LAMBERT: Layout-Aware language Modeling for information extraction

Preprint · February 2020

Łukasz Garncarek
Applica.ai
13 PUBLICATIONS **47** CITATIONS
SEE PROFILE

Rafał Powalski
3 PUBLICATIONS **0** CITATIONS
SEE PROFILE

Tomasz Stanisławek
APPLICA.AI LTD
12 PUBLICATIONS **65** CITATIONS
SEE PROFILE

Piotr Halama
1 PUBLICATION **0** CITATIONS
SEE PROFILE

LAMBERT: Layout-Aware Language Modeling for Information Extraction

Łukasz Garncarek[1]*, Rafał Powalski[1], Tomasz Stanisławek[1], Bartosz Topolski[1], Piotr Halama[1], Michał Turski[1,2], and Filip Graliński[1,2]

[1] Applica.ul. Zajęcza 15, 00-351 Warszawa (Poland)
`{firstname.lastname@applica.ai}`
[2] Adam Mickiewicz University, ul. Wieniawskiego 1, 61-712 Poznań (Poland)

Abstract. We introduce a new simple approach to the problem of understanding documents where non-trivial layout influences the local semantics. To this end, we modify the Transformer encoder architecture in a way that allows it to use layout features obtained from an OCR system, without the need to re-learn the language semantics from scratch. We augment the input of the model only with the coordinates of token bounding boxes, avoiding the use of raw images. This leads to a layout-aware language model which can be then fine-tuned on downstream tasks.

The model is evaluated on an end-to-end information extraction task using four publicly available datasets: Kleister NDA, Kleister Charity, SROIE and CORD. We show that it achieves superior performance on datasets consisting of visually rich documents, at the same time outperforming the baseline RoBERTa on documents with flat layout (NDA F_1 increase from 78.50 to 80.42). Our solution ranked 1st on the public leaderboard for the Key Information Extraction from the SROIE dataset, improving the SOTA F_1-score from 97.81 to 98.17.

Keywords: Language model · Layout · Key information extraction · Transformer · Visually rich document

1 Introduction

The sequential structure of text leads to the usual practice of treating it as a sequence of tokens, characters, or more recently, subword units. In many problems related to Natural Language Processing (NLP), this linear perspective was enough to enable significant breakthroughs, such as the introduction of the Transformer neural architecture [29]. In this setting, the task of computing token embeddings is solved by Transformer encoders, such as BERT [6] and its derivatives, achieving top scores on the GLUE benchmark [30].

They all deal with problems arising in texts defined as sequences of words. However, in many cases a structure more intricate than just a linear ordering of tokens is available. This is the case for printed or richly formatted documents,

* The first four authors have equally contributed to the paper.

where relative vertical and horizontal positions of tokens contained in tables, spacing between paragraphs, or different styles of headers, all carry useful information. After all, the very goal of endowing texts with non-trivial layout and formatting is to improve readability.

In this article we present one of the first attempts to enrich the state-of-the-art methods of NLP with layout understanding mechanisms, contemporaneous with [32], to which we compare our model. Our approach is based on injecting the layout information into a pretrained instance of RoBERTa. Afterwards, we fine-tune the augmented model on a dataset consisting of documents with non-trivial layout.

We evaluate our model on the end-to-end information extraction task, where the training set consists of documents and target values of the properties to be extracted, without any additional annotations specifying the locations where the information on these properties can be found in the documents. We compare the results with a baseline RoBERTa model, which relies only on the sequential order of tokens obtained from the OCR (and does not use the layout features), and with the solution of [32,31]. LAMBERT achieves superior performance on visually rich documents, without sacrificing results on more linear texts.

1.1 Related work

There are two main lines of research on understanding documents with non-trivial layout. The first one is Document Layout Analysis (DLA), the goal of which is to identify contiguous blocks of text and other non-textual objects on the page and determine their function and order in the document. The obtained segmentation can be combined with the textual information contained in the detected blocks. Such a method has recently been employed in [19].

Many services utilize DLA functionality for OCR (which requires document segmentation), table detection or form field detection, and their capabilities are still being expanded. The most notable examples are Amazon Textract [1], Google Cloud Document Understanding AI platform [8], and Microsoft Cognitive Services [22]. However, they have their limitations, such as the need to create rules for extracting information from the tables recognized by the system, or use training datasets with annotated document segments.

Some more recent works on information extraction using DLA include, among others, [16,3,11,2,21,24,27]. They mostly concentrate on specific types of documents, such as invoices or forms, where the layout plays a relatively greater role than for more general documents, which might contain tables, but also large amounts of unstructured text.

The second idea is to directly combine the methods of Computer Vision and NLP, for instance by representing a text-filled page as a multi-channel image, with channels corresponding to the features encoding the semantics of the underlying text, and then using convolutional networks. This method was used, among others, in [17,5]. On the other hand, [32] used the image recognition features of the page image itself.

Our idea is also related to the one used in [26], although in a different setting. Namely, they considered texts accompanied by audio-visual signal injected into a pretrained BERT instance, by combining it with the input embeddings.

