Skim-Attention: Learning to Focus via Document Layout

Laura Nguyen [◇⋆] **Thomas Scialom**[◇⋆] **Jacopo Staiano**[⋆] **Benjamin Piwowarski**[◇]
[◇] Sorbonne Université, CNRS, LIP6, F-75005 Paris, France
[⋆] reciTAL, Paris, France
{laura,thomas,jacopo}@recital.ai
benjamin.piwowarski@lip6.fr

Abstract

Transformer-based pre-training techniques of text and layout have proven effective in a number of document understanding tasks. Despite this success, multimodal pre-training models suffer from very high computational and memory costs. Motivated by human reading strategies, this paper presents *Skim-Attention*, a new attention mechanism that takes advantage of the structure of the document and its layout. *Skim-Attention* only attends to the 2-dimensional position of the words in a document. Our experiments show that *Skim-Attention* obtains a lower perplexity than prior works, while being more computationally efficient. *Skim-Attention* can be further combined with long-range Transformers to efficiently process long documents. We also show how *Skim-Attention* can be used off-the-shelf as a mask for any Pre-trained Language Model, allowing to improve their performance while restricting attention. Finally, we show the emergence of a document structure representation in *Skim-Attention*.

1 Introduction

More and more companies have started automating their document processing workflows by leveraging artificial intelligence techniques. This had lead to the emergence of a dedicated research topic, Document Intelligence[1] (DI), which encompasses the techniques used to read, interpret and extract information from business documents. Such documents span multiple pages and contain rich multi-modal information that include both text and layout. Earliest approaches to analyzing business documents rely on rule-based algorithms (Lebourgeois et al., 1992; Amin and Shiu, 2001), but the success of deep learning has put computer vision and natural language processing (NLP) models at the heart of contemporary approaches (Katti et al., 2018; Denk and Reisswig, 2019). With the massive impact of large pre-trained Transformer-based language models (Devlin et al., 2019; Radford et al., 2019), DI researchers have recently started leveraging Transformers.

At the core of the Transformer architecture is self-attention, a powerful mechanism which contextualizes tokens with respect to the whole sequence. While being the key to the success of Transformers, it is also its bottleneck: the time and memory requirements of self-attention grow quadratically with sequence length. As a consequence, only short sequences can be processed (512 tokens or 1,024 at most), making it impossible to capture long-term dependencies. This is an important issue for DI since texts can be very dense and long in business documents. To allow efficient training on very long sequences, there has been growing interest in building model architectures that reduce the memory footprint and computational requirements of Transformers (Dai et al., 2019; Kitaev et al., 2020; Beltagy et al., 2020). This plethora of long-range Transformers lie in one specific research direction: capturing long-range dependencies by reducing the cost of self-attention.

These Transformer architectures all operate on serialized texts, i.e. one-dimensional sequences of words, completely disregarding the document layout. However, layout, i.e. the physical organization of a document's contents, carries useful information about the semantics of the text and has a significant impact on readers' understanding (Wright, 1999). Thus, ignoring the document layout leads to a considerable loss of information. To address this issue, an orthogonal direction that has gained traction recently is based on integrating layout information into Transformer-based language models. Joint pre-training of text and layout has allowed models to reach state-of-the-art performance in several downstream tasks concerning layout-rich documents (Xu et al., 2020b; Pramanik et al., 2020; Xu et al., 2020a). Despite their effec-

[1] https://sites.google.com/view/di2019

Findings of the Association for Computational Linguistics: EMNLP 2021, pages 2413–2427

tiveness, these approaches all "read" the contents token by token to compute attention. We claim that one does not need to have read each word in a document page to be able to understand a specific paragraph. Thus, we argue that, to efficiently process long documents, it is a waste of effort and computation to contextualize a token with respect to the entire input sequence.

To shift towards processing long documents with awareness of their structure, we propose to take into account layout in a more intuitive and efficient way. First, we present a quick cognitive experiment wherein we show that layout plays a fundamental role in humans' comprehension of documents. In light of this experiment, we claim that one can already gather a lot of information from the layout alone. As a consequence, we propose Skim-Attention, a new self-attention mechanism that is solely based on the 2-D position of tokens in the page, independently from their semantics. To exploit this mechanism, we introduce Skimformer and SkimmingMask, two frameworks for integrating Skim-Attention into Transformer models. Skimformer is an end-to-end Transformer language model that replaces self-attention with Skim-Attention. Based on the tokens' spatial locations, Skimformer computes the Skim-Attention scores only once, before using them in each layer of a text-based Transformer encoder. Skimformer can also be adapted to long-range Transformers to model longer documents. Conversely, SkimmingMask uses Skim-Attention as a mask to sparsify attention in any Transformer language model. Each token is restricted to its k most attended tokens, as indicated by Skim-Attention, which allows for a smaller context length.

