A Hybrid Approach for Table Detection in Document Images*

Sunil Kumar Vengalil[1], Kevin Xavier[2], Konda Amith Sai[1], Sree Harsha[1], Ganesh Barma[1], and Neelam Sinha[1]

[1] International Institute of Information Technology, Bangalore, India
[2] Razorthink Technologies, Bangalore, India
sunilkumar.vengalil@iiitb.org
kevin.xavier@razorthink.com

Abstract. One of the crucial requirements of document analysis is accurate detection of tables and data present in it. It is easier to parse and extract information from tables as it is more structured compared to paragraphs. However, detecting tables in document images is a challenging task due to different layouts. Since image processing and machine learning approaches like Logistic Regression and SVM did not result in good detection, several deep learning architectures along with labelled datasets were experimented resulting in better segmentation performance. But, the deep learning models suffer from a major drawback of lack of explainability of prediction. In this work, we use features computed on word segments in order to segment tables in document images using Logistic Regression and SVM with linear and RBF kernels. All these models are more explainable compared to Deep learning models. We further augment a recent deep learning model, TableNet, with the features as additional inputs and report the comparison studies on a publicly available dataset Marmot. Performance of all models were compared by running segmentation on 100 test images in Marmot dataset. Among the machine learning models that were trained using word alignment features, SVM with RBF kernel gave the best word-level Dice score of 83.1% with a precision of 91.4% and recall of 76.1%. The augmented deep learning model gave a pixel-wise Dice score of 95.2% with a precision of 99.98% and recall of 91.44%. The same model without augmentation gave a pixel-wise Dice score of 94.8% with a precision of 99.99% and recall of 90.7%. The proposed hybrid model results in a reduction of 0.7% of false positives.

Keywords: Document Image Analysis · Table Detection · TableNet

1 Introduction

Detection of structural elements like tables, paragraphs and headers in images of digital documents is one of the core problems in automatic document processing. The problem of table detection is almost as complex as object detection

* Supported by Mphasis CSR grant

in natural images and hence traditional image processing and machine learning approaches do not perform well. Using image processing approaches, coming up with a perfect algorithm that works in all possible scenarios is infeasible as it requires tuning a number of parameters and thresholds [1]. Further these parameters highly depend on the type and quality of documents. The challenge in using Machine Learning(ML) approaches is coming up with the right set of features. Evidence from object detection [2] [3], tells us that deep learning approaches should perform better than algorithmic and ML approaches. However, the challenge here is the non-availability of a huge number of annotated samples for table detection similar to ImageNet [7] for object detection. Most of the publicly available datasets [5] [6] for table detection tasks are either 1) small in size typically of the order of a few thousands or even less or 2) restricted to certain types of documents[6]. However, one of the recent dataset, TableBank[13], has 417K images of word and latex documents taken from the internet.

One of the feature based approaches [1] uses morphological operations in order to segment structures like text blocks and lines. Another popular image processing based approach uses the fact that spacing between columns will be larger than spacing between words [8]. Like any other image processing based algorithm, these methods also suffered from the drawback that the algorithm is difficult to generalize across document types. Due to non-generalizability of algorithmic approaches, Machine Learning based techniques were introduced. T Kasar et al. [9] utilizes SVM classifier with features like horizontal and vertical lines to segment tables in document images. Embley et al. [10] provides an excellent survey of various table detection approaches using image processing and Machine Learning. Anh et al. uses Graph Neural Networks [11] to segment tables in images of invoice documents. Azka Gilani et al. uses faster-RCNN [2] for segmenting tables in document images after separating out text and non-text regions[12].

Even though Deep Learning based approaches give better performance, they require large annotated data and suffer from lack of explainability.

