1-2022

# Handwritten English Word Recognition Using a Deep Learning Based Object Detection Architecture

Riktim Mondal
*Jadavpur University*

Samir Malakar
*Asutosh College*

Elisa H. Barney Smith
*Boise State University*

Ram Sarkar
*Jadavpur University*

**Title:** Handwritten English word recognition using a deep learning based object detection architecture

**Authors:**

1. **Riktim Mondal**, Department of Computer Science and Engineering, Jadavpur University, Kolkata, India. Email- riktimrules@gmail.com
2. **Samir Malakar**, Department of Computer Science, Asutosh College, Kolkata, India. Email- malakarsamir@gmail.com
3. **Elisa H. Barney Smith**, Department of Electrical and Computer Engineering**,** Boise State University**,** Boise, Idaho, USA, Email- ebarneysmith@boisestate.edu
4. **Ram Sarkar**, Department of Computer Science and Engineering, Jadavpur University, Kolkata and Pin-700032, India. Email- raamsarkar@gmail.com

**Corresponding Author**

Samir Malakar,
Assistant Professor
Department of Computer Science,
Asutosh College, Kolkata, India
Email- malakarsamir@gmail.com

# Handwritten English word recognition using a deep learning based object detection architecture

Riktim Mondal[1], Samir Malakar[2,*], Elisa H. Barney Smith[3], Ram Sarkar[1]

[1]Department of Computer Science and Engineering, Jadavpur University, Kolkata, India

[2]Department of Computer Science, Asutosh College, Kolkata, India

[3] Department of Electrical and Computer Engineering, Boise State University, Boise, Idaho, USA

[*]Corresponding author's email id: malakarsamir@gmail.com, contact number: +91 9933053452

**Abstract-** Handwriting is used to distribute information among people. To access this information for further analysis the page needs to be optically scanned and converted to machine recognizable form. Due to unconstrained writing styles along with connected and overlapping characters, handwriting recognition remains a challenging task. Most of the methods in the literature use lexicon-based approaches and train their models on large datasets having near 50K word samples to achieve good results. This results in high computational requirements. While these models use around 50K words in their dictionary when recognizing handwritten English text, the actual number of words in the dictionary is much higher than this. To this end, we propose a handwriting recognition technique to recognize handwritten English text based on a YOLOv3 object recognition model that is lexicon-free and that performs sequential character detection and identification with a low number of training samples (only 1200 word images). This model works well without any dependency on writers' style of writing. This is tested on the IAM dataset and it is able to achieve 29.21% Word Error Rate (WER) and 9.53% Character Error Rate (CER) without a predefined vocabulary, which is on par with the state-of-the-art lexicon-based word recognition models.

***Keywords:*** *Handwriting recognition, Character spotting, IAM dataset, YOLOv3*

## 1. Introduction

Creation and distribution of handwritten documents remain prevalent in society, even while people increasingly communicate with online typed media in the era of a technology enhanced society. The volume of handwritten documents is still increasing. Numerous domains exist where direct use of digital technologies are yet to be fully approved (e.g., court proceedings) and where the medium of handwriting is still omnipresent (e.g., personal notes, content managed in schools and educational institutions, etc.). There is also a large archive of historical handwritten documents that need to be recognized to allow current access. The understanding of content from handwritten document images remains a current research problem due to its many applications like making handwritten manuscripts digitally searchable [1][2], postal automation [3], digitizing handwritten forms [4][5] and reading bank checks [6]. Therefore, methods to understand the text in these handwritten documents are needed to enable access to the vast information present there.

One of the most popular and well accepted solutions of understanding the handwritten content in the document images is to convert them into machine editable form (i.e., converting them to Unicode representation). Traditionally, this conversion of handwritten documents is performed following a three-step process. In the first step, document images are segmented into handwritten word images. There are many page segmentation techniques used for this step [7] [8]. Next, each segmented word image is recognized to convert it into machine editable form. The process of converting handwritten word images into their machine encoded form is known as handwritten word recognition and this step has been marked as the most challenging and crucial one in the literature of document image processing. Finally, recognized words are assembled to understand the content of the entire document image. The work in this paper contributes to handwritten word recognition.

Handwritten words often have largely connected character sequences if they are written in cursive, and even non-cursive writing will have inadvertent connectedness. In traditional approaches, characters present in a word image are extracted first and then the extracted isolated characters are recognized using a character recognition model to obtain the final recognized word. This solution is largely affected by the segmentation ambiguity i.e., during word segmentation many under-/over-segmented characters are generated from a word image [9][10]. It becomes very difficult to set a tradeoff between over- and / or under-segmented characters. Sometimes, these models use slant and skew correction [11][12] to reduce the segmentation errors and it increases the overall processing time by a margin.

For handwritten word recognition, it is very difficult to locate a character inside a word image directly owing to the cursive nature of the writing.

Handwritten word recognition methods have evolved to skip the stage that explicitly segments the words into characters or character parts by using hidden Markov models (HMMs) [13] or recurrent neural networks (RNNs), especially bi-directional long short-term memory (BLSTM) [14] or multi-dimensional long short-term memory (MDLSTMs) [15], along with connectionist temporal classification (CTC) loss [16]. These methods have been used because of their inherent abilities to process temporal sequences. A hybrid scheme using a convolutional recurrent architecture was proposed in [17], where the convolutional layers are used for feature extraction, and the features are subsequently passed on to a BLSTM network. This with CTC loss works better than other schemes.

However, all these models mentioned above use lexicon-driven models [13][14][15][17] where the recognized string is supposed to form a valid word based on the underlying dictionary that is included in the network model. As a result, these methods require a large amount of training data to generate good recognition accuracy. For example, Sueiras et al. [18] and Bhattacharya et al. [19] used 47952 training samples to recognize handwritten words written in English, while 80421 samples were used in the work by Pham et al. [20]. In these works, the authors use LSTM based word recognition models. Not only do these methods incur a huge computational cost, but they also demand a large number of annotated samples. It is not always possible to manage both. In addition to this, while the dictionary size for English is considered to contain at most 50K English words, the number of words in reality is much higher than that when tense and case variations are included. Therefore, it can safely be stated that there is a need for a lexicon free word recognition model to meet the real goal of handwritten word recognition with lower training cost as well as fewer training samples. A better approach would be to localize characters in a word, recognize the characters, and then form the word. This approach can identify all words, even if they are not present in the training set or if they have no lexical meaning.

Recently, Majid and Barney Smith [21] proposed an architecture for recognizing handwritten Bangla word images. Bangla is a connected script. They sequentially search for each letter (consonant, diacritic, or conjunct) in the Bangla alphabet to determine their presence in a subject word. They detect individual characters and associated diacritics separately. For the detection, they used separate networks for the consonents and the diacritics. The networks make use of a faster R-CNN [22] for the character spotting. This method does not require the segmentation of the word into characters (or consonants from diacritics) during the recognition stage and also does not limit the response to a fixed vocabulary. It trains with a relatively small dataset, requiring repetition of patterns on the character level, not the word level. The method we propose is similar in using sequential character spotting, but instead of a VGG16 Faster R-CNN we use the YOLOv3 (the third version of the You Only Look Once network) [23] deep learning based object detection technique, and we are operating on the Latin character set using the IAM dataset.

In the present work, we have taken the approach of character segmentation (localization) and classification of that character (identification) simultaneously inside a word boundary. This avoids the segmentation or localization of characters within a word and the classification of those isolated characters. By working on the character level the data requirement is significantly lower that recognition on the word level. Concisely, the contributions and findings of our work are as follows.

**Contributions:**

- We design an end-to-end lexicon-free handwritten cursive word recognition model following a character spotting protocol.
- We perform localization and identification of characters simultaneously in a handwritten word image using a YOLOv3 network.
- The model is designed so that it can work well even if a large number of ground truth images are not available during training, thereby minimizing the training cost.

**Major Findings:**

- No preprocessing techniques such as skew or slant correction are required.
- The test set used is much larger (25 times) than the training set size, because a smaller training set can be used.

- The model attains state-of-the-art results on the standard IAM dataset without using any dictionary-based error correction, unlike state-of-the-art works so this model can be used to recognize misspelled and out-of-dictionary words. It also makes it more easily transferrable to other Latin script languages.

The rest of this article is organized as follows: section 2 provides a brief description of some state-of-the-art methods related to handwritten word recognition. In section 3 we describe the YOLOv3 object detection network, and then the modifications needed to apply it to the present problem. In section 4 the results and analysis are provided including subsections on database description, experimental setup, comparative results and error case analysis. Finally, we conclude in section 5 and give some future directions for the present work.

## 2. Related Work

Offline handwriting recognition has been approached with many methods. Some of the prominent ones for the Latin script are described next. It is most common for offline handwriting recognition to require words to be contained in a vocabulary.

