DocParser: End-to-end OCR-free Information Extraction from Visually Rich Documents

Mohamed Dhouib*[1,2][0000−0002−5587−1028], Ghassen Bettaieb[1][0000−0003−3314−867X], and Aymen Shabou[1][0000−0001−8933−7053]

[1] DataLab Groupe, Credit Agricole S.A, Montrouge , France
[2] Ecole polytechnique, Palaiseau, France
mohamed.dhouib@polytechnique.edu*
{ghassen.bettaieb,aymen.shabou}@credit-agricole-sa.fr
https://datalab-groupe.github.io/

Abstract. Information Extraction from visually rich documents is a challenging task that has gained a lot of attention in recent years due to its importance in several document-control based applications and its widespread commercial value. The majority of the research work conducted on this topic to date follow a two-step pipeline. First, they read the text using an off-the-shelf Optical Character Recognition (OCR) engine, then, they extract the fields of interest from the obtained text. The main drawback of these approaches is their dependence on an external OCR system, which can negatively impact both performance and computational speed. Recent OCR-free methods were proposed to address the previous issues. Inspired by their promising results, we propose in this paper an OCR-free end-to-end information extraction model named DocParser. It differs from prior end-to-end approaches by its ability to better extract discriminative character features. DocParser achieves state-of-the-art results on various datasets, while still being faster than previous works.

Keywords: Information Extraction · Visually Rich Documents · OCR-free · End-to-end · DocParser

1 Introduction

Information extraction from visually rich documents (VRDs) is an important research topic that continues to be an active area of research [18,19,47,35,17,22,4,43,28] due to its importance in various real-world applications.

The majority of the existing information extraction from visually rich documents approaches [17,10,36,14] depend on an external deep-learning-based Optical Character Recognition (OCR) [2,1] engine. They follow a two-step pipeline: First they read the text using an off-the-shelf OCR system then they extract the fields of interest from the OCR'ed text. These two-step approaches have significant limitations due to their dependence on an external OCR engine. First

* The corresponding author

of all, these approaches need positional annotations along with textual annotations for training. Also, training an OCR model requires large scale datasets and huge computational resources. Using an external pre-trained OCR model is an option, which can degrade the whole model performance in the case of a domain shift. One way to tackle this is to fine-tune these off-the-shelf OCR models which is still a delicate task. In fact, the documents full annotations are generally needed to correctly fine-tune off-the-shelf OCR models, which is time-consuming and difficult to obtain. OCR post-correction [38,21] is an option to correct some of the recognition errors. However, this brings extra computational and maintenance cost. Moreover, these two-step approaches rarely fully exploit the visual information because incorporating the textual information is already computationally expensive.

Recent end-to-end OCR-free information extraction approaches [12,4,20] were proposed to tackle some of the limitations of OCR-dependant approaches. The majority of these approaches follow an encoder-decoder scheme. However, the used encoders are either unable to effectively model global dependence when they are primarily composed of Convolutional neural network (CNN) blocks [22,12] or they don't give enough privilege to character-level features extraction when they are are primarily composed of Swin Transformer [30] blocks [20,5]. In this paper, we argue that capturing both intra-character local patterns and inter-character long-range connections is essential for the information extraction task. The former is essential for character recognition and the latter plays a role in both the recognition and the localization of the fields of interest.

Motivated by the issues mentioned above, we propose an end-to-end OCR-free information extraction model named DocParser. DocParser has been designed in a way that allows it to efficiently perceive both intra-character patterns and inter-character dependencies. Consequently, DocParser is up to two times faster than state-of-the-art methods while still achieving state-of-the-art results on various datasets.

2 Related Work

2.1 OCR-dependant Approaches

Most of the OCR-dependant approaches simply use an off-the-shelf OCR engine and only focus on the information extraction task.

Prior to the development of deep learning techniques, earlier approaches [37,34,3] either followed a probabilistic approach, relied on rules or used manually designed features which often results in failure when applied to unfamiliar templates. The initial deep learning approaches only relied on textual information and simply used pre-trained language models [7,29]. Later, several approaches tried to take the layout information into consideration. First, [18] proposed Chargrid, a new type of text representation that preserves the 2D layout of a document by encoding each document page as a two-dimensional grid of characters. Then, [6] added context to this representation by using a BERT language model.

Later, [19] improved the Chargrid model by also exploiting the visual information. Graph-based models were also proposed to exploit both textual and visual information [28,39].

To successfully model the interaction between the visual, textual and positional information, recent approaches [17,10,36,14] resorted to pre-training large models. First [46] tried to bring the success of large pre-trained language models into the multi-modal domain of document understanding and proposed LayoutLM. LayoutLMv2 [45] was later released where new pre-training tasks were introduced to better capture the cross-modality interaction in the pre-training stage. The architecture was also improved by introducing spatially biased attention and thus making the spatial information more influential. Inspired by the Vision Transformer (ViT) [23], [17] modified LayoutLMv2 by using patch embeddings instead of a ResNeXt [44] Feature Pyramid Network [27] visual backbone and released LayoutLMv3. Pre-training tasks were also improved compared to previous versions. [10] proposed LAMBERT which used a modified RoBERTa [29] that also exploits the layout features obtained from an OCR system. [36] proposed TILT, a pre-trained encoder-decoder model. [14] tried to fully exploit the textual and layout information and released Bros which achieves good results without relying on the visual features. However, the efficiency and the computational cost of all the previously cited works are still hugely impacted by the used OCR system.