In summary, our main contributions are as follows:

- We introduce Skim-Attention, a new attention mechanism that leverages layout.
- We design two frameworks for integrating Skim-Attention into Transformer models, and show that they are more time and memory efficient than LayoutLM.
- To the best of our best knowledge, this is the first time layout is considered as a means for reducing the cost of self-attention.

2 Related Work

2.1 Cognitive Background

The layout of a document, which refers to the arrangement and organization of its visual and textual elements, has a significant influence on readers' behavior and understanding (Wright, 1999; Kendeou and Van Den Broek, 2007; Olive and Barbier, 2017). It has been shown that a well-designed layout results in less cognitive effort (Britton et al., 1982; Olive and Barbier, 2017) and facilitates comprehension of the conveyed information by helping identify the document type and its constituents, as well as providing cues regarding relationships between elements (Wright, 1999). Semiotic research assumes that readers scan the document before taking a closer look at certain units (Kress et al., 1996), a claim supported by eye-tracking experiments on newspapers (Leckner, 2012). For all these reasons, layout is a critical element for document understanding, which motivates its integration into modeling. Inspired by these research findings, our work focuses on exploiting layout in a similar fashion as humans, since this can be key to a successful model coping with long and complex documents.

2.2 Long-range Transformers

In the field of natural language processing, Transformers have become the go-to component in the modern deep learning stack. In recent years, there has been a substantial growth in the number of Transformer variants (*long-range Transformers*) that improve computational and memory efficiency, making it possible to extend the maximum sequence length and to incorporate long-term context. Models such as Longformer (Beltagy et al., 2020), Reformer (Kitaev et al., 2020), and Performer (Choromanski et al., 2020) are able to process sequences of thousands of tokens or longer. Although these models are highly efficient in reducing time and memory requirements, they consider long documents as huge one-dimensional blocks of texts: Reformer, for instance, has to read the 4,096 elements contained in the input sequence in order to create buckets of similar elements. Hence, all information about the document structure is lost.

Our approach is orthogonal to long-range Transformers; instead of focusing only on architecture optimization, we propose to leverage layout-rich information.

2.3 Multi-modal Pre-training Techniques for Document Understanding

Recently, multi-modal pre-training techniques have become increasingly popular in the document understanding area (Xu et al., 2020b; Pramanik et al., 2020; Garncarek et al., 2020; Wu et al., 2021). This research direction consists in jointly pre-training on textual and layout/visual information from a large and heterogeneous collection of unlabeled documents, learning cross-modal interactions in an end-to-end fashion. Based on the BERT architecture, Xu et al. (2020b) build LayoutLM, a multi-modal Transformer model that ties spatial information with tokens through a point-wise summation to learn pre-trained embeddings for document understanding tasks.

LayoutLM, along with most approaches in prior-art, is not motivated by efficiency and cognitive perspectives. The layout information is rather considered as an additional feature, and this approach requires to "read" each individual token one by one. As opposed to LayoutLM, in our proposed approach, attention is computed exclusively on spatial positions. This leads to improvements on time and memory efficiency. In addition, our approach can be plugged into any textual language model, making it more flexible than LayoutLM, which requires both text and layout to be learnt jointly in an extensive pre-training stage.

3 Preliminary Experiments: Human Evaluation

How much does the document layout help in comprehending long textual contents? How faster is it for humans to find information in documents when layout is provided? To answer these questions, we conduct a simple cognitive experiment wherein we measure the amount of time needed for human annotators to retrieve information from both formatted and plain-text documents. Half of the time, they are given access to the full layout, and the other half, to plain text only (i.e., no layout nor formatting).[2]

Table 1 reports the average time needed to retrieve information from the documents. We find that it is $2.5\times$ faster to answer questions from the formatted documents, and that the variability in the results is much lower in this case. These results support the hypothesis that *less cognitive effort* is spent when the document is formatted, emphasizing the importance of layout information in reading comprehension.

