Layout Analysis on Challenging Historical Arabic Manuscripts using Siamese Network

1st Reem Alaasam
The Department of Computer Science
Ben-Gurion University of the Negev
Beer Sheva, Israel
rym@post.bgu.ac.il

2nd Berat Kurar Barakat
The Department of Computer Science
Ben-Gurion University of the Negev
Beer Sheva, Israel
berat@post.bgu.ac.il

3rd Given Name Surname
The Department of Computer Science
Ben-Gurion University of the Negev
Beer Sheva, Israel
el-sana@cs.bgu.ac.il

***Abstract*—This paper presents layout analysis for historical Arabic documents using siamese network. Given pages from different documents, we divide them into patches of similar sizes. We train a siamese network model that takes as an input a pair of patches and gives as an output a distance that corresponds to the similarity between the two patches. We used the trained model to calculate a distance matrix which in turn is used to cluster the patches of a page as either main-text, side-text or a background patch. We evaluate our method on challenging historical Arabic manuscripts dataset and report the fmeasure. We show the effectiveness of our method by comparing with other works that use deep learning approaches, and show that we have the state of art results.**

***Index Terms*—layout analysis, siamese network, historical Arabic documents, clustering**

I. Introduction

Historical handwritten documents have many important information for scholars to study. Many libraries around the world are digitizing their documents to make them available for more people and to preserve their physical copies from deterioration. To access the contents of these documents there is a need for document image processing algorithms, since the raw form of a document image is not machine-readable.

Document layout analysis is the process of identifying and classifying the various regions in a page image. It is an important preprocessing step for many document image processing tasks.

Using deep learning algorithms gave state of art in many computer vision fields, and it is shown that learned feature are stronger than handcrafted features [1], therefore we decided to use deep learning algorithm for layout analysis.

In this paper we present layout analysis method for historical handwritten Arabic documents using a siamese network. Siamese network consists of two identical convolutional neural networks (CNN). It takes as an input a pair of images or patches, extracts their features and ranks the similarity between them. Given pages of a historical Arabic manuscript, we segment them to patches of similar size and train siamese network model. Using the trained model we build a distance matrix among the patches of each testing page. Then we use the distance matrix to cluster the patches into three classes:main-text, side-text and background. We evaluate our work on challenging historical Arabic manuscripts dataset. The dataset was first introduced by [2] and used later in theses work [3] and [4], it contains various writing styles within complex layout.

Rest of the paper is organized as follows, we review the related work on page layout analysis in section II, then describe the method in section III. Section IV shows the results and in Section V we draw concluding remarks.

II. Related Work

Page segmentation algorithms are divided into two types: bottom-up and top-down algorithm. Bottom-up algorithms aggregate elements into regions. Elements can be any part of a page image such as pixels, patches or connected components. Top-down algorithms segment a whole page into regions.

Standard page layout analysis algorithms are applied to modern binary documents in a top-down manner and have assumptions regarding the document structure. These algorithms could be applied to documents that have a Manhattan layout, which means that the regions are rectangles with vertical and horizontal lines.

Classification or clustering based algorithms are the most popular type of algorithms [5]. Below we include some of these algorithms which are applied on historical documents. They work in a bottom-up manner and are applied to a colored or gray-scale document image. These algorithms work for any documents without any assumptions concerning the layout, text alignment and text orientation.

Texture is a low-level feature, which is used to describe regularity and coarseness. Extraction texture features can be done by using Grey Level Co-occurrence Matrix (GLCM), Gabor filters, auto-correlation function and etc. Mehri et al. [6] and Journet et al. [7] use texture clustering for page segmentation. They use a sliding window to extract texture attributes for each pixel. Segmented regions are build by clustering together the pixels that form homogeneous regions.

Bukhari et al. [2] consider relative distance, foreground area, orientation, normalized height and neighborhood information of the connected components as features. Then to classify connected components into side note and main body texts a Multilayer Perceptron (MLP) is used. However, this approach is outperformed by Asi et al. [3]. They proposed a learning free approach to detect the main text area. Using Gabor texture

filter they segment the main text and update segmentation by minimizing an energy function. Their function gives a higher probability to closer pairs of components and tries to assign them to the same label.