Sueiras et al. [18] proposed an attention-based sequence to sequence model for recognizing handwritten English word images. In their work, they used an attention model during decoding in the sequence to sequence model. Bhattacharya et al. [19] also proposed a sequence to sequence model like Sueiras et al. [18]. However, in this work word images were preprocessed using a fuzzy-membership based pseudo segmentation method prior to feeding those to a patch based sequence to sequence model. In both the works, a LeNet5 architecture was used for feature extraction from patches of word images and trained with a large amount of data. Pham et al. [20] showed that the performance of a recurrent neural network (RNN) within an LSTM cell based handwritten word recognition model can be improved by integrating dropout therein. They applied dropout on each RNN cell of an existing LSTM based word recognition model for handwritten word recognition to obtain better performance than the existing ones. That means the authors tested the performance of an existing LSTM word recognition model by incorporating dropout. The system proposed by Doetsch et al. [24] used an LSTM embedding RNN model with an additional parameter that controls the shape of the squashing functions in the gating units. These research attempts used the IAM dataset [25] that contains images of handwritten English words. Graves and Schmidhuber [15] used a hierarchy of MD LSTM-RNNs, which comprises a hierarchical structure, similar to [16], of two models: MD RNN and MD LSTM. In another work, Stahlberg and Vogel [26] relied on a fully connected deep neural network (DNN) along with pixel and segment based features. All these works performed fine when provide large training samples, however their performance is not ensured for smaller training sets.

The technique that Bluche et al. [27] used was a voting scheme called recognizer output voting error reduction (ROVER) for combining four models to recognize word images written in the Latin script. Two of the four models are based on BDLSTM-RNNs, and the other two are based on deep multi-Layer perceptrons (MLPs). In another work, Menasri et al. [28] proposed two handwritten word recognition models: (i) a single RNN model and (ii) a classifier combination based system that uses seven recognizers (one grapheme based MLP-HMM, two variants of sliding window based GMM-HMMs, and four variants of their proposed RNNs). The RIMES dataset [28] was used to evaluate their method. The work proposed by Almazán et al. [29] encoded the input word image as Fisher vectors (FV), i.e., as an aggregation of the gradients of a Gaussian mixture model (GMM) and scale invariant feature transformation (SIFT). Use of multiple models for obtaining state-of-the-art results made the systems [27] [28] [29] costly in terms of computation. Moreover, in all these methods train and test samples were taken from the same dataset. Hence, performance of these models is unknown if we consider testing samples from different datasets.

There are some models in the literature that did not follow directly the encoder-decoder based models. An HMM system was proposed by Azeem and Ahmed [30] where the number of foreground pixels and the histogram of gradient values are considered as feature values. In another work, Gui et al. [31] used an adaptive context-aware reinforced agent-based model with low computation overhead for word recognition. This model needs pre-segmented text line images and paragraph images along with the words to learn. Recently Wu et al. [32] proposed a handwritten word recognition method that combines position embedding with residual networks (ResNets) and BLSTM network. In the first step, ResNets were employed to extract satisfactory features from the input image and in the later step the output position embedding was fed to a BLSTM as indices of the character sequence corresponding to a word. The proposed model achieved better results on two public corpora (RIMES and the ICDAR 2011 competition on isolated word recognition [33]) in a lexicon-free approach. This method uses two heavyweight CNN models such as ResNet and BLSTM, as well as large training sets. A comparative study of all the above mentioned works (along with ours) listing technical overview, detailed stat of dataset in use, the recognition accuracy obtained, pros and cons is provided in Table 1.

From the above discussion, it is clear that unconstrained handwritten word recognition is still considered an open research problem to the research community. Most of the works (e.g., [18] [20] [27][34]) on the IAM dataset have obtained good recognition accuracy (see Table 1). Some of the notable observations from the records in Table 1 are

- The accuracies obtained by the works [18] [20] [19][27][34] use a fixed length lexicon and a large number of training samples i.e., with a huge training time.
- The recognition accuracies of the works [18][20][19] decrease when tested in a lexicon-free approach.
- Dictionary dependent approaches cannot process unseen or incorrectly spelled words.

Keeping the above facts in mind, we follow a different approach where characters are identified for transcription of an entire word. This does not require the word to be segmented into individual characters. By looking for modeled characters in a word, any word, even if it is not present in the training set or is spelled wrongly, the word can be recognized by our model. Moreover, it needs fewer training samples as the recognized objects are characters, not words, so it can be adapted to new circumstances easier.

In this paper, we recognize an unconstrained handwritten word image in a lexicon free approach using an object detection based CNN architecture. This choice is motivated by the incredible advancement in object detection and recognition [35][36] owing to the success of underlying deep learning models. The present work uses deep learning based YOLOv3 architecture [23] for the detection and recognition of characters present in a word image. In this context, we would like to mention that we leverage the application of the YOLO model by suitably tailoring the model to fit the current research problem. This model needs significantly fewer training samples as compared to state-of-the-art methods to generate satisfactory word recognition results. Additionally, more than 50% of the training samples are taken from an in-house dataset i.e., not from the IAM dataset on which our model is tested.

**Table. 1** Summary of the state-of-the-art techniques. In this table WER and CER represent *word error rate* and *character error rate*, while *Type 1* and *Type 2* results represent "results with no vocabulary based correction or no language model are used" like in the preposed work, and "results when some vocabulary based correction or language model or both are used" respectively. The WER/CER scores the handwritten word database(s) in use have been cited.

| **Method** | **Technique, Learner and Handwritten Word Database(s) Used** | **Database split (Train/ Validation/ Test)** | **Result type and WER/ CER (in %)** | **Remark** |
|---|---|---|---|---|
| Sueiras et al. [18] | **Method:** Attention based sequence to sequence (an encoder decoder approach)<br>**Learner:** BLSTM<br>**Dataset:** IAM and RIMES | **IAM:**<br>47952 / 20306 / 7558<br>**RIMES:**<br>51739 / 7464 / 7776 | <u>**Type 1:**</u><br>**IAM:**<br>23.8 ± 0.05/ 8.8 ± 0.02<br>**RIMES:**<br>15.9 ± 0.4/<br>4.8 ± 0.1<br><u>**Type 2:**</u><br>**IAM:**<br>19.7±0.03/ 9.5 ±0.03<br>**RIMES:**<br>13.1 ± 0.2/ 5.7 ± 0.1 | **Pros:** First time use of attention mechanism for word recognition where the authors performed feature extraction from image patches using LeNet-5, followed by sequence to sequence model to decode the feature into recognized word.<br>**Cons:** The model works on word level only where it cannot identify words outside the closed dictionary used. Also due to the usage of multiple CNN, Encoder (RNN) and decoder (RNN), the model becomes heavy to train. |
| Bhattacharya et al. [19] | **Method: Recognition model is s**imilar to the work [18]. However, word images were pseudo segmented with fuzzy membership function.<br>**Learner:** BLSTM<br>**Dataset:** IAM | 47952/ 20306/ 7558 | <u>**Type 1:**</u><br>31.3/ 13.2<br><u>**Type 2:**</u><br>18.6 / 9.7 | **Pros:** Fuzzy membership based pseudo segmentation improves the base classifier result.<br>**Cons:** Underlying segmentation approach limits it generalization. |
| Pham et al. [20] | **Method:** RNN-LSTM with dropout<br>**Learner:** RNN-LSTM | **IAM:**<br>80421/ 7991/ 16770<br>**RIMES:** | <u>**Type 1:**</u><br>**IAM:**<br>35.1 / 10.8<br>**RIMES:** | **Pros:** First time application of dropout to multiple layers of RNN to recognize handwritten words and achieve comparable performance to state-of-the-art works. |

| | | | | |
|---|---|---|---|---|
| | **Database:** IAM, RIMES and OpenHaRT | 51739/ 7464/ 7776 **OpenHaRT:** 524196/ 57462/ 48308 | 28.5 / 6.8 **OpenHaRT:** 30.3 / 7.3 **Type 2:** **IAM:** 13.6 / 5.1 **RIMES:** 12.3 / 3.3 **OpenHaRT:** 18.0 / 4.7 | **Cons:** Fixed dictionary-based language model means unknown words cannot be recognized. Needs a high number of training samples. |
| Bluche et al. [27] | **Method:** ROVER **Learner:** Combination of BDLSTM-RNN and MLP **Database:** IAM and RIMES | **IAM:** 47952/ 20306 / 7558 **RIMES:** 51739/ 7464/ 7776 | **Type 2:** **IAM:** 11.90 / 4.90 **RIMES:** 11.8 / 3.7 | **Pros:** Performs well regardless of the type of features (handcrafted or pixel intensity values) and the neural network optical model (Deep MLP or RNN). **Cons:** Fixed dictionary based approach. Usage of an extra language model. |
| Doetsch et al. [24] | **Method:** LSTM-RNN with an additional parameter **Learner:** LSTM-RNN **Database:** IAM and RIMES | **IAM:** 47952/ 20306 / 7558 **RIMES:** 51739/ 7464/ 7776 | **Type 2:** **IAM:** 12.2 / 4.7 **RIMES:** 12.9 / 4.3 | **Pros:** Use of layer-specific gate weights in LSTM-RNNs for improving recognition accuracies. **Cons:** Lexicon based fixed dictionary approach with 50K words. |
| Almazán et al. [29] | **Method:** gradients of a Gaussian mixture model (GMM) and scale invariant feature transformation (SIFT) **Learner:** FV **Database:** IAM | 47952/ 20306 / 7558 | **Type 2:** 20.01 / 11.27 | **Pros:** Embedding of lexical string and a word image to represent it in a common vector space. Embedding is very fast to compute especially, to compare among different strings and words from images. **Cons:** The quality of the attribute models is quite dependent on the available number of training samples. The models for rare characters in rare positions are not particularly good. |
| Menasri et al. [28] | **Method:** A single RNN model with another classifier combination model **Learner:** RNN, MLP-HMM, GMM-HMMs **Database:** RIMES | 51739/ 7464 / 7776 | **Type 2:** **WER:** 15.2 | **Pros:** Design of weighted finite state transducers for classifier combination. A language and lexicon independent method. **Cons:** Usage of explicit word segmentation with the help of fixed vocabulary and language model leads to model failure in |

