**Handwritten Arabic words detection using Faster R-CNN in IFN/ENIT dataset**

May Mowaffaq Al-Taee, Sonia Ben Hassen Neji, Mondher Frikha

ATISP Research Lab, École Nationale d’Électronique et de Télécommunications de Sfax, University of Sfax, Sfax, Tunisia

**ABSTRACT**

Recognizing Arabic offline handwritten words still faces various challenges because of the diversity of writing styles and the overlap between the words and characters. Therefore, building an effective system to solve these challenges has always been difficult, which has led to a lack of published research in this field. This study introduces two new models to recognize handwritten Arabic words based on the Faster region-convolution neural network (Faster R-CNN). These models employ two pre-trained networks during the feature extraction phase: The visual geometry group-16 (VGG-16) network and the residual network (ResNet50) network. To help with overlapping detections and make localization more accurate, a soft non-maximum suppression (Soft-NMS) strategy is used in post-processing. Models are independently trained and tested on two groups of data from the Institut Für Nachrichtentechnik/Ecole Nationale d’Ingénieurs de Tunis (IFN/ENIT) dataset. The first group includes one word in each image, while the second contains multiple words. Test results showed that the proposed models give excellent results compared to others. The results of VGG16 and ResNet50 with the first dataset reached accuracy rates of 100% and 99.5%, respectively. Meanwhile, the accuracy of the second group reached 91.4% and 100% with VGG16 and ResNet50, respectively.

This is an open access article under the CC BY-SA license.

**1. INTRODUCTION**

The ability to read handwriting is becoming increasingly important as it substantially facilitates completing office work quickly and has fixed several issues that save time and effort. The major goal of the handwriting recognition system is to convert handwritten text documents from digital image format into documents with encoded character formats that can be changed and read by word processing application systems [1]. Handwriting recognition is used in many fields, including document processing, office automation, writer identification, automatic check processing in banks, postal code recognition, and signature verification [2]–[6]. Arabic has received much less attention from the Latin language despite the hundreds of millions of handwritten Arabic manuscripts in libraries, which carry a large amount of information [4], [7], [8]. Because Arabic handwriting has various challenges, like various sizes, styles of writing, and overlaps between the words characters, developing automatic methods for text recognition is difficult [9].

Recently, convolutional neural networks (CNNs) have been the best choice in many fields, such as image classification, localization of objects in images, face recognition, fingerprint analysis, computer-
assisted diagnostics, facial expression analysis, and handwritten recognition [10]–[13]. In particular, CNNs have outperformed object detection in recent years, where detecting objects typically involves searching for an object in the image and locating it using the bounding box [14]. Deep learning (DL)-based target detection algorithms can be separated into single-stage and two-stage techniques. Single shot detection (SSD) [15] and you only look once (YOLO) [16] are today’s most prominent one-stage target detection algorithms. While the region-convolution neural network (R-CNN) [17], Fast R-CNN [18], and Faster R-CNN [19] are examples of the algorithms for the target’s two-stage detection. Although the two-stage methods are slightly more computationally complex than the one-stage models, they provide higher recognition accuracy [20]. Faster R-CNN excels in object detection by combining region proposal networks (RPNs) and Fast R-CNN, where it provides end-to-end training for a specific job by utilizing innovative RPNs, building upon region proposal approaches and Fast R-CNN [21]. Several methods have been suggested for Arabic handwriting recognition using the Institut Für Nachrichtentecnhik/Ecole Nationale d’Ingénieurs de Tunis (IFN/ENIT) database.

Maalej and Kherallah [22] proposed an offline Arabic handwriting recognizer using a multi-dimensional long short-term memory network (MDLSTM) with rectified linear units (ReLUs) to solve vanishing gradient problems and dropout to prevent overfitting. Evaluated on the IFN/ENIT database, the systems achieved a label error rate of 11.40%. Elleuch et al. [23] introduced a deep CNN (DCNN) approach for classification using inception-v3, ResNet, and visual geometric group-16 (VGG16) architectures. They used the transfer learning technique and refined pre-trained models for deep features extraction. In the test stage, they used two groups of data: 10 words and 56 characters from the IFN/ENIT dataset. The proposed approach achieved accuracy rates of 95.70%, 98.99%, and 98.10% for word classification using inception-v3, ResNet, and VGG16, respectively.