To circumvent the above issues, in this article we propose a hybrid approach where we augment a recent deep learning model, TableNet [4], by adding additional input channels with word level features. In particular, our major contributions in this study are:

1. We introduce a set of features based on alignment and spacing between words in order to detect tabular structures in document images.
2. We further illustrate the significance of these features using multiple ML classifiers Logistic Regression and SVM.
3. We introduce a hybrid model where a recent deep learning model, TableNet [4], is augmented by providing an additional input image with words segmented.

2 Proposed Method

We perform table segmentation using the following three approaches:

1. Classify each word as table word Vs non-table word using binary classifiers, Logistic regression and SVM, with word alignment features mentioned in section 2.2 as the input to the classifier.
2. Using a deep learning model TableNet[4] to segment the table boundaries.
3. A hybrid approach by augmenting the TableNet with an additional input channel. A binary image with all words masked, as shown in Figure 5, is fed at this input channel.

In the case of SVM and logistic regression we use 12 word level features derived from the word patch segmentation output. SVM and logistic regression directly predicts whether each word belongs to a table or not. Even though the word level binary classifier does not give exact table boundaries, it is beneficial in two ways- 1) It gives a quantitative measure of how significant the features are for table detection. 2) It can be used to refine the prediction from other two approaches. We leave this work as a future work and in this work we just report the results of word level binary classifier using the proposed features. For the DL based approach, the images are directly provided as input. For the proposed hybrid approach, we use binary images with segmented words, see Figure 5 as input. The DL and hybrid approach generate pixel level segmentation for table regions from which word level classification is computed for comparing the approaches. The overall pipeline of the proposed method is illustrated in Figure 1.

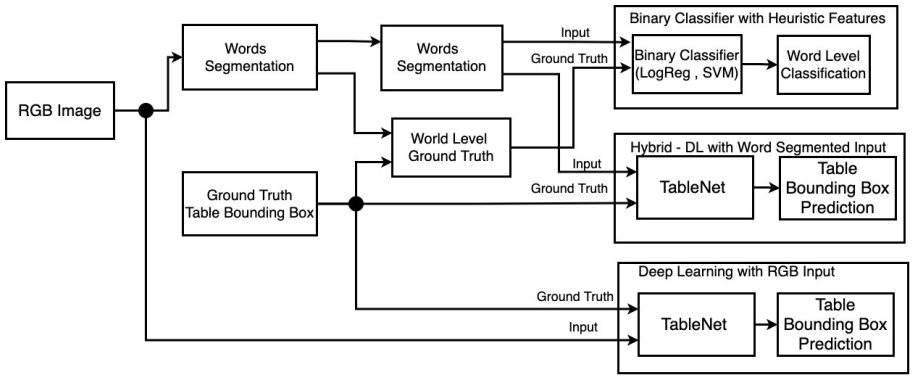

Fig. 1. Pipeline of the proposed approaches

2.1 Dataset

We performed experiments on the marmot dataset. The data set consists of images of 2000 pages extracted from research article documents of which 1000 are for Chinese and 1000 are for English language. For each language, 500 documents are with, possibly multiple, tables and 500 documents are without any tables. For

generating the training set, we used only 500 English document images which had at least one table present in it. A sample image from Marmot dataset is shown in Figure 2a.

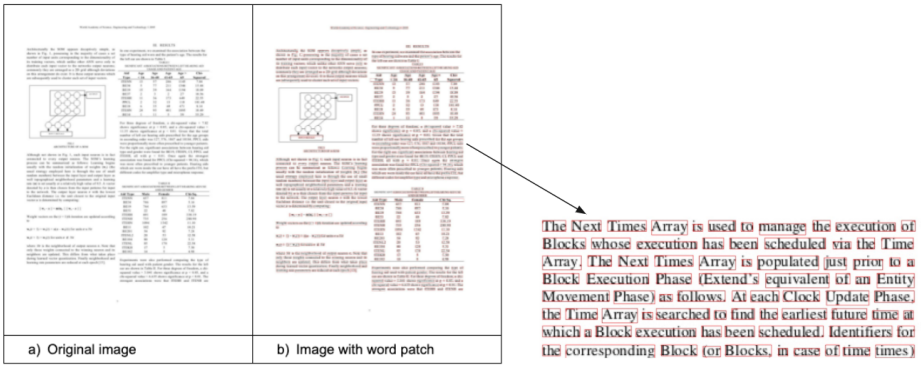

Fig. 2. Sample image from Marmot dataset and corresponding output of word segmentation.