| | | | | |
|---|---|---|---|---|
| | | | | Type of overlapping, cramped and touching handwriting. |
| Graves and Schmidhuber [15] | **Method:** A hierarchy of MD LSTM-RNN model<br>**Learner:** MD RNN and MD LSTM<br>**Database:** 2007 ICDAR Arabic recognition contest (IFN/ENIT database) | **IFN/ENIT**<br>30000/ 2492/<br>set ‘f’: 8671<br>set ‘s’: 1573 | <u>**Type 2:**</u><br>**On set ‘f’:**<br>8.57 (WER)<br>**On set ‘s’:**<br>21.17 (WER)<br>(Top-1) | **Pros:** Uses a globally trained offline handwriting recognizer. No alphabet specific preprocessing is required and can be used for any language.<br>**Cons:** Trained with a huge dataset. Model is slow at training due to the combination of heavy MD recurrent model. |
| Stahlberg and Vogel [26] | **Method:** Fully connected DNN along with pixel and segment based features<br>**Learner:** DNN<br>**Database:** IFN/ENIT database and KHATT corpus | **IFN/ENIT**<br>30000/ 2492/<br>set ‘f’: 8671<br>set ‘s’: 1573<br>**KHATT**<br>31716/ 6635/ 26921 | <u>**Type 2:**</u><br>**IFN/ENIT**<br>**On set ‘f’:**<br>6.8 (WER)<br>**On set ‘s’:**<br>11.9 (WER)<br>**KHATT corpus**<br>30.9 (WER) | **Pros:** Improvement in state-of-the-art methods using intensive text image normalization. Feature extraction using raw grayscale pixel values or foreground segmentation. Language model independent.<br>**Cons:** The use of pixel and segment features makes it a data driven method. |
| Azeem and Ahmed [30] | **Method:** HMM system with foreground pixels and HOG as feature values<br>**Learner:** HMM<br>**Database:** IFN/ENIT database | **IFN/ENIT**<br>30,000/ 2,492/<br>set ‘f’: 8,671<br>set ‘s’: 1,573 | <u>**Type 2:**</u><br>**On set ‘f’:**<br>6.9 (WER)<br>**On set ‘s’:**<br>15.2 (WER) | **Pros:** Enhancement in the pre-processing stage and extracting concavity features from the whole image. Skew correction using fusion of multiple HMM.<br>**Cons:** Use of handcrafted features for language model. Closed dictionary based recognition. |
| Gui et al. [31] | **Method:** Adaptive context-aware reinforced agent based model<br>**Learner:** Reinforcement learning agent<br>**Database:** IAM, KHATT and RIMES | **IAM**<br>47952/ 20306 / 7558<br>**RIMES**<br>51739/ 7464/ 7776<br>**KHATT**<br>31716/ 26635/ 26921 | <u>**Type 2:**</u><br>**IAM**<br>5.45/3.10<br>**RIMES**<br>2.97 / 1.45<br>**KHAT**<br>6.93 (CER) | **Pros:** Proposes a novel adaptive context-aware reinforced agent for handwritten text recognition which can generalize both in isolated words and line sentences.<br>**Cons:** Multi-lines sentences and paragraphs need to be pre-segmented for application of this method. |
| Wu et al. [32] | **Method:** Combining a position embedding with residual networks (ResNets) and BLSTM network | **Esposalles**<br>31501 (5-fold cross validation) | <u>**Type 2:**</u><br>**Esposalles**<br>0.49 (CER)<br>**RIMES** | **Pros:** Novel unconstrained off-line handwritten word recognition method with no language resource. |

| | | | | |
|---|---|---|---|---|
| | **Learner:** PE-ResNets-BiLSTM<br>**Database:** ICDAR 2017 (Esposalles) and RIMES | **RIMES** | 1.79 (CER) | **Cons:** Classification is case sensitive and the majority of misclassification occurs in the case of mixed cased words |
| **Proposed** | **Method**: Character localization and recognition to recognize handwritten text word from images<br>**Learner:** YOLOv3<br>**Dataset:** Test: IAM Train: Self-made + IAM | 1200/ 300/ 30000 | **Type 1:**<br>29.21 / 9.53 | **Pros:** Lexicon-free handwritten cursive word recognition, performs localization and identification of characters simultaneously, used much fewer training samples, easily adaptable to other languages<br>**Cons:** Since manual threshold values are used for recognizing a character that makes it a user decision parameter. |

## 3. Proposed Method

It has already been mentioned that in the present work we use an object detection based model (in other words, a character spotting technique) to recognize a handwritten word image. The process of detecting an object (here, a character) from a pool of objects (here, set of characters forming a word image) consists of two separate sub-processes: (i) localization of an object and (ii) classification or identification of that object among all other objects. By the nature of the YOLOv3 architecture the character spotting (localization) and character classification (realization) occur simultaneously. It is to be noted that handwriting recognition is difficult due to the presence of overlapping characters and variation in the size and shapes of the characters when we consider unconstrained handwritten word images. Consequently, the training dataset must have multiple samples of each object class (i.e., characters), written in different styles.

The entire character spotting based word recognition technique is shown in Fig. 1. The left part of Fig. 1 depicts the training process while the testing process is shown on the right. During the training phase the model learns the character boundaries (also, its class) in the training word images with the help of associated ground truth information and during the testing phase, the character boundaries are detected with the help of the trained model. The ground truth for an input training/validation word sample contains bounding box information along with the class of all the characters present in the word sample. In the training phase of our framework, all the images are first resized to a size of $416 \times 416$ with data augmentation like skew, hue, exposure, etc. applied to them. Then the standard procedure of training deep learning models is followed where each batch of images (for our case, the batch size is 64) are fed to the model and learning proceeds by finding the loss between the model predicted output and the ground truth output, updating the model weight through back propagation. The model updates its weight for all the image batches. This is repeated for the entire training dataset for a certain number of epochs after which we get a trained model which can be used during testing. It is to be noted that after each epoch a validation step is performed where the model predicts the word images in the validation set and the accuracy is calculated. However, no updating of the model weights is performed during this phase. This validation set helps us to see how the model is generalizing across different handwritten word images. It also helps us to avoid the overfitting problem that might occur during the model training.

Our testing phase is quite straight forward where the test dataset images are sent to the trained model created during the training phase after resizing the images to 416 × 416. The trained model predicts the bounding boxes along with the respective class of the characters which are then post processed (refer to section 4.4 for details) to obtain the digital text of the handwritten word. In the following subsections, we describe how YOLOv3 spots the characters in a handwritten word image and its associated parts.

