Supplementary Material
TEXTRON: Weakly Supervised Multilingual Text Detection through Data Programming

1. Why TEXTRON

We showcase TEXTRON as a framework to carry out multilingual text detection through Data Programming, where users can plug various text detection methods into a weak supervision-based learning framework. Following are a few salient features of weakly supervised TEXTRON that benefit the use-case of text detection.

- **Less Dependency on Labeled Data** As described in our methodology, we do not employ the usage of labeled data for carrying out text detection. We use unsupervised quantitative assessment of Labeling Functions (LFs) for effective training of parameters and qualitative inspection of unlabeled validation sets for determining best hyperparameters which is explained in Section 6.

- **Faster than conventional methods** Conventionally to perform text detection we need to train a DL model that is hungry for labeled data. And in scenarios where labeled data is limited or unavailable, we need manual intervention to annotate data. This is far more time-consuming and requires a lot of effort and expertise than our Data programming-based paradigm that TEXTRON offers. Inference of TEXTRON can help in faster generation of labeled data or directly assist in the detection of text of multiple scripts.

- **Enforcing the power of Weak Supervision** Since documents by themselves come in a wide range of layouts, fonts, styles, languages, and modalities, it is acceptable to have a little bit of qualitative noise in the labels for the data on which new models are intended to be trained. TEXTRON is an accurate example of combining several noisy rule-based LFs to generate high-quality data faster, by reducing the challenges associated with the availability of manually labeled data. Eventually, it is the interplay of these noisy weak labels generated by different text detection methods that contribute to a more accurate and generic text detection pipeline, hence having a weak supervision-based approach benefits the use-case of TEXTRON.

- **More control to the data programmer** Unlike a strict (1 or 0) pixel-based voting mechanism (like our MBV baselines) or labeled data-dependent traditional ML-based ensemble approaches, TEXTRON offers to induce strengths and weaknesses of LFs in the aggregation methods. The data programmer who designs the LFs is well aware of what kind of cases it labels in the documents and is also aware of when and how it performs efficiently or fails to perform well. LF designers control the region of influence of LFs by making it trigger for specific scenarios and abstaining from the rest. This also makes the system more flexible by designing countless LFs and using their strengths to detect text. The control of the data programmer is also exercised by setting up quality guides by mentioning to the TEXTRON aggregator the fraction of times the LF will trigger and label pixels appropriately.

- **Curation of datasets for low-resource scripts** Estimating a good set of hyperparameters for LFs followed by the inference of TEXTRON$_{8LF}$ model can help in creating text detection datasets for low-resource languages which can be used for further downstream training tasks.

2. Datasets Statistics

This section gives tabular information on various training sets, validation sets, and test sets used. Tables 1, 2 and 3 give the concerned information respectively. There are no common samples between training, validation, and test sets. We use training and validation set images in unlabeled form. We have used the labels on the test sets only to report our performance.

3. Complementary LFs Design

Labeling Functions (LFs) are conventionally weak supervision based functions that generate noisy labels. In our context, LFs are used to mark regions of document image as TEXT or NON-TEXT regions. Similar to the five LFs described in Figure 1, we have also defined and used the corresponding five complementary LFs which basically are associated with the NON-TEXT class, the outputs of which are presented in Figure 2. As stated, the complementary

Name	Langauge Modality	Image Composition	Size	Use of Labels
T1	English Printed	Randomly selected from Docbank [4]	22454	No
T2	English Printed	244 of Docbank [4], 100 of Funsd [3], 100 of CTDAR [2]	444	No
T3	Sanskrit Printed	Sanskrit Book [1]	215	No
T4	Indian Handwritten	Kannada and Telugu from PhDIndic [6]	41	No
T5	Mixed	Composed of T2, T3 and T4 defined above	700	No