Wei et al. [8] treat the segmentation problem as a pixel classification problem. Each pixel is represented as a vector of features based on the color of the image. Then Gaussian Mixture Models (GMM), Multi-Layer Perceptrons (MLP) and SVM are used to classify the pixels into decoration, background, periphery and text pixels. SVM and MLP are shown to generally outperformed GMM in the segmentation problem. However, Chen et al. [9] outperform this work by representing each pixel with more features. They use more texture and color features such as smoothness, Laplacian, Gabor Dominant Orientation Histogram, Local Binary Patterns and color variance. In addition, they use a feature selection algorithm to remove irrelevant features. A similar experiment was carried out by Wei et al. [10] with an improved feature selection algorithm. This algorithm is the combination of the genetic selection and the greedy forward selection. They showed that feature selection decreases the size of the feature vector and improves the performance with significant features.

Chen et al. [11] further study the problem of extracting significant features. They use unsupervised learning method, which is convolutional autoencoder instead of handcrafted features which are used by the algorithms above. Given unlabeled training set Convolutional autoencoder learns feature extractor on a randomly selected set of image patches. The feature extractor then used to train SVM that can classify the pixels into background, periphery, decoration pixels and text block. Comparing to their previous work [9], Wei et al. [12] show superior improvement. They increased the classification accuracy and reduce the dimension of the features by using a feature selection algorithm based on [10].

Kurar et al. [4] present binarization free Layout Analysis using fully Convolutional Network. They train an FCN to predict the class of each pixel. They evaluate their work on challenging historical Arabic manuscripts dataset. The same dataset is used in [3] and [2]. In this paper, we also use the same dataset to evaluate our method. Kurar et al. [4]' results are comparable with [3], in addition they present a binarization free method.

III. Method

Our method is composed of three steps, the first one is converting our dataset which consists of images of pages, to a dataset of patches taken from each page image from the original dataset. The second step is training siamese network model that can predict the similarity between any two patches from our dataset. The third step is building a distance matrix using the trained model and classifying each patch of the same page to one of the following classes: main-text, side-text or background.

A. Siamese network

Siamese network was firstly introduced in [13], it consists of two branches that share the same Convolutional Neural Network (CNN) architecture and the same weights. The input is a pair of images and the output is a distance in the range $[0, 1]$, which corresponds to the similarity of the input pair.

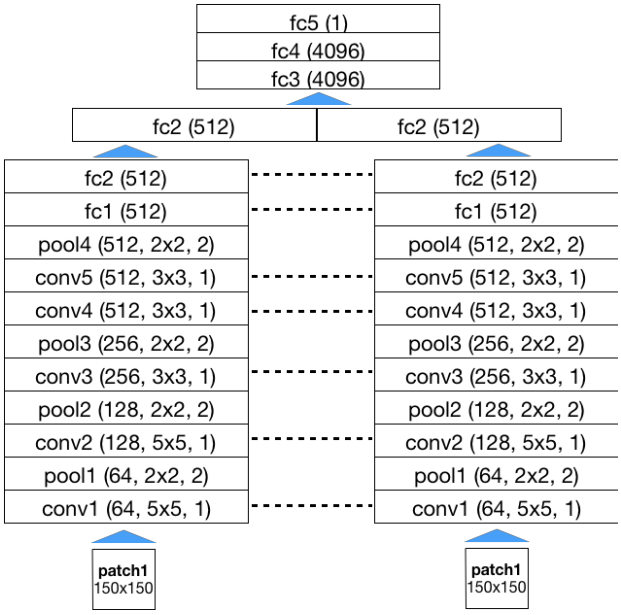

Fig. 1. Siamese architecture for pair similarity. Dotted lines stand for identical weights, conv stands for convolutional layer, fc stands for fully connected layer and pool is a max pooling layer.

Many hyperparameters in CNN architecture are not learned and are given manually. Starting from hyperparameters that are taken from a similar task is preferable [14]. Hence we first based the branches of the siamese network model on Alexnet [15] and through experiments we tune the hyperparameter to fit our task. The final architecture that we use is shown in Figure 1. It contains two branches of CNN, each of the branches has five convolutional layers. Dotted lines indicate identical weights. The numbers in parentheses are the number of filters, filter size and stride. All convolutional and fully connected layers are followed by ReLU activation functions except fc5 which feeds into a sigmoid binary classifier. The learning rate is 0.00001 and the optimizing algorithm we use is ADAM.

We trained this model from scratch and reached 90% accuracy (Figure 2) on validation set patches. The dataset that is used is explained in more details in the following section.

