applied sciences

MDPI

Article

Investigating Attention Mechanism for Page Object Detection in Document Images

Shivam Naik [1], Khurram Azeem Hashmi [1,2,3,*], Alain Pagani [3], Marcus Liwicki [4], Didier Stricker [1,3] and Muhammad Zeshan Afzal [1,2,3]

[1] Department of Computer Science, Technical University of Kaiserslautern, 67663 Kaiserslautern, Germany; naik@rhrk.uni-kl.de (S.N.); didier.stricker@dfki.de (D.S.); muhammad_zeshan.afzal@dfki.de (M.Z.A.)
[2] Mindgarage, Technical University of Kaiserslautern, 67663 Kaiserslautern, Germany
[3] German Research Institute for Artificial Intelligence (DFKI), 67663 Kaiserslautern, Germany; alain.pagani@dfki.de
[4] Department of Computer Science, Luleå University of Technology, 971 87 Luleå, Sweden; marcus.liwicki@ltu.se
* Correspondence: khurram_azeem.hashmi@dfki.de

Abstract: Page object detection in scanned document images is a complex task due to varying document layouts and diverse page objects. In the past, traditional methods such as Optical Character Recognition (OCR)-based techniques have been employed to extract textual information. However, these methods fail to comprehend complex page objects such as tables and figures. This paper addresses the localization problem and classification of graphical objects that visually summarize vital information in documents. Furthermore, this work examines the benefit of incorporating attention mechanisms in different object detection networks to perform page object detection on scanned document images. The model is designed with a Pytorch-based framework called Detectron2. The proposed pipelines can be optimized end-to-end and exhaustively evaluated on publicly available datasets such as DocBank, PublayNet, and IIIT-AR-13K. The achieved results reflect the effectiveness of incorporating the attention mechanism for page object detection in documents.

Keywords: attention mechanism; page object detection; transfer learning; document image analysis

Citation: Naik S.; Hashmi, K.A.; Pagani, A.; Liwicki, M.; Stricker, D.; Afzal, M.Z. Investigating Attention Mechanism for Page Object Detection in Document Images. *Appl. Sci.* **2022**, *12*, 7486. https://doi.org/10.3390/app12157486

Academic Editor: Byung-Gyu Kim

Received: 29 April 2022
Accepted: 22 July 2022
Published: 26 July 2022

1. Introduction

In the age of the digital era, the inflow of rich, informative documents has been immense. The number of publications is growing day by day, causing a large corpus of data to be available for analysis [1]. As a result, the need to extract information from documents has skyrocketed. Modern rule-based methods [2] have been effective for some time in extracting text. However, these methods operate with certain shortcomings, such as failure to capture the structure of the source object, e.g., tables, or to handle layout variations. With the advent of the deep-learning era, modern data-driven systems have overcome these shortcomings [3]. They bridge the gap between a traditional and a robust document detection and analysis system. Apart from text information that can be parsed with efficient OCR-based systems [4], the documents also contain graphical items, such as tables, figures, or formulas, that graphically summarize the aspects of the topic in discussion. Hence, the processing of these graphical objects has caught the research community's attention.

In general, the problem of page object detection has been addressed in previous works, such as [5–9]. A general object detection pipeline similar to [10,11] is followed to localize different types of objects, i.e., equations, tables, and figures, which make up a large portion of graphical objects present in the documents. The performance observed in these studies conveys that a deep backbone such as ResNet-152 [12] is required to localize these objects robustly. Hence, a model with a deep architecture is needed to deal with the inherent complex structures of the graphical objects and the similarities between them. Tables and

figures are a few such examples of graphical objects. To reduce the depth in the backbone without compromising the robustness of object detection, a self-attention mechanism can be employed after the feature extraction phase.

Attention mechanisms became widely popular when it was first incorporated by Bahdanau et al. [13] into the sequence to sequence (seq2seq) model, originally proposed by Kalchbrenner et al. [14], Luong et al. [15], and Cho et al. [16]. It was not until 2017, when the Transformer was introduced by Vaswani et al. [17], that the self-attention mechanism was widely adopted. Since then, self-attention mechanisms have been used in many deep-learning methods. The reason behind its popularity is its innate nature to provide semantic information to the model by providing saliency over important regions of the features. Hence, they are now an integral part of techniques in the field of Natural Language Processing (NLP) and Computer Vision (CV). DynamicHead [18], developed by Microsoft Inc., is one such variant of attention mechanisms used in this study, which combines a set of attention modules, such as scale awareness, task awareness, and spatial awareness.

The inclusion of the attention mechanism allows the model to pay attention to the input images across all the scales, and correlate between similar objects across spatial locations. It also concurrently focuses on different tasks such as detection, and segmentation. We train the model end-to-end on a benchmark dataset, i.e., DocBank [19]. The dataset contains 500,000 annotated images, with 400,000, 50,000, and 50,000 images in training, testing, and validation sets, respectively. The main goal of this paper is to provide insights into the improvements gained by incorporating an attention mechanism into a basic deep learning pipeline. To highlight the contrasting differences, a lightweight backbone such as ResNet-50 [12], combined with a standard object detection head, e.g., Faster-RCNN [10], is used as a baseline model in all the experiments. The benefits of residual blocks meant to deal with vanishing and exploding gradients, while containing a relatively small number of parameters, make ResNet-50 the ideal candidate as a backbone for the baseline. The attention mechanism is then incorporated into the baseline configuration to enable direct comparison between a model with and without the attention module. Additionally, more sets of backbones and detection heads are used in the experiments to showcase the performance with stronger backbones, discussed in detail in Section 5.

Due to the diverse distribution of labels in DocBank, including rare objects such as section, title, and reference, along with the more commonly found annotations, i.e., table, figure, and paragraph, a comprehensive study is carried out to identify the labels which attain the maximum attention and the ability of the model to learn the representation of rarely appearing labels. For the sake of fair comparison, the whole pipeline is trained and tested further with the datasets PublayNet [20] and IIIT-AR-13K [21], again containing annotated document images. The samples from each dataset are shown in Figure 1. A detailed overview of the results of the carried out experiments can be found in Section 5. To further examine the effects of DynamicHead, a sparse method called Sparse-RCNN [22] is used as an object detection head, along with a dense-to-sparse head, such as Faster-RCNN.

The motivation to choose a sparse head lies in the fact that the document images contain a fairly sparse number of objects in contrast with natural images. The number of objects can be dense, on various scales, and in unexpected locations. The head exploits the back-propagation on the proposal boxes, i.e., the proposal boxes are learned during training, which completely avoids the use of Region Proposal Network, anchor placements, and any need for post-processing, such as non-maximum suppression. Furthermore, proposal features are learned during training that captures the semantic information about the objects, helping the head learn its representation and generalize based on the understanding of the characteristics of the objects. Hence, the head tunes its proposals to best match the dataset. In Section 5, a comparison between baselines for each dataset and our model using the above mentioned detection heads are carried out.

The following chapters are organized as follows: Section 2 gives a brief insight on the advent of attention mechanism to image processing, and its developments and Section 3 covers the main methodology behind the model pipeline along with technical details about

DynamicHead, followed by the Experiments, Results and Discussion, and Conclusions and Future Work in Sections 4–6, respectively.

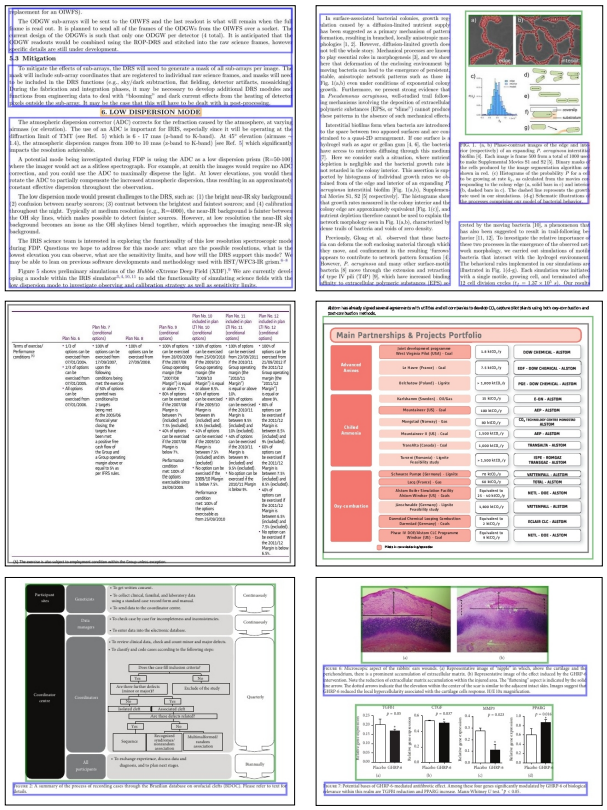


Figure 1. Visual representation of the detections obtained by the proposed model on publicly available benchmark datasets, i.e., DocBank [19], PublayNet [20], and IIIT-AR-13K [21]. Colors are used as visual embeddings to distinguish various detected objects. Blue, orange, green, and black represent paragraphs, title, figures, and tables, respectively. The first, second, and third rows represent samples from DocBank, IIIT-AR-13K, and PublayNet, respectively.