2.2 End-to-end Approaches

In recent years, end-to-end approaches were proposed for the information extraction task among many other Visually-Rich Document Understanding (VRDU) tasks. [12,22] both used a CNN-based encoder and a recurrent neuronal network coupled with an attention mechanism decoder. However, the accuracy of these two approaches is limited and they perform relatively badly on small datasets. [4] proposed TRIE++, a model that learns simultaneously both the text reading and the information extraction tasks via a multi-modal context block that bridges the visual and natural language processing tasks. [41] released VIES which simultaneously performs text detection, recognition and information extraction. However, both TRIE++ and VIES require the full document annotation to be trained. [20] proposed Donut, an encoder-decoder architecture that consists of a Swin Transformer [30] encoder and a Bart [25]-like decoder. [5] released Dessurt, a model that processes three streams of tokens, representing visual tokens, query tokens and the response. Cross-attention is applied across different streams to allow them to share and transfer information into the response. To process the visual tokens, Dessurt uses a modified Swin windowed attention that is allowed to attend to the query tokens. Donut and Dessurt achieved promising results, however, they don't give enough privilege to local character patterns which leads to sub-optimal results for the information extraction task.

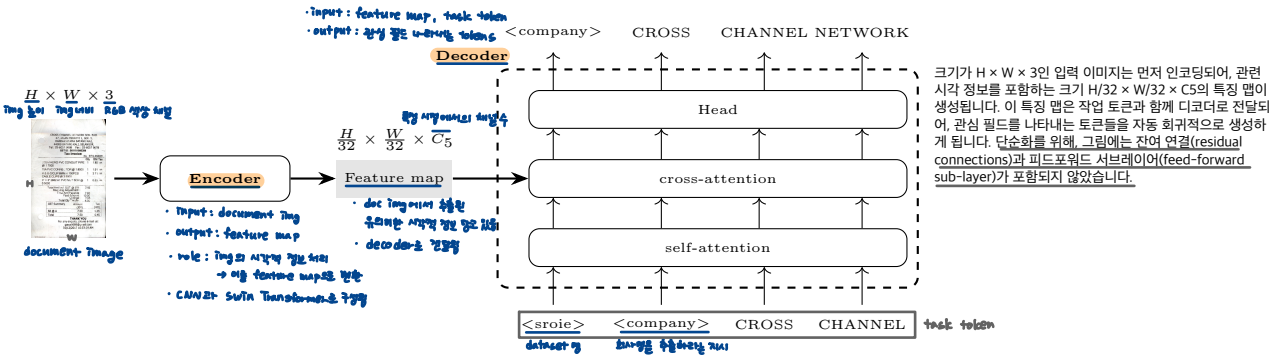


Fig. 1. An overview of DocParser's architecture. The input image of size $H \times W \times 3$ is first encoded to generate a feature map of size $\frac{H}{32} \times \frac{W}{32} \times C_5$ containing relevant visual information. The feature map is then fed to the decoder along with a task token to auto-regressively generate tokens that represent the fields of interest. For the purpose of simplification, the figure does not include residual connections and the feed-forward sub-layer.

3 Proposed Method

This section introduces DocParser, our proposed end-to-end information extraction from VRDs model.

Given a document image and a task token that determines the fields of interest, DocParser produces a series of tokens representing the extracted fields from the input image. DocParser architecture consists of a visual encoder followed by a textual decoder. An overview of DocParser's architecture is shown on figure 1. The encoder consists of a three-stage progressively decreased height convolutional neural network that aims to extract intra-character local patterns, followed by a three-stage progressively decreased width Swin Transformer [30] that aims to capture long-range dependencies. The decoder consists of n Transformer layers. Each layer is principally composed of a multi-head self-attention, sub-layer followed by a multi-head cross-attention sub-layer and a feed-forward sub-layer as explained in [40].

3.1 Encoder

The encoder is composed of six stages. The input of the encoder is an image of size $H \times W \times 3$. It is first transformed to $\frac{H}{4} \times \frac{W}{4}$ patches of dimension C_0 via an initial patch embedding. Each patch either represents a fraction of a text character or a fraction of a non-text component of the input image. First, three stages composed of ConvNext [31] blocks are applied at different scales for character-level discriminative features extraction. Then three stages of Swin Transformer blocks are applied with varying window sizes in order to capture long-range dependencies. The output of the encoder is a feature map of size $\frac{H}{32} \times \frac{W}{32} \times C_5$ that contains multi-grained features. An overview of the encoder architecture is illustrated in figure 2.

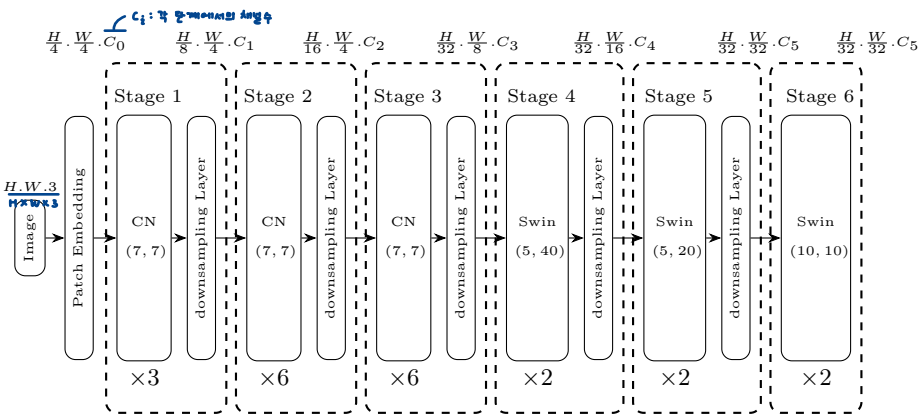


Fig. 2. The architecture of DocParser's encoder. H and W represent the height and the width of the input image. C_i, $i \in [0..5]$ represent the number of channels at different stages. The first three stages are composed of **C**onv**N**ext (CN) blocks with a filter size of $(7, 7)$. The last three stages are composed of Swin Transformer blocks with different attention window sizes. The windows' heights and widths are respectively equal to (5,40), (5,20) and (10,10).