B. Data preparation

Training and testing were done with challenging historical Arabic manuscripts dataset which was first used in [2], and later it was also used in other work [3] and [4]. The dataset contains various writing styles and different layout structures as shown in Figure 3. It contains 32 documents from 7 different historical Arabic manuscripts. It is available online

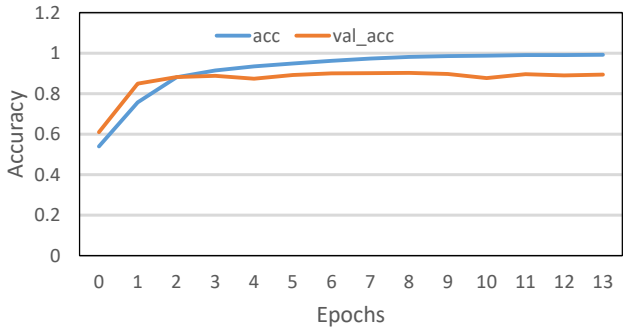

Fig. 2. Accuracy over the epochs of model training.

for downloading[1]. We used 24 pages for training and 8 pages for testing following the split used in [4], since we compare our work with them. Main text and side text are labeled in pixel level.

Fig. 3. three pages from the train set

We did not preprocess the images for size or mean normalization since one of the stated goals was to train a model that would be resistant to non-binarized document images. We generated patches of 150×150 pixels using a sliding window over images of pages. In our dataset the average size of a page image is 2800×3900, hence each page gives us on average around 470 patches. If a patch contains more than 23% main-text labeled pixels, it is labeled as main-text patch, else if it contains more than 23% side-text labeled pixels, it is labeled as side-text patch, else it is labeled as background patch (Figure 4). We picked the percent through experiments.

As mentioned before the input for a siamese network is a pair of images, in our case, it is a pair of patches. To prepare the pairs, we first take each patch from the same class and pair it with all the other patches of the same class. This method generates the maximum number of positive pairs, which mean pairs that are composed of patches of the same class. For each positive pair a negative pair was generated by randomly pairing patches from different classes. The number of positive and negative pairs are equal. This guarantees that the dataset is balanced. For all the data in train and validation set this process of pairing is used.

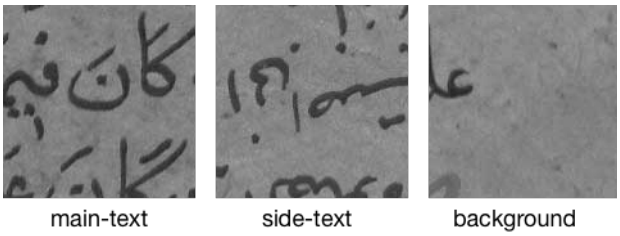

Fig. 4. Sample patches from each class.

C. Clustering

Hierarchical cluster is a nested clusters set which are represented as a tree. It transforms proximity matrix into a sequence of nested clusters [16]. Generally hierarchical cluster falls into two types of strategies [17]:

- Divisive clustering: this is a top-down approach, which starts as one cluster, then splits recursively while moving down in the hierarchy.
- Agglomerative clustering: this is a bottom-up approach, in which each element starts in its own cluster, then merge pairs of clusters while moving up in the hierarchy.

In this paper we use agglomerative clustering, which clusters the patches of one page using distance matrix.

D. Evaluation

To evaluate our method we use 8 pages (around 3800 patches) taken from three different manuscripts. For each test page we first divide it to patches of size 150×150 using a sliding window. Then using the trained siamese model we build a distance matrix between all the patches of the same page, which in average are 470 patches. The distance matrix is fed to the agglomerative clustering algorithm, which for each patch assigns one of the classes: main-text, side-text or background. We apply postprocessing after clustering the patches, which we explain in the postprocessing section. For each segmented page we compute F-measure and compare our work with [2] and [4], we discuss this in more details in section IV.

E. Postprocessing

After clustering we apply postprocessing step that refines the labels of a page' patches. For each patch in a page image, we consider its 8-neighbors, if the label of the patch is not background, then it is labelled the same as the majority label of the 8-neighbors. In case of equal number of neighbors from main-text and side-text we consider two patches from left, right, up and down, and label the patch according to majority. An example is shown in Figure 5.