Table 1. Overview of different training datasets

Name	Langauge Modality	Image Composition	Size	Use of Labels
V1	English Printed	45 of Docbank [4], 40 of Funsd [3] and 40 of CTDAR [2]	125	No
V2	Malayalam Printed	Malayalam Textbooks [5]	10	No
V3	Sanskrit Printed	Sanskrit Book [1]	85	No
V4	Indian Handwritten	Kannada and Telugu from PhDIndic [6]	90	No

Table 2. Overview of different validation datasets

LFs use exactly the same algorithm proposed in the fundamental LFs. However, the only difference is that they help to label the concerned set of pixels with NON-TEXT class. All ten LFs are triggered on the black region pixels, while they abstain from labeling the white region pixels.

4. Unsupervised Assessment

While choosing the best-performing TEXTRON LF set, we performed extensive experiments using different combinations of CV-based LFs as well as the LFs derived from pre-trained DL models on different sizes of non-overlapping subsets of training set T1 (Refer Table 1). We analyzed the unsupervised performance measures of the ten LFs designed as mentioned in Section 3. Figure 3 shows coverage, overlaps, and conflicts for each LF as explained in the review copy. Ideally, a good LF will have significant coverage, high overlap, and fewer conflicts. While experimenting we saw that the Image Edge-Based LF had an extremely low coverage for TEXT pixels. Further, the coverage and overlap for this LF were approximately the same for almost all images. In other words, almost all the pixels that were labeled as TEXT by this LF were labeled by some other LF too. This also makes sense as the black pixels in Figure 1f are triggered by many other LFs. Also, the corresponding complementary LF had relatively high conflicts. This is also qualitatively observed in Figure 2f, as a lot of these black triggered pixels also point to regions of another class. So we dropped this LF and its corresponding complementary LF from the best-performing set and analyzed further for the remaining eight LFs. These eight LFs included four fundamental LFs namely pre-trained DBNet, Tesseract-based LF, Contour-based LF, and Canny Filter-based LF along with their complementary LFs. Figure 4 presents the three unsupervised performance measures for these LFs on a random image of the Docbank dataset. The fundamental LFs that labeled the text pixels had less (yet significant) coverage which was a favorable scenario as the amount of TEXT pixels in any generic document page is much less as compared to the NON-TEXT pixels. Similarly, all our chosen complementary LFs have a high percentage of coverage and overlap, both of which are above 80% for almost every image. We have performed a similar kind of analysis with various unlabeled images from Training sets T1 and parts of T2, T3, and T4 as described in Table 1 for handwritten and printed Indian language text training sets as well. All kinds of analysis and visual inspection of outputs have convinced us to use TEXTRON$_{8LF}$.

5. Subsequent Training

With this newly defined TEXTRON$_{8LF}$ configuration, we train our graphical model (parameterized by θ) for the training objective on train set T5 mentioned in Table 1. We try to optimize the training objective for 50 epochs by keeping a constant learning rate of 0.01 for each *unlabeled image* in the train set. Once the graphical model is trained for these 700 unlabeled images, we end up freezing the parameters and thus θ remains untouched for further experimentation.