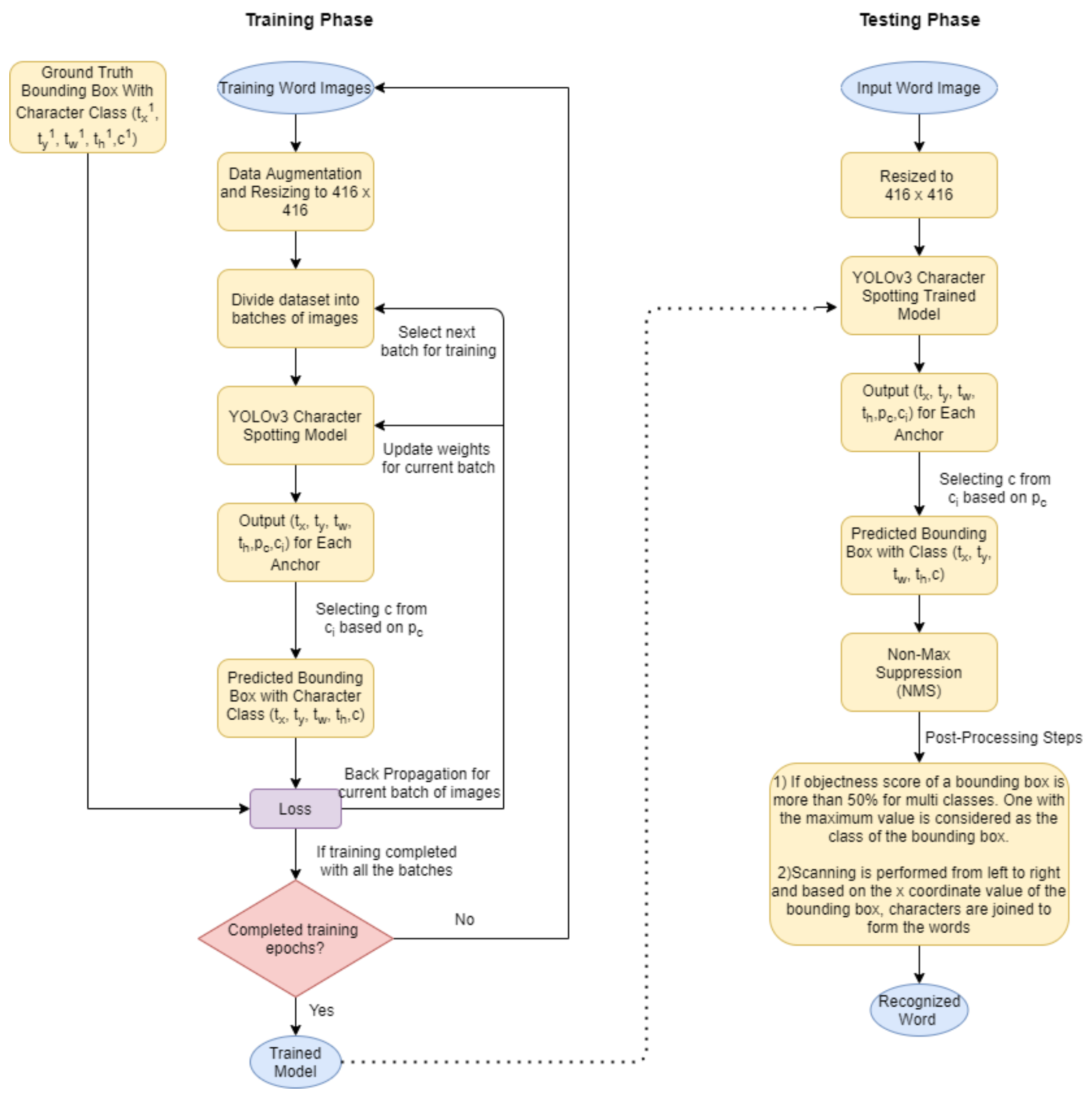

Training Phase
Testing Phase
Ground Truth Bounding Box With Character Class ($t_x^1$, $t_y^1$, $t_w^1$, $t_h^1$, $c^1$)
Training Word Images
Data Augmentation and Resizing to 416 x 416
Divide dataset into batches of images
Select next batch for training
YOLOv3 Character Spotting Model
Update weights for current batch
Output ($t_x$, $t_y$, $t_w$, $t_h$, $p_c$, $c_i$) for Each Anchor
Selecting c from $c_i$ based on $p_c$
Predicted Bounding Box with Character Class ($t_x$, $t_y$, $t_w$, $t_h$, c)
Back Propagation for current batch of images
Loss
If training completed with all the batches
Completed training epochs?
No
Yes
Trained Model
Input Word Image
Resized to 416 x 416
YOLOv3 Character Spotting Trained Model
Output ($t_x$, $t_y$, $t_w$, $t_h$, $p_c$, $c_i$) for Each Anchor
Selecting c from $c_i$ based on $p_c$
Predicted Bounding Box with Class ($t_x$, $t_y$, $t_w$, $t_h$, c)
Non-Max Suppression (NMS)
Post-Processing Steps
1) If objectness score of a bounding box is more than 50% for multi classes. One with the maximum value is considered as the class of the bounding box.
2)Scanning is performed from left to right and based on the x coordinate value of the bounding box, characters are joined to form the words
Recognized Word

**Fig. 1**: Diagram of the word recognition framework. Training is shown on the left and test on the right.

## 3.1 Overview of YOLOv3 model

As image classification models improve researchers also come up with additional image recognition models. Since we aim to spot the characters in a handwritten word image to recognize the entire word image, we design the present word recognition model to perform object (i.e., character) detection and recognition simultaneously with the help of the YOLOv3 model.

Object detection methods are divided into two categories based on the number of stages used to perform the task. The object detection and recognition models like R-CNN [37], Fast R-CNN [38], Faster R-CNN [22], and Mask R-CNN [39] work in two stages, and in general, different networks are used in these stages. In the first stage, the region is searched to determine whether the object is present or not using the region proposal scheme. If there are any objects present in the proposed region, the object classification model is used to identify which object is present in that region. Similar approaches are found in [21][40] for recognizing handwritten words [21] and digit strings [40]. Although these two stage methods give very promising results, these models are very slow to train and often cannot be used in real time applications due to the high computational resource requirement. Another variant falls under the category of single stage object detection and recognition model. Some of the well-known single stage object detection models are single shot detection model (SSD) [41], RetinaNet [42] and YOLO family [23]. YOLO based models only pass the input image through their single network once, thus the name YOLO. This makes the training of YOLO based models faster with very high state-of-the-art results for real time usage and influenced our decision to use it.

The YOLO predicts object (here, characters) boundaries at three resolutions. The feature maps used for the detection purpose are generated following the FPN architecture. In FPN, initially, the special dimension of the feature maps decreases like the usual cases and in the later stages, the dimension of the feature map increases again through up-sampling and gets concatenated with preceding feature maps with corresponding sizes. This process is repeated three times and each concatenated feature map is fed to an object detection process. Thus, the object detector in use processes an input word image at a different spatial resolution. This process is shown in Fig. 2.

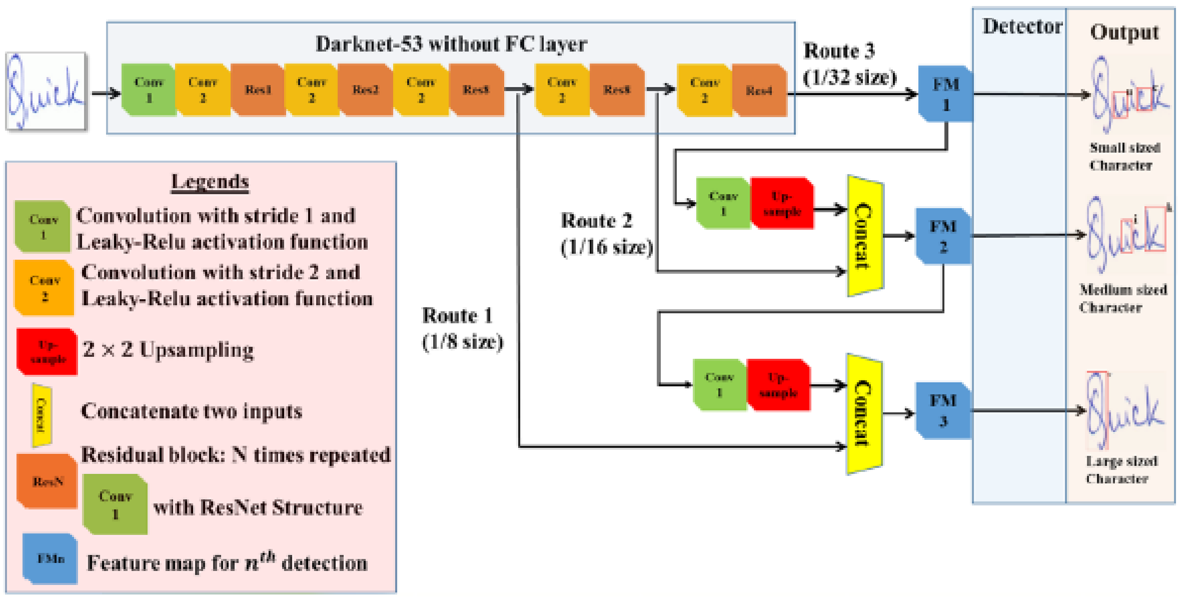


**Fig. 2:** The block diagram of the YOLOv3 architecture used here for handwritten word recognition. In the present work, the dimension (i.e., width × height ) of the input word is 416 × 416 . Hence, the dimensions of the feature maps represented by FM1, FM2 and FM3 are 13 × 13, 26 × 26 and 52 × 52 respectively. This means the detection of constituting characters of a word is performed at three different scales (viz.,13 × 13, 26 × 26 and 52 × 52). The Conv2 blocks (i.e., convolutional operation with stride 2) are responsible for reducing the dimension of the feature maps. The residual blocks use skip connection like the ResNet architecture.

In the YOLOv3 architecture, the presence of a residual network in the backbone network, a skip connection before the detection layer, and the up-sampling of layers before concatenating feature maps of the same dimension help to detect objects of various scales. The YOLOv3 architecture (see Fig. 2) used in this work predicts character boundaries inside a word image at three different spatial resolutions: $13 \times 13$, $26 \times 26$ and $52 \times 52$. The feature maps of dimension (i.e., $width \times height$) $13 \times 13$ have a larger context at a smaller resolution in comparison with other feature maps. These feature maps help detect relatively large sized character(s). Similarly, the feature maps of dimension $52 \times 52$ help detect smaller characters(s). This is the key reason behind the success of the YOLOv3 in recognizing words with fewer training samples as compared to other state-of-the-art methods.

The detection (prediction) attributes are stored in a depth-wise fashion and the shapes of the attribute are $1 \times 1 \times (B \times (5 + C))$, where B is the number of bounding boxes a cell in the feature map can predict, C is the number of classes, 5 corresponds to 4 bounding box attributes and 1 object confidence score value. In this work, following the suggestion of [23], we set the values of B as 3. C is 52 for the letters a-z and A-Z.

The model, shown in Fig. 2, takes input images in batches of shape ($m$, 416, 416, 3) where $m$ is the batch size. The output is a list of bounding boxes and their class prediction score in the form ($pc$, $b_x$, $b_y$, $b_h$, $b_w$, $c$) for each cell in the prediction map. $pc$ is the object confidence score of a bounding box in the output image, $b_x$, $b_y$, $b_h$, $b_w$ are the respective values of the center position in the $x$ direction, $y$ direction and height and width of the predicted bounding box in the output image. Here for our task of English text recognition the number of character classes, $c$, is 52 (26 lower case letters and 26 upper case letters). Punctuation, numerals and special characters could also be included, but are not considered in this work.

