ASAR 2018 Competition Page Layout Analysis Using Fully Convolutional Networks

Ahmad Droby*
Ben-Gurion University of the Negev
Beer-Sheva, Israel
Email: drobya@post.bgu.ac.il

Berat Kurar Barakat*
Ben-Gurion University of the Negev
Beer-Sheva, Israel
Email: berat@post.bgu.ac.il

Jihad El-Sana
Ben-Gurion University of the Negev
Beer-Sheva, Israel
Email: el-sana@cs.bgu.ac.il

***Abstract*—This technical report presents a Fully Convolutional Network based method for layout analysis of benchmarking dataset provided by the competition. The document image is segmented into text and non-text zones by dense pixel prediction. Convolutional part of the network can learn useful features from the document images and is robust to uncontrained layouts. We have evaluated the zone segmentation with average black pixel rate, over-segmentation error, under-segmentation error, correct-segmentation, missed-segmentation error, false alarm error, overall block error rate whereas the zone classification with precision, recall, F1-measure and average class accuracy on both pixel and block levels.**

I. Introduction

Page layout analysis is an important pre-processing step for various document image processing algorithms. Page layout analysis process consists of page segmentation and zone classification. Page segmentation segments an image into homogeneous zones and zone classification classifies the regions into predefined classes such as text, graphic, and picture.

Computational solutions devoted to layout analysis of Arabic documents are few and not comparable due to the differences in data and evaluation metrics. Therefore we participate this competition which provides a benchmarking dataset and specifies evaluation metrics.

In this report we present a Fully Convolutional Network (FCN) based approach that segments texts and non-text zones. A FCN is trained to predict the class of each pixel. We have used this method for page layout analysis of historical manuscripts with complex layout in a paper submitted to ASAR 2018.

In the following, Section II describes the method, Section III presents the experimental results and finally concluding remarks are given in Section IV.

II. Method

We used FCN for segmenting side text and main text in Arabic documents with complex layout. FCN has made great improvements in object segmentation field [1]. It is an end to end segmentation framework that extracts the features and learns the classifier function simultaneously.

A. FCN architecture

The FCN architecture (Figure 1) we used is based on the FCN proposed for object segmentation [1]. First five blocks follow the design of VGG 16-layer network [2] except the discarded final layer. This is a conventional Convolutional Neural Network (CNN) and is called the encoder part of the FCN. Through the encoder, input image is down-sampled and filters can see coarser information with larger receptive field. Then decoder part of FCN up-samples coarse outputs to dense pixels. Up-sampling with a factor f is applying a convolution filter with a stride equal to $\frac{1}{f}$ and is called transpose convolution. Up-sampling filters are also learned during the training.

FCN32 up-samples the final layer of encoder back to input size in a single step, which limits the scale of detail in the predictions. Therefore we used FCN8 which combines final layer of encoder with lower layers with finer information (Figure 1). Default input size of VGG is 224×224, however to include more context we changed the input size to 320×320. We also changed the output channel to 3 which is the number of classes, text, non-text and background.

B. Dataset

We used publicly available BCE-Arabic V1 dataset [3] for training and validation. This dataset is composed of scanned pages from different Arabic books with normal (V1) layout. We tested the resulting model on complex (V2) layouts of the same dataset, namely the benchmarking.

Benchmarking dataset contains 90 images in 3 equal sets. The 90 images contain 927 text zones, 185 non-text zones:

1) Set A: 30 images of single column normal layouts (297 text blocks, 53 image blocks)
2) B: 30 images of double column simple layouts (379 text blocks, 54 image blocks, 19 graphics blocks)
3) C: 30 images of complex layouts (251 text blocks, 42 image blocks, 17 graphics blocks)

C. Pre-processing

Training and validation images are binarized. Then we randomly generate 100.000 and 15.000 patches of 320×320

