Image-based Character Embedding for Arabic Document Classification

Mahmoud Daif Shunsuke Kitada Hitoshi Iyatomi

Major in Applied Informatics, Graduate School of Science and Engineering, Hosei University
{mahmoud.daif.8h@stu., shunsuke.kitada.8y@stu., iyatomi@}hosei.ac.jp

Abstract

This work introduces an image-based character embdeddings model for Arabic text classification. The problem with Arabic text classification using classical techniques is its dependency on complex morphological analysis and hand crafted feature engineering. Using character-level embeddings eliminates the need for complex morphological analysis and feature engineering. We propose a new Arabic document classification model using the CE-CLCNN, where text is represented as an array of character images, and the class-balanced loss. The CE-CLCNN consists of two parts, a character autoencoder (CE) and a character-level CNN (CLCNN). The CE learns to encode image based character embeddings, and the CLCNN is then used to classify the document using those embeddings. We created two datasets to test the effectiveness of our system. The first one is the Arabic Wikipedia title dataset (AWT), and the second one is the Arabic Poetry dataset (APD). The proposed model outperforms the classical SVM by 21.23% and 13.82% for the micro and macro F-score respectively for the ADP dataset, and by 4.02% and 3.95% for the AWT dataset. To the best of our knowledge, this is the first time an image based character embedding model was used to address the problem of Arabic text classification. Also, the first time a text classification deep learning model is tested on datasets that contain the three types of Arabic.

1 Introduction

Arabic is one of the six official languages of the United Nations and the official language of 28 states. Arabic is spoken by as many as 422 million people making it the fifth most popular language around the world. In addition to that, classical or old Arabic is the official religious language of 1.8 billion Muslims worldwide. Arabic has 28 basic characters, 15 with dots and 13 without. Arabic has 3 different vowels and is written from right to left. According to the Internet World Statistics, as of 2017, Arab users represent 4.8% of internet users[1].

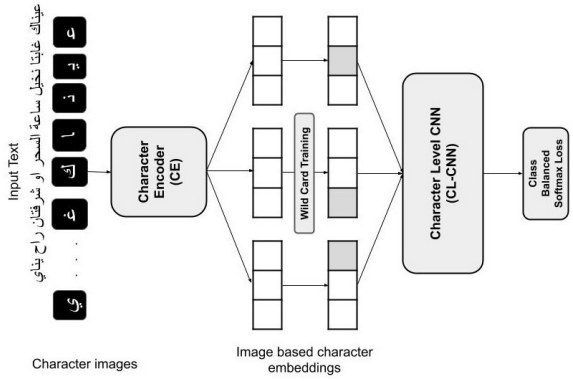


Figure 1: CE-CLCNN, end-to-end image-based text classification model composed of Character Encoder (CE) and Character CNN (CLCNN).

Text classification importance has recently increased due to the increase of textual data on the internet like social network services (SNS) and news. It has many applications like sentiment analysis, spam detection and news categorization. Arabic text classification is particularly challenging because of its complex morphological analysis. Arabic can be classified into three different types each having its own purpose and morphology. The modern standard Arabic, colloquial or dialectal Arabic and the classical or old Arabic. The modern standard Arabic is the official language used in media, government, news papers and is taught in schools. Colloquial Arabic varies between countries and regions. Old or classical Arabic survives nowadays in religious scriptures and old poetry.

Most conventional research on Arabic text classification have used classical techniques in their feature extraction and classification [1]. Feature extraction techniques for text classification require complex morphological analysis. Such as part of speech tagging [2], stemming [3], and segmentation [4]. Arabic segmentation is especially complex because unlike

[1]Arabic Speaking Internet Users and Population Statistics. https://www.internet-worldstats.com/stats19.html Accessed: 16-Dec-2018,

languages like English, Arabic words are not always separated by white spaces, some words could be separated into several words. It also includes embedding process with some hand crafted features like document term matrix with tf-idf scores or word count and negation handling.

Arabic text classification have been often done using classical algorithms like support vector machines (SVM) or naive bayes [1]. Those conventional combinations left two major issues that could not be ignored. First, the performance highly depends on the performance of morphological analysis, which is quite difficult on Arabic. The same problem is indicated and addressed in some languages such as Japanese [5]. Second, obtaining appropriate embedding (i.e. building hand crafted features) is difficult.