A bounding box prior is assigned to each cell of the grid on the detection feature map from the set of anchors [22] which overlaps the maximum area with the ground truth bounding box at the time of training. For this work we use the anchors, i.e., bounding box priors, taken from the COCO dataset [45]: $(10 \times 13)$, $(16 \times 30)$, $(33 \times 23)$, $(30 \times 61)$, $(62 \times 45)$, $(59 \times 119)$, $(116 \times 90)$, $(156 \times 198)$ and $(373 \times 326)$.

In YOLOv3, prediction of the center ($t_x$, $t_y$), height ($t_h$) and width ($t_w$) of a bounding box is handled as a regression problem. These are transformed to $b_x$, $b_y$, $b_h$, $b_w$ by

$$b_x = sigmoid(t_x) + c_x \qquad (1)$$

$$b_y = sigmoid(t_y) + c_y \quad (2)$$

$$b_w = p_w * e(t_w) \quad (3)$$

$$b_h = p_h * e(t_w) \quad (4)$$

for final prediction of bounding boxes in the output image. In these equations, $c_x$ and $c_y$ are the top left coordinates of the cell in the grid of the feature map and $p_w$ and $p_h$ are the anchor dimensions for the bounding box under consideration. We can see from equations 1 and 2 that $t_x$ and $t_y$ are kept between 0 and 1 using a sigmoid function and an extra offset is provided by adding constants $c_x$ and $c_y$. This is done to bring the center of the bounding box of an object inside a cell that is trying to predict that object.

### 3.1.3 Loss function

In YOLOv3 cross-entropy loss (logistic regression) is used to predict an object's (here, characters) confidence score, which gives the probability of whether an object is present or not within the bounding boxes of a grid cell, and if present, the class of this predicted object and the values of $t_x$, $t_y$, $t_w$ and $t_h$. Here the class scores are predicted through a sigmoid layer using logistic regression for multi label classification. The loss function of YOLOv3 is a combination of different loss functions. Mean square error (MSE) is used for bounding box detection, and for classifying the characters binary cross entropy (BCE) is used:

$$\text{MSE}(t_x, t_x^1) + \text{MSE}(t_y, t_y^1) + \text{MSE}(t_w, t_w^1) + \text{MSE}(t_h, t_h^1) + \text{BCEobj}(p_c, p_c^1) + \text{BCEno-obj}(p_c, p_c^1) + \text{For\_all\_class}_{ci}(BCE(P(ci^1), P(ci^2))) \quad (5)$$

where $t_x^1, t_y^1, t_w^1, t_h^1$ and $p_c^1$ are the respective ground truth values for our predicted $t_x, t_y, t_w, t_h$ and $p_c$ respectively for a particular grid cell in the detection feature map. Here the object confidence score ($p_c$) is used for two types of losses: the presence of any object (BCEobj($p_c$, $p_c^1$)) and the absence of any object (BCEno-obj ($p_c$, $p_c^1$)). P($ci^1$) and P($ci^2$) are the class sigmoid scores of predicted and ground truth objects respectively. Although we use 52 classes where the characters are exclusive in nature ('A' is different than 'a'), keeping the original loss function of YOLOv3 in mind we have used BCE for individual classes separately instead of a single softmax function for classifying a character.

## 3.2 Character detection

In this stage, we first train the YOLOv3 model described above and then feed the unseen handwritten word images to the model. The training dataset is created by manually preparing the bounding boxes (detailed in section 4.1). Before supplying the images to the training pipeline, all images are resized to dimension 416 × 416 for computational ease. Images are divided into several batches. Each batch is trained for a different number of epochs. Our character spotting model predicts the bounding box of each character and these are compared with the ground truth bounding box to calculate the loss. Using back propagation, the loss is used to update the weights of the model for the current batch. Similarly, all the batches are trained to update the model weights until we reach the maximum number of training epochs. More details about the training process are described in section 4.2. During the training, different parameters are adjusted based on the performance of the trained model on a given validation set i.e., validation loss is used to decide the best setup. The best performing model is used to recognize the test word image samples.

The test word images are first resized to $416 \times 416$ to feed into the model built during training. The bounding box of each of the characters of a text is predicted through a non-max suppression algorithm and linear scanning based character selection to get the recognized word. The test process returns the identified character boundaries with its class and objectiveness score based on which character has been recognized. The final set of identified characters that the model returns depends on a threshold value applied for the objectiveness score. We set this threshold value a bit less than suggested in [23] with the goal that no character remains undetected.

### 3.3 Post-processing

From the previous stage, we obtain the detected characters' boundary information along with the confidence score at which they are recognized. The choice of lower value of threshold results in detecting multiple overlapped charactes. Therefore, during the post-processing stage, we handle the overlapped characters. We scan the character boundaries of a word image from left to right and calculate the overlapped regions between two consecutive character boundaries. The overlapped region ($\Delta$) is calculated by

$$\Delta = \frac{number\ of\ columns\ shared\ by\ two\ consecutive\ character\ boundaries}{number\ of\ columns\ occupied\ by\ the\ smaller\ character}. \tag{6}$$

If we find $\Delta > 0.5$ for two consecutive characters, then we suppress the detected characters with the lower objectiveness score. We use the final set of characters to generate the text of the handwritten word image.

## 4. Results and Analysis

In this section, we first describe the data preparation and training process of our method and then we present the experimental results of using the YOLOv3 model customized to do character spotting for handwritten word recognition. Finally, we compare our results with some other state-of-the-art methods trained and tested on our datasets and the performance reported of some methods on the IAM dataset.

### 4.1 Data collection and preparation

For the experiments, we have used word images from the IAM handwriting dataset [25]. Each of the handwritten word images is programmatically cropped from the offline document images written by different writers. In a writer independent way we have randomly selected 700 training and 30000 testing word images from the entire IAM pre-segmented word image dataset. Along with this, we have also included 800 additional writing samples from an in-house handwritten English word image dataset to the training set to improve the coverage of the alphabet. We have drawn the bounding boxes of those 1500 training word images using the tool provided in [46]. We have used 300 word images as a validation set during training. Hence, we have used only 1200 word images for learning with YOLOv3. Some of the sample images of the training dataset are shown in Fig. 3.

### 4.2 Performance Metrics

To assess the performance of the present end-to-end word recognition model, we use two popularly used performance metrics: word error rate (WER) and character error rate (CER) [21]. WER is the percentage of total test set word samples that are incorrectly recognized while CER represents the percentage of the characters present in the entire test set recognized wrongly. WER and CER are calculated by

$$WER = \frac{Number\ of\ words\ incorrectly\ recognized}{Total\ number\ of\ words\ present\ in\ the\ test\ set} \tag{7}$$

$$CER = \frac{Number\ of\ erroneously\ recognized\ characters}{Total\ number\ of\ characters\ present\ in\ the\ test\ set\ words}. \tag{8}$$

### 4.3 YOLOv3 Training and Parameter Tuning

We train the model using the Darknet framework [44] that follows a transfer learning strategy starting with the YOLOv3 architecture pre-trained on the COCO dataset, and the Darknet-53 architecture pre-trained on the ImageNet dataset. The YOLOv3 model is trained using the "steps" policy as the learning algorithm. The batch size is 64. The initial learning rate is 0.001 and the momentum of 0.9 is used. For learning rate decay in each iteration, a decay rate of 0.005 is used. The model is trained for 15000 iterations. All the train and test images are resized to $416 \times 416$. We evaluated other learning algorithms namely, "random", "polynomial" and, "stochastic gradient descent (SGDR)", but the "step" policy produced the best results. Table 3 shows the performance of the present end-to-end word recognition model in terms of WER and CER while using different learning methods keeping other parameters fixed.

For data augmentation, we manually rotate some of the input images by an angle of 5 to 10 degrees to introduce slant to the characters and annotate it as such. We also randomly vary the saturation, exposure and hue of the image inputs to introduce noise to make the model robust to outliers, since the background of the text in the datasets can be white, grey or yellow. We compared training with different data augmentation processes and the results are listed in Table 4. Results in this table indicate the benefits of using the said augmentation. It is to be mentioned here that all the results listed in this table are the performance of the model while evaluated on validation set word samples.

We also test the performance on the other classification training procedures where cropping is introduced for better classification which leads to more time in training, our work does not use cropping since our model can detect characters (small, medium and large characters) with the help of the 3-scale detection strategy. Unlike the original YOLOv3 model where a 0.7 threshold is used to detect objects, we find 0.5 is the best value for our purposes. The performance of the present word recognition model (evaluated on validation set word images) while using different threshold values are shown in Table 5. The results in this table show that 0.5 is the best threshold value. The threshold values less than 0.5 segment a character into many smaller components or detect a part of a character written boldly as new characters and thereby they pass through the condition of suppressing unnecessary characters (see section 3.3). However, when using the larger threshold values, it misses many characters present in the word image.