2.2 Preprocessing and Feature Detection

The preprocessing and feature detection pipeline is shown in Figure 3. The document image is first converted to binary image by Ostu's thresholding. After this preprocessing stage, text lines in the image are segmented using the spacing between each line. The text lines are further segmented into words, again using the character spacing, and the top, bottom, left and right coordinates of each word are obtained.

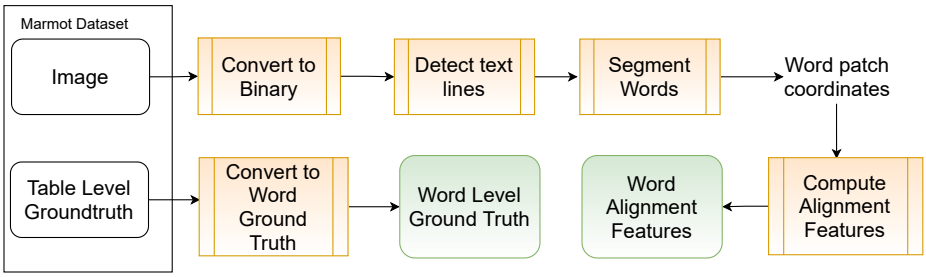

Fig. 3. Feature detection pipeline

Segmenting text lines and word patches Text and non-text regions in the document were separated out first using a DL model that was trained using synthetically generated images and labels. Each text block in the document was further broken down into individual text lines. This is done by counting the number of back-ground pixels along each row. The count is normalized with respect to the width of the text block and if the normalized value is less than a threshold the entire row is labelled as blank row. All the consecutive blank rows are merged into a single region which corresponds to the spacing between two rows. Rectangular regions between line spacing is taken as the bounding box coordinates of text lines.

Once each text line is segmented, a similar algorithm is used to segment the word patches in each text line. This time the number of back-ground pixels along each column of a text line is found and normalized with respect to the height of the text line. If the normalized value is less than a threshold the column is marked as blank column and all consecutive blank columns are merged to get the spacing between two word patches.

Since the above mentioned algorithm is based on thresholds which can vary from document to document, we obtained additional evidences for word patches using two more third party tools:

1. Tesseract - An Open Source OCR library from Google that also gives bounding box coordinates of word patches
2. Google Cloud Vision - A cloud based API that parses a document image and provides information like word patch bounding box and text inside each word patch image

We combined the word patch bounding box information from all three sources (our algorithm, Tesseract and Google Cloud Vision) using a max voting scheme and generated the final word patch output.

The entire document is represented by a collection of words W, where each $w \in W$ is a 4-tuple (t, l, b, r) representing the top, left, bottom and right coordinates of the word patch image. See Figure 2b for a sample image with boundaries marked.

Table 1. Sample features for left alignment. Similar features are added for right and center alignment also

Feature Name	**Description**
top_left_aligned	Number of words left aligned to the word lying above the word
continous_top_left_aligned	Number of words left aligned to the word lying immediately above the word contiguously
bottom_left_aligned	Number of words left aligned to the word lying below the word
continous_bottom_left_aligned	Number of words left aligned to the word lying immediately below the word contiguously

Word Alignment Features Using the word patch coordinates, we computed 12 derived features that capture left and right alignment between words and spacing between words. Features computed based on left alignment are given in Table 1. Similar features are computed for right and center alignment also. Each word is represented by a 12-dimensional feature vector.