1.2 Contribution

Our main contribution is the introduction of a *Layout-Aware Language Model*, a general-purpose language model that views texts not simply as sequences of words, but instead as collections of tokens distributed over two-dimensional pages. Thus, it is able to process documents containing not only plain text, but also tables, headers, forms and various other visual elements. The implementation of the model will be made available at `https://github.com/applicaai`.

A key feature of this solution is that it retains the crucial advantage of language models, which is the ability to learn in an unsupervised setting, thus leveraging the vast abundance of publicly available unannotated documents, and allows transferring the learned representation to downstream tasks.

An important advantage is the simplicity of this approach, which requires only augmenting the input text with bounding boxes of tokens. In particular no images are needed directly by the model. This eliminates an important performance factor in real-world scenarios (actually encountered by our team), where large volumes of documents have to be sent over a network between distributed processing services.

Another contribution is an extensive ablation study of the impact of augmenting RoBERTa with various types of additional positional (both sequential and layout) embeddings on model performance on the SROIE [14], CORD [23], Kleister NDA and Kleister Charity datasets [9].

Finally, we created a new dataset for unsupervised training of layout-aware language models. We will share a 200k document subset, amounting to 2M visually rich pages, accompanied by a classification of documents into two classes: business/legal documents with complex structure and others. Due to accessibility problems of the IIT-CDIP Test Collection dataset[3] [18], this would constitute the largest widely available dataset for training layout-aware language models, allowing researchers to compare the performance of their solutions not only on the same test sets, but also with the same training set. The dataset will be published at `https://github.com/applicaai`.

2 Proposed method

We inject the layout information into the model in two ways. Firstly, we modify the input embeddings of the original RoBERTa model by adding the layout term. We also experiment with completely removing the sequential embedding term. Secondly, we apply relative attention bias, used in [28,12,25] in the context of sequential position. The final architecture is depicted in Figure 1.

[3] the link `https://ir.nist.gov/cdip/` seems dead (access on Feb 17, 2021)

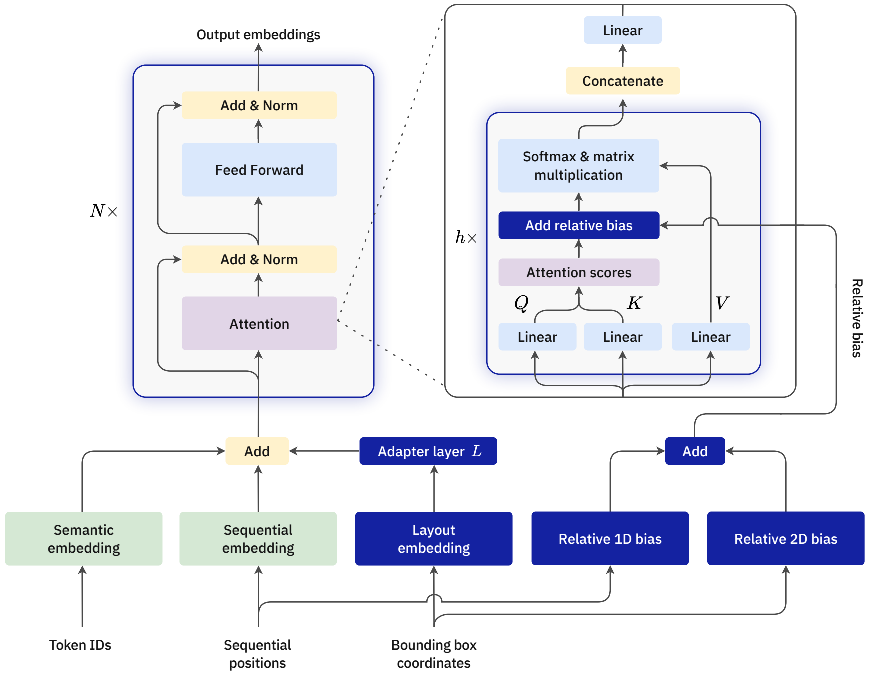

Fig. 1: LAMBERT model architecture. Differences with the plain RoBERTa model are indicated by white text on dark blue background. $N = 12$ is the number of transformer encoder layers, and $h = 12$ is the number of attention heads in each encoder layer. Q, K, and V are respectively the queries, keys and values obtained by projecting the self-attention inputs.

2.1 Background

The basic Transformer encoder, used in, for instance, BERT [6] and RoBERTa [20], is a sequence-to-sequence model transforming a sequence of input embeddings $x_i \in \mathbb{R}^n$ into a sequence of output embeddings $y_i \in \mathbb{R}^m$ of the same length, for the input/output dimensions n and m. One of the main distinctive features of this architecture is that it discards the order of its input vectors, enabling parallelization level unattainable for recurrent neural networks.

In such a setting, the information about the order of tokens is preserved not by the structure of the input, but instead passed explicitly to the model, by defining the input embeddings as

$$x_i = s_i + p_i, \tag{1}$$

where $s_i \in \mathbb{R}^n$ is the semantic embedding of the token at position i, taken from a trainable embedding layer, while $p_i \in \mathbb{R}^n$ is a *positional embedding*, depending only on i. In order to avoid confusion, henceforth we will use the term *sequential embeddings* instead of *positional embeddings*, as the latter might be understood as relating to the 2-dimensional position on the page, which we will deal with separately.