Patch Embedding

Similar to [8], we use a progressive overlapping patch embedding. For an input image of size $W \times H \times 3$, a 3×3 convolution with stride 2 is first applied to have an output of size $\frac{W}{2} \times \frac{H}{2} \times \frac{C_0}{2}$. It is then followed by a normalization layer and another 3×3 convolution with stride 2. The size of the final output is $\frac{W}{4} \times \frac{H}{4} \times C_0$.

ConvNext-based Stages

The first three stages of DocParser's encoder are composed of ConvNext blocks. Each stage is composed of several blocks. The kernel size is set to 7 for all stages. At the end of each stage, the height of the feature map is reduced by a factor of two and the number of channels C_i, $i \in [1, 2, 3]$ is increased to compensate for the information loss. The feature map width is also reduced by a factor of two at the end of the third stage. The role of these blocks is to capture the correlation between the different parts of each single character and to encode the non-textual parts of the image. We don't reduce the width of the feature map between these blocks in order to avoid encoding components of different characters in the same feature vector and thus allowing discriminative character features computation. We note that contrary to the first encoder stages where low-level features extraction occurs, encoding components of different characters in the same feature vector doesn't affect performance if done in the encoder last stages where high-level features are constructed. This is empirically demonstrated in

section 5. We chose to use convolutional blocks for the early stages mainly due to their good ability at modeling local correlation at a low computational cost.

Swin Transformer-based Stages

The last three stages of the encoder are composed of Swin Transformer blocks. We modify Swin's window-based multi-head self-attention to be able to use rectangular attention windows. At the output of the fourth and fifth stages, the width of the feature map is reduced by a factor of two and the number of channels is increased to compensate for the information loss. The role of these layers is to capture the correlation between the different characters of the input image or between textual and non-textual components of the image. In the forth and fifth stage, the encoder focuses on capturing the correlation between characters that belong to adjacent sentences. This is accomplished through the use of horizontally wide windows, as text in documents typically has an horizontal orientation. In the last stage, the encoder focuses on capturing long-range context in both directions. This is achieved through the use of square attention windows. As a result, the output of the encoder is composed of multi-grained features that not only encode intra-character local patterns which are essential to distinguish characters but also capture the correlation between textual and non-textual components which is necessary to correctly locate the fields of interest. We note that positional embedding is added to the encoder's feature map before the encoder's forth stage.

3.2 Decoder

The decoder takes as input the encoder's output and a task token. It then outputs autoregressively several tokens that represent the fields of interest specified by the input token. The decoder consists of n[1] layers, each one is similar to a vanilla Transformer decoder layer. It consists of a multi-head self-attention sub-layer followed by a multi-head cross-attention sub-layer and a feed-forward sub-layer.

Tokenization We use the tokenizer of the RoBERTa model [29] to transform the ground-truth text into tokens. This allows to reduce the number of generated tokens, and so the memory consumption as well as training and inference times, while not affecting the model performance as shown in section 5. Similar to [20], special tokens are added to mark the start and the end of each field or group of fields. Two additional special tokens $< item >$ and $< item/ >$ are used to separate fields or group of fields appearing more than once in the ground truth. An example is shown in figure 3.

At Training Time When training the model, we use a teacher forcing strategy. This means that we give the decoder all the ground truth tokens as input. Each

[1] For our final model, we set n=1

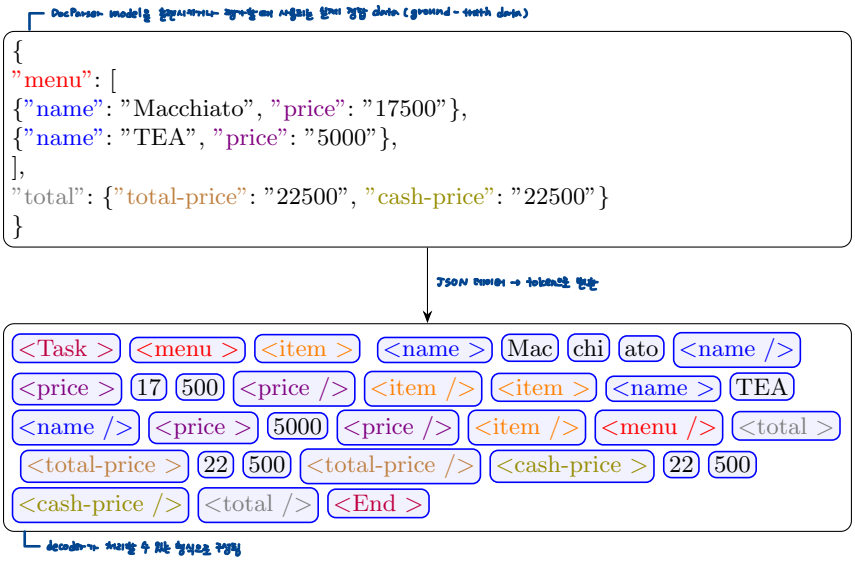


Fig. 3. An illustration of the processing applied to the textual ground-truth.

input token corresponding last hidden state is used to predict the next token. To ensure that each token only attends to previous tokens in the self-attention layer, we use a triangular attention mask that masks the following tokens.