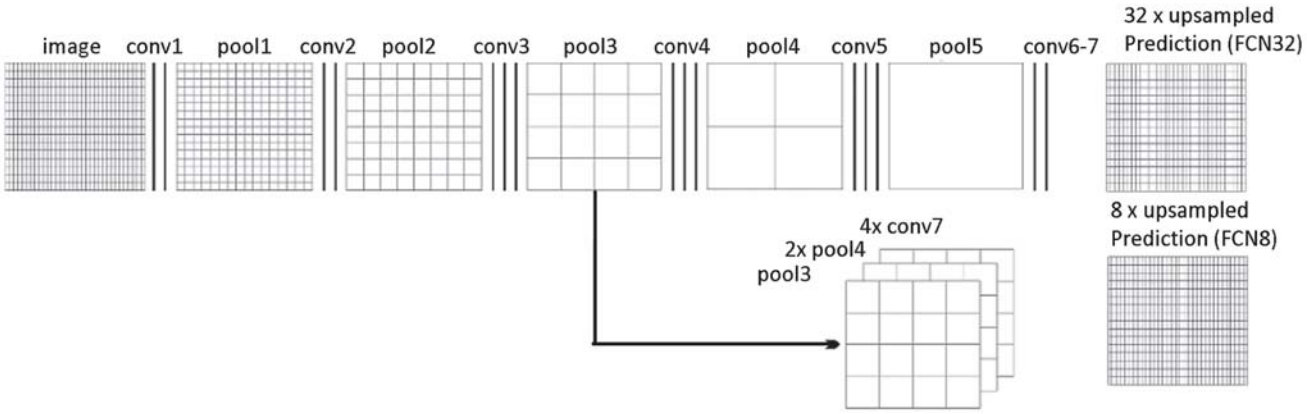

Fig. 1. The FCN architecture. Pooling and prediction layers are shown as grids that show relative coarseness. Convolutional layers are shown as vertical lines. FCN32 upsamples the final layer back to input size in a single step. FCN8 4 times upsamples the final layer, 2 times upsamples the pool4 layer and combine them with pool3 layer to upsample to input size.

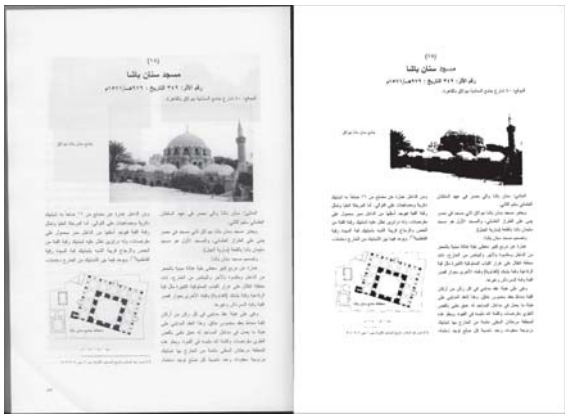

Fig. 2. Original test image and pre-processed test image.

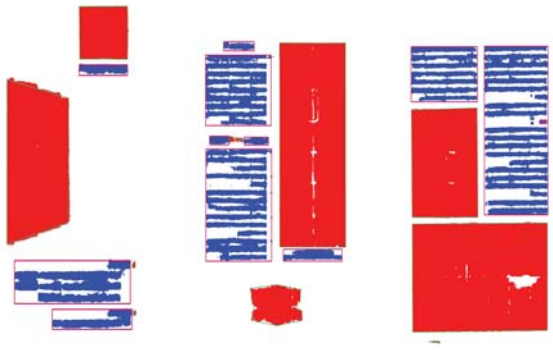

Fig. 3. Examples results of the post-processing. Green contours define the non-text classified zones, Red bounding boxes define text classified zones

size for training and validation sets respectively. Random patches potentially increases the variance that can accelerate convergence [1]. Testing images are binarized and margins are trimmed by 3% of the rows and 10% of the columns. Horizontal lines are removed using morphological operations. It is worth to note that during prediction of test patches, marginal regions that are less than patch size were filled with background pixels. A pre-processed testing image can be seen in Figure 2.

D. Post-processing

The resulting classification of the FCN is often noisy and a class zone has a jagged edges, thus we employed a post-processing method to denoise the results and extract a well defined classified zones. Each class (e.g. text and non-text) are considered separately. First, using morphological operations small classified zones are removed and ragged edges are smoothed. Second, the contours of each connected component (i.e. classified zones) are extracted. Then for each extracted contours the following values are considered: AP - total number of pixels inside the contours, CP - number of classified pixels inside the contours. Only the connected components that satisfy $AP \leq M$ and $CP/AP \leq \alpha$ are considered, where M and α are constants (We used $M = 100$ and $\alpha = 0.5$). To avoid over-segmentation if two contours for different connected components intersect, the connected component with the smaller area is discarded. Third, for text classified connected components we define it using its bounding box and for non-text classified connected components we define it using its simplified contours by applying RamerDouglasPeucker algorithm on the extracted contours (with $\epsilon = 0.1 \cdot AL$, where AL is the arc length of the extracted contours). Figure 3 show 3 examples of the output of the post-processing. Where the input is the output of the FCN (pixel level classified image); reds are pixels classified as non-text, and blue are pixels classified as text. Output, the red bounding boxes defines the text classified regions and the green contours defined the non-text classified regions.

TABLE I
SEGMENTATION RESULTS ON SETS A, B AND C.

Segmentation	Set A	Set B	Set C
text avrbpr	0.68	0.74	0.58
text ose	3.47	2.77	3.66
text use	1.57	2.80	0.23
text cs	5.80	6.93	4.80
text mse	2.40	2.63	1.60
text fa	0.33	0.47	0.33
nontext avrbpr	0.86	0.79	0.82
nontext ose	0.20	0.30	0.13
nontext use	0.37	1.33	0.63
nontext cs	1.63	2.20	2.10
nontext mse	0.03	0.20	0.23
nontext fa	1.53	0.57	1.40

III. EXPERIMENTS

A. Training

We train by Stochastic Gradient Descent (SGD) with momentum equals to 0.9 and learning rate equals to 0.001. We initialize VGG with its publicly available pre-trained weights. We trained on the randomly sampled dataset until least validation loss. All the experiments are conducted on Keras [4] and run on a single Nvidia 1080GTX.