Since in RoBERTa, on which we base our approach, the embeddings p_i are trainable, the number of pretrained embeddings (in this case 512) defines a limit on the length of the input sequence. In general, there are many ways to circumvent this limit, such as using predefined [29] or relative [4] sequential embeddings.

2.2 Modification of input embeddings

We replace the input embeddings defined in (1) with

$$x_i = s_i + p_i + L(\ell_i). \tag{2}$$

Here, $\ell_i \in \mathbb{R}^k$ stands for *layout embeddings*, which are described in detail in Section 2.3. They carry the information about the position of the i-th token on the page.

The dimension k of the layout embeddings is allowed to differ from the input embedding dimension n, and this difference is dealt with by a trainable linear layer $L\colon \mathbb{R}^k \to \mathbb{R}^n$. However, our main motivation to introduce the adapter layer L was to gently increase the strength of the signal of layout embeddings during training, initially avoiding presenting the model with inputs it was not prepared to deal with. Moreover, in theory, in the case of non-trainable layout embeddings, the adapter layer may be able to learn to project ℓ_i onto a subspace of the embedding space that reduces interference with the other terms in (2). For instance, it is possible for the image of the adapter layer to learn to be approximately orthogonal to the sum of the remaining terms. This would minimize the information loss caused by adding multiple vectors. While this was our theoretical motivation, and it would be interesting to investigate in detail how much of it actually holds, such detailed considerations of a single model component exceed the scope of this paper. We included the impact of using the adapter layer in the ablation study.

We initialize the weight matrix of L according to a normal distribution $\mathcal{N}(0, \sigma^2)$, with the standard deviation σ being a hyperparameter. We have to choose σ carefully, so that in the initial phase of training, the $L(\ell_i)$ term does not interfere too much with the already learned representations. We experimentally determined the value $\sigma = 0.02$ to be near-optimal[4].

2.3 Layout embeddings

In our setting, a document is represented by a sequence of tokens t_i and their bounding boxes b_i. To each element of this sequence, we assign its layout embedding ℓ_i, carrying the information about the position of the token with respect

[4] we tested the values 0.5, 0.1, 0.02, 0.004, and 0.0008

to the whole document. This could be performed in various ways. What they all have in common is that the embeddings ℓ_i should depend only on the bounding boxes b_i and not on the tokens t_i.

We base our layout embeddings on the method originally used in [7], and then in [29] to define the sequential embeddings. We first normalize the bounding boxes by translating them so that the upper left corner is at $(0, 0)$, and dividing their dimensions by the page height. This causes the page bounding box to become $(0, 0, w, 1)$, where w is the normalized width.

The layout embedding of a token will be defined as the concatenation of four embeddings of the individual coordinates of its bounding box. For an integer d and a vector of scaling factors $\theta \in \mathbb{R}^d$, we define the corresponding embedding of a single coordinate t as

$$\mathrm{emb}_\theta(t) = (\sin(t\theta); \cos(t\theta)) \in \mathbb{R}^{2d}, \tag{3}$$

where the sin and cos are performed element-wise, yielding two vectors in $\mathbb{R}^d$. The resulting concatenation of single bounding box coordinate embeddings is then a vector in $\mathbb{R}^{8d}$.

In [29, Section 3.5], and subsequently in other Transformer-based models with precomputed sequential embeddings, the sequential embeddings were defined by emb_θ with θ being a geometric progression interpolating between 1 and 10^{-4}. Unlike the sequential position, which is a potentially large integer, bounding box coordinates are normalized to the interval $[0, 1]$. Hence, for our layout embeddings we use larger scaling factors (θ_r), namely a geometric sequence of length $n/8$ interpolating between 1 and 500, where n is the dimension of the input embeddings.

2.4 Relative bias

Recall that in a typical Transformer encoder, a single attention head transforms its input vectors into three sequences: queries $q_i \in \mathbb{R}^d$, keys $k_i \in \mathbb{R}^d$, and values $v_i \in \mathbb{R}^d$. The raw attention scores are then computed as $\alpha_{ij} = d^{-1/2} q_i^T k_j$. Afterwards, they are normalized using softmax, and used as weights in linear combinations of value vectors.

The idea of relative bias is to modify the computation of the raw attention scores by introducing a bias term: $\alpha'_{ij} = \alpha_{ij} + \beta_{ij}$. In the sequential setting, $\beta_{ij} = W(i-j)$ is a trainable weight, depending on the relative sequential position of tokens i and j. We will refer to this form of attention bias as *sequential attention bias*.