Ali and Mallaiah [24] proposed a model for Arabic handwriting recognition using CNN and support vector machine (SVM). They applied dropout to the model and showed its efficiency in many Arabic scripts. The model was tested using the IFN/ENIT, handwritten Arabic characters database (HACDB), Arabic handwriting database (AHDB), and Arabic handwritten character dataset (AHCD) databases. The test results using IFN/ENIT datasets showed that the proposed model with dropout achieved 98.58%, while the model without dropout achieved 96.50%. Gader and Echi [25] proposed an attention-based convolutional long short-term memory model following that a connectionist temporal classification (CNN-Att-ConvLSTM-CTC) architecture for extracting handwritten Arabic words. The model used attention-based CNN to extract text-line features, feeds them into ConvLSTM network to learn a mapping between them, and then uses a CTC to learn the alignment between images and transcription. The model was trained on three databases, KHATT, AHDB, and IFN/ENIT. The extraction rate was 94.1% in IFN/ENIT. Moreover, Hamida et al. [4], a new image processing approach incorporating three descriptors, the gabor filter (GF), histogram of oriented gradients (HOG), and local binary pattern (LBP) for the feature extraction step, was developed. The model was tested 100 classes from the IFN/ENIT dataset, training the k-nearest neighbor (k-NN) algorithm for each feature extraction descriptor, and they used the best k-NN model to classifying Arabic handwriting images using a majority-voting technique. The model achieved a 99.88% recognition rate.

More recently Gader et al. [5] developed a model with three components: CNN, RNN (LSTM), and CTC. CNN for feature extraction, while RNN was used for spatiotemporal prediction. Moreover, the CTC was applied to infer information from the input image. The recognition rate is approximately 99.01%, 95.05%, and 96.57% for abc-d, abcd-e, and abcd-e-f, respectively. Finally, Lamtougui et al. [26] suggested a DL model to recognize handwritten lines and text using CNN, bidirectional long short term memory (BLSTM), and CTC. To improve the data quality, a data augmentation approach throughout the model’s training phase. The method was train and tested on KHATT and IFN/ENIT datasets, achieving a 92.11% accuracy rate in IFN/ENIT.

Although these methods are efficient, they have certain limitations. Indeed, they need substantial training samples, resulting in high computing costs. In addition, some methods use handcrafted algorithms to extract features from images, significantly increasing computational complexity and execution time, while other models use different regularization methods to prevent overfitting. This work aims to take advantage of the properties of Faster R-CNN in object detection, where it is used to recognize handwritten Arabic words, by building two models that use pre-trained networks. The major contributions of this paper include:

- This is the first attempt to use the new DL model based on the Faster R-CNN approach with the IFN/ENIT dataset.
- Using Faster R-CNN in this model gives several major benefits, such as speeding up detection frame creation and reducing training time and testing through sharing convolutional features for region proposals with object detection networks.
- Applying soft-non-maximum suppression (Soft-NMS) to improve word detection efficacy in the final stage, which treats the problem of multiple detections of the same item in an image, improving...
localization accuracy, managing overlapping detections, and enabling bounding box selection and confidence score fine-tuning.

- We manually created the bounding box annotations since the IFN/ENIT datasets lacked them and were crucial for the training process, reducing the range of searches for object features and the time needed for searches.
- This model can give excellent results using less data than various models.

The paper’s structure is as follows: in section 2, we will describe the components of the proposed technique. Section 3 will showcase the experimental findings. Section 4 will conclude with a presentation of the conclusions and suggestions for future work.

2. MATERIALS AND METHOD

2.1. Dataset (IFN/ENIT)

The IFN/ENIT dataset is one of the most commonly used datasets for studies on recognizing handwritten Arabic text [27], which include 32492 images written by over 1000 writers and represent Arabic words handwritten. These words represent the names of 937 Tunisian cities and villages. Many research groups have openly used the IFN/ENIT dataset [26], [28]. Figure 1 displays examples of images from the IFN/ENIT dataset. Researchers faced several challenges while using the IFN/ENIT dataset. The most significant ones are word overlap (Figure 2(a)), incorrect letter writing (Figure 2(b)), and differences in writing style depending on the writer (Figure 2(c)).