**Table 3**: The loss of the end-to-end word recognition system when evaluated on validation set word images using different learning algorithms. The initial learning rate=0.001, momentum=0.9, decay rate=0.005 and number of iterations=15000. No data augmentation is used.

| Learning algorithm | Performances (in %) | |
|---|---|---|
| | CER | WER |
| Random | 11.97 | 26.67 |
| Polynomial | 12.68 | 27.33 |
| SGDR | 10.82 | 25.67 |
| **Steps** | **08.92** | **23.33** |

**Table 4:** The performance of the end-to-end system with varying augmentation models. The initial learning rate=0.001, momentum=0.9, decay rate=0.005, number of iterations=15000 and learning algorithm="steps".

| Data augmentation procedure | Performances (in %) | |
|---|---|---|
| | CER | WER |
| No augmentation | 08.97 | 23.33 |
| Only rotation | 07.85 | 21.67 |
| Only saturation, exposure and hue change | 07.79 | 21.33 |
| Rotation and saturation, exposure, and hue change | **05.58** | **18.33** |

**Table 5**: The performance of the end-to-end system with varying YOLOv3 threshold values. The initial learning rate=0.001, momentum=0.9, decay rate=0.005, number of iterations=15000 and learning algorithm="steps". Bold faced numbers indicate the best scores.

| YOLOv3 threshold value to predict the presence of a character | Performances (in %) | |
|---|---|---|
| | CER | WER |
| 0.3 | 28.73 | 44.67 |
| 0.4 | 17.09 | 30.33 |
| 0.5 | **5.58** | **18.33** |
| 0.6 | 15.78 | 24.67 |
| 0.7 | 23.49 | 33.33 |
| 0.8 | 31.49 | 47.33 |

## 4.4 Results and Evaluation

The model predicts multiple bounding boxes, which need to be filtered. Boxes having an object confidence score below a threshold (0.5 or 50%) are filtered out first. After which, non-maximum suppression (NMS) [47] is used to remove cases of multiple bounding boxes for the same character in the image, based upon an intersection over union (IOU) threshold (0.45 or 45%) [48]. Even after removing such bounding boxes if the confidence score of more than one class for the same bounding box is above 0.5 for the object confidence threshold and above 0.45 for IOU threshold, then we consider the class with the maximum score as the most probable class for that character (see Fig. 4).

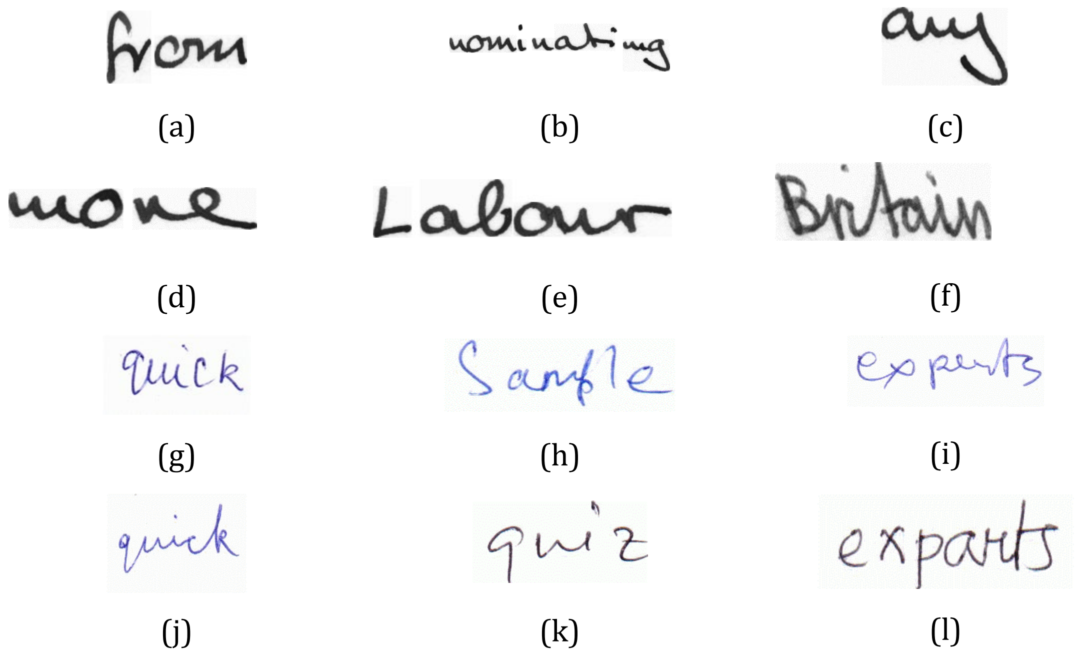


**Fig. 3**: Some specimens from the training dataset. Images (a-f) are taken from IAM and (g-l) from the in-house datasets.

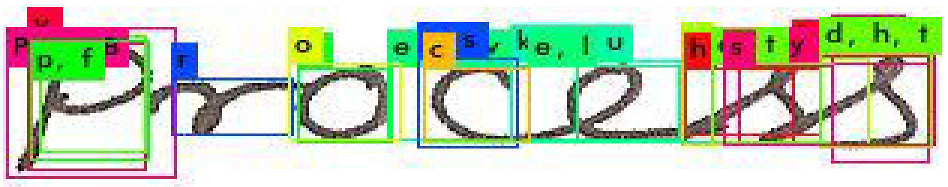
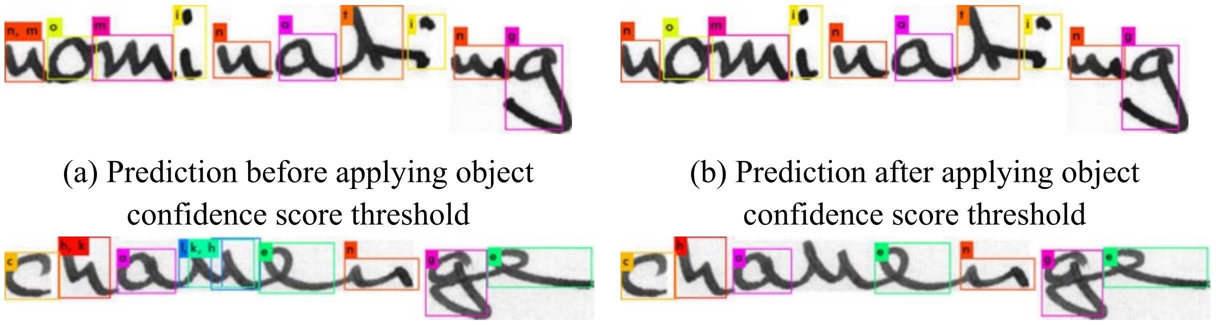


**Fig. 4**: Multiple bounding boxes are produced in an intermediate stage for an image, from which the best box is selected based on the object confidence score, the NMS algorithm and the maximum class confidence score.

For example, in Fig. 5(a) the first character ‘n’ has been predicted both as n and m with object confidence scores of 92% and 32% respectively. We have used an object confidence score-based threshold which should be above 50% to consider a predicted bounding box object to be an actual character, which removes the ‘m’ and leads to correct recognition of the word which is “nominating” (see Fig. 5(b)). Likewise, in Fig. 5(c) the word “challenge” has been predicted considering the first ‘h’ as ‘h’ and ‘k’ with object confidence scores of 89% and 63% respectively. Since both are above the object score threshold, we select the class with the maximum score to identify the character. The sub-string “ll” in that word has been completely misclassified as the object confidence score is below 50% for this case and consequently, we have rejected it in the final predicted word (see Fig. 5(d)) and hence the string "ll" remains non-detected which is a limitation of the present work.

To convert the detections into the OCRed output, we traverse the image from left to right. This way the positions of the characters can be maintained and converted into digital form. We compare the ground truth text with our predicted text using two metrics for evaluating the performance of our model namely WER and CER [21]. We achieve a **29.21%** WER i.e., our system fails to correctly recognize only 8761 of 30000 test word images. However, our system possesses only a **9.53**% CER i.e., out of 135,316 characters only 12896 of the characters are misclassified according to the characters in the ground truth text. Some of the successfully recognized word images are shown in Fig. 6 and error cases are shown in Fig. 7.



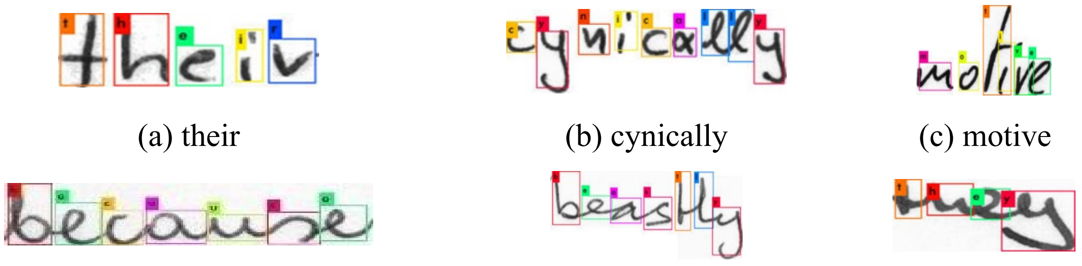
(a) Prediction before applying object confidence score threshold

(b) Prediction after applying object confidence score threshold

(c) Character regions prediction before applying object confidence threshold and NMS

(d) Prediction after applying object confidence threshold and NMS

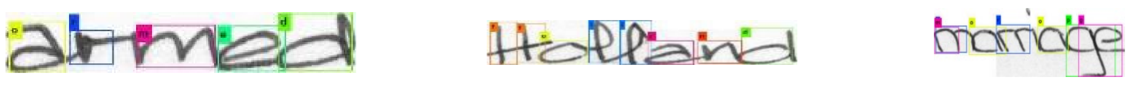
**Fig. 5**: Illustration of output processing before converting the sample into machine-encoded form.