[1]https://www.cs.bgu.ac.il/ berat

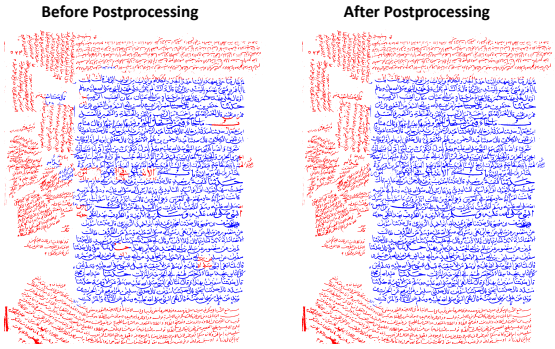

Fig. 5. A page image before and after postprocessing

IV. Results

We test our method using 8 pages taken from different manuscripts, which is the same test set used by [2] and [4]. The test set contains different layout structures and various writing styles since it is taken from different manuscripts. For each page, we segment it to patches and build distance matrix using the trained siamese model. This distance matrix is used to cluster the patches into three classes, then we calculate the F-measure for all the pages in the test set. In the following section, we explain in more details the F-measure.

A. Metrics

We evaluate the performance of our method by measuring the F-measure metric, since it is the metric that is used in the works we compare with [2] and [4]. F-measure outputs a single scalar after combining recall and precision values. Recall value is in the range $[0, 1]$, a perfect score of 1 indicates that all relevant pixels were predicted. However, it does not provide any information regarding the number of false predictions. Precision value is also in the range $[0, 1]$, a perfect score of 1 indicates that no pixel was falsely predicted. However, it does not provide any information regarding whether all relevant pixels were predicted. F-measure combine these two values and guarantees that both of them are high. Recall and precision are calculated according to the following equations:

$$Recall = \frac{TP}{TP + FN}$$
$$Precision = \frac{TP}{TP + FP}$$

where

- True-Positive(TP): number of pixels of class C_i predicted as pixels of class C_i.
- False-Positive(FP): number of pixels of class C_i predicted as pixels of class C_j.
- False-Negative(FN):number of pixels of class C_j predicted as pixels of class C_i.

TABLE I
Comparison with F-measures, MT(Main Text) and ST(Side Text)

	MT F-measures(%)	ST F-measures(%)
Bukhari et al [2]	95.02	94.68
Karur et al. [4]	95	80
Proposed method	**98.59**	**96.89**

Using the recall and precision values, F-measure is calculated according to the following equation:

$$Fmeasure = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

We provide F-measure score for both main-text and side-text.

B. Comparison

We compare our work with [2] and [4], both of these works are done on the same dataset and they are also binarization free approaches. Bukhari et al [2] uses Multilayer Perceptron (MLP), while Kurar et al. [4] uses fully Convolutional Network. As shown in table I, we outperform their works in both the main text and side text segmentation. In Figure 6, we show some visual results of input and output.

As shown in Figure 6, the input images have complex layout structures. In addition, they have a variety of skewed and curved lines, bleed-through and noise. However, we show the effectiveness of our method even in these conditions.

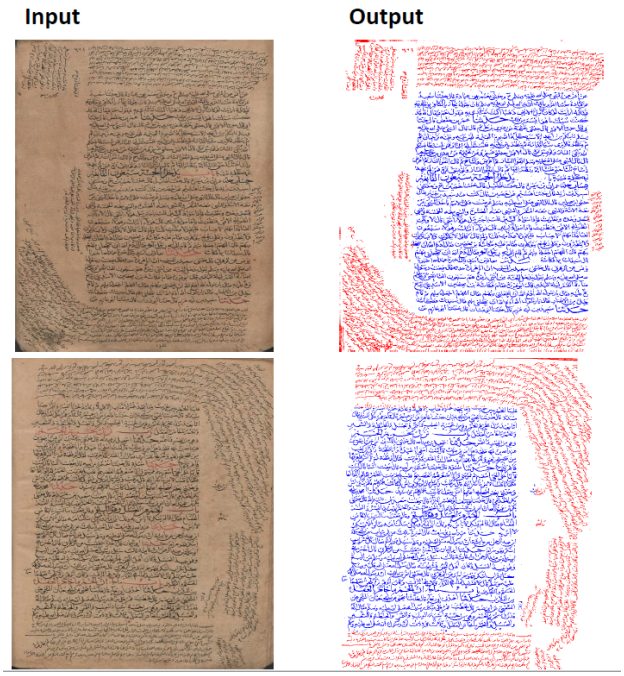

Fig. 6. Example of input and output using our method

V. Conclusion

In this paper we present binarization free layout analysis on challenging historical Arabic documents. Our method is divided into two parts, the first one is training a siamese network model on patches of pages. Siamese network takes

as an input a pair of patches and gives as an output a distance that corresponds to the similarity between the two patches. The second part of our method is using the trained model in part one, and building a distance matrix between all the patches of the same page. The distance matrix is fed into a clustering algorithm, which distributes the patches into three classes: main text, side text and background. We test our method on challenging historical Arabic manuscripts dataset. The dataset contains various writing styles and complex layout structures, in addition to a variety of degradation. We compare our work with others and show the effectiveness of our method. In future work, we plan to include datasets from other languages and show that our method works regardless of the language used on the documents.