![Figure 1. Examples of images in IFN/ENIT dataset](image)

![Figure 2. Problems facing researchers in the IFN/ENIT dataset; (a) overlapping, (b) writing incorrectly, and (c) different style](image)

2.2. Faster model

Faster R-CNN includes feature extraction, RPN, and the Fast R-CNN method (detector) [19]. Figure 3 shows the general outline of the Faster R-CNN.

![Figure 3. General outline for the Faster R-CNN](image)

2.2.1. Features extraction using the shared CNN

A crucial aspect of the overall performance of the Faster R-CNN algorithm is the feature extraction step. Our approach uses CNN, a leading-edge object detection technique that employs a set of convolutions and pooling operations to extract essential features from images. Each image and its corresponding annotations are fed to the ResNet50 [29] or the VGG16 [30] pre-trained networks to ensure efficient and effective extraction of image features.
The ResNet50 architecture comprises two modules: convolution and identity blocks. Because the convolution block’s input and output dimensions differ, they cannot be linked in series. Therefore, the network’s dimension should be changed. The input dimension of the identity block is the same as the output dimension, which may be connected in series and used to deepen the network [31], [32].

The VGG16 network has 13 convolution layers activated by ReLU and three fully connected layers. It also has pooling layers. The last fully linked layer is eliminated, keeping only the front portion of the convolutional layer, which forms the network core [33], [34]. There are several advantages to the Faster R-CNN feature extraction step, including transfer learning for improved performance and Faster convergence, shared features for efficient computing, hierarchical feature representation, and end-to-end training for task-specific adaptability. With these benefits, Faster R-CNN is more successful at precisely and computationally efficiently identifying objects in images [35].

2.2.2. RPN

The RPN is a fully convolutional network (FCN) that generates exact regional proposals using shared full-image convolutional features. It uses a 3x3 sliding window approach to process input and create a feature vector, with 9 anchors generated at each image point with three aspect ratios (1:1, 2:1, 1:2) and three scales (128, 256, and 512) in the center. Two fully connected layers process proposals to determine the likelihood of an object being present in the proposed window. One layer predicts the object’s bounding box coordinates, while the other determines if the proposal is an object (a word) or a background. The RPN can be trained from end to end and feeds these proposals into the Fast R-CNN for detection [36]. In (1) is used to find the intersection over union (IoU), a key object detection indication.

\[
IoU = \frac{\text{Anchor} \cap \text{GTBox}}{\text{Anchor} \cup \text{GTBox}}
\]

Where: the \( IoU \) value represents the ratio of the intersection area of the anchor with the ground truth bounding box (GTBox) to their union area. Anchors are suggested outputs assigned an objectness score determined by \( IoU \) score [14].

The RPN uses two types of anchors: negative and positive. When the \( IoU \) score for each ground truth area is below 0.3, the negative anchor is assigned, whereas the positive anchor is assigned when the \( IoU \) score for any ground truth box exceeds 0.7. The training loss is not influenced by anchors, with scores ranging from 0.3 to 0.7; instead, the subsequent network module is trained using the remaining negative and positive anchors [37]. To determine whether the anchors are negative or positive based on the threshold value, we use (2):

\[
p = \begin{cases} 
-1 & \text{if } IoU < 0.3 \\
1 & \text{if } IoU > 0.7 
\end{cases}
\]

The loss function of the whole network is given by (3):

\[
L([p_i], [t_i]) = \frac{1}{N_{cls}} \sum_i L_{cls}(p_i, p_i^*) + \lambda \frac{1}{N_{reg}} \sum_i L_{reg}(t_i, t_i^*)
\]

where \( i \) is the index of anchors, \( p_i \) is the probability that the \( i \)-th anchor is predicted to be the true label, \( p_i^* \) is the presence or absence of a target for the anchor, \( t_i \) is the prediction of the bounding box regression parameter of the \( i \)-th anchor, \( t_i^* \) is the ground truth box corresponding to the \( i \)-th anchor, \( N_{cls} \) is the batch size, \( N_{reg} \) is the number of anchor positions, and \( \lambda \) is the balance parameter. \( L_{cls} \) is a binary log loss and \( L_{reg} \) is a smoothed L1 loss. Faster R-CNN can be trained end-to-end by back-propagation using the stochastic gradient descent (SGD) for the optimization of the loss function [19], [36].