2. Related Work

2.1. Page Object Detection

In the initial days, traditional computer vision techniques were employed on documents to detect graphical objects. The classic methodologies in the past employed image processing techniques, such as binarization and linked component analysis. Contrarily, deep learning-based approaches leverage the Convolutional Neural Network (CNN) as a backbone to extract the features from document images [23,24].

2.1.1. Traditional Approach

To achieve table detection, the earlier approaches have established a specific underlying structure for tables in a document. Tupaj et al. [25] utilized Optical Character Recognition (OCR) to extract tabular information. The algorithm attempts to determine probable table regions by evaluating the keywords and white spaces. The primary drawback of this strategy is its total reliance on the presumptions about the tables' structure and the gathering of the employed keywords. In the area of table analysis, Wang et al. [26] introduced a new approach. To detect table lines, it uses the distance between consecutive words. Following that, table entity candidates are proposed by grouping adjacent vertical lines with successive horizontal words. However, the fundamental premise is that a table can have no more than two columns. As a result, three different layouts (single, double, and mixed columns) are created using this method. This approach has the disadvantage of application to a small number of designed tables.

2.1.2. Deep Learning Approach

In terms of the deep-learning approach, the initial attempts toward page object detection were carried out by Saha et al. [27] by building a GOD-framework to detect tables and figures. The model trains on a vast corpus of document images and obtains a trained model capable of distinguishing the graphical objects with good accuracy. Furthermore, a comparison is carried out between the traditional rule-based techniques and the GOD-framework to prove the effectiveness of the deep-learning approach in identifying and localizing document images, without any prior assumptions. Furthermore, the CasTab-DetectoRS network [6] takes the task a step further by incorporating Cascade networks for identifying tables. It also introduced the concept of the Recursive Pyramid Network (R.P.N.), an extension of the well-known Region Proposal Network [10], and Switchable Atrous Convolutions [28]. Altogether, it experiments with a wide range of datasets and outperforms most models in detecting tables.

In 2019, Huang et al. [29] authored a YOLO-based table detection method. The contrasting difference between natural images and graphical objects is addressed in their work. An anchor optimization strategy is employed to identify anchor sizes that are optimized to fit the tables. This enables the model to find the exact locations of the tables, making it more robust. Fairly recent research by Ma et al. [30] introduced RobusTabNet to localize tables and further proposed a split-and-merge method for recognizing the structure of a table by detecting the separation lines. This enables robust identification of the tables, in any orientation, e.g., warped orientation. They also introduced a new region proposal network called CornerNet to generate high-quality table proposals for the detection heads.

2.2. Attention Mechanism

Since the advent of the attention mechanism, many researchers found the application of attention in Computer Vision. The advancements in vision-based attention can be divided into four phases. The first phase introduces reinforcement learning-based attention via Recurrent Neural Networks (RNN) [31]. The main idea is to estimate the optimal local features through a moving local window. The process is learned by the model via rewards, given when an important feature is selected by the moving window. In the second phase, the first global attention mechanism is introduced [32]. This work applies an affine transformation to the features, to translate the features to a consistent orientation, discarding the confusion of the model over similar features. In the third phase, channel-based attention is introduced [33], where the squeeze-and-excitation process assigns weights to the feature's channels. This process tailors the saliency of the features to adapt to the problem being addressed by the model. Finally, the fourth phase introduces self-attention [34]. The main idea is to judge any local feature by its global context.

The method discussed in this paper follows the concepts from the fourth phase. This approach attempts to capture the dependencies at a long range, by computing activation at a position by a weighted sum of all the features at all the locations of the feature maps. The general idea of non-local dependency computation for a feature map at a given location is to apply pairwise multiplication of the local features at the given location with the local features at other spatial locations. Hence, self-attention is obtained, allowing the network to learn from the different areas of the image.

3. Method

This section presents the unified framework for page object detection with the help of existing models [10,18,22]. The overall architecture can be seen in Figure 2. The entire model is built on Detectron2 [35], a Pytorch-based deep learning framework. Due to its easy integration, dynamic backbone attachment, and a dedicated text-based configuration format to control the pipeline, Detectron2 plays a major role in providing a structured way of building a model. As the first step of the pipeline, a standard backbone, e.g., ResNet [12], Swin Transformer [36], or ConvNeXt [37], is used for extracting semantic features from the input. This is known as the bottom-up pathway. The feature maps obtained during the

bottom-up from the backbone stages are fed into the Feature Pyramid Network (FPN) [38], forming the top-down pathway of the backbone. The final features $\mathcal{P} = \{\mathcal{P}_i\}_{i=2}^{5}$ (where $\mathcal{P}_i$ denotes a feature extracted from the block i of the top-down pathway) of different scales are extracted and fed into the self-attention mechanism, i.e., DynamicHead, the details of which are discussed in Section 3.1.

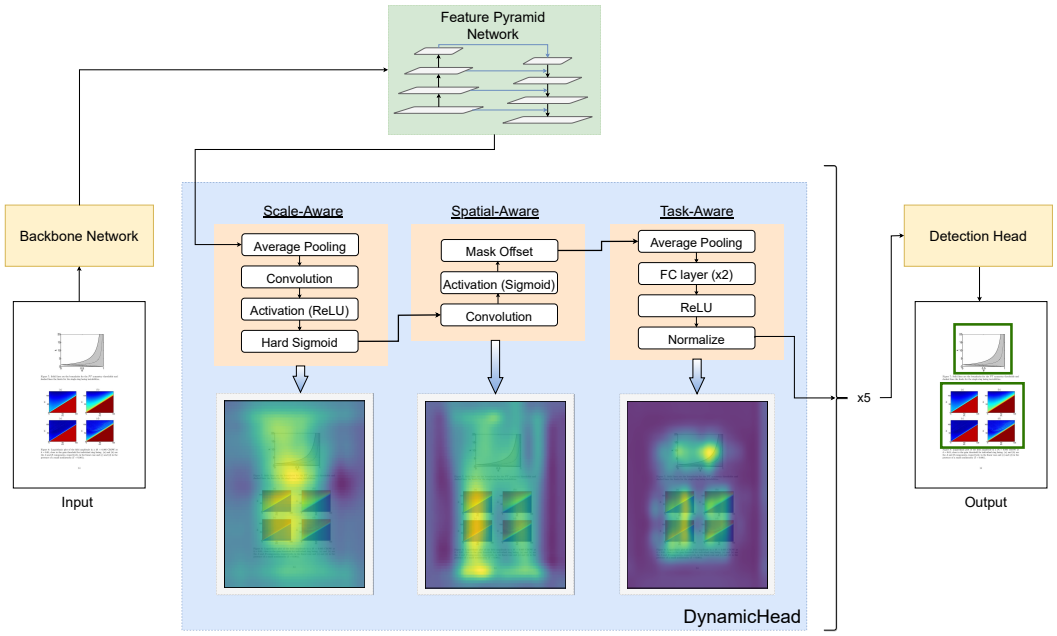


Figure 2. System design: the pipeline for page object detection using attention-mechanism. It consists of one of the backbones (i.e., ResNet-50, ConvNeXt, or Swin Transformer) which acts as the bottom-up feature extractor for the Feature Proposal Network (FPN). The top-down pathway of FPN performs deconvolution in four blocks, each block with varying scale. Then, the multi-scale features from different blocks of FPN are fed into the attention module, DynamicHead. The DynamicHead module sequentially processes the fed features through scale-aware, spatial-aware, and task-aware sub-modules. To obtain optimal attention, the DynamicHead module is applied to the features multiple times (i.e., five times). The gradient maps extracted from each sub-module within DynamicHead are indicated by the blue arrows. Finally, the features are passed on to the detection head (e.g., Faster-RCNN or Sparse-RCNN) to obtain the bounding box predictions.