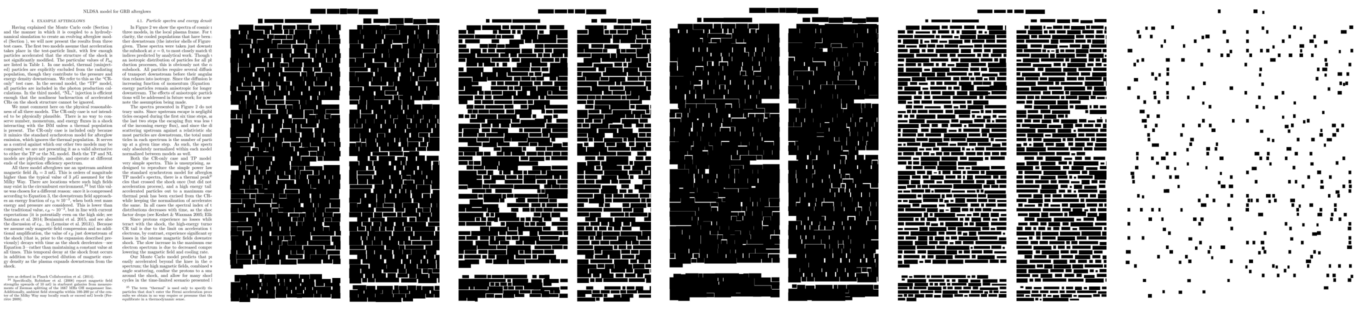
6. Role of Validation Datasets

We further experimented on our various validation sets described in Table 2. This included tweaking the LF quality guides and their hyperparameters for an already trained TEXTRON$_{8LF}$ model. The reason we need validation sets is to tune the LF hyperparameters which include the following:

- **Width Shrinkage:** The width shrinkage takes a value between 0 and 1 indicating the factor to which the widths of all the bounding boxes need to be shrunk during the LF application stage. Ideally, for documents

Name	Langauge Modality	Image Composition	Size	Use of Labels
D1	English Printed	Docbank [4] Test Sample	100	*Yes
D2	Malayalam Printed	Malayalam Books [1]	200	*Yes
D3	Tamil Printed	Tamil Books [1]	225	*Yes
D4	Gujarati Printed	Gujarati Books [1]	323	*Yes
D5	Devanagari Handwritten	Devanagari from PhDIndic [6]	220	*Yes

Table 3. Overview of different test sets where * indicates usage of labels only for evaluation purpose



(a) Input Image (b) DBNet based LF (c) Contour based LF (d) Canny Filter LF (e) Tesseract based LF (f) Image Edges LF

Figure 1. Binary Image Outputs for different Labeling Functions

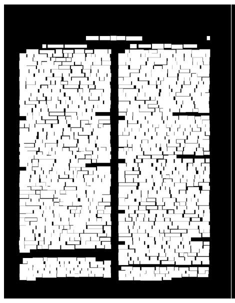
in which words in the same line have less amount of space between them, we need more width shrinkage to preserve word level demarcation in LF outputs.

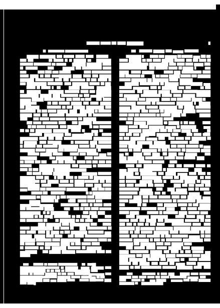
- **Height Shrinkage:** The height shrinkage takes a value between 0 and 1 indicating the factor to which the heights of all the bounding boxes need to be shrunk during the LF application stage. Ideally, for documents in which words in consecutive lines are closely spaced, we need more height shrinkage to preserve line-level demarcation of words in LF outputs. If this shrinkage is insufficient, then TEXTRON might end up giving a single box engulfing words of different lines that are closely spaced as seen in Figure 5
- **Contour Thickness:** This is a hyperparameter used by 2 LFs namely the contour-based LF to mark the TEXT region and complementary contour-based LF to mark the NON-TEXT region. The thickness (usually set to 4) determines how thick contours will engulf the text pixels joining the pixels having the same intensity. Lesser thickness will lead to fragmented boxes or character-level boxes. Setting unreasonably greater thickness will lead to a loss of word-level demarcation as it might engulf more closely spaced words in one box. The optimal thickness will generate contours thick enough to engulf the entire word together.
- **Edge Thickness:** This is a hyperparameter similar in significance to *contour thickness* as described above. The only difference is that it is used by 2 other LFs namely the canny filter-based LF to mark the TEXT region and complementary canny filter-based LF to mark the NON-TEXT region. The thickness is usually set either equal to *contour thickness* or one less than it. Because the canny filter already detects edges along the words before contour-based post-processing, it does add to the amount of thickness of edges detected along the word pixels beforehand. Hence, it is advisable to keep this parameter lesser than or equal to the *contour thickness*.