Finally, table ground truth coordinates in the original Marmot dataset [5] were used to give a binary label, table Vs non-table, to each word in the document. Table 2 summarizes the number of table and non-table words used in training and validation sets.

Table 2. Dataset used for training ML models. Features detected on word patches extracted from Marmot dataset is used for training and validation

	Table Words	**Non-table Words**	**Total**
Training	39478	39478	78956
Validation	16919	159625	176544
Total	56397	199103	255500

2.3 Classification of Words

Figure 1 shows the architecture of the overall training and prediction system.

Following three approaches were followed to train a model for classifying each word as table or non-table word:

1. Binary Machine Learning classifiers trained using 12-dimensional feature vectors corresponding to each word.
2. Using Deep learning model TableNet [4] to find the bounding box of each table. Each word lying inside the bounding box predicted by TableNet is classified as table words and all other words are marked as non-table words.
3. A hybrid approach where the TableNet model is augmented with an additional input channel carrying information about boundaries of word segments.

2.4 Machine Learning Approach

Figure 4 shows the block diagram of ML training pipeline. The word level features mentioned in table 1 and the label, table Vs non-table, is used to train binary classifiers Logistic Regression and SVM.

2.5 Deep Learning Approach

We trained a TableNet [4] model using raw images normalized and resized to 256×256. TableNet is an end-to-end deep learning model that leverages the

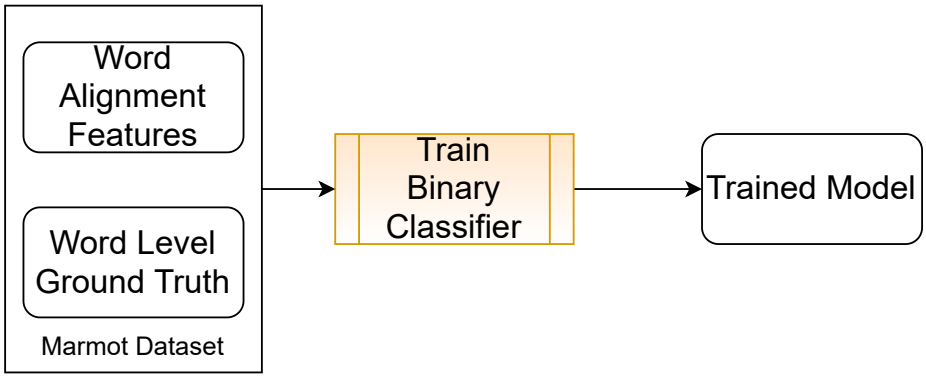

Fig. 4. Block diagram of the machine learning training pipeline. We used Logistic Regression and SVM.

inherent interdependencies between table discovery and table structure identification. This model uses an initialized base network with pre-trained VGG19 functionality. Two decoder branches follow for 1) Table area segmentation and 2) Column segmentation within table area. Next, rule-based row extraction is used to extract data into individual table cells. The model is trained using a multi-task loss function which is a combination of table loss and column loss. The model takes a single input image and produces two output images labeled with different semantics for tables and columns. The models share the VGG19 coding layer for the table and column detectors, while the decoders for the two tasks are separate. The shared common level is trained repeatedly from the gradient received from the table and column detectors while the decoder is trained independently. Semantic information about the underlying data type is then used to further improve model performance. Using VGG19 as the core network, pre-trained over one million images from the ImageNet database. ImageNet datasets enable the exploitation of prior knowledge in the form of low-level features learned through training on ImageNet.

2.6 Hybrid Approach

We followed the same approach as in the DL approach, but instead of training the image using original images we used images with word patches masked shown in Figure 5.

3 Results and Discussions

In this section we discuss the experimental results for table segmentation on Marmot dataset using the three approaches discussed in section 2.3. Table 3 provides a comparison of precision, recall and F1 score at word level classification using all three approaches.