4 Expriments and Results

4.1 Pre-training

We pre-train our model on two different steps :

Knowledge Transfer Step Using an $L2$ Loss, we teach the ConvNext-based encoder blocks to produce the same feature map as the PP-OCR-V2 [9] recognition backbone which is an enhanced version of MobileNetV1 [15]. A pointwise convolution is applied to the output of the ConvNext-based blocks in order to obtain the same number of channels as the output of PP-OCR-V2 recognition backbone. The goal of this step is to give the encoder the ability to extract discriminative intra-character features. We use 0.2 million documents from the IIT-CDIP [24] dataset for this task. We note that even though PP-OCR-V2 recognition network was trained on text crops, the features generated by its backbone on a full image are still useful thanks to the translation equivariance of CNNs.

Masked Document Reading Step After the knowledge transfer step, we pre-train our model on the task of document reading. In this pre-training phase,

the model learns to predict the next textual token while conditioning on the previous textual tokens and the input image. To encourage joint reasoning, we mask several 32×32 blocks representing approximately fifteen percent of the input image. In fact, in order to predict the text situated within the masked regions, the model is obliged to understand its textual context. As a result, DocParser learns simultaneously to recognize characters and the underlying language knowledge. We use 1.5 million IIT-CDIP documents for this task. These documents were annotated using Donut. Regex rules were applied to identify poorly read documents, which were discarded.

4.2 Fine-tuning

After the pre-training stage, the model is fine-tuned on the information extraction task. We fine-tune DocParser on three datasets: SROIE and CORD which are public datasets and an in-house private Information Statement Dataset.

SROIE : A public receipts dataset with 4 annotated unique fields : company, date, address, and total. It contains 626 receipts for training and 347 receipts for testing.

CORD : A public receipts dataset with 30 annotated unique fields of interest. It consists of 800 train, 100 validation and 100 test receipt images.

Information Statement Dataset (ISD) : A private information statement dataset with 18 annotated unique fields of interest. It consists of 7500 train, 3250 test and 3250 eval images. The documents come from 15 different insurers, each insurer has around 4 different templates. We note that for the same template, the structure can vary depending on the available information. On figure 4 we show 3 samples from 3 different insurers.

4.3 Evaluation Metrics

We evaluate our model using two metrics:

Field-level F1 Score The field-level F1 score checks whether each extracted field corresponds exactly to its value in the ground truth. For a given field, the field-level F1 score assumes that the extraction has failed even if one single character is poorly predicted. The field-level F1 score is described using the field-level precision and recall as:

$$\text{Precision} = \frac{\text{The number of exact field matches}}{\text{The number of the detected fields}} \tag{1}$$

$$\text{Recall} = \frac{\text{The number of exact field matches}}{\text{The number of the ground truth fields}} \tag{2}$$

$$\text{F1} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{3}$$

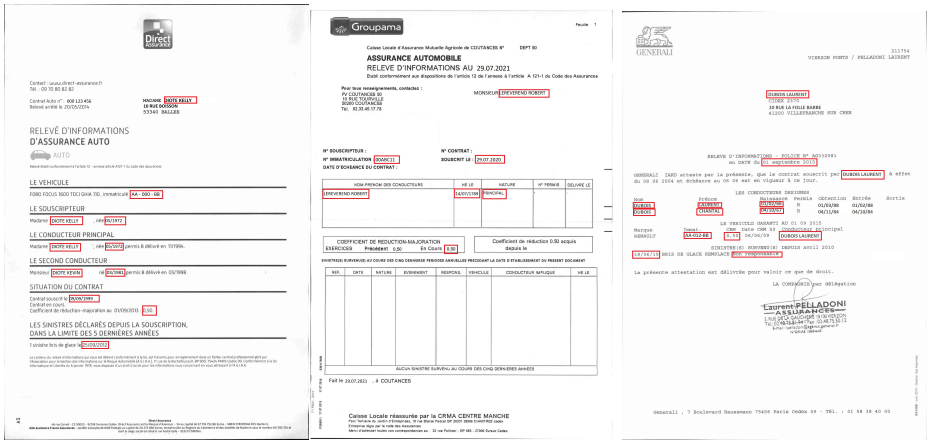


Fig. 4. Anonymized samples from our private in-house dataset. The fields of interest are located within the red boxes.

Document Accuracy Rate (DAR) This metric evaluates the number of documents that are completely and correctly processed by the model. If for a given document we have even one false positive or false negative, the DAR assumes that the extraction has failed. This metric is a challenging one, but requested in various industrial applications where we need to evaluate at which extent the process is fully automatable.

4.4 Setups

The dimension of the input patches and the output vectors of every stage C_i, $i \in [0 \,..\, 5]$ are respectively set to 64, 128, 256, 512, 768, and 1024. We set the number of decoder layers to 1. This choice is explained in Section 5. For both pre-training and fine-tuning we use the Cross-Entropy Loss, AdamW [33] optimizer with weight decay of 0.01 and stochastic depth [16] with a probability equal to 0.1. We also follow a light data augmentation strategy which consists of light re-scaling and rotation as well as brightness, saturation, and contrast augmentation applied to the input image. For the pre-training phase, we set the input image size to 2560×1920. The learning rate is set to $1e-4$. The pre-training is done on 7 A100 GPUs with a batch size of 4 on each GPU. We use gradient accumulation of 10 iterations, leading to an effective batch size of 280. For the fine-tuning, the resolution is set to 1600×960 for CORD and SROIE datasets and 1600×1280 for the Information Statement Dataset. We pad the input image in order to maintain its aspect ratio. We also use a Cosine Annealing scheduler [32] with an initial learning rate of $3e-5$ and a batch size of 8.