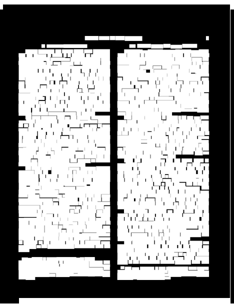
6.1. Why do we need hyperparameters for LFs

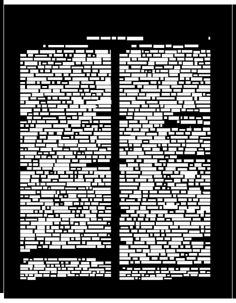
All the above-mentioned hyperparameters are responsible for LF outputs and thereby greatly influence the quality of TEXTRON output. Even though we have trained the model TEXTRON$_{8LF}$ with parameters θ learned for every LF, it is still not sufficient to capture the variety of writing styles in different forms of documents. This is because the TEXTRON$_{8LF}$ model works on aggregating the outputs of different LFs with parameters θ but in order to capture word-level demarcation of diverse document images, some changes are needed in the LF application stage itself. These changes are controlled by the LF hyperparameters that influence LF outputs. These altered LF outputs (binary maps) are in turn provided to TEXTRON$_{8LF}$ for the aggregation stage. For example, handwritten text may not be able to

(a) Input Document Image


(b) Complementary DBNet based LF


(c) Complementary Contour-based LF


(d) Complementary Canny Filter LF


(e) Complementary Tesseract based LF

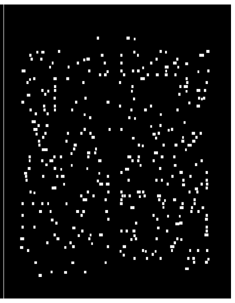

(f) Complementary Image Edges LF

Figure 2. Binary Image Outputs for Complementary Labeling Functions

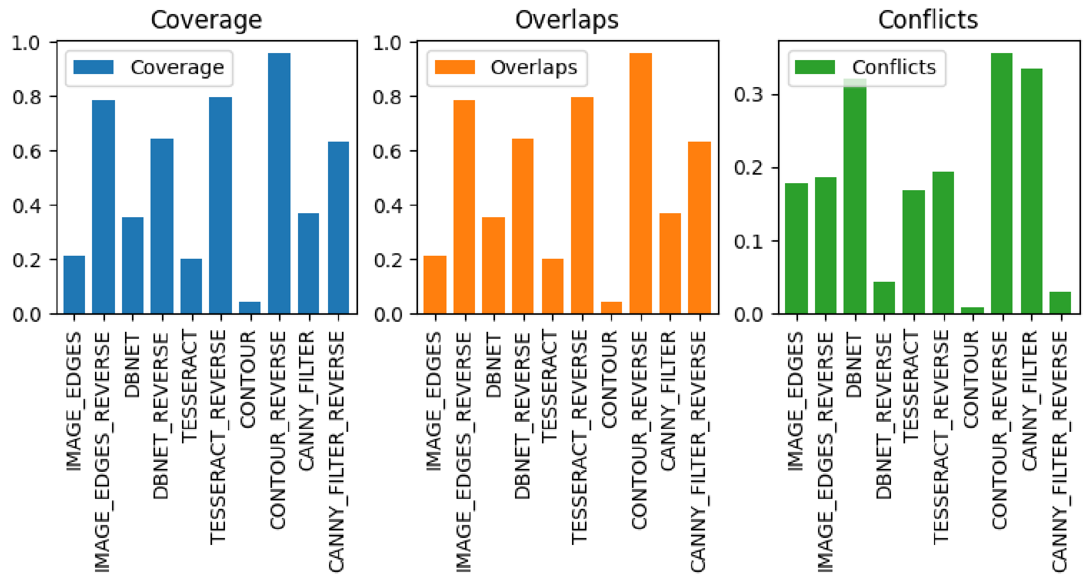


Figure 3. Unsupervised quantitative assessment on the ten LFs included in the experimentation displaying coverage, overlaps, and conflicts for each LF

maintain lines of words with uniform spacing and can get inclined so for detecting these words, we might need more height shrinkage. On the other hand, printed words have uniform line spacing and can do with lesser height shrinkage. Since a plethora of documents, scripts, fonts, and modalities exist we need some domain knowledge to leverage multilingual text detection. This is precisely the role of validation set which can give you a set of effective hyperparameters for certain languages or kinds of documents with particular layouts and styles of writing. This also minimizes the need to have multiple physical models for different languages.