ACKNOWLEDGMENT

This research was supported in part by Frankel Center for Computer Science at Ben-Gurion University of the Negev. One of the authors, Reem Alaasam, is a fellow of the Ariane de Rothschild Women Doctoral Program, and would like to thank them for their support.

REFERENCES

[1] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, "Reading digits in natural images with unsupervised feature learning," in *NIPS workshop on deep learning and unsupervised feature learning*, no. 2, 2011, p. 5.

[2] S. S. Bukhari, T. M. Breuel, A. Asi, and J. El-Sana, "Layout analysis for arabic historical document images using machine learning," in *Frontiers in Handwriting Recognition (ICFHR), 2012 International Conference on*. IEEE, 2012, pp. 639–644.

[3] A. Asi, R. Cohen, K. Kedem, J. El-Sana, and I. Dinstein, "A coarse-to-fine approach for layout analysis of ancient manuscripts," in *Frontiers in Handwriting Recognition (ICFHR), 2014 14th International Conference on*. IEEE, 2014, pp. 140–145.

[4] B. Kurar and J. El-Sana, "Binarization free layout analysis for arabic historical documents using fully convolutional networks," in *2nd International Workshop on Arabic and Derived Script Analysis and Recognition (ASAR)*. IEEE, 2018, pp. 151–155.

[5] S. Eskenazi, P. Gomez-Krämer, and J.-M. Ogier, "A comprehensive survey of mostly textual document segmentation algorithms since 2008," *Pattern Recognition*, vol. 64, pp. 1–14, 2017.

[6] M. Mehri, P. Héroux, P. Gomez-Krämer, A. Boucher, and R. Mullot, "A pixel labeling approach for historical digitized books," in *Document Analysis and Recognition (ICDAR), 2013 12th International Conference on*. IEEE, 2013, pp. 817–821.

[7] N. Journet, J.-Y. Ramel, R. Mullot, and V. Eglin, "Document image characterization using a multiresolution analysis of the texture: application to old documents," *International Journal on Document Analysis and Recognition*, vol. 11, no. 1, pp. 9–18, 2008.

[8] H. Wei, M. Baechler, F. Slimane, and R. Ingold, "Evaluation of svm, mlp and gmm classifiers for layout analysis of historical documents," in *Document Analysis and Recognition (ICDAR), 2013 12th International Conference on*. IEEE, 2013, pp. 1220–1224.

[9] K. Chen, H. Wei, J. Hennebert, R. Ingold, and M. Liwicki, "Page segmentation for historical handwritten document images using color and texture features," in *Frontiers in Handwriting Recognition (ICFHR), 2014 14th International Conference on*. IEEE, 2014, pp. 488–493.

[10] H. Wei, K. Chen, R. Ingold, and M. Liwicki, "Hybrid feature selection for historical document layout analysis," in *Frontiers in Handwriting Recognition (ICFHR), 2014 14th International Conference on*. IEEE, 2014, pp. 87–92.

[11] K. Chen, M. Seuret, M. Liwicki, J. Hennebert, and R. Ingold, "Page segmentation of historical document images with convolutional autoencoders," in *Proceedings of the International Conference on Document Analysis and Recognition, ICDAR*, vol. 2015-Novem, 2015, pp. 1011–1015.

[12] H. Wei, M. Seuret, K. Chen, A. Fischer, M. Liwicki, and R. Ingold, "Selecting autoencoder features for layout analysis of historical documents," in *Proceedings of the 3rd International Workshop on Historical Document Imaging and Processing*. ACM, 2015, pp. 55–62.

[13] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah, "Signature verification using a" siamese" time delay neural network," in *Advances in Neural Information Processing Systems*, 1994, pp. 737–744.

[14] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[15] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," Tech. Rep. [Online]. Available: http://code.google.com/p/cuda-convnet/

[16] A. K. Jain and R. C. Dubes, *Algorithms for clustering data*. Prentice-Hall, 1988.

[17] L. Rokach and M. Oded, "Clustering methods," *Data mining and knowledge discovery handbook*, pp. 321–352, 2005.