To solve those problems, several character-based approaches have been proposed utilizing deep learning techniques used mainly in image processing [6, 7, 8].

Zhang et al. [6] introduced a character-level CNN (CLCNN) that treats text as a raw signal at character level. The CNN then learns the language morphology and extracts appropriate features for text classification. Their method mitigated the issue of complex morphological analysis.

Later Shimada et al. [7] proposed image based character embeddings for text classification. They were the first to handle a character as an image and obtained character-embedding with their character encoder (CE) which is an auto-encoder. Their method not only reduces morphological analysis difficulty, but also obtaining efficient embeddings.

Their system had the advantage of extracting shape features for Japanese and Chinese characters. It is also suitable for extracting features from text on social media sites containing a lot of "emojis" and image based text. In addition to that, their method has the advantage of overcoming some of the difficulties in data augmentation by using wild card training. They train both the character encoder (CE) and CL-CNN separately.

Kitada et al. [8] proposed CE-CLCNN that concatenated Shimada's CE and CL-CNN so as to make the system an end-to-end fashion and introduced random erasing on image domain as another data augmentation method. They demonstrated state-of-the art document classification performance on several common dataset.

Another problem with text classification is that large datasets usually suffer from long tailed data distribution problem. Meaning few classes make up most of data, while minority classes have far fewer candidates, which often reduces the learning algorithm's accuracy on the minority classes.

To solve the long tailed distribution issue, Cui et al. [9] introduced class balanced loss based on effective number of classes which re-weights the loss by the inverse of effective number of classes. Other techniques like oversampling were also introduced to deal with the same problem. Basically this problem is addressed either by using re-sampling or re-weighting the cost function.

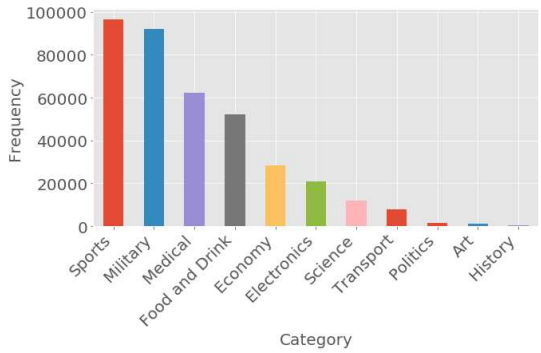
Our proposed framework for Arabic text classification adopts the CE-CLCNN end-to-end model of character encoder and character CNN classifier with some modifications to tune the model to Arabic language as well as using the class balanced loss to solve the class imbalance problem. We introduce two datasets. The first one is the Wikipedia Title dataset and the second one is the Arabic Poetry dataset. Those two datasets contain the three different types of Arabic. They were introduced in the hope of being a benchmark for Arabic text classification algorithms.

To the best of our knowledge, this is the first time an image based character embedding model was used to address the problem of Arabic text classification. Also, the first time a deep learning based model is tested on datasets that contain the three types of Arabic language. This shows that our method could be used to overcome Arabic's complicated morphological analysis with all kinds of Arabic.

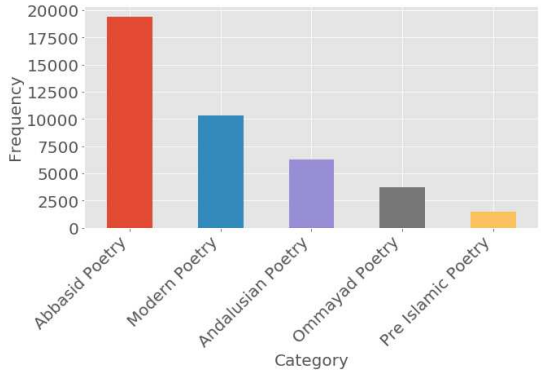
2 Datasets

The problem with Arabic NLP is the absence bench marking datasets like it's the case in other languages like English, Japanese and Chinese for example. This is due to how expensive it is and time consuming to annotate a dataset for text classification, and the fact that deep learning algorithms require relatively large datasets. Our solution to this problem is crawling two large enough datasets that do not require manual annotation. Sections 2 - 2 describe how we construct both datasets. Hopefully our datasets would be a benchmark for Arabic text classification. The first one is the Arabic Wikipedia Title (AWT) dataset and the second one is the Arabic Poetry (APD) dataset.