6.2. How to perform Hyperparameter tuning with unlabeled Validation Sets

Our experimentation involved tuning hyperparameters. Width shrinkage was done by 10% (shrinkage factor was set as 0.9) for all the experiments and was considered opti-

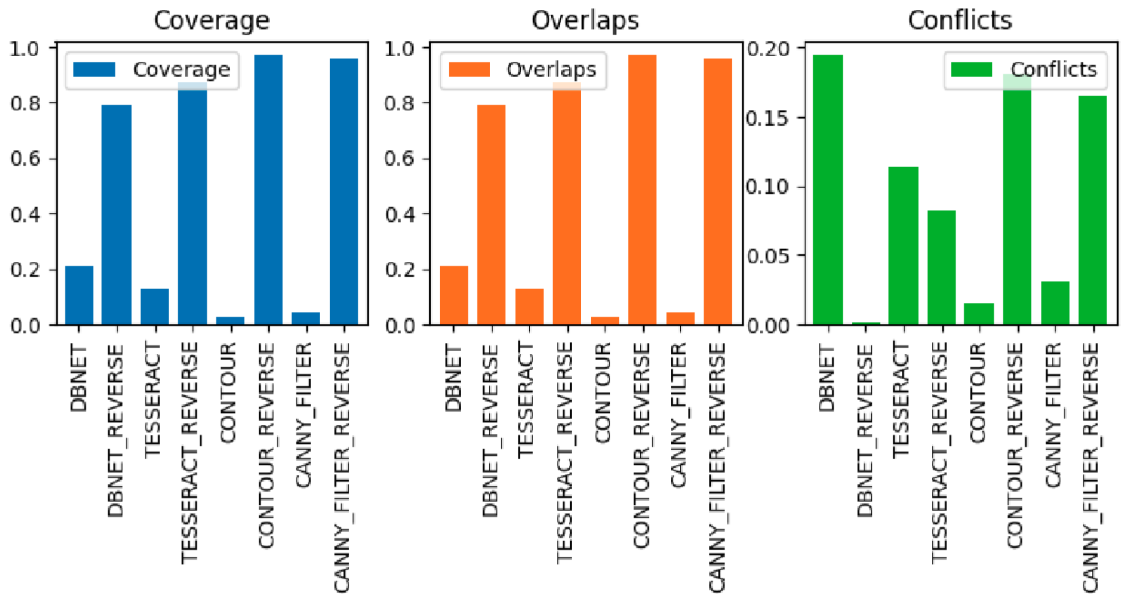
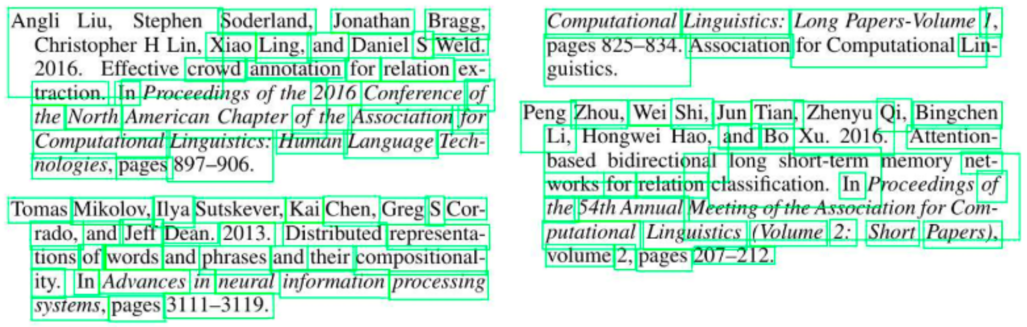