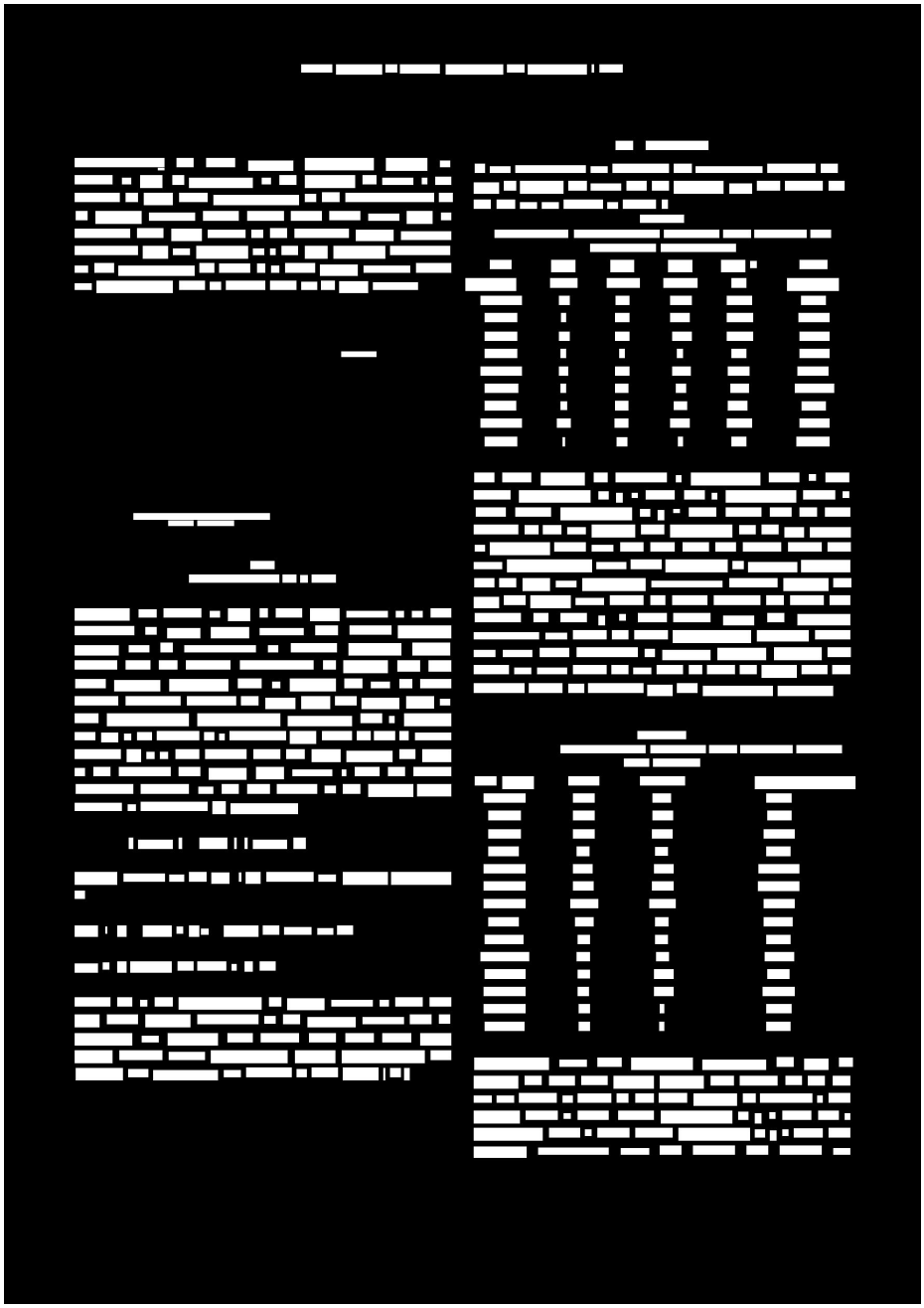

Fig. 5. Sample image from Marmot dataset with segmented word patches used for hybrid approach

Among the ML classifiers SVM with RBF kernel gave the best word-wise F1 score of 83.1% on test images. Linear Regression resulted in lesser performance since they do not add any non-linear transformation which might be important for table detection task. This should also explain why SVM with linear kernel gave lesser performance compared SVM with RBF kernel.