In our case, the bias β_{ij} depends on the relative positions of the tokens. More precisely, let $C \gg 1$ be an integer resolution factor (the number of cells in a grid used to discretize the normalized coordinates). If $b_i = (x_1, y_1, x_2, y_2)$ is the normalized bounding box of the i-th token, we first reduce it to a 2-dimensional position $(\xi_i, \eta_i) = (Cx_1, C(y_1 + y_2)/2)$, and then define

$$\beta_{ij} = H(\lfloor \xi_i - \xi_j \rfloor) + V(\lfloor \eta_i - \eta_j \rfloor), \tag{4}$$

where $H(\ell)$ and $V(\ell)$ are trainable weights defined for integer $\ell \in [-C, C)$. A good value for C should allow distinguishing between consecutive lines and tokens, without unnecessarily impacting performance. For a typical document $C = 100$ is enough, and we fix this in our experiments. This form of attention bias will be referred to as *2D attention bias*.

3 Experiments

All experiments were performed on 8 NVIDIA Tesla V100 32GB GPUs. As our pretrained base model we used RoBERTa in its smaller, base variant (125M parameters, 12 layers, 12 attention heads, hidden dimension 768), which was also employed as the baseline, after additional training on the same dataset we used for LAMBERT. The implementation and pretrained weights from the *transformers* library [13] were used.

In the LAMBERT model, we used layout embeddings of dimension $k = 128$, and initialized the adapter layer L with standard deviation $\sigma = 0.02$, as mentioned in Section 2. For comparison, in our experiments, we also included the published version of the LayoutLM model [32], which is of a similar size.

The models were trained on a masked language modeling objective extended with layout information (with the same settings as the original RoBERTa [20]); and subsequently, on downstream information extraction tasks. In the remainder of the paper, these two stages will be referred to as *training* and *fine-tuning*, respectively.

The training was performed on a collection of PDFs extracted from Common Crawl containing a variety of documents (we randomly selected up to 10 documents from any single domain). The documents were processed with an OCR system, `Tesseract 4.1.1-rc1-7-gb36c`, to obtain token bounding boxes. The final model was trained on the subset of the corpus consisting of business documents with non-trivial layout, filtered by an SVM binary classifier, totaling to approximately 315k documents (3.12M pages). The SVM model was trained on 700 manually annotated PDF files to distinguish between business (e.g. invoice, form) and non business documents (e.g. poetry, scientific text).

In the training phase, we used the Adam optimizer with the weight decay fix from [13]. We employed a learning rate scheduling method similar to the one used in [6], increasing the learning rate linearly from 0 to 1e−4 for the warm-up period of 10% of the training time and then decreasing it linearly to 0. The final model was trained with batch size of 128 sequences (amounting to 64K tokens) for approximately 1000k steps (corresponding to training on 3M pages for 25 epochs), which took about 5 days to complete a single experiment.

After training our models, we fine-tuned and evaluated them independently on multiple downstream end-to-end information extraction tasks. Each evaluation dataset was split into training, validation and test subsets. The models were extended with a simple classification head on top, consisting of a single linear layer, and fine-tuned on the task of classifying entity types of tokens. We employed early stopping based on the F_1-score achieved on the validation part of the dataset. We used the Adam optimizer again, but this time without the learning rate warm-up, as it turned out to have no impact on the results.

The extended model operates as a tagger on the token level, allowing to classify separate tokens, while the datasets contain only the values of properties we are supposed to extract from the documents. Therefore, further processing of output is required. To this end, we use the pipeline described in [9].

Every contiguous sequence of tokens tagged as a given entity type is treated as a recognized entity and assigned a score equal to the geometric mean of the scores of its constituent tokens. Then, every recognized entity undergoes a normalization procedure specific to its general data type (e.g. date, monetary amount, address, etc.). This is performed using regular expressions, e.g. the date `July, 15th 2013` found in the document is converted to `2013-07-15`. Afterwards, duplicates are aggregated by summing their scores, leading to a preference towards entities detected multiple times. Finally, the highest-scoring normalized entity is selected as the output of the information extraction system. The predictions obtained this way are compared with target values provided in the dataset using F_1-score as the evaluation metric. See [9] for more details.

4 Results

We evaluated our models on four public datasets containing visually rich documents. The Kleister NDA and Kleister Charity datasets are part of a larger Kleister dataset, recently made public in [9] (many examples of documents, and detailed descriptions of extraction tasks can be found therein). The NDA set consists of legal agreements, whose layout variety is limited. It should probably be treated as a plain-text dataset. The Charity dataset on the other hand contains reports of UK charity organizations, which include various tables, diagrams and other graphical elements, interspersed with passages of text. All Kleister datasets come with predefined train/dev/test splits, with 254/83/203 documents for NDA and 1729/440/609 for Charity.

The SROIE [14] and CORD [23] datasets are composed of scanned and OCRed receipts. Documents in SROIE are annotated with four target entities to be extracted, while in CORD there are 30 different entities. We use the public 1000 samples from the CORD dataset with the train/dev/test split proposed by the authors of the dataset (800/100/100 respectively). As for SROIE, it consists of a public training part, and test part with unknown annotations. For the purpose of ablation studies, we further subdivided the public part of SROIE into training and test subsets (546/80 documents; due to lack of a validation set in this split, we fine-tuned for 15 epochs instead of employing early stopping); we refer to this split as SROIE*, while the name SROIE is reserved for the original SROIE dataset, where the final evaluation on the test set is performed through the leaderboard [15].