Figure 4. Unsupervised quantitative assessment on the eight LFs included in the experimentation displaying coverage, overlaps, and conflicts for each LF


Angli Liu, Stephen Soderland, Jonathan Bragg, Christopher H Lin, Xiao Ling, and Daniel S Weld. 2016. Effective crowd annotation for relation extraction. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 897–906.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Computational Linguistics: Long Papers-Volume 1, pages 825–834. Association for Computational Linguistics.

Peng Zhou, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao, and Bo Xu. 2016. Attention-based bidirectional long short-term memory networks for relation classification. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 207–212.

Figure 5. TEXTRON output for Contour Thickness 5 for Docbank Image

mal for all languages in Validation sets V1 to V4 (described in Table 2). For printed text, the height shrinkage factor was set to 0.8 for all the page outputs of TEXTRON$_{8LF}$, and the LFs were able to keep boxes of distinct lines separate as required. However, for hand handwritten text validation set V4, we observed a similar problem encountered as shown in Figure 5. Hence we needed to tune this factor and shrink the heights of bounding boxes more for getting good validation results for handwritten text. Height shrinkage by 30% (setting the value to 0.7) was suitable for handwritten text. Note that, as this validation set was unlabeled, we relied completely on visual inspection of outputs for determining the quality of text detection. As this is a weakly supervised process, some qualitative noise was acceptable during the inspection of final outputs. This not only helps to determine language-specific or font-specific parameters for a new or similarly displayed text but also reduces the need for labeled data. Hence all four of our validation sets were considered candidates to determine effective thickness and shrinkage factors for similar modalities and scripts of text. Further, we also experimented with thickness factors where *contour thickness varied from 2 to 6* and *edge thickness* varied from 1 to 5 on validation sets V1 to V4. After visual inspection of outputs as shown in Figure 6, we determined the concerned thicknesses for different types of documents. After tuning, we chose the hyperparameters of Validation Set V1 to test on Docbank test set D1 to report our results. As both V1 and D1 contained the same type of printed English documents, we believe the hyperparameters tuned for V1 will help enhance the performance of Test set D1. Sim-

Dataset	**Height Shrunk By**	**Contour Thickness**	**Edge Thickness**
Docbank	20%	4	2
Malayalam	20%	5	4
Tamil	20%	4	3
Gujarati	20%	4	3
Devanagari	30%	5	5

Table 4. Hyperparameters of TEXTRON$_{8LF}$ for different datasets

ilarly, printed Indian language text was considered one of the document types for which we used Validation sets V2 and V3 to determine the best hyperparameters. As a result, the validation set V2 (Printed Malayalam) was considered to be a candidate set of hyper-parameter tuning for Test Set D2 (all test sets are described in Table 3)). Also, the results of test sets D3 (Printed Tamil) and D4 (Printed Gujarati) were evaluated with hyperparameters tuned for Validation set V3. Finally, for handwritten text, we used Validation set V4 (Telugu and Kannada handwritten) to determine the best shrinkage and thickness parameters. These parameters were used for the inference stage of TEXTRON$_{8LF}$ on Test set D5 composed of Devanagari handwritten text. Once the model was able to perform well through hyperparameter tuning for a sufficiently large number of images from Validation sets, we eventually used this saved TEXTRON$_{8LF}$ model and tuned hyperparameters for inference on unseen test sets.

