Artificial Intelligence Applications*, vol. 3, pp. 45-63, 2012.
[5] Chang Y. and Lee K., "Bayesian feature selection for sparse topic model. *IEEE International Workshop on Machine Learning for Signal Processing*, China, pp. 1-6, 2011.
[6] Faria L., Akbik A., Sierman B. and Ras M. "Automatic preservation watch using information extraction on the Web: a case study on semantic extraction of natural language for digital preservation," *10th International Conference on Preservation of Digital Objects*, Lisbon, Portugal, 2013.
[7] Alghamdi H.M., Selamat A., Abdul Karim N.S., "Arabic web pages clustering and annotation using semantic class features," *Journal of King Saud University – Computer and Information Sciences*, vol. 26, pp. 388-397, 2014.
[8] Ibn Taymiya. 1996. *Book of Al Iman*. 5th ed.
[9] Hochreiter S., Schmidhuber J., "Long short-term memory". *Neural computation*, vol. 9, no. 8, pp. 1735-1780, 1997.
[10] Józefowicz R., Vinyals O., Schuster M., Shazeer N. and Wu Y., "Exploring the limits of language modelling," 2016. arXiv preprint, 1602.02410 arxiv.org/abs/1602.02410.
[11] Zoph B., Vaswani A., May J. and Knight K., "Simple, fast noise-contrastive estimation for large rnn vocabularies," *The 15th Annual Conference of the North American Chapter of the Association for Computational Linguistics,* 2016.
[12] Ling W., Luís T., Marujo L., Astudillo R.F., Amir S., Dyer C., Black A.W. and Trancoso I. "Finding function in form: Compositional character models for open vocabulary word representation, *Conference on Empirical Methods in Natural Language Processing.* Lisbon, Portugal, 2015.
[13] Kang M., Ng T. and Nguyen L. "Mandarin word-character hybridinput neural network language model," *The 12th Annual Conference: International Speech Communication Association*, pp. 625-628, 2011.
[14] Dos Santos C.N. and Zadrozny B., "Learning character-level representations for part-of-speech tagging," *The Proceeding of the 31st International Conference on Machine Learning*. Beijing, China, pp. 1818-1826, 2014.
[15] Bojanowski P., Joulin A., and Mikolov T., "Alternative structures for character-level RNNs," 2015.
[16] Luong M.T., and Manning C.D., "Achieving open vocabulary neural machine translation with hybrid word-character models," *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, German pp. 1054-1063, 2016.
[17] Zhang Z., Xie Y., and Yang L. "Photographic text-toimage synthesis with a hierarchically-nested adversarial network". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. pp. 6199-6208. 2018.
[18] Ahmed S. B., Malik Z., Razzak M. I. "Sub-sampling Approach for Unconstrained Arabic Scene Text Analysis by Implicit Segmentation based Deep Learning Classifier". *Global Journal of Computer Science and Technology*, 2019.
[19] Sutskever I., Martens J. and Hinton G. "Generating text with recurrent neural networks," *The 28th International Conference on Machine Learning.* Washington, USA, 2011.
[20] Graves A., "Generating sequences with recurrent neural networks, 2013. arXiv preprint arXiv:1308.0850.
[21] https://www.arabworldbooks.com/
[22] http://shamela.ws/index.php/main
[23] https://www.hindawi.org/