In Table 1, we present the evaluation results achieved on downstream tasks by the trained models. With the exception of Kleister Charity dataset, where only 5 runs were made, each of the remaining experiments was repeated 20 times, and

Model	Params	Our experiments				External results	
		NDA	Charity	SROIE*	CORD	SROIE	CORD
RoBERTa [20]	125M	77.91	76.36	94.05	91.57	92.39[b]	—
RoBERTa (16M)	125M	78.50	77.88	94.28	91.98	93.03[b]	—
LayoutLM [32]	113M	77.50	77.20	94.00	93.82	94.38[a]	94.72[a]
	343M	79.14	77.13	96.48	93.62	97.09[b]	94.93[a]
LayoutLMv2 [31]	200M	—	—	—	—	96.25[a]	94.95[a]
	426M	—	—	—	—	97.81[b]	**96.01**[a]
LAMBERT (16M)	125M	80.31	79.94	96.24	93.75	—	—
LAMBERT (75M)	125M	**80.42**	**81.34**	**96.93**	**94.41**	**98.17**[b]	—

Table 1: Comparison of F_1-scores for the considered models. Best results in each column are indicated in bold. In parentheses, the length of training of our models, expressed in non-unique pages, is presented for comparison. For RoBERTa, the first row corresponds to the original pretrained model without any further training, while in the second row the model was trained on our dataset. [a]result obtained from relevant publication; [b]result of a single model, obtained from the SROIE leaderboard [15]

the mean result was reported. We compare LAMBERT with baseline RoBERTa (trained on our dataset) and the original RoBERTa [20] (without additional training); LayoutLM [32]; and LayoutLMv2 [31]. The LayoutLM model published by its authors was plugged into the same pipeline that we used for LAMBERT and RoBERTa. In the first four columns we present averaged results of our experiments, and for CORD and SROIE we additionally provide the results reported by the authors of LayoutLM, and presented on the leaderboard [15].

Since the LayoutLMv2 model was not publicly available at the time of preparing this article, we could not perform experiments ourselves. This is why some of the results are missing. For CORD, we present the scores given in [31], where the authors did not mention whether they come from averaging over multiple runs, or from a single model. A similar situation occurs for LayoutLM; we presented the average results of 20 runs (best run of LAMBERT attained the score of 95.12), which are lower than the scores presented in [31]. The difference could be attributed to using a different end-to-end evaluation pipeline, or averaging (if the results in [31,32] come from a single run).

For the full SROIE dataset, most of the results were retrieved from the public leaderboard [15], and therefore they come from a single model. For the base variants of LayoutLM and LayoutLMv2, the results were unavailable, and we present the scores from the corresponding papers.

In our experiments, the base variant of LAMBERT achieved top scores for all datasets. However, in the case of CORD, the result reported in [31] for the large variant of LayoutLMv2 is superior. If we consider the best scores of LAM-

BERT (95.12) instead of the average, and the scores of LayoutLM reported in [32], LAMBERT slightly outperforms LayoutLM, while still being inferior to LayoutLMv2. Due to lack of details on the results of LayoutLM, it is unknown which of these comparisons is valid.

For Kleister datasets, the base variant (and in the case of Charity, also the large variant) of LayoutLM did not outperform the baseline RoBERTa. We suspect that this might be the result of LayoutLM being better attuned to the evaluation pipeline used by its authors, and the fact that it was based on an uncased language model, while in the Kleister dataset performance for entities such as names may depend on using case.

5 Hyperparameters and ablation studies

In order to investigate the impact of our modifications to RoBERTa, we performed an extensive study of hyperparameters and various components of the final model. We investigated the dimension of layout embeddings, impact of the adapter layer L, size of training dataset, and finally performed a detailed ablation study of the embeddings and attention biases we augmented the baseline model with.

In the studies, every model was fine-tuned and evaluated 20 times on each dataset, except for Kleister Charity dataset, on which we fine-tuned the models 5 times due to the much longer time it took. For each combination of model and dataset, the mean score was reported, together with the two-sided 95% confidence interval, computed using the corresponding t-value. We considered differences to be significant when the corresponding intervals were disjoint. All the results are presented in Table 2, which is divided into sections corresponding to different studies. The F_1-scores are reported as *increases* with respect to the reported mean baseline score, to improve readability.

5.1 Baseline.

As a baseline for the studies we use the publicly available pretrained base variant of the RoBERTa model with 12 layers, 12 attention heads, and hidden dimension 768, which we additionally trained on our training set, and fine-tuned on the evaluation datasets in a manner analogous to our models.