3.1. DynamicHead

DynamicHead [18], referred to as DyHead in this paper, is a unified attention mechanism comprising three different attentions, each focusing on different aspects of the features. As the document images contain similar objects in various scales, e.g., paragraphs, tables, figures, and references, DynamicHead helps the overall model to transform its weights for the objects in various scales and in different spatial locations to be identified.

To have a concatenated feature set to be fed into DynamicHead, the FPN features are resized to a median scale by using either up-sampling or down-sampling on each feature level. The concatenated features are denoted by $\mathcal{R}$. These features are then fed into a series of blocks comprising scale-aware, spatial-aware, and task-aware attentions in the mentioned order.

3.1.1. Scale-Aware Attention

The features $\mathcal{R}$ are given as input to this sub-module. It performs attention by computing combined features that are obtained by interpreting the standard features across all scales, eventually activating the critical regions. The equation for applying attention, as described in [18], is given by Equation (1), where H, W, and C are height, width, and channel, respectively, f is the linear function, and σ is the Hard-Sigmoid function [39]. The final

features $\mathcal{R}_{scale}$ are obtained by combining $Attn_{scale}(\mathcal{R})$ and the input features $\mathcal{R}$ via dot product, as shown in Equation (2):

$$Attn_{scale}(\mathcal{R}) = \sigma\left(f\left(\frac{1}{\text{HWC}}\sum_{H,W,C}\mathcal{R}\right)\right) \tag{1}$$

$$\mathcal{R}_{scale} = Attn_{scale}(\mathcal{R}) \cdot \mathcal{R} \tag{2}$$

3.1.2. Spatial-Aware Attention

This sub-module takes $\mathcal{R}_{scale}$ as input and generates output features $\mathcal{R}_{spatial}$ by activating the regions of $\mathcal{R}_{scale}$, where similar objects exist across various spatial locations in the features. This allows the detection heads to focus on various locations of the input images to robustly detect them. The attention is applied as follows:

$$Attn_{spatial}(\mathcal{R}) = \frac{1}{L}\sum_{i=1}^{L}\sum_{j=1}^{S}\mathcal{R}\left(i;(\alpha_j) + \text{offset}(\alpha_j);\eta\right)\cdot\Delta_j \tag{3}$$

$$\mathcal{R}_{spatial} = Attn_{spatial}(\mathcal{R}) \cdot \mathcal{R} \tag{4}$$

In Equation (3), as originally introduced in [18], L is the number of levels of $\mathcal{R}_{scale}$, S is the number of samples obtained by applying deformable convolutions [40] on the features to make the features sparse, offset() is the spatial offset function that focuses on the important regions of the features, α_j denotes spatial features at sample j, η denotes that the attention is applied over all the channels of the features, and Δ_j is a scalar to improve the attention which is learned via back-propagation. The attention is applied to all the spatial samples, and later to all the levels. The final features $\mathcal{R}_{spatial}$, given by Equation (4), are obtained by the dot-product of $Attn_{spatial}(\mathcal{R}_{scale})$ and $\mathcal{R}_{scale}$.

3.1.3. Task-Aware Attention

This is the last module that actively learns to activate the required channels from the features, tailored specifically to the tasks to be carried out, such as bounding-box regression and segmentation. The attention is originally stated in [18], and given by Equation (5) in this paper, where $\mathcal{R}_c$ is the c^{th} channel of the features and A^1, A^2, B^1, and B^2 are the hyper-functions that control the weights and thresholds for activation. The hyper-function is implemented by conducting global average pooling on the feature dimensions for dimensionality reduction, and by applying shifted sigmoid function (range: $[-1, 1]$). Similar to the spatial and scale-aware sub-modules, the final features $\mathcal{R}_{task}$, given by Equation (6), are obtained by the dot-product of $Attn_{task}(\mathcal{R}_{spatial})$ and $\mathcal{R}_{spatial}$:

$$Attn_{task}(\mathcal{R}) = \max\left(A^1(\mathcal{R})\cdot\mathcal{R}_c + B^1(\mathcal{R}), A^2(\mathcal{R})\cdot\mathcal{R}_c + B^2(\mathcal{R})\right) \tag{5}$$

$$\mathcal{R}_{task} = Attn_{task}(\mathcal{R}) \cdot \mathcal{R} \tag{6}$$

The final step involves attaching the three mechanisms in the order of $Attn_{scale}$, $Attn_{spatial}$, and $Attn_{task}$, which make up a single DynamicHead block. These blocks can be stacked together after the FPN to improve the overall attention effectiveness. The output feature maps obtained after passing through a series of DynamicHead blocks are passed to the object detection head. The effectiveness of DynamicHead is discussed in Section 5, where various detection heads are used.

3.2. Detection Head

The feature maps obtained from DynamicHead blocks are passed onto the object detection head. There are two object-detection heads mainly used by the model, i.e., Faster-RCNN and Sparse-RCNN.

3.2.1. Faster-RCNN

Faster-RCNN receives the features from the DyHead block. The detailed pipeline of Faster-RCNN is given by Figure 3. Due to the multi-scale property of features, one anchor level per feature scale is assigned to the features instead of multiple anchor levels per feature scale, as used in the original Faster-RCNN [10] implementation. To be precise, five features, each of different scales received from the DyHead block, are assigned anchors of areas 32, 64, 128, and 256 in the given order. Furthermore, multiple aspect ratios are used at each anchor level, i.e., (0.5, 1.0, 2.0). Finally, the features are aligned to equal sizes using ROI-Pooling and passed to the regressor and classifier for predictions.

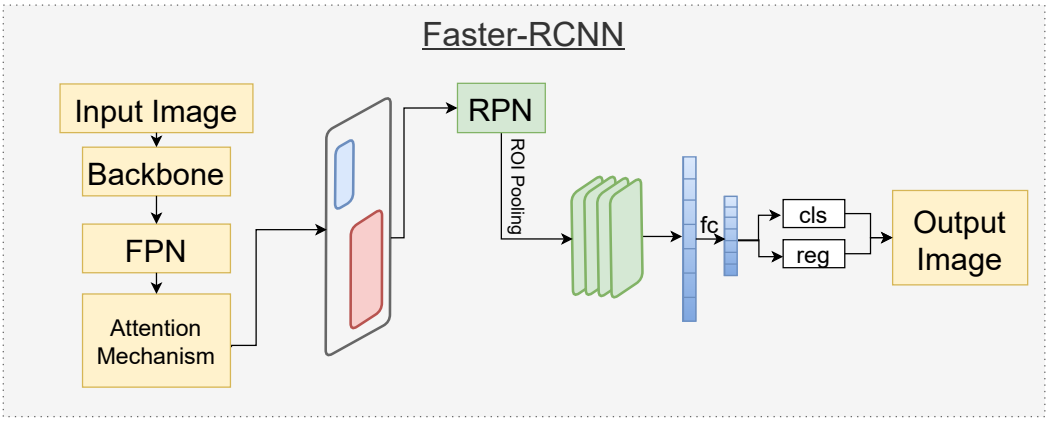


Figure 3. Flow diagram of page object detection with Faster-RCNN in detail. The features passed to the Faster-RCNN head initially passed through the Region Proposal Network (RPN) for generating proposal boxes of different dimensions. They are then aligned via ROI Pooling, to obtain fixed-size features. These features are then passed as fully connected layers (FC). Finally, the detected object is classified and a bounding box is computed.

3.2.2. Sparse-RCNN

The Sparse-RCNN head performs an ROI-aligned operation to obtain equal-sized features from the features fed by the DyHead block with the help of an initial set of proposal boxes (randomly initialized). Figure 4 describes the pipeline followed by Sparse-RCNN. The number of proposals given by Sparse-RCNN is set to 300 for this study. The proposal features are learned during the training process by learning the semantics of the objects present. Hence, learned proposals help the detection head to tailor the ROI boxes and reduce the computational cost for post-processing techniques such as non-maximum suppression.