Table 1. Performance comparisons on the three datasets. The field-level F1-score and the extraction time per image on an Intel Xenion W-2235 CPU are reported. In order to ensure a fair comparison, we exclude parameters related to vocabulary. Additional parameters α^* and time t^* for the OCR step should be considered for LayouLM-v3. For the ISD dataset t^* is equal to 3.6 seconds.

			SROIE		CORD		ISD	
	OCR	Params(M)	F1(%)	Time(s)	F1(%)	Time(s)	F1(%)	Time(s)
LayoutLM-v3	✓	$87 + \alpha^*$	77.7	$2.1 + t^*$	80.2	$2.1 + t^*$	90.8	$4.1 + t^*$
Donut		149	81.7	5.3	84	5.7	95.4	6.7
Dessurt		87	84.9	16.7	82.5	17.9	93.5	18.1
DocParser		**70**	**87.3**	3.5	**84.5**	3.7	**96.2**	**4.4**

4.5 Results

We compare DocParser to Donut, Dessurt and LayoutLM-v3. The results are summarized in table 1. A comparison of inference speed on an NVIDIA Quadro RTX 6000 GPU is presented in table 2. Per-field extraction performances on our Information Statement Dataset can be found in table 3. DocParser achieves a new state-of-the-art on SROIE, CORD and our Information Statement Dataset with an improvement of respectively 2.4, 0.5 and 0.8 points over the previous state-of-the-art. In addition, Docparser has a significantly faster inference speed and less parameters.

Table 2. Comparison of inference speed on GPU. Extraction time (seconds) per image on an NVIDIA Quadro RTX 6000 GPU is reported. Additional time t^* for the OCR step should be considered for LayouLM-v3. For the ISD dataset t^* is equal to 0.5 seconds.

	SROIE	CORD	ISD
LayoutLM-v3	$0.041 + t^*$	$0.039 + t^*$	$0.065 + t^*$
Donut	0.38	0.44	0.5
Dessurt	1.2	1.37	1.39
DocParser	0.21	0.24	**0.25**

Regarding the OCR required by the LayoutLM-v3 approach, we use, for both SROIE and CORD datasets, Microsoft Form Recognizer[2] which includes a document-optimized version of Microsoft Read OCR (MS OCR) as its OCR engine. We note that we tried combining a ResNet-50 [13]-based DBNet++ [26] for text detection and an SVTR [8] model for text recognition and fine-tuned

[2] https://learn.microsoft.com/en-us/azure/applied-ai-services/form-recognizer/concept-read?view=form-recog-3.0.0

Table 3. Extraction performances on our Information Statement Dataset. Per field (field-level) F1-score, field-level F1-score mean, DAR, and extraction time per image on an Intel Xenion W-2235 CPU are reported. The OCR engine inference time t^* should be considered for LayouLM-v3.

Fields	LayoutLM	DONUT	Dessurt	**DocParser**
Name of first driver	85.7	91.3	89.7	**92.9**
Name of second driver	82.7	90.1	89.1	**92.1**
Name of third driver	76.8	94.2	91.7	**94.7**
Bonus-Malus	96	98.5	98.1	**99**
Date of birth of first driver	97.1	97.1	97.3	**98.3**
Date of birth of second driver	95.8	96.9	96.3	**97.2**
Date of birth of third driver	91.5	**93.2**	90.8	92.8
Date of termination of contract	92	97	95.7	**97.8**
Date of the first accident	95.6	96	96.3	**97.2**
Date of the second accident	94.2	95.2	94.8	**96.1**
Subscription date	94.7	97.9	95.7	**98.5**
Date of creation of the document	95.6	97.5	96.8	**98**
Matriculation	95.2	94	94.6	**96.7**
Name of first accident's driver	85.9	85.9	85.4	**86.1**
Name of second accident's driver	84.1	**87.1**	84.9	85.2
Underwriter	92.2	92.2	92.3	**92.8**
Responsibility of the first driver of the accident	89.8	96.5	95.7	**97.2**
Responsibility of the second driver of the accident	89.5	95	94	**95.3**
Mean F1 (%)	90.8	95.4	93.5	**96.2**
DAR (%)	58.1	74	70.6	**77.5**
Time (s)	4.1+t^*	6.7	18.1	**4.4**

them on the fields of interest of each dataset. However, the obtained results are worse than those obtained with Microsoft Form Recognizer OCR engine. For the Information Statement Dataset, we don't use MS OCR for confidentiality purposes. Instead, we use an in-house OCR fine-tuned on this dataset to reach the best possible performances. Even though the best OCRs are used for each task, LayoutLM-v3 extraction performances are still lower than those of OCR-free models. This proves the superiority of end-to-end architectures over the OCR-dependent approaches for the information extraction task. We note that for Donut, we use the same input resolution as DocParser. For Dessurt, we use a resolution of 1152×768, which is the resolution used to pre-train the model.

5 Primary Experiments and Further Investigation

5.1 Primary Experiments

In all the experiments, the tested architectures were pre-trained on 0.5 Million synthesized documents and fine-tuned on a deskewed version of the SROIE dataset. We report the inference time on a an Intel Xenion W-2235 CPU, as we aim to provide a model suited for low resources scenarios.

Table 4. Comparison of different encoder architectures. The dataset used is a deskewed version of the SROIE dataset. The field-level F1 score is reported.

Encoder architecture	F1(%)
EasyOCR-based encoder	54
PP-OCRv2-based encoder	77
Proposed encoder	81

On the Encoder's Architecture The table 4 shows a comparison between an EasyOCR[3]-based encoder, a PP-OCRv2 [9]-based encoder and our proposed DocParser encoder. Concerning the EasyOCR and PP-OCRv2 based encoders, each one consists of its corresponding OCR's recognition network followed by few convolutional layers that aim to further reduce the feature map size and increase the receptive field. Our proposed encoder surpasses both encoders by a large margin.