Table segmentation using TableNet gave a precision, recall and pixel-wise F1 score of 99.9%, 90.08% and 94.78% respectively.

The original TableNet model was trained with all three RGB channels of the image. However, document images are binary in nature and further the details of character shapes are immaterial for table task. We anticipate better results in binary images as the number of possible states for each pixel is reduced to two, whether the pixel is part of word patch or not. This prompted us to modify the input image with a binary mask image with word patches segmented resulting in a hybrid approach. Using this approach the pixel-wise recall increased by 0.7% which resulted in an increased F1 score of 95.2%.

Figure 6 shows the sample prediction results using SVM. The words predicted as table words are marked in green and non-table words are marked in blue. Figure 7 shows the table mask predicted by the hybrid deep learning model for a sample test image. Figure 8 shows the table masks predicted by the TableNet model. It is evident from Figure 7 and Figure 8 that the boundaries of the predicted table mask using hybrid model is more sharper and rectangular as compared to the prediction of TableNet model.

It is observed that the proposed hybrid approach outperformed the vanilla TableNet model. The assumption of hybrid approach is that the gray values of pixels in a document image are distributed bimodal. This hybrid approach can further be improved with the addition of additional word level features. Apart from this CRF can also be experimented and is expected to outperform SVM based classifiers as CRF models the correlation between labels of neighbouring words. Though we have used marmot data, further research can be done with TableBank [13] dataset which contains much larger number of document images with tables.

Table 3. Comparison of different binary classifiers' performance on classifying each word as table vs non-table

	Precision	**Recall**	**F1 Score**
Logistic Regression	85.2	67.8	75..5
SVM Linear	84.2	66.9	74.6
SVM RBF	91.4	76.2	83.1
Hybrid	99.98	94.33	97.07