The Iterative Head from Sparse-RCNN takes the ROI-aligned features and integrates the learned proposal features independently, i.e., ROI_box_k and $\text{proposal\_feature}_k$ for all $k \in \text{N}$ are integrated, where N denotes the number of proposal boxes. Finally, the integrated feature maps are passed via fully connected layers (FC) and passed to the classifier and box predictor. A detailed overview of the experiments carried out is discussed in Section 4.

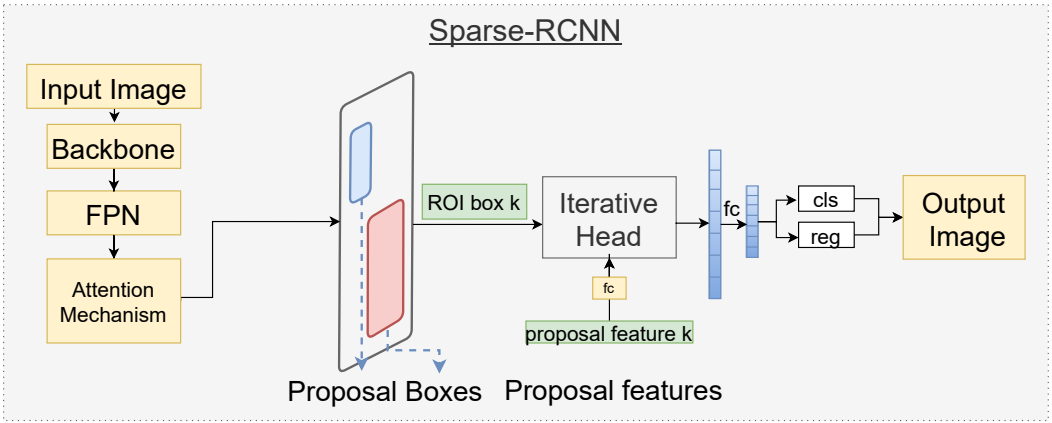


Figure 4. Flow diagram of page object detection with Sparse-RCNN in detail. Sparse-RCNN takes features from DyHead, and learns and proposes the optimal Region of Interest (ROI) boxes for the features, instead of a Region Proposal Network (RPN). The features are concatenated with latent proposal features, passed as FC layers, to predict and classify the objects.

4. Experiments

4.1. Experimental Setup

In this section, details are discussed about various experiments carried out using the previously elaborated model with different datasets. The model is trained and tested on DocBank, IIIT-AR-13K, and PublayNet. The image dimensions vary slightly intra-dataset and inter-dataset. To solve this issue, all the images are modified by resizing the shortest edge to a fixed edge length, while retaining the aspect ratio. The model is implemented using a Pytorch-based framework called Detectron2 and trained on eight RTX3090 GPUs.

4.2. Evaluation Metrics

The model performance is assessed using the following metrics: Average Precision (AP), Precision, Recall, and F1-score. The results are presented for different IoU thresholds, to fairly evaluate the model's performance.

4.2.1. Precision

Precision [41] is defined as the fraction of correctly predicted samples among the total predictions predicted as positive. The mathematical formula is given by:

$$Precision = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} \tag{7}$$

4.2.2. Recall

Recall [41] is defined as the fraction of correctly predicted samples among all the predicted samples that are positive in Ground Truth. The mathematical formula is given by:

$$Recall = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \tag{8}$$

4.2.3. F1-Score

F1-Score [41] is the measure of the overall accuracy of the model, based on the precision and recall, i.e., the harmonic mean of Precision and Recall. The mathematical formula is given by:

$$\text{F1-Score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{True Positives} + \text{Precision} + \text{Recall}} \tag{9}$$

4.2.4. Average Precision (AP)

Average Precision [42] is given by the arithmetic mean of precision scores for each sample retrieved. It essentially summarizes the precision-recall curve in a value. The mathematical formula is given by:

$$AP = \frac{\sum_{rec} P@rec}{R} \tag{10}$$

where $\sum_{rec} P@rec$ is the summation of precision values for each recall value and R is the total recall value.

4.2.5. Intersection over Union (IoU)

Intersection over Union [43] is the measure of the portion of the predicted area compared to the ground-truth area. The mathematical formula is given by:

$$IoU(X, Y) = \frac{\text{Intersection of areas of A and B}}{\text{Union of areas of A and B}} = \frac{|A \cap B|}{|A \cup B|} \tag{11}$$

4.3. Dataset Details

Three publicly available datasets are chosen for the experiments shown in this paper, i.e., DocBank, PublayNet, and IIIT-AR-13K. These datasets are chosen primarily due to their focus on documents. Since these datasets are relatively new, this paper hopes to provide a good insight into the dataset, its distributions, and its applications in page object detection. The distribution of data objects in each of the datasets is given by Table 1.

Table 1. The data distribution of datasets used for the experiments. These datasets are divided into train and test sets, with major classes, generally found in documents, spanning over both training and test sets.

Dataset	No of Classes	No of Images	
		Train	Test
DocBank	13	400,000	50,000
PublayNet	5	335,703	11,245
IIIT-AR-13K	5	9333	1955

Furthermore, the experimental results are distributed into two categories. The first category, elaborated in Section 5, contains the results obtained by training the original datasets on different model variants. The second category contains the results of the model trained on the sub-set of each dataset, containing common classes across the three datasets, i.e., table, figures, and paragraphs. The second category aims to convey to the readers about the cross-dataset detection capabilities, with a detailed explanation in Section 5.4.

5. Results and Discussion

This section discusses the experimental results of the model trained using various backbones and object detection heads. The standard metrics, i.e., Precision (7), Recall (8), F1-score (9), and AP (10) for IoU (11) levels with a range of [0.5, 0.95], are used to compare baselines and the proposed pipeline. All the models are trained for a 1x schedule (12 epochs) with a batch size of 8, on RTX3090 GPUs. Lightweight backbones, such as ResNet-50 or the tiny versions of ConvNeXt and Swin-Transformer, are used for the experiments. The reason for choosing lightweight backbones is to adhere to the goal of this paper to show the importance of the attention mechanism with a minimal backbone. Furthermore, tiny versions of ConvNeXt and Swin Transformers contain a similar number of parameters as ResNet-50, while providing additional benefits of improved saliency over the extracted features. For all the experiments discussed in this section, the baseline model is chosen as Faster-RCNN, with ResNet-50 as the backbone to adhere to the goal of this paper.

5.1. DocBank

Table 2 shows the experimental results obtained from the baseline model and the attention-based models. All the results shown are calculated for bounding boxes to fairly compare with the baseline model. To the best of my knowledge, this work is the first at computing the metrics on DocBank. Therefore, a comparison with the prior literature is not available.

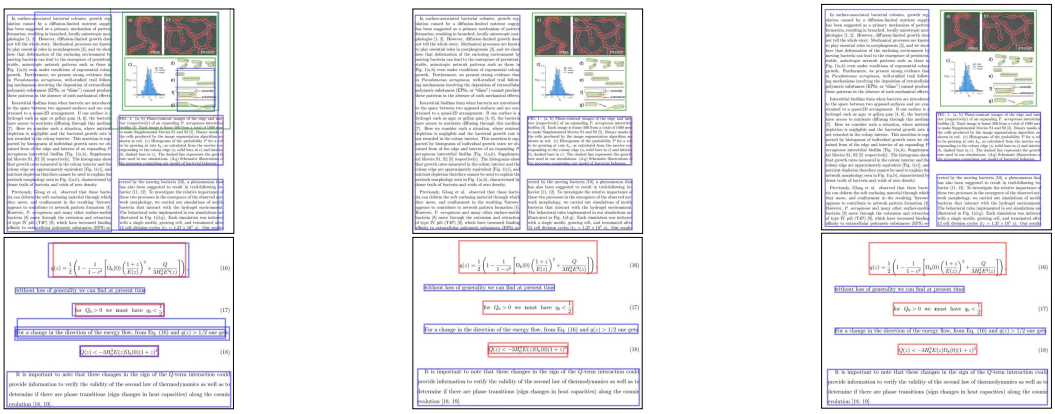
Table 2. Performance evaluations of DocBank using Faster-RCNN with different backbones. Results with Sparse-RCNN using ResNet-50 backbone are also presented. For Faster-RCNN+ResNet-50, the inclusion of the attention mechanism improves the mAP by 5.18. The Faster-RCNN+DyHead+ConvNeXt (tiny) combination outperforms baseline model and the other model variants containing DyHead by a large margin.