2.2.3. NMS

NMS is essential for object detection models that reduce RPN proposal redundancy. It reduces suggestions while preserving detection accuracy by selecting the detection box with the greatest classification score and eliminating boxes with excessive overlap [38]. Soft-NMS is used instead of the NMS to fix multiple detections of the same thing in an image, improve localization accuracy, deal with overlapping detections, and let the bounding box selection and confidence score fine-tuning happens [39].

2.2.4. Fast R-CNN detector

A detection network receives the feature map and the regions of interest generated by the previous networks as input. This part comprises two major steps. First, the RoI pooling selects a specific area from the feature map and resizes it to a fixed size. After processing the feature maps and proposals, the information is

---

Handwritten Arabic words detection using Faster R-CNN in IFN/ENIT dataset (May Mowaffaq Al-Taee)
aggregated and used to generate proposal feature maps of fixed sizes. These maps are then transformed into vectors and input into fully connected layers [21]. The second is the classification and regression layer, which comprises a fully connected layer that displays the class assigned to each word. The bounding box regression generates a box that shows the last position of the recognized word [21], [36], [40], [41].

2.3. Implementation of the proposed model

This section will display the steps followed to implement the proposed model, which includes the data collection and preparation phase and the phases of training and testing of the model. Figure 4 displays the general outline of the proposed model.

![Figure 4. General outline of the proposed model](image)

2.3.1. Data collection

Our work uses two groups of data from the IFN/ENIT dataset: the first set includes one class in each image, and the number of classes used is 11. The second group includes multiple classes in each image, and the number of classes used is 20. Table 1 shows the names of the classes used in the first and second groups. The data is divided into 80% of images for training and 20% for testing. The first group, which included 1100 images, was split into 880 for training and 220 for testing, while the images of the second group were 1000, split into 800 for training and 200 for testing.

<table>
<thead>
<tr>
<th>Classes name for group 1</th>
<th>Zanosh</th>
<th>Chamakh</th>
<th>Sh‘al</th>
<th>Nqah</th>
<th>Nahal</th>
<th>Mareth</th>
<th>Alchraae</th>
<th>Alchwamkh</th>
<th>Alkhaaleej</th>
<th>Almachhame A</th>
<th>Akouala</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classes name for group 2</td>
<td>Rabayie- siddihahr</td>
<td>Sidi-Bwo- Bakr</td>
<td>Tel- Algizlan</td>
<td>Sab’at</td>
<td>Abar</td>
<td>Rais- aldra’a’a</td>
<td>Hais- Alsalah</td>
<td>Douwar- Allouwata</td>
<td>Be’r- Marwa</td>
<td>Awlad- Alchamkh</td>
<td>Awlad- Hafwz</td>
</tr>
</tbody>
</table>

2.3.2. Preprocessing

Because of the different original image sizes, we used preprocessing to resize the images to a fixed size without distortion through several steps.
- Remove all white areas from the images.
- Resize the image to 256*64.
- Add a white area with the size of 6 pixels on each side of the image to facilitate the labeling process around each class in the image, resulting in a final image with a size of 262×72 pixels used during the training and testing phases for the model.

2.3.3. Data augmentation

We applied the data augmentation approach to solve the data imbalance problem in the training phase. Three techniques are employed to increase the data: original image (Figure 5(a)), erosion (Figure 5(b)), dilatation (Figure 5(c)), and contrast (Figure 5(d)). After the data augmentation step, the images used in the training stage became 3,520 for the first group, while the training images became 3,200 for the second group.
2.3.4. Data labelling
The LabelImg tool [25] was used to create the bounding box (bbox) annotations manually around each class in the image used in the training phases. The bbox annotations contain each class’s name and bbox values (xmax, xmin, ymax, ymin, height, and width). Each image has its own extensible markup language (XML) file, and then the XML files are grouped into one comma separated values (CSV) file and then converted to a TXT file used in the training phases. Figure 6 illustrates an example of creating the bounding box for the classes in the image.