B. Evaluation

We evaluated the results using the evaluator program provided by the competition organizers. The evaluation code takes the original image, its processed result and ground truth as input. It outputs the following metrics for segmentation:

1) The average black pixel rate (AvgBPR) represents the number of black pixels contained in segmented blocks compared to the corresponding blocks in the ground truth image.
2) The over-segmentation error (OSE) compares the number of over-segmented blocks to the number of ground truth blocks.
3) The under-segmentation error (USE) compares the number of under-segmented blocks to the number of ground truth blocks.
4) The correct-segmentation (CS) metric compares the number of correctly segmented blocks to the number of ground truth blocks.
5) The missed-segmentation error (MSE) compares the number of missed segments to the total number of ground truth blocks.
6) The false alarm error (FA) compares the number of false alarms to the total number of ground truth blocks.

Evaluator program outputs Precision (Pr), Recall (Rec), F1-measure (F1) and average class accuracy (Acc) on both pixel and block levels, for classification.

C. Results

Table I and Table II show the segmentation and classification results on the sets A, B and C. Proposed method achieved poor results on the 4 test samples in set A (Figure 4). We argue that this was due to their low resolution in relative to the resolution of training samples (Table III).

TABLE II
CLASSIFICATION RESULTS ON SETS A, B AND C.

Classification	Set A	Set B	Set C
block pr	0.99	0.98	0.99
block rec	0.83	0.97	0.85
block f1	0.88	0.97	0.90
block acc	0.97	0.99	0.93
pixel pr	0.82	0.88	0.71
pixel rec	0.94	0.94	0.95
pixel f1	0.87	0.90	0.81
pixel acc	0.80	0.87	0.75

TABLE III
4 TEST SAMPLES IN SET A HAVE LOW RESOLUTION IN RELATIVE TO TRAINING SAMPLES.

Sample name	Resolution
KIC Document 0002 (19) Page 06	904×1265
KIC Document 0002 (19) Page 08	935×1264
KIC Document 0002 (19) Page 14	949×1262
KIC Document 0002 (19) Page 18	905×1270
Training samples	1654×2338

Fig. 4. One of the low resolution samples in set A and its poor segmentation using the FCN trained on samples with higher resolution.

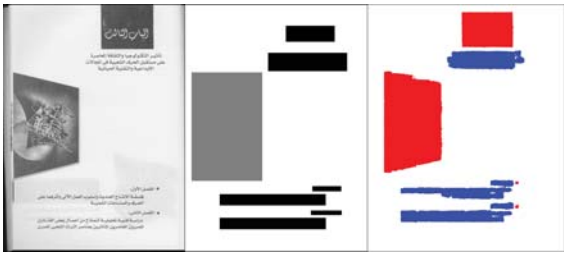

Fig. 5. Test sample (left) is not precisely annotated using rectangular bounding box (middle). Whereas FCN segments these cases more precisely (right).

Other common error cases happened due to inconsistencies in data annotation. Figure 5 shows that rectangular nontext annotation is not as precise as the actual zone, bullets are annotated as text zone whereas FCN segments both of these cases correctly. Figure 6 shows that a complete nontext part is not annotated whereas is recognized by FCN.

IV. CONCLUSION

This technical report gives the quantitative results of page layout analysis using FCN on the benchmarking dataset published by ASAR 2018 page layout competition. Also common

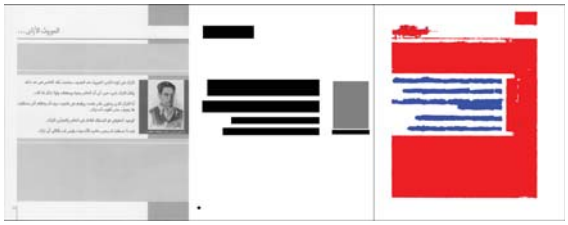

Fig. 6. Test sample (left) contains complete non-text zones which is not annotated (middle) but recognized by FCN (right).

error cases are visualised qualitatively. FCN is robust to complex layouts but prediction prediction performance has a direct ratio with the consistency of the test samples' resolution with training samples' resolution.

ACKNOWLEDGMENT

The authors would like to thank the Lynn and William Frankel Center for Computer Sciences at Ben-Gurion University for the support in this research.

REFERENCES

[1] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3431–3440.

[2] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[3] R. S. Saad, R. I. Elanwar, N. Kader, S. Mashali, and M. Betke, "Bce-arabic-v1 dataset: Towards interpreting arabic document images for people with visual impairments," in *Proceedings of the 9th ACM International Conference on PErvasive Technologies Related to Assistive Environments*. ACM, 2016, p. 25.

[4] F. Chollet *et al.*, "Keras," https://github.com/fchollet/keras, 2015.