5.2 Embeddings and biases.

In this study we disabled various combinations of input embeddings and attention biases. The models were trained on 2M pages for 8 epochs, with 128-dimensional layout embeddings (if enabled). The resulting models were divided into three groups. The first one contains sequential-only combinations which do not utilize the layout information at all, including the baseline. The second group consists of models using only the bounding box coordinates, with no access to sequential token positions. Finally, the models in the third group use

Train epochs and pages	Embeddings dimension	Inputs				Datasets			
		sequential	seq. bias	layout	2D bias	NDA	Charity	SROIE*	CORD
8×2M	128	●				78.50$_{\pm1.16}$	77.88$_{\pm0.48}$	94.28$_{\pm0.42}$	91.98$_{\pm0.62}$
			●			**1.94**$_{\pm0.46}$	−0.82$_{\pm0.74}$	0.33$_{\pm0.22}$	−0.15$_{\pm0.49}$
		●	●			**2.42**$_{\pm0.61}$	0.52$_{\pm0.64}$	0.79$_{\pm0.17}$	0.03$_{\pm0.57}$
				●		1.25$_{\pm0.59}$	**2.62**$_{\pm0.80}$	**1.86**$_{\pm0.15}$	0.89$_{\pm0.83}$
					●	−0.49$_{\pm0.62}$	2.02$_{\pm0.48}$	0.53$_{\pm0.28}$	−0.17$_{\pm0.62}$
				●	●	0.88$_{\pm0.50}$	**3.00**$_{\pm0.37}$	**1.94**$_{\pm0.16}$	0.68$_{\pm0.62}$
		●		●		1.74$_{\pm0.67}$	0.06$_{\pm0.93}$	**1.94**$_{\pm0.18}$	**1.42**$_{\pm0.53}$
		●			●	1.73$_{\pm0.60}$	2.02$_{\pm0.53}$	**2.09**$_{\pm0.22}$	**1.93**$_{\pm0.71}$
		●		●	●	0.54$_{\pm0.85}$	1.84$_{\pm0.42}$	**2.08**$_{\pm0.38}$	**2.15**$_{\pm0.65}$
		●	●	●		1.66$_{\pm0.76}$	0.32$_{\pm1.35}$	**1.75**$_{\pm0.35}$	1.06$_{\pm0.54}$
		●	●		●	0.85$_{\pm0.91}$	1.84$_{\pm0.27}$	**2.01**$_{\pm0.24}$	**1.95**$_{\pm0.46}$
		●	●	●	●	**1.81**$_{\pm0.60}$	**2.06**$_{\pm0.69}$	**1.96**$_{\pm0.16}$	**1.77**$_{\pm0.46}$
8×2M	128	●		●		1.74$_{\pm0.67}$	0.06$_{\pm0.93}$	**1.94**$_{\pm0.18}$	**1.42**$_{\pm0.53}$
	384	●		●		0.90$_{\pm0.54}$	0.70$_{\pm0.40}$	**1.86**$_{\pm0.22}$	**1.51**$_{\pm0.60}$
	768	●		●		0.71$_{\pm1.04}$	0.50$_{\pm0.85}$	**2.18**$_{\pm0.25}$	**1.54**$_{\pm0.51}$
	768[b]	●		●		0.77$_{\pm0.58}$	**2.30**$_{\pm0.20}$	0.37$_{\pm0.15}$	**1.58**$_{\pm0.52}$
8×2M	128	●	●	●	●	**1.81**$_{\pm0.60}$	2.06$_{\pm0.26}$	1.96$_{\pm0.18}$	1.77$_{\pm0.46}$
8×2M[a]		●	●	●	●	**1.86**$_{\pm0.66}$	1.92$_{\pm0.19}$	**2.60**$_{\pm0.18}$	1.59$_{\pm0.61}$
25×3M[a]		●	●	●	●	**1.92**$_{\pm0.50}$	**3.46**$_{\pm0.21}$	**2.65**$_{\pm0.13}$	**2.43**$_{\pm0.19}$

Table 2: Improvements of F_1-score over the baseline for various variants of LAMBERT model. The first row (with grayed background) contains the F_1-scores of the baseline RoBERTa model. The other grayed row corresponds to full LAMBERT. Statistically insignificant improvements over the baseline are grayed. In each of three studies, the best result together with all results insignificantly smaller are in bold. [a]filtered datasets; [b]model with disabled adapter layer

both sequential and layout inputs. In this group we did not disable the sequential embeddings. It includes the full LAMBERT model, with all embeddings and attention biases enabled.

Generally, we may observe that none of the modifications has led to a significant deterioration of performance. Among the considered models the only one which reported a significant improvement for all four datasets—and at the same time, the best improvement—was the full LAMBERT.

For the Kleister datasets the variance of results was relatively higher than in the case of SROIE* and CORD, leading to wider confidence intervals, and reducing the number of significant outcomes. This is true especially for the Kleister NDA dataset, which is the smallest one. In Kleister NDA, significant improvements were achieved for both sequential-only models, and for full LAMBERT. The differences between these increases were insignificant. It would seem that for sequential-only models the sequential attention bias is responsible for the improvement, but after adding also the layout inputs, it no longer leads to improvements when unaccompanied by other modifications. Still, achieving better results on sequential-only inputs may be related to the plain text nature of the Kleister NDA dataset.