Figure 5. Data augmentation; (a) original image, (b) erosion, (c) dilation, and (d) contrast

2.3.5. Training model
The training phase was performed separately for each group. The first group was trained with 50 epochs and 3,520 iterations, consuming 48 hours for the VGG16 model and 23 hours for the ResNet50 model. Similarly, the second data group was trained with 50 epochs and 3,200 iterations, with the VGG16 model requiring 46 hours and the ResNet50 model requiring only 23 hours. The learning rate was 1e-5, and we changed the RPN setup by modifying the three scales to (32, 64, and 128) to enhance the precision of detecting small-sized objects while maintaining aspect ratios of (1:1, 2:1, and 1:2) and the code written in Python was executed on an NVIDIA processor core i9. The entire training is done on a central processing unit (CPU). However, using a graphical processing unit (GPU) environment can considerably decrease training time. Figure 7(a) illustrates the total loss for VGG16, and Figure 7(b) illustrates the total loss for ResNet50 after training the model with 50 epochs and 3520 iterations on the first group of data. Figure 8(a) illustrates the total loss for VGG16, and Figure 8(b) illustrates the total loss for ResNet50 after training the model with 50 epochs and 3200 iterations on the second group of data.

Figure 6. Labeling words in an image

Figure 7. The total loss for the; (a) total loss for VGG16 and (b) total loss for ResNet50

Handwritten Arabic words detection using Faster R-CNN in IFN/ENIT dataset (May Mowaffaq Al-Taee)
3. RESULTS AND DISCUSSION

To evaluate the efficiency and effectiveness of the proposed models, we use several evaluation metrics, including recall, accuracy, F1_Score, and precision. These metrics are defined as (4) to (7) [38], [42], [43]:

\[
Recall = \frac{TP}{TP + FN}
\]  

\[
Accuracy = \frac{TP + TN}{TP + TN + FN + FP}
\]
F1\_Score = \frac{2\times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{6}

\text{Precision} = \frac{TP}{TP + FP} \tag{7}

Where \( TN, FP, TP, \) and \( FN \) stand for true negative, false positive, true positive, and false negative respectively. The IoU value is a critical parameter used to determine whether a predicted bounding box is a true positive or a false positive. In addition, we use the mean average precision (mAP) to evaluate the proposed model, which is a crucial metric in target detection. The value of (mAP) is calculated by using (8):

\[
mAP = \frac{1}{M} \sum_{q=1}^{M} AP_q
\]

Where \( AP_q \) is the average precision of the \((q\)-th\) class and \( M \) is the total number of classes. Testing the models on the first group’s images produced the best results after 25 epochs with VGG16 and 35 with ResNet50. Table 2 shows the total value of accuracy and mAP, while Table 3 displays the test results for each class, where the evaluation metrics used were precision, recall, and F1\_Score.

Table 2. Test results for the first group

<table>
<thead>
<tr>
<th>Network</th>
<th>Number of epochs</th>
<th>Best epoch</th>
<th>Accuracy (%)</th>
<th>mAP (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>VGG16</td>
<td>50</td>
<td>25</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>ResNET50</td>
<td>50</td>
<td>35</td>
<td>99.5</td>
<td>100</td>
</tr>
</tbody>
</table>

Table 3. Test results for each class in the first group

<table>
<thead>
<tr>
<th>Class name</th>
<th>VGG16 Precision</th>
<th>Recall</th>
<th>F1_Score</th>
<th>ResNet50 Precision</th>
<th>Recall</th>
<th>F1_Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Akoudah</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Aldakhaniya</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Alkhaleej</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Alchiwamkh</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Alchraa</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Mareth</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Nahal</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.95</td>
<td>0.97</td>
</tr>
<tr>
<td>Nq\h</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Sh'al</td>
<td>1</td>
<td>1</td>
<td>0.95</td>
<td>1</td>
<td>1</td>
<td>0.97</td>
</tr>
<tr>
<td>Chamakh</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Zanosh</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

In contrast, testing the models on the second group’s images produced the best results after 25 epochs with VGG16 and 40 with ResNet50. Table 4 shows the total value of accuracy and mAP, while Table 5 displays the test results for each class, where the evaluation metrics used were precision, recall, and F1\_Score. Table 6 compares the results obtained by applying the proposed models with the other methods using the IFN/ENIT dataset.