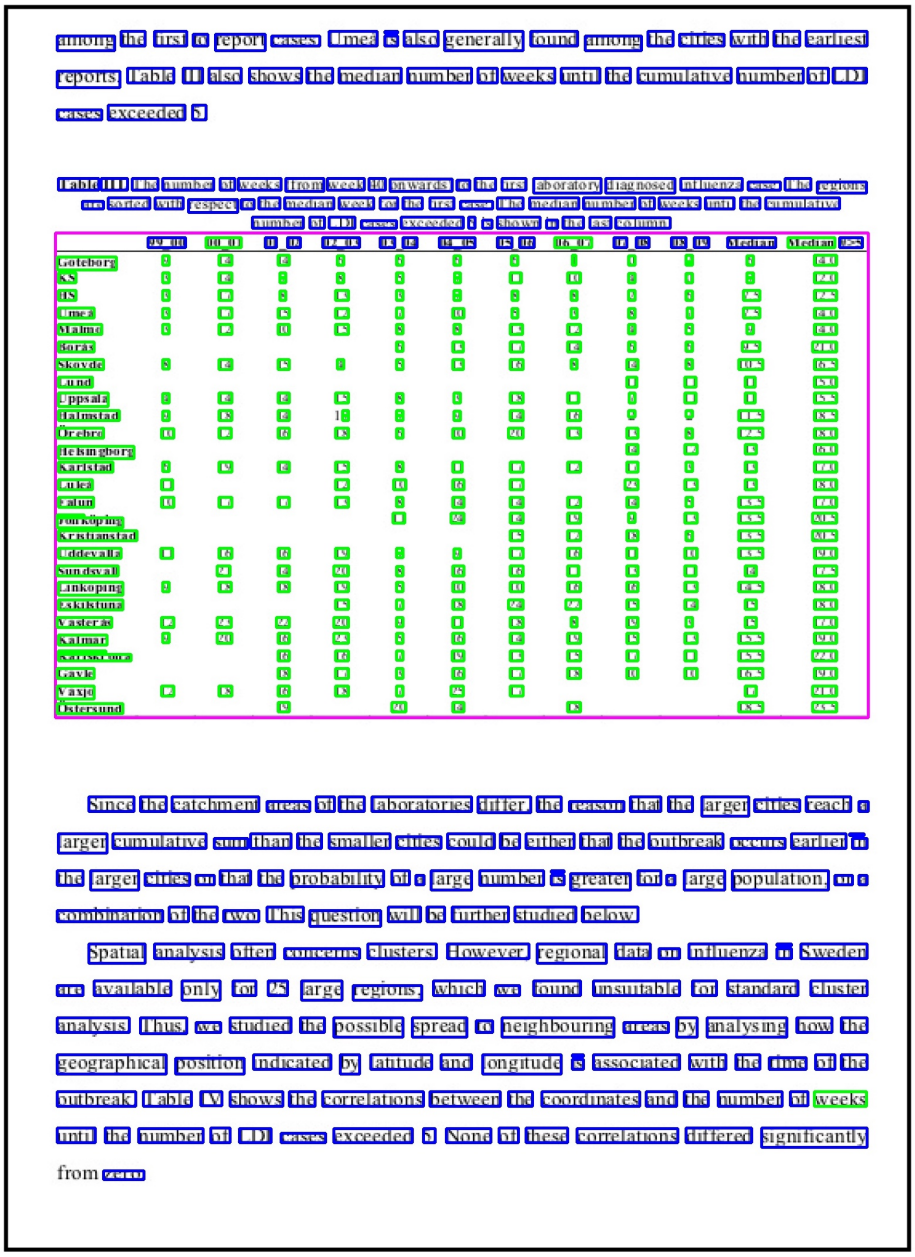

Fig. 6. Prediction results using SVM with RBF kernel. Table words are marked in green and non-table words are marked in blue.

4 Conclusion

In this study we propose new word level features that can be used for detecting tabular structures in document images. We illustrate the significance of the proposed features by training binary classifier, Logistic Regression and SVM, to predict whether each word belongs to a table or not. We get an F1 score of 83.1% for SVM with RBF kernel on Marmot dataset. We further propose a mechanism to augment a popular deep learning model, TableNet, by providing word segmentation information at the input. The proposed augmented model outperforms the original TableNet model with an increase in recall by 0.7%. This resulted in an increase in F1-score from 94.8% to 95.2%. The hybrid model can further be improved by adding more word level features.

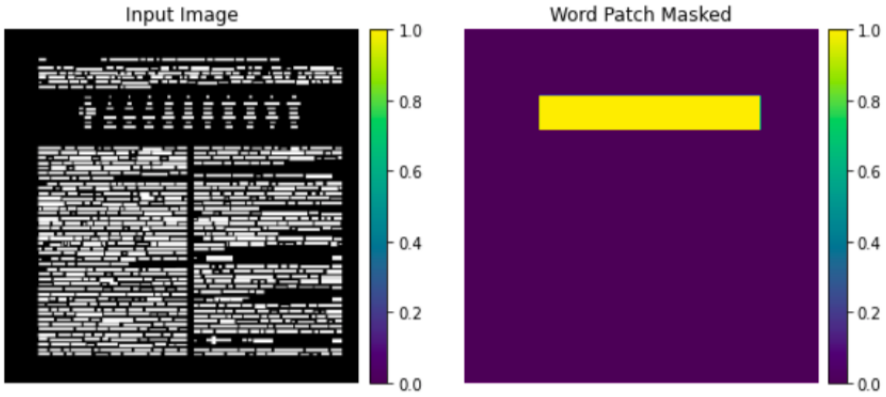

Fig. 7. Pixel level prediction of table mask using hybrid model.