On the Feature Map Width Reduction While encoding the input image, the majority of the text recognition approaches reduce the dimensions of the feature map mainly vertically [8] [1]. Intuitively, applying this approach for the

[3] https://github.com/JaidedAI/EasyOCR/blob/master/easyocr

Table 5. The effect of decreasing the width of the feature map in various stages of DocParser's encoder. The dataset used is a deskewed version of the SROIE dataset. The field-level F1-score and the extraction time per image on an Intel Xenion W-2235 CPU are reported.

Encoder stages where the feature map width is reduced	Decoder	Inference time (seconds)	F1(%)
(3,4,5) (proposed)	Transformer	3.5	81
(3,4,5)	LSTM + Additive attention	3.3	58
(1,2,3)	Transformer	2.1	69
(1,2,3)	LSTM + Additive attention	1.9	52
No reduction	Transformer	6.9	81
No reduction	LSTM + Additive attention	6.6	81

information extraction task may seem relevant as it allows different characters to be encoded in different feature vectors. Our empirical results, however, show that this may not always be the case. In fact, we experimented with reducing the encoder's feature map width at different stages. As a decoder, we used both a one layer vanilla Transformer decoder and a Long Short-Term Memory (LSTM) [11] coupled with an attention mechanism that uses an additive attention scoring function [42]. Table 5 shows that reducing the width of the feature map in the early stages affects drastically the model's accuracy and that reducing the width of the feature map in the later stages achieves the the best speed-accuracy trade-off. Table 5 also shows that while the LSTM-based decoder struggles with a reduced width encoder output, the performance of the vanilla Transformer-based decoder remains the same in both cases. This is probably due to the multi-head attention mechanism that makes the Transformer-based decoder more expressive than an LSTM coupled with an attention mechanism.

On the Tokenizer Choice In addition to the RoBERTa tokenizer, we also tested a character-level tokenizer. Table 6 shows that the RoBERTa tokenizer allows faster decoding while achieving the same performance as the character-level tokenizer.

On the Number of Decoder Layers Table 7 shows that increasing the number of decoder layers doesn't improve DocParser's performance. Therefore, using one decoder layer is the best choice as it guarantees less computational cost.

On the Data Augmentation Strategy Additionally to the adopted augmentation techniques, we experimented with adding different types of blur and noise

Table 6. Comparison between different tokenization techniques. The dataset used is a deskewed version of the SROIE dataset. The field-level F1-score and the decoding time per image on an Intel Xenion W-2235 CPU are reported.

Tokenizer	Decoding inference time (s)	F1(%)
RoBERTa tokenizer	0.6	81
Character-level tokenizer	1.2	81

to the input images for both the pre-training and the fine-tuning. We concluded that this does not improve DocParser's performance. The lack of performance improvement when using blur may be attributed to the fact that the datasets used for evaluating the model do not typically include blurred images. Additionally, it is challenging to accurately create realistic noise, thus making the technique of adding noise to the input images ineffective.

Table 7. Effect of the number of decoder layers on the performance and the decoding inference time of DocParser. The dataset used is a deskewed version of the SROIE dataset. The field-level F1-score and the decoding time per image on an Intel Xenion W-2235 CPU are reported.

Decoder layers	Decoding inference time (s)	F1(%)
1	0.6	81
2	0.8	81
4	1.2	81

5.2 Further Investigation

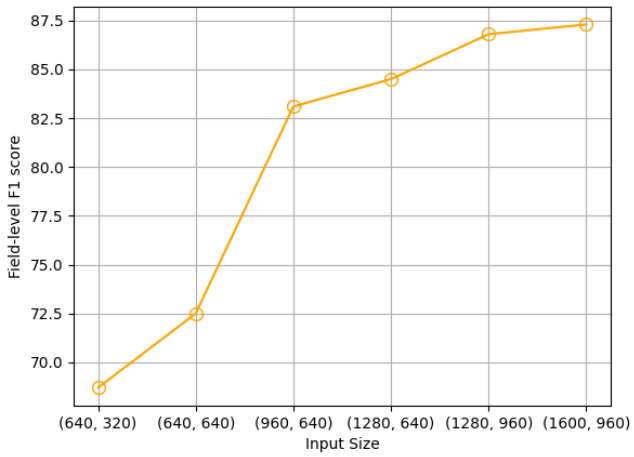
Table 8. Comparison between different pre-training strategies. All the models are pre-trained for a total of 70k steps. The field-level F1-score is reported.

Pre-training tasks	SROIE	CORD	ISD
Knowledge transfer	77.5	75	89.7
Knowledge transfer + Document reading	84.7	83.7	95.6
Knowledge transfer + Masked document reading	**84.9**	**84.2**	**95.9**

On the Pre-training Strategy Table 8 presents a comparison between different pre-training strategies. To reduce compute used, all the models were pre-trained for 70k back-propagation steps, with 7k knowledge transfer steps in the case of two pre-training tasks. The results show that masking text regions during the document reading pre-training task does effectively lead to an increase in performance on all three datasets. It also confirms, as demonstrated in [20] and [5], that document reading, despite its simplicity, is an effective pre-training task.

On the Input Resolution Figure 5 shows the effect of the input resolution on the performance of DocParser on the SROIE dataset. DocParser shows satisfying results even with a low-resolution input. It achieves 83.1 field-level F1 score with a 960×640 input resolution. The inference time for this resolution on an Intel Xenion W-2235 CPU is only 1.7 seconds. So, even at this resolution, DocParser still surpasses Donut and LayoutLM-v3 on SROIE while being more than three times faster. However, if the input resolution is set to 640×640 or below, the model's performance shows a drastic drop. This may be due to the fact that the characters start to be illegible at such a low resolution.