(a) their (b) cynically (c) motive

(d) because (e) beastly (f) they

**Fig. 6**: Examples of images where the entire word is recognized perfectly.

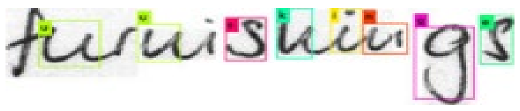


(a) "armed" is recognized as "ormed"

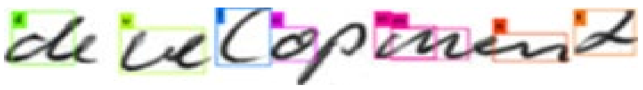
(b) "Holland" recognized as "ttolland"

(c) "marriage" is recognized as "moropg"

(d) "positions" is recognized as "prkons"



(e) "furniskings" is recognized as "uuskinge"



(f) "development" is recognized as "dvlommnt"

**Fig. 7**: Examples of images where our system fails to recognize the entire word correctly.

## 4.5 Comparison with State-of-the-art Methods

In Section 2 Table 1 showed several state-of-the art methods and their performance on several data sets. Here we provide additional comparison performed following two different approaches. In the first approach, we evaluate the methods proposed by Sueiras et al. [18] and Bhattacharya et al. [19] on the present dataset with identical splits. We consider three variants depending on choice of lexicon of these two lexicon-based methods. The underlying lexicons are constructed from words of the entire IAM dataset and words of the test set. We also evaluate both these models without any lexicon. Such choices help us to build 6 different word recognition systems. We train these models along with our model on the present training set for five times and record the results on the test set each time. The results are shown in Tables 6 and 6.

We also perform a statistical test to ensure that the results of our method are statistically significant compare to the other methods considered here for comparison. The goal is to determine whether there is enough evidence to "reject" a hypothesis: the proposed model does not perform well as compared to other methods. Here, we calculate the p-value from the error rates provided in Tables 6 and 7. If the calculated $p - value < 0.05$, then we reject the null hypothesis at a 5% significance level. To determine the p-value, we use the Wilcoxon rank-sum test [49][50], which is a non-parametric statistical test process. Here, we calculate pairwise p-values and the estimated p-values are recorded in Table 8. From the test results provided in Table 6 (in terms of WER score) and Table 7 (in terms of CER score), we can conclude that the null hypothesis can be rejected for any pair of tests, which means our method outperforms the others on the present dataset.

In the second approach of comparison, we revisit the performance scores reported in the literature by the authors of several state-of-the-art methods. The performance scores are provided in Table 9, which is a simplification of Table 1. It is to be noted that these results were generated using larger training and validation sets and fewer test samples than the present scenario. For example, the methods proposed by Sueiras et al. [18] and Bhattacharya et al. [19] obtain around 23.80% (better than the current model) and 31.30% (worse than the current model) WER respectively on the IAM dataset while no dictionary based correction is made like ours. Their CER scores are 8.80% (better than the current model) and 13.20% (worse than the current model) respectively. But the number of train, validation and test samples used in these works are 47952 (nearly 40 times than ours), 20306 (nearly 68 times than ours) and 7558 (nearly 0.25 times than ours) respectively. However, when these two methods are trained and tested on our current train, validation and test datasets the performances are poor (the current model provides more than 2 times WER and 6 times CER) as compared to the current model (see Tables 6 and 7). Hence, all this information ensures that our method uses significantly smaller training samples than the state-of-the-art methods while producing comparable results with state-of-the-art methods and there is no lexicon in use.

**Table 6**: The CER and WER scores computed from 3 methods (Sueiras et al. [18], Bhattacharya et al. [19], and the proposed method) over five independent runs on the present dataset. Here, zero lexicon means no lexicon driven correction is made while the test set and IAM lexicon indicate that the lexicon is formed using words of the test set and all words of the IAM dataset.

| | | Method | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | Sueiras et al. [18] | | | Bhattacharya et al. [19] | | | **Proposed** |
| | Lexicon | Zero | Test set | IAM | Zero | Test set | IAM | Zero |
| CER (in %) | Maximum | 68.03 | 66.25 | 63.16 | 66.67 | 64.98 | 63.60 | 9.23 |
| | Average | 68.84 | 67.29 | 64.61 | 68.64 | 66.35 | 64.65 | 9.53 |
| | Standard deviation | 0.57 | 0.56 | 0.91 | 1.05 | 0.86 | 0.60 | 0.19 |
| WER | Minimum | 71.35 | 69.18 | 64.74 | 70.85 | 68.68 | 63.71 | 28.99 |
| | Average | 72.72 | 70.23 | 66.24 | 72.80 | 69.61 | 65.30 | 29.21 |

**Table 7**: Shows the statistical test reports for WER (CER): calculated p-values of the Wilcoxon rank sum test of the proposed method with two other methods with their three variants considered here. The p-values 'X(Y)' represents X is the p-value considering WER whereas p-value considering CER is Y.

| | Sueiras et al. [18] with lexicon | | | Bhattacharya et al. [19] with lexicon | | |
|---|---|---|---|---|---|---|
| | Zero | Test set | IAM | Zero | Test set | IAM |
| WER Proposed Method | 0.0045 | 0.0045 | 0.0045 | 0.0045 | 0.0045 | 0.0045 |
| CER Proposed Method | 0.0045 | 0.0045 | 0.0045 | 0.0045 | 0.0045 | 0.0045 |

**Table 8**: Comparison of state-of-the-art recognition results on the IAM dataset. Note that these results were generated using a significantly larger train and validation set and fewer test samples as compared to ours.

| Method | Number of train / test / validation word images | Vocabulary size used | Recognition accuracy (in %) | |
|---|---|---|---|---|
| | | | WER | CER |
| Sueiras et al. [18] | 47952 / 20306 / 7558 | 50K | 19.7 ±0.03 | 9.5 ±0.03 |
| Bhattacharya et al. [19] | Do | Words of entire IAM dataset | 18.6 | 9.70 |
| Bluche et al. [27] | Do | 50K | 11.90 | 4.90 |
| Doetsch et al. [24] | Do | 50K | 12.20 | 4.70 |
| Almazán et al. [29] | Do | 50K | 20.01 | 11.27 |
| Pham et al. [20] | 80421 / 17991 / 16770 | 50K | 13.60 | 5.10 |
| Sueiras et al. [18] | 47952 / 20306 / 7558 | No vocabulary | 23.80 ±0.05 | 8.8 ±0.02 |
| Bhattacharya et al. [19] | Do | No vocabulary | 31.30 | 13.20 |
| Pham et al. [20] | 80421 / 17991 / 16770 | No vocabulary | 35.10 | 10.80 |
| **Proposed** | 1200 / 30000 / 300 | No vocabulary | 29.21 | 9.53 |

## 5. Conclusion and future scope

In this work, we have changed the approach to handwriting recognition to locate all the possible characters in a document and from that form the recognized words, instead of recognizing all the words. We used the YOLOv3 model for segmentation as well as recognition of the characters in handwritten word images. Although different OCR models have been used to recognize handwriting, they fail massively when the handwriting is either cursive or the characters overlap too much. Since the arrival of YOLOv3, researchers have used it to solve many complex computer vision problems like human activity recognition, for recognizing different types of objects present in a given video or image. By taking inspiration from such examples, we have approached our task with YOLOv3. We have designed an end-to-end lexicon-free handwritten cursive word recognition model. The key advantage of this model is that it performs localization and identification of characters simultaneously in a handwritten word image using the YOLOv3 network. The model is designed in such a way that it works well even if we use a minimum number of ground truth images during training, thereby minimizing the training cost and moreover the cost of collecting labeled training data significantly. It has been observed that our model has performed significantly better with fewer training samples (around 2% and 1% of that used in [18] and [20]) and thus, minimizing the training cost.

We have obtained competitive results compared to state-of-the-art methods when evaluating on the standard IAM dataset. It is to be noted that we have not used any preprocessing techniques such as skew or slant correction which are commonly used in other research works.

Another notable observation is that this model can be used to recognize misspelled and out-of-dictionary words. If a scenario were to exist with a closed vocabulary, that could be used at the end to improve the results further. In spite of the good results obtained by the proposed model, there are some scopes of improvement. For example, we have obtained lower WER scores because in many cases upper case characters have been recognized erroneously. The reason might be the lower number of uppercase characters present in the training images. So, in the future inclusion of more upper case characters in the dataset can be considered. In some cases the model fails to detect some characters within a word image. So we need more variants of the character samples in the training word images. Besides, the loss function used here is inspired by the YOLOv3 architecture. We will modify the existing architecture to adjust the loss function to be more specific to the task of handwritten word recognition. Also, we will try to replace the manual threshold values by adaptive threshold values. Apart from these, we aim to develop a real prototype which will enhance the capability of modern OCR systems to recognize of any type of text used in our daily life. We will also apply this system to the recognition of handwritten text written in other scripts.

**Acknowledgment**

We would like to thank the Centre for Microprocessor Applications for Training, Education and Research (CMATER) research laboratory of the Computer Science and Engineering Department, Jadavpur University, Kolkata, India for providing us the infrastructural support.

**Conflict of Interest**

We declare that we have no conflict of interest.