Table 4. Test results for the second group

<table>
<thead>
<tr>
<th>Network</th>
<th>Number of epochs</th>
<th>Best epoch</th>
<th>Accuracy (%)</th>
<th>mAP (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>VGG16</td>
<td>50</td>
<td>25</td>
<td>91.4</td>
<td>99.3</td>
</tr>
<tr>
<td>ResNET50</td>
<td>50</td>
<td>40</td>
<td>100</td>
<td>99.4</td>
</tr>
</tbody>
</table>

Our models achieved excellent outcomes compared to some well-known techniques. Even though other models are efficient, they have certain limitations. Indeed, they need substantial training samples, resulting in high computing costs. In addition, some methods use handcrafted algorithms to extract features from images, significantly increasing computational complexity and execution time, while other models use different regularization methods to prevent overfitting. Our proposed models are not computationally expensive because they do not require significant training samples and do not require the integration of organizational techniques, dictionaries, or linguistic models. Despite these advantages, manual labeling or annotation of the training images is relatively complex.
In future work, we will improve the VGG16 network to handle small classes such as "و" and "ح" as the results were low compared to other classes in this work. In future work, we will improve the VGG16 network to handle small classes and use another dataset to train and test models. We will also develop a model for recognizing letters and apply alternative pre-trained networks instead of the models used in the currently proposed method.

4. CONCLUSION

The result shows that the proposed model performs excellently in detecting and recognizing handwritten Arabic words. The first group achieved the best result after 25 epochs of 100% accuracy for VGG16 and 99.5% after 35 epochs of ResNet50. Meanwhile, the second group achieved the best results: 91.4% accuracy for VGG16 after 25 epochs and 100% accuracy for ResNet50 after 40 epochs. In addition, the results show that handling overlapping detections after the classification stage using Soft-NMS rather than NMS increases the number of class detections and improves recognition accuracy. Also, creating the bounding box annotations reduces the range of searches for object features and the time needed for searches during the training stage, despite these advantages, manual labeling or annotation of the training images is relatively complex. In addition, using Faster R-CNN is not computationally costly since it only needs a few training samples and does not use organizational strategies, dictionaries, or linguistic models. In contrast, the results show the VGG16 network suffers when dealing with small classes such as "و" and "ح" as the results were low compared to other classes in this work. In future work, we will improve the VGG16 network to handle small classes and use another dataset to train and test models. We will also develop a model for recognizing letters and apply alternative pre-trained networks instead of the models used in the currently proposed method.

ACKNOWLEDGEMENTS

The authors extend their gratitude for the encouragement received from the National School of Electronics and Communications in Sfax.

REFERENCES


Bulletin of Electr Eng & Inf, Vol. 13, No. 5, October 2024: 3568-3578
Handwritten Arabic word detection using Faster R-CNN in IFN/ENIT dataset (May Mowaffaq Al-Taei)


---

**BIOGRAPHIES OF AUTHORS**

**May Mowaffaq Al-Taee** learned a B.Sc. in Computer Science from Baghdad University, College of Education Ibn Al-Haitham in 2000 and an M.Sc. in Computer Science from College of Science, University of Baghdad in 2004. She is work at the Ministry of Higher Education and Scientific Research in Iraq. Her research areas are image processing, information security, and artificial intelligence. She has published a many research papers in local and international refereed journals as an author or co-author. Currently a doctoral student at a National School of Electronic and Telecommunication of Sfax, University of Sfax, Tunisia. She can be contacted at email: may.tai@enetcom.u-sfax.tn.

**Sonia Ben Hassen Neji** was born in Sfax, Tunisia, in 1986. She received the engineering degree (Hons) in signals and systems and the M.Sc. degree (Hons) in telecommunication from the Tunisia Polytechnic School, in 2009 and 2010, respectively, and the Ph.D. degree (Hons) in telecommunication from the National Engineers School of Tunis in 2016. She is currently an Assistant Professor in the National school of Electronic and Telecommunications of Sfax (Department of Telecommunication) and member of the Advanced Technologies for Image and Signal (ATISP) laboratory. Her research interests include statistical signal processing, array processing with an emphasis on direction of arrival estimation for wireless communications, image processing, and pattern recognition. She can be contacted at email: sonia.benhassen@enetcom.usf.tn.

**Prof. Dr. Mondher Frihka** is currently a full professor at the National School of Electronics and Telecommunications, University of Sfax, Tunisia. He is also a director of the ’Advanced Technologies of Image and Signal Processing’ research lab. His research interests include digital signal and image processing, speech and audio processing, pattern recognition, and IA applications. He received the master of applied sciences in electrical engineering from the university of Ottawa Canada in 1991. He then worked as a head project at the Industriel Land Agency in Tunisia. In 2003, he started pursing his graduate research and obtained in 2007 his Ph.D. degree from the National School of Engineering of Sfax, Tunisia. He can be contacted at email: mondher.frihka@enetcom.usf.tn.