Fig. 5. The impact of the input resolution on DocParser's performance on the SROIE dataset. The field-level F1 score is reported.



6 Conclusion

We have introduced DocParser, a fast end-to-end approach for information extraction from visually rich documents. Contrary to previously proposed end-

to-end models, DocParser's encoder is specifically designed to capture both intra-character local patterns and inter-character long-range dependencies. Experiments on both public and private datasets showed that DocParser achieves state-of-the-art results in terms of both speed and accuracy which makes it perfectly suitable for real-world applications.

Acknowledgments The authors wish to convey their genuine appreciation to Prof. Davide Buscaldi and Prof. Sonia Vanier for providing them with valuable guidance. Furthermore, the authors would like to express their gratitude to Paul Wassermann and Arnaud Paran for their assistance in proofreading previous versions of the manuscript.

References

1. Baek, J., Kim, G., Lee, J., Park, S., Han, D., Yun, S., Oh, S.J., Lee, H.: What is wrong with scene text recognition model comparisons? dataset and model analysis. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 4715–4723 (2019)
2. Baek, Y., Lee, B., Han, D., Yun, S., Lee, H.: Character region awareness for text detection. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 9365–9374 (2019)
3. Cesarini, F., Francesconi, E., Gori, M., Soda, G.: Analysis and understanding of multi-class invoices. Document Analysis and Recognition **6**, 102–114 (2003)
4. Cheng, Z., Zhang, P., Li, C., Liang, Q., Xu, Y., Li, P., Pu, S., Niu, Y., Wu, F.: Trie++: Towards end-to-end information extraction from visually rich documents. arXiv preprint arXiv:2207.06744 (2022)
5. Davis, B., Morse, B., Price, B., Tensmeyer, C., Wigington, C., Morariu, V.: End-to-end document recognition and understanding with dessurt. In: Computer Vision–ECCV 2022 Workshops: Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part IV. pp. 280–296. Springer (2023)
6. Denk, T.I., Reisswig, C.: Bertgrid: Contextualized embedding for 2d document representation and understanding. In Workshop on Document Intelligence at NeurIPS. (2019)
7. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: Pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers). pp. 4171–4186. Association for Computational Linguistics, Minneapolis, Minnesota (Jun 2019)
8. Du, Y., Chen, Z., Jia, C., Yin, X., Zheng, T., Li, C., Du, Y., Jiang, Y.G.: Svtr: Scene text recognition with a single visual model. In: Raedt, L.D. (ed.) Proceedings of the Thirty-FirstInternational Joint Conference on Artificial Intelligence, IJCAI-22. pp. 884–890. International Joint Conferences on Artificial Intelligence Organization (7 2022), main Track
9. Du, Y., Li, C., Guo, R., Cui, C., Liu, W., Zhou, J., Lu, B., Yang, Y., Liu, Q., Hu, X., Yu, D., Ma, Y.: Pp-ocrv2: Bag of tricks for ultra lightweight ocr system. ArXiv **abs/2109.03144** (2021)

10. Garncarek, Ł., Powalski, R., Stanisławek, T., Topolski, B., Halama, P., Turski, M., Graliń ski, F.: LAMBERT: Layout-aware language modeling for information extraction. In: Document Analysis and Recognition ICDAR 2021, pp. 532–547. Springer International Publishing (2021)
11. Graves, A., Graves, A.: Long short-term memory. Supervised sequence labelling with recurrent neural networks pp. 37–45 (2012)
12. Guo, H., Qin, X., Liu, J., Han, J., Liu, J., Ding, E.: Eaten: Entity-aware attention for single shot visual text extraction. In: 2019 International Conference on Document Analysis and Recognition (ICDAR). pp. 254–259 (2019)
13. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
14. Hong, T., Kim, D., Ji, M., Hwang, W., Nam, D., Park, S.: Bros: A pre-trained language model focusing on text and layout for better key information extraction from documents. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 36, pp. 10767–10775 (2022)
15. Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H.: Mobilenets: Efficient convolutional neural networks for mobile vision applications (2017)
16. Huang, G., Sun, Y., Liu, Z., Sedra, D., Weinberger, K.Q.: Deep networks with stochastic depth. In: Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14. pp. 646–661. Springer (2016)
17. Huang, Y., Lv, T., Cui, L., Lu, Y., Wei, F.: Layoutlmv3: Pre-training for document ai with unified text and image masking. In: Proceedings of the 30th ACM International Conference on Multimedia. p. 4083–4091. MM '22, Association for Computing Machinery, New York, NY, USA (2022)
18. Katti, A.R., Reisswig, C., Guder, C., Brarda, S., Bickel, S., Höhne, J., Faddoul, J.B.: Chargrid: Towards understanding 2D documents. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. pp. 4459–4469. Association for Computational Linguistics, Brussels, Belgium (Oct-Nov 2018)
19. Kerroumi, M., Sayem, O., Shabou, A.: Visualwordgrid: Information extraction from scanned documents using a multimodal approach. In: Document Analysis and Recognition–ICDAR 2021 Workshops: Lausanne, Switzerland, September 5–10, 2021, Proceedings, Part II. pp. 389–402. Springer (2021)
20. Kim, G., Hong, T., Yim, M., Nam, J., Park, J., Yim, J., Hwang, W., Yun, S., Han, D., Park, S.: Ocr-free document understanding transformer. In: Avidan, S., Brostow, G., Cissé, M., Farinella, G.M., Hassner, T. (eds.) Computer Vision – ECCV 2022. pp. 498–517. Springer Nature Switzerland, Cham (2022)
21. Kissos, I., Dershowitz, N.: Ocr error correction using character correction and feature-based word classification. In: 2016 12th IAPR Workshop on Document Analysis Systems (DAS). pp. 198–203 (2016)
22. Klaiman, S., Lehne, M.: Docreader: bounding-box free training of a document information extraction model. In: Document Analysis and Recognition–ICDAR 2021: 16th International Conference, Lausanne, Switzerland, September 5–10, 2021, Proceedings, Part I 16. pp. 451–465. Springer (2021)
23. Kolesnikov, A., Dosovitskiy, A., Weissenborn, D., Heigold, G., Uszkoreit, J., Beyer, L., Minderer, M., Dehghani, M., Houlsby, N., Gelly, S., Unterthiner, T., Zhai, X.: An image is worth 16x16 words: Transformers for image recognition at scale. In: International Conference on Learning Representations (2021)