**References**

1. Malakar S, Ghosh M, Sarkar R, Nasipuri M (2020) Development of a Two-Stage Segmentation-Based Word Searching Method for Handwritten Document Images. J Intell Syst 29: . doi: 10.1515/jisys-2017-0384
2. Majumder S, Ghosh S, Malakar S, et al (2021) A voting-based technique for word spotting in handwritten document images. Multimed Tools Appl preprint:1–24 . doi: https://doi.org/10.1007/s11042-020-10363-0
3. Pal U, Roy RK, Kimura F (2012) Multi-lingual city name recognition for Indian postal automation. In: Proceedings of the International Workshop on Frontiers in Handwriting Recognition. IEEE, pp 169–173
4. Ghosh S, Bhattacharya R, Majhi S, et al (2018) Textual Content Retrieval from Filled-in Form Images. In: Proceedings of the Workshop on Document Analysis and Recognition. Springer, pp 27–37

5. Bhattacharya R, Malakar S, Ghosh S, et al (2020) Understanding contents of filled-in Bangla form images. Multimed Tools Appl 80:3529–3570

6. Singh S, Kariveda T, Gupta J Das, Bhattacharya K (2015) Handwritten words recognition for legal amounts of bank cheques in English script. In: Proceeding of the 2015 8th International Conference on Advances in Pattern Recognition. IEEE, pp 1–5

7. Singh PK, Mahanta S, Malakar S, et al (2014) Development of a page segmentation technique for Bangla documents printed in italic style. In: Proceedings of the 2nd International Conference on Business and Information Management (ICBIM 2014)

8. Sarkar R, Malakar S, Das N, et al (2011) Word extraction and character segmentation from text lines of unconstrained handwritten Bangla document images. J Intell Syst 20:227–260 . doi: 10.1515/JISYS.2011.013

9. Malakar S, Sarkar R, Basu S, et al (2020) An image database of handwritten Bangla words with automatic benchmarking facilities for character segmentation algorithms. Neural Comput Appl Preprint:1–20 . doi: https://doi.org/10.1007/s00521-020-04981-w

10. Malakar S, Ghosh P, Sarkar R, et al (2011) An improved offline handwritten character segmentation algorithm for Bangla script. In: Proceedings of the 5th Indian International Conference on Artificial Intelligence (IICAI 2011)

11. Bera SK, Chakrabarti A, Lahiri S, et al (2019) Normalization of unconstrained handwritten words in terms of Slope and Slant Correction. Pattern Recognit Lett 128:488–495

12. Bera SK, Kar R, Saha S, et al (2018) A One-Pass Approach for Slope and Slant Estimation of Tri-Script Handwritten Words. J Intell Syst 29:688–702 . doi: 10.1515/jisys-2018-0105

13. Marti U-V, Bunke H (2001) Using a statistical language model to improve the performance of an HMM-based cursive handwriting recognition system. In: Hidden Markov models: applications in computer vision. World Scientific, pp 65–90

14. Graves A, Liwicki M, Fernández S, et al (2008) A novel connectionist system for unconstrained handwriting recognition. IEEE Trans Pattern Anal Mach Intell 31:855–868

15. Graves A, Schmidhuber J (2009) Offline Handwriting Recognition with Multidimensional Recurrent Neural Networks. In: Proceedings of the Advances in

Neural Information Processing Systems. pp 545–552

16. Graves A, Fernández S, Gomez F, Schmidhuber J (2006) Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In: Proceedings of the 23rd International Conference on Machine learning. ACM, pp 369–376

17. Shi B, Bai X, Yao C (2016) An End-to-End Trainable Neural Network for Image-Based Sequence Recognition and Its Application to Scene Text Recognition. IEEE Trans Pattern Anal Mach Intell 39:2298–2304

18. Sueiras J, Ruiz V, Sanchez A, Velez JF (2018) Offline continuous handwriting recognition using sequence to sequence neural networks. Neurocomputing 289:119–128

19. Bhattacharya R, Malakar S, Schwenker F, Sarkar R (2021) Fuzzy-Based Pseudo Segmentation Approach for Handwritten Word Recognition Using a Sequence to Sequence Model with Attention. In: Pattern Recognition. ICPR International Workshops and Challenges: Virtual Event, January 10–15, 2021, Proceedings, Part II. Springer International Publishing, pp 582–596

20. Pham V, Bluche T, Kermorvant C, Louradour J (2014) Dropout improves recurrent neural networks for handwriting recognition. In: Proceedings of the 14th International Conference on Frontiers in Handwriting Recognition. IEEE, pp 285–290

21. Majid N, Barney Smith EH (2019) Segmentation-free Bangla offline handwriting recognition using sequential detection of characters and diacritics with a Faster R-CNN. In: 2019 International Conference on Document Analysis and Recognition (ICDAR). IEEE, pp 228–233

22. Ren S, He K, Girshick R, Sun J (2015) Faster R-CNN: Towards real-time object detection with region proposal networks. In: Proceedings of the Advances in neural information processing systems. pp 91–99

23. Redmon J, Farhadi A (2018) Yolov3: An incremental improvement

24. Doetsch P, Kozielski M, Ney H (2014) Fast and Robust Training of Recurrent Neural Networks for Offline Handwriting Recognition. In: Proceedings of the 14th International Conference on Frontiers in Handwriting Recognition. IEEE, pp 279–284

25. Marti U V, Bunke H (2002) The IAM-database: an English sentence database for offline handwriting recognition. Int J Doc Anal Recognit 5:39–46

26. Stahlberg F, Vogel S (2015) The QCRI recognition system for handwritten Arabic. In: International Conference on Image Analysis and Processing. Springer, pp 276–286

27. Bluche T, Ney H, Kermorvant C (2014) A comparison of sequence-trained deep neural networks and recurrent neural networks optical modeling for handwriting recognition. In: International Conference on Statistical Language and Speech Processing. Springer, pp 199–210

28. Menasri F, Louradour J, Bianne-Bernard A-L, Kermorvant C (2012) The A2iA French handwriting recognition system at the Rimes-ICDAR2011 competition. In: Proceedings of the Document Recognition and Retrieval XIX. International Society for Optics and Photonics, p 82970Y

29. Almazán J, Gordo A, Fornés A, Valveny E (2014) Word spotting and recognition with embedded attributes. IEEE Trans Pattern Anal Mach Intell 36:2552–2566

30. Azeem SA, Ahmed H (2013) Effective technique for the recognition of offline Arabic handwritten words using hidden Markov models. Int J Doc Anal Recognit 16:399–412

31. Gui L, Liang X, Chang X, Hauptmann AG (2018) Adaptive Context-aware Reinforced Agent for Handwritten Text Recognition. In: British Machine Vision Conference. p 207

32. Wu X, Chen Q, You J, Xiao Y (2019) Unconstrained Offline Handwritten Word Recognition by Position Embedding Integrated ResNets Model. IEEE Signal Process Lett 26:597–601

33. Grosicki E, El-Abed H (2011) ICDAR 2011 - French Handwriting Recognition Competition. In: Proceedings of the International Conference on Document Analysis and Recognition. IEEE, pp 1459–1463

34. Poznanski A, Wolf L (2016) CNN-N-Gram for Handwriting Word Recognition. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition. pp 2305–2314

35. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp 770–778

36. Szegedy C, Ioffe S, Vanhoucke V, Alemi AA (2017) Inception-v4, inception-ResNet and the impact of residual connections on learning. In: Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence. pp 4278–4284

37. Girshick R, Donahue J, Darrell T, Malik J (2014) Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of IEEE Conference on Computer Vsion and Pattern Recognition. pp 580–587

38. Girshick R (2015) Fast R-CNN. In: Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV). pp 1440–1448

39. Zhang Y, Nie S, Liu W, et al (2019) Sequence-to-sequence domain adaptation network for robust text image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp 2740–2749

40. Chakraborty A, De R, Malakar S, et al (2021) Handwritten Digit String Recognition using Deep Autoencoder based Segmentation and ResNet based Recognition Approach. In: 2020 25th International Conference on Pattern Recognition (ICPR). IEEE, pp 7737–7742

41. Liu W, Anguelov D, Erhan D, et al (2016) SSD: Single Shot MultiBox Detector. In: European Conference on Computer Vision. Springer, pp 21–37

42. Lin T-Y, Goyal P, Girshick R, et al (2017) Focal Loss for Dense Object Detection. In: Proceedings of the IEEE International Conference on Computer Vision. pp 2980–2988

43. Kirillov A, He K, Girshick R, Dollár P (2017) A unified architecture for instance and semantic segmentation. http://presentations.cocodataset.org/COCO17-Stuff-FAIR.pdf

44. Redmon J YOLO: Real-Time Object Detection. https://pjreddie.com/darknet/yolo/

45. Lin T-Y, Maire M, Belongie S, et al (2014) Microsoft COCO: Common Objects in Context. In: European Conference on Computer Vision. Springer, pp 740–755

46. AlexeyAB GUI for marking bounded boxes of objects in images for training neural network Yolo v3 and v2. https://github.com/AlexeyAB/Yolo_mark

47. Neubeck A, Van Gool L (2006) Efficient Non-Maximum Suppression. In: Proceedings of the 18th International Conference on Pattern Recognition (ICPR'06). IEEE, pp 850–855

48. Rezatofighi H, Tsoi N, Gwak J, et al (2019) Generalized intersection over union: A metric and a loss for bounding box regression. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp 658–666

49. Zhang Z, Ding S, Sun Y (2020) A support vector regression model hybridized with

chaotic krill herd algorithm and empirical mode decomposition for regression task. Neurocomputing 410:185–201

50. Hong W-C, Fan G-F (2019) Hybrid empirical mode decomposition with support vector regression model for short term load forecasting. Energies 12:1093