Arabic Wikipedia Title Dataset Liu et al. [10] introduced the Wikipedia Title dataset for Japanese, Chinese and Korean by making use of Wikipedia's recursive hierarchical structure to crawl 12 different Wikipedia categories and assigning them as label to all article titles under this category, its subcategories and its subcategories subcategories and so on. We crawl 11 different categories of the Arabic Wikipedia. A total of 444,911 different titles with a number of 4,196,127 different words were crawled. Details of the dataset category distribution can be found in



(a) Arabic Wikipedia Title Dataset (AWT)



(b) Arabic Poetry Dataset (APD)

Figure 2: The category distribution for the ATW dataset and APD dataset.

Figure 2a. We assume that an article would exist in one category. Meaning that if an article was under both categories we assign it to one of them. This is somewhat a source of noisy annotations, however we chose categories as distinctive in nature as possible to reduce this problem.

Arabic Poetry Dataset The Arabic Poetry Dataset crawled from Adab Website[2] contains Arabic poetry from the 6th to 21st centuries unlike the Wikipedia title dataset which contains only modern standard Arabic. The APD dataset can test the proposed framework's ability to learn the different kinds of Arabic languages. The dataset consists of 41,264 poems of five different kinds of poetry from different centuries. Details about category distribution can be found in Figure 2b.

3 Methodology

We adopt a similar end-to-end model of a character encoder and a character CNN as the one used by Kitada et al. [8]. The architecture of our model can be found in Figure 1. Details of the character encoder and character CNN can be found in sections 3–3. Wildcard training introduced by Shimada et al. [7] is used as a data augmentation method. Since both our datasets are both unbalanced, we used class balanced loss which is shown to improve performance.

[2]Adab website for Arabic poetry from 6th to 21st centuries. `http://www.adab.com/`.

Table 1: Character Encoder (CE) Architecture

Layer	Configuration
Conv2D	(c= 1, k = 3x3, f=32) + ReLU
Max-Pool2D	(k=2x2)
Conv2D	(c=32, k = 3x3, f=32) + ReLU
Max-Pool2D	(k=2x2)
Conv2D	(c=32, k = 3x3, f=32) + ReLU
Fully Connected	(800,128) + ReLU
Fully Connected	(128,128) + ReLU

Table 2: Character Level Convolutional Neural Network (CLCNN) Architecture

Layer	Configuration
Conv1D	(c= 128, k = 3, f=512) + ReLU
Max-Pool1D	(k=3)
Conv1D	(c=512, k=3, f=512) + ReLU
Max-Pool1D	(k=3)
Conv1D	(c=512, k = 3, f=512) + ReLU
Conv1D	(c=512, k = 3, f=512) + ReLU
Fully Connected	(1024,1024) + ReLU
Fully Connected	(1024,128) + ReLU

CE The character encoder (CE) is a convolutional auto-encoder where the convolution is performed in a depth wise manner. The architecture can be found in Table 1. Each character image of size 36x36 pixels is encoded into a 128 dimension vector.

CLCNN The CLCNN is a CNN classifier for an input text of an array of characters where each character is encoded into 128 dimension vector. The detailed architecture of the CLCNN can be found in Table 2.

Class Balanced Loss Both our datasets suffer from the long tailed distribution problem as shown in Figure 2a and 2b. To deal with that we use state-of-the-art method, the class balanced loss [9]. The idea is to increase the loss value for classes with fewer number of candidates and decrease the loss for classes with a larger number of candidates, so one mistake becomes more costly for minority classes.

4 Experiments

To train our classifier both datasets are divided into 80% training and 20% testing. The maximum char-

Table 3: Classification results.