	Average	Standard Deviation
Formatted	6.05	1.73
Plain-text	15.18	9.06

Table 1: Average (std) time (in seconds) required to answer questions from documents, depending on whether layout is provided.

We believe that machines could benefit from the the document layout, just like humans, as a strategy to retrieve information faster while expending less effort. In particular, layout information could be of great help in reducing the cost of self-attention in Transformer models.

4 Proposed Approach

Using common sense, and in light of the cognitive experiment previously reported, it is clear that the layout is of utmost importance for humans to understand long documents. We propose to take into account layout by introducing *Skim-Attention*, a self-attention module that computes attention solely based on spatial positions. To process long and layout-rich documents, we propose different ways of integrating this mechanism into Transformer architectures.

4.1 Background on Transformers

We first provide an overview of the well-established Transformer, an encoder-decoder architecture composed by stacking a series of Transformer blocks on top of each other. Each block is characterized by a self-attention module. Given an input sequence encoded as a matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$, the operation for a single layer is defined as:

$$\boldsymbol{\alpha} = \text{Softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d}}\right)\mathbf{V} \qquad (1)$$

where $\mathbf{Q}$, $\mathbf{K}$ and $\mathbf{V}$ are the Query, Key and Value matrices obtained by a linear transformation of $\mathbf{X}$. More intuitively, the attention matrix, $\mathbf{A} = \mathbf{Q}\mathbf{K}^\top$, provides text-based similarity scores for all pairs of tokens in the sequence, while each row in $\text{Softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d}}\right)$ represents a distribution that indicates how we need to aggregate information from the input tokens ($\mathbf{V}$) for the corresponding output token ($\mathbf{Q}$).

[2]For additional details regarding the experimental protocol, documents, questions and results, see Section A in the appendix.

It is clear that the main limitation of Transformers lies in the computational and memory requirements of the attention: to obtain the attention matrix, inner products between each key and each query need to be computed, resulting in a quadratic complexity w.r.t. the input sequence length. This operation is repeated at each layer, hence processing longer sequence quickly becomes computationally challenging. Finally, the standard Transformer architecture considers documents as serialized sequences of texts, leading to a severe loss of information when it comes to layout-rich documents.

4.2 Skim-Attention Overview

Our novel attention mechanism, Skim-Attention, views documents as collections of boxes distributed over a two-dimensional space, i.e., the page. In the following, we provide details on how to encode spatial positions into layout embeddings, before describing our attention module.

Layout Embeddings Layout embeddings carry information about the spatial position of the tokens. Following LayoutLM (Xu et al., 2020b), the spatial position of a token is represented by its bounding box in the document page image, (x_0, y_0, x_1, y_1), where (x_0, y_0) and (x_1, y_1) respectively denote the coordinates of the top left and bottom right corners. We discretize and normalize them to integers in $[0, ..., 1000]$. Four embedding tables are used to encode spatial positions: two for the coordinate axes (x and y), and the other two for the bounding box size (width and height). The final layout embedding of a token, $\boldsymbol{\ell} \in \mathbb{R}^{d_\ell}$, located at position (x_0, y_0, x_1, y_1) is defined by:

$$\begin{aligned}\boldsymbol{\ell} = &\text{LayoutEmb}_x(x_0) + \text{LayoutEmb}_y(y_0) \\ &+ \text{LayoutEmb}_x(x_1) + \text{LayoutEmb}_y(y_1) \\ &+ \text{LayoutEmb}_w(x_1 - x_0) \\ &+ \text{LayoutEmb}_h(y_1 - y_0)\end{aligned} \tag{2}$$

Skim-Attention We propose Skim-Attention, an attention mechanism that leverages document layout in a novel way. As opposed to standard self-attention, Skim-Attention does not depend on the text semantics (i.e. token representations), as it calculates the attention using *only* the spatial positions of the tokens, i.e. their layout embeddings $\boldsymbol{\ell}$.