7. Hyperparameters for Inference

In this section, we mention the best-performing set of hyperparameters for TEXTRON$_{8LF}$ that have been tuned on validation sets as described in Section 6. For all the test sets mentioned in Table 3, we apply 10% width shrinkage as the preprocessing during the LF application stage. Additionally, the hyperparameters of *height shrinkage*, *contour thickness*, and *edge thickness* that were determined for different themes of documents are mentioned in Table 4. Using these TEXTRON$_{8LF}$ configurations, we have inferred and reported the results on unseen test sets in the review copy. We also present additional results and insights in the following Section 8.

8. Results and Discussions

In this section, we present the performance of our TEXTRON$_{8LF}$ model using various combinations of quality guides and hyperparameters on the various test sets described in Table 3.

8.1. Classwise Results on Docbank

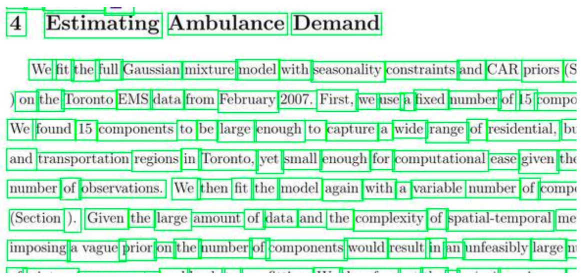
The Docbank test set D1 as described in Table 3 has 100 images and every page has bounding boxes annotated for several classes. We assign 'text' predictions of TEXTRON$_{8LF}$ with Docbank classes based on the class of the ground truth that has the highest IOU with TEXTRON$_{8LF}$ prediction. We evaluate and present the results of these eleven classes. Table 5 presents a comparison of the classwise F1 scores of the word-level bounding boxes detected by the DBNet and MBV baseline against TEXTRON$_{8LF}$. We have carried out this evaluation with an IOU threshold of 0.5. Our model not only performs better for half of the classes but also shows an overall improvement compared to the baseline performances. However, TEXTRON$_{8LF}$ performs poorly on the title class as titles generally have a different and larger font; CV-based LFs fail to unify title words in one bounding box. We can tweak the quality guides or include more relevant LFs in the paradigm to work on such limitations. Besides, there is also a class imbalance observed in the test set since classes such as date, author and title make up less than 1 percent of all bounding boxes, owing to which performance drop for those classes seems significant.

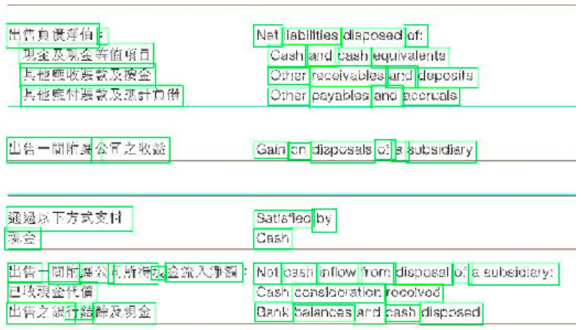
Class	**Samples**	**DBNet**	**MBV**	**TEXTRON$_{8LF}$**
Date	9	**100.00**	**100.00**	**100.00**
Author	45	**98.88**	88.17	96.70
Title	71	**91.18**	54.19	75.00
Section	435	92.63	81.73	**97.45**
List	478	87.29	**89.03**	88.40
Abstract	740	91.30	92.51	**92.68**
Footer	870	91.35	86.01	**96.00**
Caption	1317	87.98	**89.67**	88.31
Table	2669	**58.24**	50.51	50.59
Equation	4190	17.66	**32.54**	22.08
Reference	5571	**94.38**	94.07	94.18
Paragraph	38828	89.39	88.76	**90.12**
Overall	55223	84.92	84.51	**85.21**

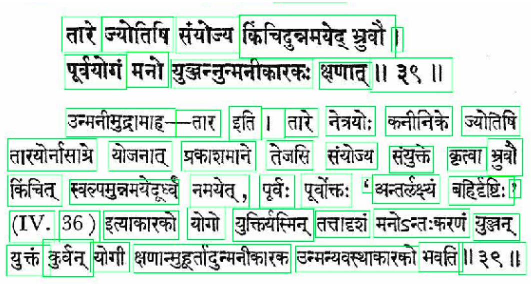
Table 5. Docbank Classwise F1 Scores (in %) for IOU 0.5