Model		**Faster-RCNN (Baseline)**	**Faster-RCNN +DyHead**			**Sparse-RCNN +DyHead**
Backbone		**ResNet-50**	**ResNet-50**	**ConvNeXt (tiny)**	**Swin (tiny)**	**ResNet-50**
Average Precision (AP)	paragraph	74.35	80.01	87.51	40.46	84.52
	equation	65.17	74.22	83.45	61.03	78.08
	abstract	83.22	86.40	92.07	90.40	91.61
	table	71.14	77.68	86.73	76.02	86.31
	list	54.88	66.82	81.31	68.71	76.97
	section	65.10	71.93	80.02	63.37	75.15
	figure	62.62	64.99	71.39	60.61	71.92
	author	44.79	55.72	76.36	72.4	74.18
	footer	64.51	70.61	81.84	77.03	78.66
	caption	71.12	75.65	83.32	75.75	80.78
	date	7.14	0	44.17	0	36.85
	title	77.27	79.80	85.96	85.45	82.44
	reference	77.31	82.18	88.13	81.91	87.03
mAP		62.97	68.15	80.18	65.63	77.27

ConvNeXt and DyHead with the standard Faster-RCNN gives the best performance of all our experiments on the DocBank dataset. The Faster-RCNN+ConvNeXt+DyHead model is trained from scratch, as ConvNeXt is a relatively new backbone compared to Swin Transformers or ResNet-50, containing publicly available pre-trained weights trained on ImageNet [44]. Furthermore, Table 3 shows the F1-score, Precision, and Recall for IoU levels of 0.5 to 0.9, using the best model, i.e., Faster-RCNN+ConvNeXt+DyHead. The model performs the best at the IoU level of 0.5, and the performance gradually declines as the IoU increases due to a reduction in False Positives. Figure 5 shows the visual improvements in the detections carried out by the baseline model, Faster-RCNN+ResNet-50+DyHead, and Faster-RCNN+ConvNeXt+DyHead variants.

Table 3. The model performance results for Faster-RCNN+ConvNeXt+DyHead (bbox) on DocBank.

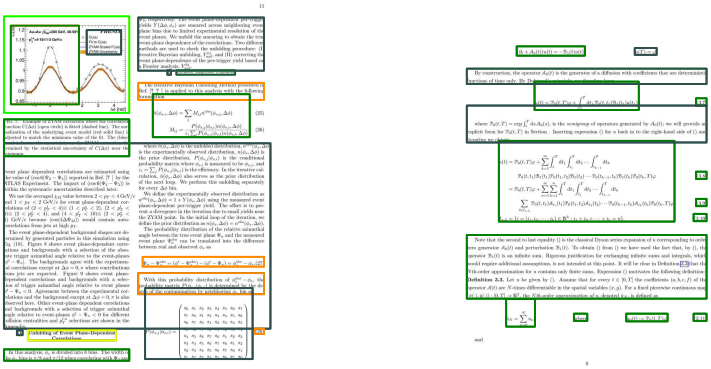
IoU	**AP**	**Precision**	**Recall**	**F1-Score**
0.5	90.95	0.94	0.91	0.86
0.6	89.99	0.93	0.90	0.85
0.7	87.64	0.91	0.88	0.84
0.8	82.85	0.87	0.83	0.81
0.9	66.79	0.72	0.67	0.72



(**a**) (**b**) (**c**)

Figure 5. Detection results of DocBank [19] obtained from the baseline, Faster-RCNN+ResNet-50+DyHead, and the Faster-RCNN+ConvNeXt+DyHead model. The bounding-box colours red, green, and blue represent equations, figures, and paragraphs, respectively. (**a**) The samples obtained from the standard Faster-RCNN model using ResNet-50 as the backbone, without attention. (**b**) The corresponding improved samples obtained when the attention mechanism is incorporated into the pipeline used in (**a**). (**c**) The samples obtained when ResNet-50 is replaced by ConvNeXt, and attention is added to the baseline model's pipeline. The level of misclassifications, and the bounding-box accuracy of the detected objects is drastically improved in (**b**) just by the addition of attention. The results are further improved in (**c**), where nested detections are discarded.

The DynamicHead module clearly improves the detection of graphical objects when compared with the baseline. Furthermore, DynamicHead coupled with ConvNeXt proves to perform the best while also dealing with the nested detections found in the Faster-RCNN+ResNet-50+DyHead variant. Although the detections performed are adequate, certain discrepancies in the DocBank dataset annotations slightly impact the performance of the model. Some of the examples are shown in Figure 6, e.g., figures are annotated as paragraphs, and annotations contain nested annotations within them. Hence, these discrepancies hinder the model's representation learning, leading to misclassifications.



(**a**) (**b**)

Figure 6. Samples chosen from the DocBank [19] dataset show that few annotations are wrongly classified or nested annotations exist. (**a**) The existence of too many nested annotations in the ground-truth, which can negatively impact the model's prediction power. (**b**) The anomalies present in the annotations, such as equation numbers (dark-green color) marked as equations, causing the model to misclassify.

5.2. PublayNet

In the case of PublayNet, the proposed architecture beats the baseline results. Table 4 shows the test results using different variants of the architecture. The inclusion of DynamicHead improves the baseline mAP scores by 1.65. Furthermore, ConvNeXt as the backbone further improves the scores by 4.22.

Table 4. Performance evaluations of PublayNet using Faster-RCNN with different backbones. Results with Sparse-RCNN using ResNet-50 backbone are also presented. For Faster-RCNN+ResNet-50, the attention mechanism contributes a slight improvement. However, the model benefits largely from the ConvNeXt backbone complemented with the attention mechanism to improve the feature saliency.

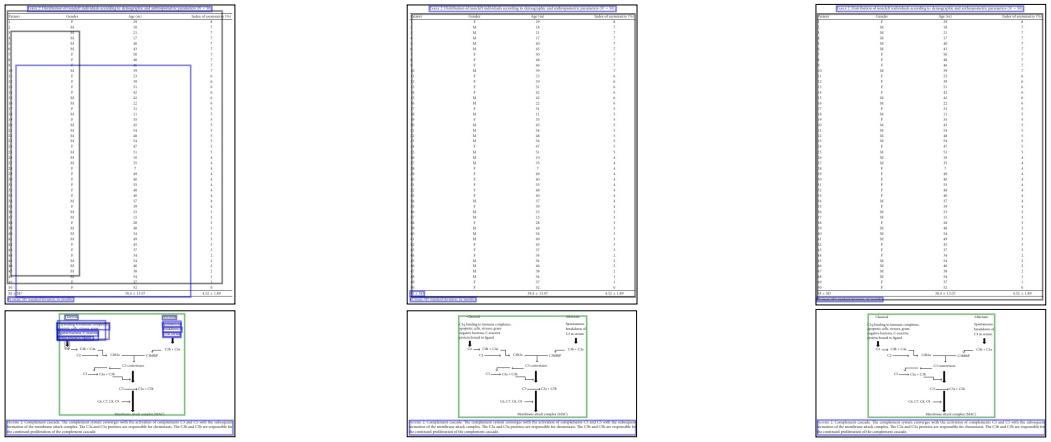
Model		**Faster-RCNN (baseline)**	**Faster-RCNN +DyHead**			**Sparse-RCNN +DyHead**
Bacbone		**ResNet-50**	**ResNet-50**	**ConvNeXt**	**Swint-tiny**	**ResNet-50**
Average Precision (AP)	text	91.3	92.72	94.29	92.23	92.71
	title	81.2	85.94	88.72	85.31	82.75
	list	88.5	89.04	94.34	87.81	87.84
	table	94.3	96.57	97.62	96.73	95.32
	figure	94.5	94.01	96.12	94.23	92.05
mAP		90	91.65	94.22	91.27	90.14

The combination of ConvNeXt and DyHead with the standard Faster-RCNN surpasses the baseline scores. Table 5 shows the F1-score, Precision, and Recall values for IoU levels of 0.5 to 0.9 for this model variant. Figure 7 shows detections obtained on PublayNet samples using different variants of the model. At first glance, it is clear that the use of DynamicHead improves the localization of the graphical objects. Similar to the results in Section 5.1, Faster-RCNN+ConvNeXt+DyHead performs the best out of all the variants.