Formally, let $\mathbf{X}^\ell = \{\boldsymbol{\ell}_0, \boldsymbol{\ell}_1, \ldots, \boldsymbol{\ell}_n\}$ be an input sequence of layout embeddings, and $\mathbf{Q}^\ell = \mathbf{W}_q^\ell \mathbf{X}^\ell, \mathbf{K}^\ell = \mathbf{W}_k^\ell \mathbf{X}^\ell$, the Queries and Keys obtained by linear transformations of the layout embeddings. For a single attention head, the Skim-Attention matrix is defined by:

$$\mathbf{A}^\ell = \text{Softmax}\left(\frac{\mathbf{Q}^\ell \left(\mathbf{K}^\ell\right)^\top}{\sqrt{\mathbf{d}^\ell}}\right) \tag{3}$$

Intuitively, $\mathbf{A}^\ell$ captures the correlation between two tokens based on their spatial positions: the more similar two tokens are in terms of layout embeddings, the higher their attention score.

Since attention is calculated only once, we want the layout embeddings to be as meaningful as possible. Therefore, to obtain better layout representations, we contextualize them by adding a small Transformer prior to computing Skim-Attention.

It is possible to combine Skim-Attention with any long-range Transformer, as these approaches are orthogonal. We adapt our approach by computing the corresponding long-range attention only once, based on layout instead of text semantics.

4.3 Skim-Attention in Transformers

We investigate two approaches to exploit Skim-Attention: *i) Skimformer*, wherein self-attention is replaced by Skim-Attention; and *ii) Skimming-Mask*, where an attention mask is built from Skim-Attention and fed to a Transformer language model.

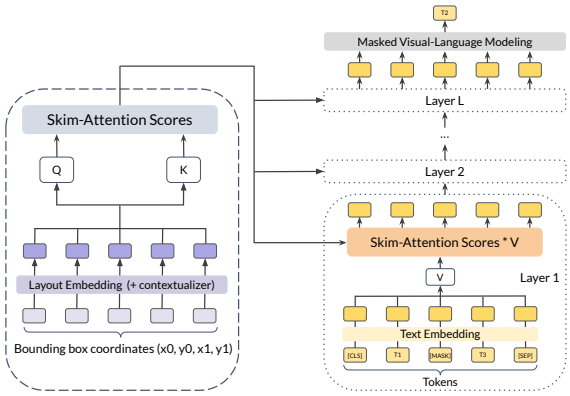
Skimformer is a two-stage Transformer that replaces self-attention with Skim-Attention. Inspired by previous work in cognitive science, the intuition behind this approach is to mimic how humans process a document by *i)* skimming through the document to extract its structure, and *ii)* reading the contents informed by the previous step. Skimformer accepts as inputs a sequence of token embeddings and the corresponding sequence of layout embeddings. The model adopts a two-step approach: first, the skim-attention scores are computed once and only once using layout information alone; then, these attentions are used in every layer of a Transformer encoder. The architecture of Skimformer is depicted in Figure 1a.

For a given encoder layer k and a single head, the traditional self-attention operation becomes:

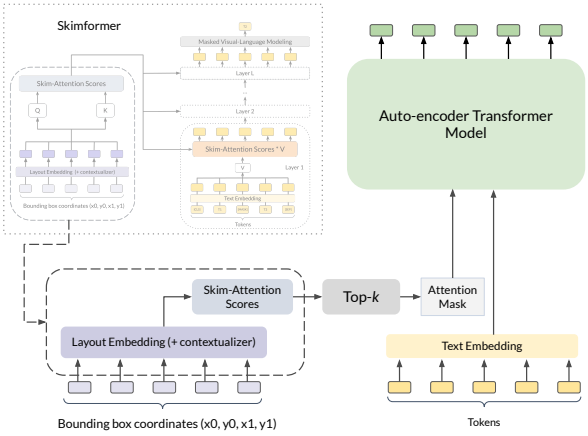
$$\boldsymbol{\alpha}'_k = \mathbf{A}^\ell \mathbf{V}_k^t \tag{4}$$

where $\mathbf{A}^\ell$ is the skim-attention matrix obtained through Eq. 3, and $\mathbf{V}_k^t = \mathbf{W}_{v,k}\mathbf{X}^t$ is the Value matrix produced by projecting the textual input[3]

[3] As opposed to BERT, we do not encode sequential positions into the text embeddings.



(a) Skimformer model architecture. L denotes the number of Transformer encoder layers. **Q** and **K** are the queries and keys obtained by projecting the layout embeddings. **V** represents the values produced by projecting the encoder layers' textual inputs. The attention is solely based on token spatial positions and computed only once. The attention scores are then distributed to each layer of a Transformer encoder.