Acknowledgement This work is supported by Mphasis CSR grant.

References

1. Tran, Dieu Ni, et al. "Table detection from document image using vertical arrangement of text blocks. "International Journal of Contents 11.4 (2015): 77-85.
2. Ren, Shaoqing, et al. "Faster r-cnn: Towards real-time object detection with region proposal networks." Advances in neural information processing systems 28 (2015): 91-99.
3. Redmon, Joseph, et al. "You only look once: Unified, real-time object detection." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.
4. Paliwal, Shubham Singh, et al. "Tablenet: Deep learning model for end-to-end table detection and tabular data extraction from scanned document images." 2019

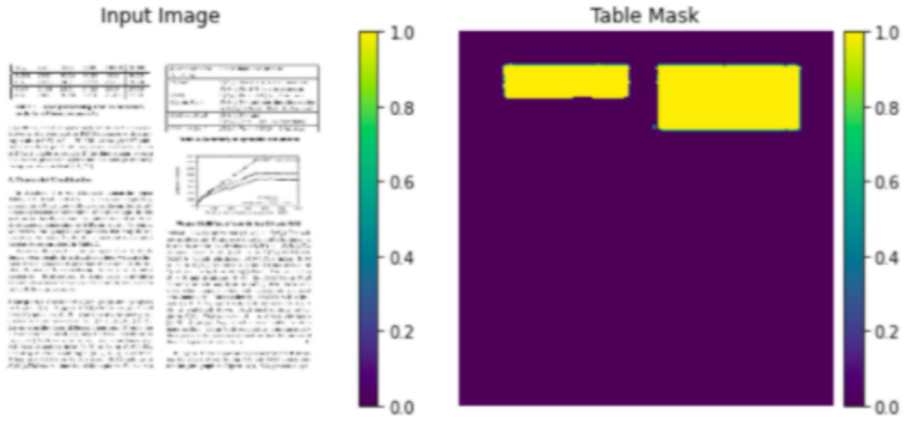

Fig. 8. Pixel level prediction of table mask using TableNet model.

International Conference on Document Analysis and Recognition (ICDAR). IEEE, 2019.
5. Fang, Jing, et al. "Dataset, ground-truth and performance metrics for table detection evaluation." 2012 10th IAPR International Workshop on Document Analysis Systems. IEEE, 2012.
6. Zhong, Xu, Jianbin Tang, and Antonio Jimeno Yepes. "Publaynet: largest dataset ever for document layout analysis." 2019 International Conference on Document Analysis and Recognition (ICDAR). IEEE, 2019.
7. Deng, Jia, et al. "Imagenet: A large-scale hierarchical image database." 2009 IEEE conference on computer vision and pattern recognition. Ieee, 2009.
8. Mandal, Sekhar, et al. "A simple and effective table detection system from document images." International Journal of Document Analysis and Recognition (IJDAR) 8.2 (2006): 172-182.
9. Kasar, Thotreingam, et al. "Learning to detect tables in scanned document images using line information." 2013 12th International Conference on Document Analysis and Recognition. IEEE, 2013.
10. Embley, David W., et al. "Table-processing paradigms: a research survey." International Journal of Document Analysis and Recognition (IJDAR) 8.2 (2006): 66-86.
11. Anh, Tran Tuan, Na In-Seop, and Kim Soo-Hyung. "A hybrid method for table detection from document image." 2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR). IEEE, 2015.
12. Gilani, Azka, et al. "Table detection using deep learning." 2017 14th IAPR international conference on document analysis and recognition (ICDAR). Vol. 1. IEEE, 2017.
13. Li, Minghao, et al. "Tablebank: A benchmark dataset for table detection and recognition." arXiv preprint arXiv:1903.01949 (2019).