While other models did not report significant improvement over the baseline, there are still some differences between them to be observed. The model using only 2D attention bias is significantly inferior to most of the others. This seems to agree with the intuition that relative 2D positions are the least suitable way to pass positional information about plain text.

In the case of the Kleister Charity dataset, significant improvements were achieved by all layout-only models, and all models using the 2D attention bias. Best improvement was attained by full LAMBERT, and two layout-only models using the layout embeddings; the 2D attention bias used alone did improve the results significantly, but did not reach the top score. The confidence intervals are too wide to make further conclusions, and many more experiments are needed to increase the significance of the results.

For the SROIE* dataset, except for two models augmented only with a single attention bias, all improvements are significant. Moreover, the differences between all the models using layout inputs are insignificant. We may conclude that passing bounding box coordinates in any way, except only through 2D attention bias, significantly improves the results. As to lack of significant improvement for 2D attention bias, we hypothesize that this is due to its relative nature. In all other models the absolute position of tokens is somehow known, either through the layout embeddings, or the sequential position. When a human reads a receipt, the absolute position is one of the main features used, to locate typical positions of entities.

For CORD, which is the more complex of the two receipt datasets, significant improvements were observed only for combined sequential and layout models. In this group, the model using both sequential and layout embeddings, augmented with sequential attention bias, did not yield a significant improvement. There were no significant differences among the remaining models in the group. Contrary to the case of SROIE*, none of the layout-only models achieved significant improvement.

5.3 Layout embedding dimension.

In this study we evaluated four models, using both sequential and layout embeddings, varying the dimension of the latter. We considered 128-, 384-, and 768-dimensional embeddings. Since this is the same as for the input embeddings of RoBERTa$_{\mathrm{BASE}}$, it was possible to remove the adapter layer L, and treat this as another variant, in Table 2 denoted as 768^{b}.

In Kleister NDA, there were no significant differences between any of the evaluated models, and no improvement over the baseline. On the other hand, in Kleister Charity disabling the adapter layer and using the 768-dimensional layout embeddings lead to significantly better performance. These results remain consistent with earlier observations that in Kleister NDA the best results were achieved by sequential-only models, while in the case of Kleister Charity, by layout-only models. It seems that in case of NDA the performance is influenced mostly by the sequential features, while in case of Charity, removing the adapter layer increases the strength of the signal of the layout embeddings, carrying the layout features which are the main factor impacting the performance.

In SROIE* and CORD all the results were also comparable, with one exception, namely on SROIE* the model with disabled adapter layer did not perform significantly better than the baseline, as opposed to the remaining models.

5.4 Training dataset size.

In this study, following the observations from [10], we considered models trained on 3 different datasets. The first model was trained for 7 epochs on 2M unfiltered (see Section 3 for more details of the filtering procedure) pages. In the second model, we used the same training time and dataset size, but this time only filtered pages were used. Finally, the third model was trained for 25 epochs on 3M filtered pages.

It is not surprising that increasing the training time and dataset size, leads to improvement of results, at least up to some point. In case of Kleister NDA dataset, there were no significant differences in the results. For Kleister Charity, the best result was achieved for the largest training dataset, consisting of 75M filtered pages. This result was also significantly better than the outcomes for the smaller dataset. In the case of SROIE* the two models trained on datasets with filtered documents achieved significantly higher score than the one trained on unfiltered documents and there was no significant difference between these two models. This supports the hypothesis that in this case filtering could be the more important factor. Finally, for CORD the situation is similar to Kleister Charity.

6 Conclusions and further research

We introduced LAMBERT, a layout-aware language model, producing contextualized token embeddings for tokens contained in formatted documents. The model can be trained in an unsupervised manner. For the end user, the only difference with classical language models is the use of bounding box coordinates as additional input. No images are needed, which makes this solution particularly simple, and easy to include in pipelines that are already based on OCR-ed documents.

The LAMBERT model outperforms the baseline RoBERTa on information extraction from visually rich documents, without sacrificing performance on documents with flat layout, as evidenced by the results for the Kleister NDA dataset. Its base variant with around 125M parameters is also able to compete with the large variants of LayoutLM (343M parameters) and LayoutLMv2 (426M parameters), in the case of Kleister and SROIE datasets achieving superior results. In particular, LAMBERT$_{\mathrm{BASE}}$ achieved 1st place in the Key Information Extraction from the SROIE dataset leaderboard [15].

The choice of particular components of LAMBERT is supported by an ablation study including confidence intervals, and is shown to be statistically significant. Another conclusion from this study is that for the visually rich documents the point where no more improvement is attained by increasing the training set has not yet been reached. Thus, LAMBERT's performance can still be improved by simply increasing the unsupervised training set. In the future we are planning to experiment with increasing the model size, and training datasets.

Further research is needed to ascertain the impact of the adapter Layer L on the model performance, as the results of the ablation study were inconclusive. It would also be interesting to understand if the mechanism through which it impacts the results is consistent with the hypotheses formulated in Section 2.2.