(b) SkimmingMask model architecture. The layout embeddings, Key and Query projections are initialized from an already pre-trained Skimformer model. By filtering the k most attended tokens for each token, the Skim-Attention scores are then converted to an attention mask and given as input to a text-based Transformer model.

Figure 1: Our proposed model architectures: Skimformer (left) and SkimmingMask (right). Both models take as input a sequence of tokens and a sequence of token bounding box coordinates. The input of each modality is converted to an embedding sequence. Only the layout embeddings are used to compute Skim-Attention.

$\mathbf{X}^t = \{\mathbf{t}_0, \mathbf{t}_1, \ldots, \mathbf{t}_n\}$ at layer k.

More intuitively, computing skim-attention scores (Eq. 3) can be interpreted as *skimming through* the document. Information about the semantics (contained in **V**) is then routed based on these similarity scores. This is done via Eq. 4 and can be seen as *reading* the contents of the document, focusing on the most relevant parts informed by the skim-attention scores.

We train Skimformer using Masked Visual-Language Modeling (MVLM), a pre-training task that extends Masked Language Modeling (MLM) with layout information. MVLM randomly masks some of the input tokens but preserves their layout embeddings. The model is then trained to recover the masked tokens given the contexts. Hence, MVLM helps capture nearby token features, leveraging both semantics and spatial information.

While we experimented with a standard Transformer model, it is worth noting that any language model can be used as the backbone of Skimformer.

SkimmingMask For each token in a sequence, Skim-Attention provides a ranking of the other tokens based on their layout-based similarity. Leveraging this, SkimmingMask uses Skim-Attention as a mask to restrict the computation of self-attention to a smaller number of elements for each token. In this setting, Skim-Attention is viewed as an independent, complementary module that can be plugged into any language model. Given a sequence of layout embeddings, the corresponding skim-attention matrix is converted to an attention mask: based on the similarity scores provided in the attention matrix, each token can only attend to its k most similar tokens. The resulting mask is then given as input to a text-based Transformer language model with standard self-attention, and is used to restrict self-attention for each element in the input text sequence. This can be viewed as sparsifying the standard self-attention matrix.

SkimmingMask is not trainable end-to-end with the Transformer model it is plugged to, as creating an attention mask from an attention matrix is not a differentiable operation (we leave this for future work). Thus, to train this model, the weights for Skim-Attention need to be already trained, and we naturally use the Skimformer weights. The overall architecture of the model is illustrated in Figure 1b.

We note that SkimmingMask is a new way to cluster tokens: all tokens belonging to the same group have a high similarity to each other regarding their respective *layout position*. This makes SkimmingMask a concurrent approach to Reformer, which reduces the cost of self-attention by clustering tokens into chunks. As opposed to the latter, the concept of similarity is not based on text semantics but on the document structure. Moreover,

SkimmingMask does not require the semantic of each token, but only their layout features. Because each token is viewed as a bounding box whose characteristics are only its size and position, the representation space of layout features is much smaller than that of the text, which spans a vocabulary of more than 30k sub-words. As a consequence, computing attention based on layout could require a smaller latent space dimension than for text, corresponding to less computational efforts. This is also the case for humans: as demonstrated in section 3, it is much easier to retrieve information from documents when the layout is provided.

5 Experiments

5.1 Data

Pre-training Data To pre-train our models on a wide variety of document formats, we select three datasets with various non-trivial document layouts: DocBank (Li et al., 2020), RVL-CDIP (Harley et al., 2015) and PubLayNet (Zhong et al., 2019). We combine them by randomly selecting 25k documents from each dataset, for a total of 75K documents. We discard the provided labels and consider these data as unannotated. The resulting dataset is referred to as MIX. As a first evaluation metric, we can compare the perplexity for the different language models on MIX.

DocBank DocBank is a large-scale dataset that contains 500K English document pages from papers extracted from arXiv.com. These articles span a variety of disciplines (e.g. Physics, Mathematics, and Computer Science), which is beneficial to train more robust models. Pages are split into a training set, validation set and test set with a ratio of 8:1:1. As the authors already extracted the text and bounding boxes using PDFPlumber,[4] there is no need for an OCR system or a PDF parser. To build our subset, we extract 25k document pages: 20k from the full training set, 2,500 from the validation set and 2,500 from the test set.