Model	Wikipedia Arabic Title		Arabic Postry	
	Micro F-score [%]	Macro F-score [%]	Micro F-score [%]	Macro F-score [%]
SVM	45.47	26.60	52.80	34.83
CLCNN w/ Onehot encoding	42.76	18.71	68.24	37.72
CE-CLCNN	47.47	26.85	**74.86**	45.61
CE-CLCNN w/ Class Balanced Softmax	**49.49**	**30.55**	74.03	**48.65**

acter length for the AWT dataset is set to 60 characters and 128 for the APD dataset. Each character is encoded into a 128 dimension vector. Adam optimizer is chosen as the optimization method with a batch size of 64, using a learning rate of 0.001. The training loss converges after approximately 40 epochs for both datasets. As a baseline, two models were used. The first is the classical SVM. As a pre-processing stage for the classical SVM, none Arabic characters, diacritics, and Arabic stop words are removed. After that the text was stemmed using khoja stemmer [2]. As for the SVM features we used a document term matrix of unigrams with TF-IDF scores. Farasa segmenter [4] was used for segmenting Arabic text. The other baseline is using the CLCNN with character one hot encodes as input. This model was used to evaluate the effect of the CE part of our network.

5 Results and Discussion

Classification results can be found in Table 3. It is noted that the classification results for CE-CLCNN exceeds those of the classical SVM without the need for the pre-processing, segmentation, stemming and feature engineering associated with classical methods. The proposed method outperforms the CLCNN with one-hot encoding as input as well. This proves that encoding Arabic character images through the CE improves performance. This kind of encoding was originally intended for characters with visual features like Chinese and Japanese. The CE is still useful for Arabic because it solves the problem of finding the best character embeddings for the classification problem. This is because both the CE and the CLCNN are trained as an end-to-end system, so the CE learns the best embeddings suitable for the CLCNN. Also, using class balanced softmax loss improves the macro F-score for both datasets which means that our model learns minority classes better which helps solving the class imbalance problem.

6 Conclusion and Future Work

The proposed image-based character embeddings framework has been shown to achieve good results for Arabic document classification. However there is more room for improvement especially in the character encoder part. It could be further tuned and redesigned to handle dependencies between Arabic characters.

References

[1] S. A. Salloum, A. Q. AlHamad, M. Al-Emran, and K. Shaalan, "A survey of arabic text mining," in *Intelligent Natural Language Processing: Trends and Applications*. Springer, 2018, pp. 417–431.

[2] S. Khoja, "Apt: Arabic part-of-speech tagger," in *Proc. of the Student Workshop at NAACL*, 2001, pp. 20–25.

[3] M. N. Al-Kabi, S. A. Kazakzeh, B. M. A. Ata, S. A. Al-Rababah, and I. M. Alsmadi, "A novel root based arabic stemmer," *Journal of King Saud University-Computer and Information Sciences*, vol. 27, no. 2, pp. 94–103, 2015.

[4] A. Abdelali, K. Darwish, N. Durrani, and H. Mubarak, "Farasa: A fast and furious segmenter for arabic," in *Proc. of NAACL*, 2016, pp. 11–16.

[5] F. Peng, X. Huang, D. Schuurmans, and S. Wang, "Text classification in asian languages without word segmentation," in *Proc. IRAL workshop*, 2003, pp. 41–48.

[6] X. Zhang, J. Zhao, and Y. LeCun, "Character-level convolutional networks for text classification," in *Proc. of NIPS*, 2015, pp. 649–657.

[7] D. Shimada, R. Kotani, and H. Iyatomi, "Document classification through image-based character embedding and wildcard training," in *Proc. of IEEE Big Data*. IEEE, 2016, pp. 3922–3927.

[8] S. Kitada, R. Kotani, and H. Iyatomi, "End-to-end text classification via image-based embedding using character-level networks," in *Proc. of IEEE Applied Imagery Pattern Recognition Workshop (AIPR)*. IEEE, 2018, pp. 1–4.

[9] Y. Cui, M. Jia, T.-Y. Lin, Y. Song, and S. Belongie, "Class-balanced loss based on effective number of samples," in *Proc. of CVPR*, 2019, pp. 9268–9277.

[10] F. Liu, H. Lu, C. Lo, and G. Neubig, "Learning character-level compositionality with visual features," in *Proc. of ACL*, 2017, pp. 2059–2068.