The model performs fairly better on PublayNet compared to DocBank. The dataset contains distinct classes which allow the model to discreetly interpret their distinct features. The F1-score at different IoU levels, as shown in Table 5, also conveys the robustness of the model to detect objects even at the highest IoU level, i.e., 0.9. Additionally, with the existence of high-quality annotations present in both train and test sets, the model does not suffer from confusion.

Table 5. The model performance results for ConvNeXt+DyHead backbone with Faster-RCNN (bbox) on PublayNet.

IoU	**AP**	**Precision**	**Recall**	**F1-Score**
0.5	97.17	0.98	0.97	0.95
0.6	96.71	0.97	0.97	0.95
0.7	95.88	0.97	0.96	0.95
0.8	93.49	0.94	0.93	0.93
0.9	84.43	0.86	0.84	0.86



(**a**) (**b**) (**c**)

Figure 7. Detection results of PublayNet from baseline, Faster-RCNN+ResNet-50+DyHead, and the Faster-RCNN+ConvNeXt+DyHead model. The colors blue, black, and green represent paragraphs, tables, and figures, respectively. (**a**) The samples obtained from the baseline model with Faster-RCNN using ResNet-50 as the backbone. (**b**) The samples obtained with the attention mechanism incorporated into the pipeline of the baseline model. (**c**) The detections by Faster-RCNN+ConvNeXt+DyHead model. The detections are clean and accurate with the inclusion of the attention mechanism. Furthermore, DyHead and ConvNeXt prove to detect the objects better compared to its counterparts with fewer nested detections.

5.3. IIIT-AR-13K

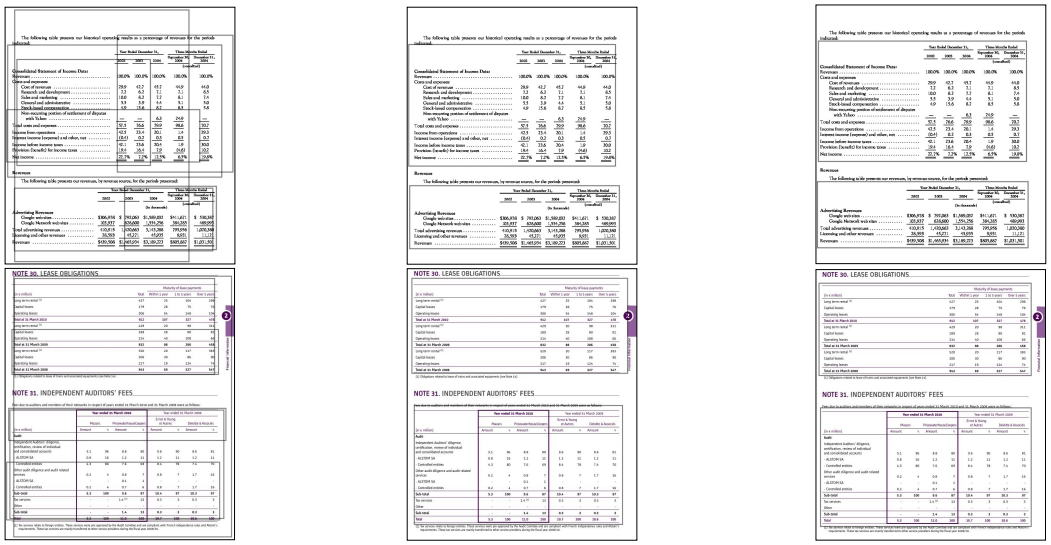
The importance of the attention mechanism is best conveyed with the IIT-AR-13K dataset, as the difference in performance between baseline model and attention-based models is large. The baseline model struggles to interpret the features when similar objects with different labels exist, i.e., figures, signatures, and natural images, whereas the attention-based architecture improves the localization capabilities of the model in comparison with the baseline model, as shown in Table 6. The results contain the estimated label-based and mean mAP scores for tests performed using different variants of the architecture. The comparison with prior methods is not provided, as the dataset is used only for table detection in other articles, unlike our experiments on all the classes.

Table 6. Performance evaluations of IIIT-AR-13K using Faster-RCNN with different backbones. Results with Sparse-RCNN using ResNet-50 backbone are also presented. The baseline model fails to achieve robustness with this dataset. Nevertheless, inclusion of DynamicHead improves the mAP score by 11.18 in comparison with baseline, confirming the drastic improvements with attention mechanism. The results are further improved with ConvNeXt backbone with the mAP score roughly double the baseline score, i.e., 33.30 to 69.74.

Model		**Faster-RCNN (baseline)**	**Faster-RCNN +DyHead**			**Sparse-RCNN +DyHead**
Backbone		**ResNet-50**	**ResNet-50**	**ConvNeXt (tiny)**	**Swin (tiny)**	**ResNet-50**
Average Precision (AP)	table	71.10	82.58	88.60	87.58	85.04
	logo	2.55	3.67	52.70	42.01	21.48
	figure	37.45	39.96	64.01	64.69	51.10
	signature	17.73	43.38	66.17	66.75	53.56
	natural_image	37.67	52.81	77.21	82.16	81.22
mAP		33.30	44.48	69.74	68.64	58.48

The ConvNeXt variant performs the best out of all the combinations of our model, the outputs of which can be visualized in Figure 8. The contrasting differences can be found between the baseline model and the Faster-RCNN+ResNet-50+DyHead variant.

However, the detections are similar between Faster-RCNN+ResNet-50+DyHead and Faster-RCNN+ConvNeXt+DyHead models. Nevertheless, when the granular details are compared, the ConvNeXt variant computes bounding boxes more accurately. Table 7 shows the F1-score, Precision, and Recall for IoU level values of 0.5 to 0.9, computed for the RCNN+ConvNeXt+DyHead model. It can be observed that the model is not robust enough to localize the objects when high level of overlap is expected between ground-truth and computed bounding boxes, i.e., at IoU levels 0.8 and 0.9.



(**a**) (**b**) (**c**)

Figure 8. Detection results of IIIT-AR-13K obtained from the baseline, Faster-RCNN+ResNet-50+DyHead, and Faster-RCNN+ConvNeXt+DyHead. The color black represents tables in the subfigures. (**a**) The samples obtained from the baseline model. (**b**) The corresponding samples obtained with attention mechanism incorporated into the pipeline followed in baseline model. (**c**) The detections on samples by the Faster-RCNN+ConvNeXt+DyHead variant. Attention mechanism-based results have no signs of erroneous detections of Tables as compared to the baseline.

Table 7. The model performance results for ConvNeXt+DyHead backbone with Faster-RCNN (bbox) on IIIT-AR-13K.

IoU	AP	Precision	Recall	F1-Score
0.50	91.44	0.93	0.91	0.89
0.6	89.09	0.91	0.89	0.87
0.7	84.21	0.86	0.84	0.84
0.8	70.65	0.74	0.71	0.75
0.9	36.22	0.41	0.36	0.47

5.4. Cross-Dataset Evaluation

To facilitate cross-dataset comparison, each dataset is reduced to the labels that are common among all three datasets, i.e., table, figure, and paragraph. The Leave-One-Out strategy is employed here, where the model is trained on one dataset and tested on the other two. This is done in order to evaluate the cross-dataset performance of these models. The purpose of cross-validation is to assess the effectiveness of the learned representation and the generalization of the model's prediction power across different data distributions. The Faster-RCNN+ConvNeXt+DyHead combination is used in this assessment, as it has been proven to consistently perform better than the other model variants. The comparison metrics are shown in Table 8.

Table 8. The cross-validation results using Leave-One-Out strategy on DocBank, PublayNet, and IIIT-AR-13K. All results are obtained from the model with Faster-RCNN+ConvNeXt+DyHead. Similarities between the DocBank and PublayNet data evidently help the model detect objects with better performance as opposed to IIIT-AR-13K. Furthermore, due to anomalies in DocBank, as compared to the clean annotations in PublayNet, training with PublayNet and testing with DocBank yields poor results compared to the opposite scenario, i.e., training on DocBank and testing on PublayNet.