8.2. Performance on Devanagari Handwritten Text Detection

In this section, we evaluate and present the performance of TEXTRON$_{8LF}$ on the test set D5 mentioned in Table 3. Tables 6, 7 and 8 show that TEXTRON$_{8LF}$ is able to maintain the best performance for Devanagari text detection even for higher IOU thresholds. This also benefits several downstream applications like document searching, information retrieval, and handwritten text recognition.


(a) Docbank Image from Validation Set V1


(b) CTDAR Image from Validation Set V1


(c) Printed Sanskrit Image from Validation Set V3

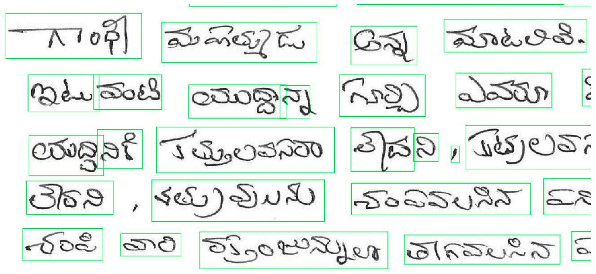

(d) Handwritten Telugu Image from Validation Set V4

Figure 6. Predictions during TEXTRON$_{8LF}$ Learning Phase from Validation Sets

Approach	P	R	F
DBNet	**72.84**	62.33	67.17
Tesseract	60.29	65.89	62.97
MBV	57.05	68.76	62.36
TEXTRON$_{8LF}$	69.24	**74.51**	**71.78**

Table 6. Results with IOU 0.5 on Handwritten Devanagari Text

Approach	P	R	F
DBNet	61.34	52.49	56.57
Tesseract	46.19	50.48	48.24
MBV	49.47	59.63	54.08
TEXTRON$_{8LF}$	**62.51**	**67.27**	**64.80**

Table 7. Results with IOU 0.6 on Handwritten Devanagari Text

Approach	P	R	F
DBNet	41.54	35.54	38.31
Tesseract	27.36	29.91	28.58
MBV	37.73	45.48	41.24
TEXTRON$_{8LF}$	**50.50**	**54.35**	**52.35**

Table 8. Results with IOU 0.7 on Handwritten Devanagari Text

References

[1] *Tamil and Malayalam Versions*. Yatharth Geeta Indian Languages, 2018. 2, 3

[2] Liangcai Gao, Yilun Huang, Hervé Déjean, Jean-Luc Meunier, Qinqin Yan, Yu Fang, Florian Kleber, and Eva Lang. Icdar 2019 competition on table detection and recognition (ctdar). In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 1510–1515. IEEE, 2019. 2

[3] Guillaume Jaume, Hazim Kemal Ekenel, and Jean-Philippe Thiran. Funsd: A dataset for form understanding in noisy scanned documents. In *2019 International Conference on Document Analysis and Recognition Workshops (ICDARW)*, volume 2, pages 1–6, 2019. 2

[4] Minghao Li, Yiheng Xu, Lei Cui, Shaohan Huang, Furu Wei, Zhoujun Li, and Ming Zhou. DocBank: A benchmark dataset for document layout analysis. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 949–960, Barcelona, Spain (Online), Dec. 2020. International Committee on Computational Linguistics. 2, 3

[5] School of Distance Education. *Malayalam: Language, Culture and Literature*. University Of Kerala, 2017. 2

[6] Obaidullah Sk, Chayan Halder, KC Santosh, Nibaran Das, and Kaushik Roy. *PHDIndic 11, page-level handwritten document image dataset of 11 official Indic scripts*. IEEE Dataport, 2021. 2, 3