References

1. Amazon: Amazon textract. `https://aws.amazon.com/textract/` (accessed November 25, 2019) (2019)
2. Bart, E., Sarkar, P.: Information extraction by finding repeated structure. In: DAS '10 (2010)
3. Cesarini, F., Francesconi, E., Gori, M., Soda, G.: Analysis and understanding of multi-class invoices. IJDAR **6**, 102–114 (2003)
4. Dai, Z., Yang, Z., Yang, Y., Carbonell, J., Le, Q.V., Salakhutdinov, R.: Transformer-XL: Attentive language models beyond a fixed-length context. In: ACL (2019)
5. Denk, T.I., Reisswig, C.: BERTgrid: Contextualized Embedding for 2D Document Representation and Understanding. In: Workshop on Document Intelligence at NeurIPS 2019 (2019)
6. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: Pre-training of deep bidirectional transformers for language understanding. In: NAACL-HLT (2019)
7. Gehring, J., Auli, M., Grangier, D., Yarats, D., Dauphin, Y.N.: Convolutional sequence to sequence learning. In: ICML (2017)
8. Google: Cloud Document Understanding AI. `https://cloud.google.com/document-understanding/docs/` (accessed November 25, 2019) (2019)
9. Graliński, F., Stanisławek, T., Wróblewska, A., Lipiński, D., Kaliska, A., Rosalska, P., Topolski, B., Biecek, P.: Kleister: A novel task for information extraction involving long documents with complex layout. ArXiv **2003.02356** (2020)
10. Gururangan, S., Marasović, A., Swayamdipta, S., Lo, K., Beltagy, I., Downey, D., Smith, N.A.: Don't stop pretraining: Adapt language models to domains and tasks. ArXiv **2004.10964** (2020)
11. Hamza, H., Belaïd, Y., Belaïd, A., Chaudhuri, B.: An end-to-end administrative document analysis system. In: 2008 The Eighth IAPR International Workshop on Document Analysis Systems. pp. 175–182 (2008)
12. Huang, Y., Cheng, Y., Bapna, A., Firat, O., Chen, M., Chen, D., Lee, H., Ngiam, J., Le, Q.V., Wu, Y., Chen, Z.: Gpipe: Efficient training of giant neural networks using pipeline parallelism. In: NeurIPS (2019)
13. Hugging Face: Transformers. `https://github.com/huggingface/transformers` (accessed November 27, 2019) (2019)
14. ICDAR: Competition on Scanned Receipts OCR and Information Extraction. `https://rrc.cvc.uab.es/?ch=13` (accessed February 21, 2021) (2019)

15. ICDAR: Leaderboard of the Information Extraction Task, Robust Reading Competition. https://rrc.cvc.uab.es/?ch=13&com=evaluation&task=3 (accessed April 7, 2020) (2020)
16. Ishitani, Y.: Model-based information extraction method tolerant of ocr errors for document images. Int. J. Comput. Process. Orient. Lang. **15**, 165–186 (2002)
17. Katti, A.R., Reisswig, C., Guder, C., Brarda, S., Bickel, S., Höhne, J., Faddoul, J.B.: Chargrid: Towards understanding 2D documents. In: EMNLP (2018)
18. 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 (2006)
19. Liu, X., Gao, F., Zhang, Q., Zhao, H.: Graph convolution for multimodal information extraction from visually rich documents. In: NAACL-HLT (2019)
20. 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 **1907.11692** (2019)
21. Medvet, E., Bartoli, A., Davanzo, G.: A probabilistic approach to printed document understanding. IJDAR **14**, 335–347 (12 2011)
22. Microsoft: Cognitive Services. https://azure.microsoft.com/en-us/services/cognitive-services/ (accessed November 25, 2019) (2019)
23. Park, S., Shin, S., Lee, B., Lee, J., Surh, J., Seo, M., Lee, H.: CORD: A Consolidated Receipt Dataset for Post-OCR Parsing. In: Document Intelligence Workshop at Neural Information Processing Systems (2019)
24. Peanho, C., Stagni, H., Silva, F.: Semantic information extraction from images of complex documents. Applied Intelligence **37**, 543–557 (12 2012)
25. Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., Liu, P.J.: Exploring the limits of transfer learning with a unified text-to-text transformer. Journal of Machine Learning Research **21**(140), 1–67 (2020)
26. Rahman, W., Hasan, M., Lee, S., Zadeh, A., Mao, C., Morency, L.P., Hoque, E.: Integrating multimodal information in large pretrained transformers. In: ACL (2020)
27. Rusinol, M., Benkhelfallah, T., Poulain d'Andecy, V.: Field extraction from administrative documents by incremental structural templates. In: ICDAR (2013)
28. Shaw, P., Uszkoreit, J., Vaswani, A.: Self-attention with relative position representations. In: NAACL-HLT (2018)
29. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: Advances in Neural Information Processing Systems 30 (2017)
30. Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., Bowman, S.R.: GLUE: A multitask benchmark and analysis platform for natural language understanding. In: Proceedings of ICLR (2019), https://gluebenchmark.com/ (accessed November 26, 2019)
31. 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. arXiv **2012.14740** (2020)
32. 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 & Data Mining. p. 1192–1200 (2020)