Train Dataset	Test Dataset	Metrics [0.5:0.95]			
		Precision	Recall	F1-Score	mAP
1. DocBank	PublayNet	0.81	0.77	0.77	76.62
	IIIT-AR-13K	0.46	0.37	0.44	36.72
2. PublayNet	DocBank	0.56	0.50	0.57	50.45
	IIIT-AR-13K	0.42	0.36	0.41	36.25
3. IIIT-AR-13K	DocBank	0.53	0.47	0.57	46.74
	PublayNet	0.65	0.60	0.66	59.53

It can be observed that the performance of the model drops with the Leave-One-Out strategy. For example, as shown in Table 4, Faster-RCNN+ConvNeXt+DyHead scores 94.22 mAP with its own test set but scores 50.45 mAP at best with other datasets. This indicates the features learned by the model are highly dependent on the data distribution and any slight change in the data distribution needs re-learning of the target distribution. Nevertheless, PublayNet receives the highest cross-validation score with an F1-Score of 77%. Interestingly, all the trained models fail to generalize on IIIT-AR-13K, due to the differences in the visual representation of graphical objects in IIIT-AR-13K. Figure 9 shows the difference between samples from IIIT-AR-13K and the other two datasets.

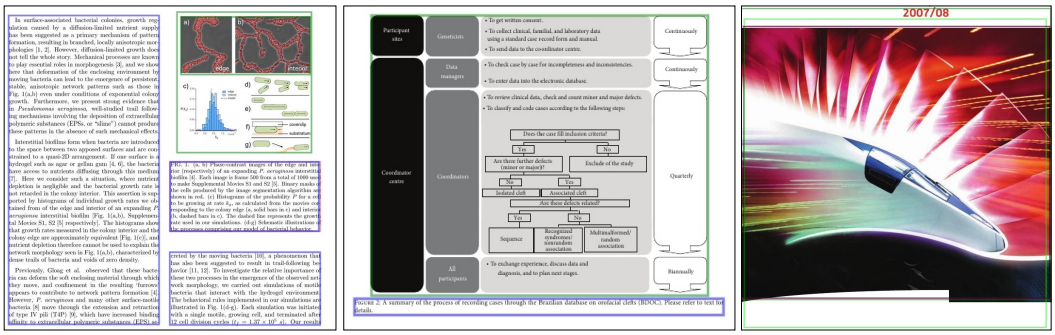


Figure 9. Different visual representations present in DocBank [19] and PublayNet [20] v/s IIIT-AR-13K [21]. Columns 1, 2, and 3 represent samples from DocBank, PublayNet, and IIIT-AR-13K, respectively. The colours blue and green represent paragraphs and figures, respectively. The vital distinction in the data distribution lies in the layout of the datasets. Figures and paragraphs are homogeneous in PublayNet and DocBank, whereas in IIIT-AR-13K, the samples follow a different layout, font-style, and document structure.

5.5. Computational Analysis

In this section, the details regarding various model properties and their corresponding runtime complexities are discussed. As the primary dataset for this paper remains DocBank, all the data presented in Table 9 are computed for the same dataset.

Table 9. The computational complexity of our algorithm with various configurations. The baseline configuration consists of a ResNet-50 backbone and Faster-RCNN as the head. All the rest use DyHead as the attention mechanism along with a backbone and the head. The addition of DyHead module adds an overhead to the runtime of the overall model, due to additional parameters. An interesting observation is the runtime of Sparse-RCNN with DyHead. Due to learned proposals, the model runs faster than its counterpart with a similar configuration, i.e., Faster-RCNN + ResNet-50 + DyHead.

Model	Backbone	Runtime (FPS)	No. of Parameters (in Millions)	GPU	Batch Size
Faster-RCNN (Baseline)	ResNet-50	9.25	41.3	RTX3090	1
Faster-RCNN + DyHead	ResNet-50	4.84	53.2	RTX3090	1
Faster-RCNN + DyHead	Swin (tiny)	4.24	59.3	RTX3090	1
Faster-RCNN + DyHead	ConvNeXt (tiny)	4.41	56.6	RTX3090	1
Sparse-RCNN + DyHead	ResNet-50	6.41	111.93	RTX3090	1

6. Conclusions and Future Work

In this paper, we propose an object detection pipeline by combining standard modules present in the object-detection world to demonstrate the effectiveness of the attention mechanism on page object detection. The model is trained using three datasets having high-quality annotations and a huge corpus of images for the model to learn adequately without over-fitting. A new backbone, ConvNeXt, was incorporated to improve the quality of model learning, showing that the combination of the right backbone and attention mechanism can improve object detection by a good margin.

DocBank and PublayNet share similar visual representations, whereas the IIIT-AR-13K images contain different visual embeddings. Hence, the model proposed in this paper, trained on either DocBank or PublayNet, never encounters graphical objects found in IIIT-AR-13K and fails to classify and segment them appropriately. For the DocBank dataset, the model performance shows an adequate understanding of the data distribution. In some areas, the model suffers from outliers while distinguishing between identical graphical objects, such as paragraphs and captions. Overall, the proposed attention-based model performs the best with the PublayNet dataset due to distinct class labels and clean annotations. The model also benefited from the huge training set. Finally, with IIIT-AR-13K, the model performance takes a big leap compared to the baseline model, although the visual embeddings are extremely ambiguous compared to the other two datasets. A cross-validation experiment is also conducted between the datasets. This experiment evaluates the generalization of models based on a reduced dataset containing the common class labels in all three datasets. The model trained on the reduced set of DocBank generalises extremely well, which is indicated in Table 8. Overall, the DynamicHead module proves to be extremely useful in providing saliency over the important regions of the documents that can help improve the models' performance.

To better understand the importance of attention, the future aspects are to evaluate the activations on the input images by extracting the gradients and assessing them. Furthermore, due to the specific visual properties of documents, compared to complex natural images, the model can over-fit easily. To avoid that, increasing the dataset further will help overcome sensitivity issues towards the outliers and help the model achieve robustness. An additional future task would include filtering the anomalies from annotations of the dataset to avoid introducing information ambiguity to the model.

Author Contributions: Writing—original draft preparation, S.N., K.A.H. and M.Z.A.; writing—review and editing, K.A.H., M.Z.A. and M.L.; supervision and project administration, A.P. and D.S. All authors have read and agreed to the submitted version of the manuscript.