24. Lewis, D., Agam, G., Argamon, S., Frieder, O., Grossman, D., Heard, J.: Building a test collection for complex document information processing. In: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. p. 665–666. SIGIR '06, Association for Computing Machinery, New York, NY, USA (2006)
25. Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., Zettlemoyer, L.: BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. pp. 7871–7880. Association for Computational Linguistics, Online (Jul 2020)
26. Liao, M., Zou, Z., Wan, Z., Yao, C., Bai, X.: Real-time scene text detection with differentiable binarization and adaptive scale fusion. IEEE Transactions on Pattern Analysis and Machine Intelligence **45**(1), 919–931 (2022)
27. Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2117–2125 (2017)
28. Liu, X., Gao, F., Zhang, Q., Zhao, H.: Graph convolution for multimodal information extraction from visually rich documents. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Industry Papers). pp. 32–39. Association for Computational Linguistics, Minneapolis, Minnesota (Jun 2019)
29. Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., Stoyanov, V.: Roberta: A robustly optimized bert pretraining approach. arXiv preprint arXiv:1907.11692 (2019)
30. Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin transformer: Hierarchical vision transformer using shifted windows. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 10012–10022 (2021)
31. Liu, Z., Mao, H., Wu, C.Y., Feichtenhofer, C., Darrell, T., Xie, S.: A convnet for the 2020s. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 11976–11986 (2022)
32. Loshchilov, I., Hutter, F.: Sgdr: Stochastic gradient descent with warm restarts. in iclr, 2017. arXiv preprint arXiv:1608.03983 (2016)
33. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. In: International Conference on Learning Representations (2017)
34. Medvet, E., Bartoli, A., Davanzo, G.: A probabilistic approach to printed document understanding. Int. J. Doc. Anal. Recognit. **14**(4), 335–347 (dec 2011)
35. Palm, R.B., Winther, O., Laws, F.: Cloudscan-a configuration-free invoice analysis system using recurrent neural networks. In: 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR). vol. 1, pp. 406–413. IEEE (2017)
36. Powalski, R., Borchmann, Ł., Jurkiewicz, D., Dwojak, T., Pietruszka, M., Pałka, G.: Going full-tilt boogie on document understanding with text-image-layout transformer. In: Document Analysis and Recognition–ICDAR 2021: 16th International Conference, Lausanne, Switzerland, September 5–10, 2021, Proceedings, Part II 16. pp. 732–747. Springer (2021)
37. Rusiñol, M., Benkhelfallah, T., dAndecy, V.P.: Field extraction from administrative documents by incremental structural templates. In: 2013 12th International Conference on Document Analysis and Recognition. pp. 1100–1104 (2013)

38. Schaefer, R., Neudecker, C.: A two-step approach for automatic OCR post-correction. In: Proceedings of the The 4th Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature. pp. 52–57. International Committee on Computational Linguistics, Online (Dec 2020)
39. Sun, H., Kuang, Z., Yue, X., Lin, C., Zhang, W.: Spatial dual-modality graph reasoning for key information extraction. arXiv preprint arXiv:2103.14470 (2021)
40. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in neural information processing systems **30** (2017)
41. Wang, J., Liu, C., Jin, L., Tang, G., Zhang, J., Zhang, S., Wang, Q., Wu, Y., Cai, M.: Towards robust visual information extraction in real world: New dataset and novel solution. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 35, pp. 2738–2745 (2021)
42. Wang, W., Yang, N., Wei, F., Chang, B., Zhou, M.: Gated self-matching networks for reading comprehension and question answering. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 189–198. Association for Computational Linguistics, Vancouver, Canada (Jul 2017)
43. Wei, M., He, Y., Zhang, Q.: Robust layout-aware ie for visually rich documents with pre-trained language models. In: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 2367–2376 (2020)
44. Xie, S., Girshick, R., Dollár, P., Tu, Z., He, K.: Aggregated residual transformations for deep neural networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1492–1500 (2017)
45. Xu, Y., Xu, Y., Lv, T., Cui, L., Wei, F., Wang, G., Lu, Y., Florencio, D., Zhang, C., Che, W., Zhang, M., Zhou, L.: LayoutLMv2: Multi-modal pre-training for visually-rich document understanding. In: Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers). pp. 2579–2591. Association for Computational Linguistics, Online (Aug 2021)
46. Xu, Y., Li, M., Cui, L., Huang, S., Wei, F., Zhou, M.: LayoutLM: Pre-training of text and layout for document image understanding. In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM (aug 2020)
47. Zhao, X., Niu, E., Wu, Z., Wang, X.: Cutie: Learning to understand documents with convolutional universal text information extractor. arXiv preprint arXiv:1903.12363 (2019)