Funding: The work leading to this publication has been partially funded by the European project INFINITY under Grant Agreement ID 883293.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. To, W.M.; Yu, B. Rise in higher education researchers and academic publications. *Emerald Open Res.* **2020**, *2*, 3. [CrossRef]
2. Smith, R. An Overview of the Tesseract OCR Engine. In Proceedings of the 9th International Conference on Document Analysis and Recognition (ICDAR 2007), Curitiba, Brazil, 23–26 September 2007; pp. 629–633. [CrossRef]
3. Vargas, R.; Mosavi, A.; Ruiz, R. Deep Learning: A Review. *Adv. Intell. Syst. Comput.* **2017**, *5*. [CrossRef]
4. Hashmi, K.A.; Ponnappa, R.B.; Bukhari, S.S.; Jenckel, M.; Dengel, A. Feedback learning: Automating the process of correcting and completing the extracted information. In Proceedings of the 2019 International Conference on Document Analysis and Recognition Workshops (ICDARW), Sydney, NSW, Australia, 22–25 September 2019; Volume 5, pp. 116–121.
5. Saha, R.; Mondal, A.; Jawahar, C.V. Graphical Object Detection in Document Images. In Proceedings of the 2019 International Conference on Document Analysis and Recognition, ICDAR 2019, Sydney, Australia, 20–25 September 2019; pp. 51–58. [CrossRef]
6. Hashmi, K.A.; Pagani, A.; Liwicki, M.; Stricker, D.; Afzal, M.Z. CasTabDetectoRS: Cascade Network for Table Detection in Document Images with Recursive Feature Pyramid and Switchable Atrous Convolution. *J. Imaging* **2021**, *7*, 214. [CrossRef]
7. Nazir, D.; Hashmi, K.A.; Pagani, A.; Liwicki, M.; Stricker, D.; Afzal, M.Z. HybridTabNet: Towards Better Table Detection in Scanned Document Images. *Appl. Sci.* **2021**, *11*, 8396. [CrossRef]
8. Hashmi, K.A.; Pagani, A.; Liwicki, M.; Stricker, D.; Afzal, M.Z. Cascade Network with Deformable Composite Backbone for Formula Detection in Scanned Document Images. *Appl. Sci.* **2021**, *11*, 7610. [CrossRef]
9. Antonacopoulos, A.; Clausner, C.; Papadopoulos, C.; Pletschacher, S. Historical document layout analysis competition. In Proceedings of the 2011 International Conference on Document Analysis and Recognition, Beijing, China, 18–21 September 2011; pp. 1516–1520.
10. Ren, S.; He, K.; Girshick, R.B.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *arXiv* **2015**, arXiv:1506.01497.
11. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R.B. Mask R-CNN. *arXiv* **2017**, arXiv:1703.06870.
12. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. *arXiv* **2015**, arXiv:1512.03385.
13. Bahdanau, D.; Cho, K.; Bengio, Y. Neural Machine Translation by Jointly Learning to Align and Translate. In Proceedings of the 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, 7–9 May 2015.
14. Kalchbrenner, N.; Blunsom, P. Recurrent Continuous Translation Models. In Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, Grand Hyatt Seattle, Seattle, WA, USA, 18–21 October 2013; pp. 1700–1709.
15. Luong, T.; Sutskever, I.; Le, Q.V.; Vinyals, O.; Zaremba, W. Addressing the Rare Word Problem in Neural Machine Translation. *arXiv* **2014**, arXiv:1410.8206.
16. Cho, K.; van Merrienboer, B.; Gülçehre, Ç.; Bougares, F.; Schwenk, H.; Bengio, Y. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. *arXiv* **2014**, arXiv:1406.1078.
17. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention Is All You Need. *arXiv* **2017**, arXiv:1706.03762.
18. Dai, X.; Chen, Y.; Xiao, B.; Chen, D.; Liu, M.; Yuan, L.; Zhang, L. Dynamic Head: Unifying Object Detection Heads with Attentions. *arXiv* **2021**, arXiv:2106.08322.
19. Li, M.; Xu, Y.; Cui, L.; Huang, S.; Wei, F.; Li, Z.; Zhou, M. DocBank: A Benchmark Dataset for Document Layout Analysis. *arXiv* **2020**, arXiv:2006.01038.
20. Zhong, X.; Tang, J.; Jimeno-Yepes, A. PubLayNet: Largest dataset ever for document layout analysis. *arXiv* **2019**, arXiv:1908.07836.
21. Mondal, A.; Lipps, P.; Jawahar, C.V. IIIT-AR-13K: A New Dataset for Graphical Object Detection in Documents. *arXiv* **2020**, arXiv:2008.02569.
22. Sun, P.; Zhang, R.; Jiang, Y.; Kong, T.; Xu, C.; Zhan, W.; Tomizuka, M.; Li, L.; Yuan, Z.; Wang, C.; et al. Sparse R-CNN: End-to-End Object Detection with Learnable Proposals. *arXiv* **2020**, arXiv:2011.12450.
23. Bhatt, J.; Hashmi, K.A.; Afzal, M.Z.; Stricker, D. A survey of graphical page object detection with deep neural networks. *Appl. Sci.* **2021**, *11*, 5344. [CrossRef]

24. Hashmi, K.A.; Liwicki, M.; Stricker, D.; Afzal, M.A.; Afzal, M.A.; Afzal, M.Z. Current Status and Performance Analysis of Table Recognition in Document Images with Deep Neural Networks. *IEEE Access* **2021**, *9*, 87663–87685. [CrossRef]
25. Tupaj, S.; Shi, Z.; Chang, C.H.; Alam, H. *Extracting Tabular Information from Text Files*; EECS Department, Tufts University: Medford, MA, USA, 1996.
26. Wang, Y.; Haralick, R.M.; Phillips, I.T. Automatic table ground truth generation and a background-analysis-based table structure extraction method. In Proceedings of the Sixth International Conference on Document Analysis and Recognition, Seattle, WA, USA, 13 September 2001; pp. 528–532.
27. Saha, R.; Mondal, A.; Jawahar, C.V. Graphical Object Detection in Document Images. *arXiv* **2020**, arXiv:2008.10843.
28. Qiao, S.; Chen, L.C.; Yuille, A. DetectoRS: Detecting Objects with Recursive Feature Pyramid and Switchable Atrous Convolution. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 20–25 June 2021; pp. 10213–10224.
29. Huang, Y.; Yan, Q.; Li, Y.; Chen, Y.; Wang, X.; Gao, L.; Tang, Z. A YOLO-Based Table Detection Method. In Proceedings of the 2019 International Conference on Document Analysis and Recognition (ICDAR), Sydney, NSW, Australia, 20–25 September 2019; pp. 813–818. [CrossRef]
30. Ma, C.; Lin, W.; Sun, L.; Huo, Q. Robust Table Detection and Structure Recognition from Heterogeneous Document Images. *arXiv* **2022**, arXiv:2203.09056.
31. Mnih, V.; Heess, N.; Graves, A.; Kavukcuoglu, K. Recurrent Models of Visual Attention. *arXiv* **2014**, arXiv:1406.6247.
32. Jaderberg, M.; Simonyan, K.; Zisserman, A.; Kavukcuoglu, K. Spatial Transformer Networks. *arXiv* **2015**, arXiv:1506.02025.
33. Hu, J.; Shen, L.; Albanie, S.; Sun, G.; Wu, E. Squeeze-and-Excitation Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2020**, *42*, 2011–2023. [CrossRef] [PubMed]
34. Wang, X.; Girshick, R.B.; Gupta, A.; He, K. Non-local Neural Networks. *arXiv* **2017**, arXiv:1711.07971.
35. Wu, Y.; Kirillov, A.; Massa, F.; Lo, W.Y.; Girshick, R. Detectron2. 2019. Available online: https://github.com/facebookresearch/detectron2 (accessed on 29 April 2022).
36. Liu, Z.; Lin, Y.; Cao, Y.; Hu, H.; Wei, Y.; Zhang, Z.; Lin, S.; Guo, B. Swin Transformer: Hierarchical Vision Transformer using Shifted Windows. *arXiv* **2021**, arXiv:2103.14030.
37. Liu, Z.; Mao, H.; Wu, C.Y.; Feichtenhofer, C.; Darrell, T.; Xie, S. A ConvNet for the 2020s. *arXiv* **2022**, arXiv:2201.03545.
38. Lin, T.; Dollár, P.; Girshick, R.B.; He, K.; Hariharan, B.; Belongie, S.J. Feature Pyramid Networks for Object Detection. *arXiv* **2016**, arXiv:1612.03144.
39. Courbariaux, M.; Bengio, Y.; David, J. BinaryConnect: Training Deep Neural Networks with binary weights during propagations. *arXiv* **2015**, arXiv:1511.00363.
40. Dai, J.; Qi, H.; Xiong, Y.; Li, Y.; Zhang, G.; Hu, H.; Wei, Y. Deformable Convolutional Networks. *arXiv* **2017**, arXiv:1703.06211.
41. Powers, D.M.W. Evaluation: From precision, recall and F-measure to ROC, informedness, markedness and correlation. *arXiv* **2020**, arXiv:2010.16061.
42. Zhang, E.; Zhang, Y. Average Precision. In *Encyclopedia of Database Systems*; Springer: Boston, MA, USA, 2009; pp. 192–193. [CrossRef]
43. Blaschko, M.B.; Lampert, C.H. Learning to Localize Objects with Structured Output Regression. In *Computer Vision—ECCV 2008*; Forsyth, D., Torr, P., Zisserman, A., Eds.; Springer: Berlin/Heidelberg, Germany, 2008; pp. 2–15.
44. Deng, J.; Dong, W.; Socher, R.; Li, L.; Li, K.; Fei-Fei, L. ImageNet: A large-scale hierarchical image database. In Proceedings of the 2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), Miami, FL, USA, 20–25 June 2009; pp. 248–255. [CrossRef]