RVL-CDIP RVL-CDIP is a large collection of 400k scanned document images from various categories (e.g. letter, form, advertisement, invoice). The wide range of layouts, as well as the low image quality, allows to train more robust models. We select 25k documents from the RVL-CDIP dataset available on Kaggle,[5] which amounts to half of the training images from the full dataset (160k images). The text and word bounding boxes are extracted using Tesseract.[6] We split the data into 80% for training, 10% for validation and 10% for test.

PubLayNet PubLayNet comprises over 360 thousand document images from PubMed Central™ Open Access. The medical publications contained in the collection have similar layouts, but the text density coupled with the small image size add to the robustness of the trained models. We extract the first training split among the 7 available on IBM Data Asset eXchange[7] and use the first 20k images as our training set. For the validation and test sets, we keep the first 2,500 images in each split. Because OCR accuracy is too low without any preprocessing, we apply a few image processing operations (i.e. rescaling, converting to grayscale, applying dilation and erosion) on each image in order to improve text extraction.

Dataset for Document Layout Analysis In addition to perplexity, we evaluate our approach on a downstream task, document layout analysis. Document layout analysis consists in associating each token with its corresponding category: abstract, author, caption, date, equation, footer, list, paragraph, reference, section, table, title and figure.[8]

We use a subset of the full DocBank dataset, created by selecting 10k document pages (distinct from the ones used for pre-training): 8,000 from the full training set, 1,000 from the validation set and 1,000 from the test set. We refer to this dataset as *DocBank-LA*. Each document page is organized as a list of words with bounding boxes, colors, fonts and labels. We use the precision, recall and F1 score defined by Li et al. (2020).

5.2 Experimental Settings

For reproducibility purposes, we make the code publicly available.[9]

Baselines We compare our models with three baselines: i) the text-only BERT, ii) the multimodal LayoutLM, and iii) the text-only Longformer for long documents. Note that the LayoutLM architecture is based on BERT, with addi-

[4] https://github.com/jsvine/pdfplumber

[5] https://www.kaggle.com/nbhativp/first-half-training

[6] https://github.com/tesseract-ocr/tesseract

[7] https://developer.ibm.com/exchanges/data/all/publaynet/

[8] We actually discard the *Figure* label, as 1) our models do not take image features into account, and 2) the text associated with such elements is always the same, making the task trivial.

[9] https://github.com/recitalAI/skim-attention

Model	Test Perplexity
BERT (Devlin et al., 2019)	357.11
LayoutLM (Xu et al., 2020b)	45.86
Skimformer	33.77
Longformer (Beltagy et al., 2020)	333.28
LongSkimformer	**32.02**

Table 2: Test perplexity on the MIX dataset after 10k optimization steps. Each model was trained from scratch. Bold denotes the best score.

tional layout components. For fair comparison, all our models designed for short sequences are based on BERT as well, as detailed below.

Pre-training For BERT, LayoutLM and Longformer, we use their default architecture. Following the BERT base model, Skimformer consists of a 12-layer Transformer encoder with 12 attention heads and a hidden size set to 768 for both text and layout embeddings, amounting to 99M parameters. We further add a 2-layer Transformer encoder to contextualize the layout embeddings, which increases the number of parameters to 113M. To test Skim-Attention on longer documents, we build LongSkimformer, a combination of Skim-Attention and Longformer. Every model is trained from scratch on the MIX dataset for 10k steps. We set the maximum sequence length to $n = 512$ for every model except for Longformer and LongSkimformer, for which $n = 2,048$. Skimformer, LongSkimformer and LayoutLM are pre-trained using MVLM, while BERT and Longformer are pre-trained with MLM. For more implementation details, see Section B in the appendix.

Document Layout Analysis As DocBank contains fine-grained token-level annotations, we consider the document layout analysis task as a sequence labeling task. Each model pre-trained on MIX is fine-tuned on this downstream task for 10 epochs. Section B of the appendix provides a detailed description of the settings used. For the SkimmingMask models, we selected the hyperparameter k on validation, i.e. the number of tokens that can be attended to.[10]

5.3 Results and Discussion

5.3.1 Perplexity

In Table 2, we report the perplexity on the MIX dataset. We observe that Skimformer and

[10]We tested $k \in [512, 384, 256, 128]$.

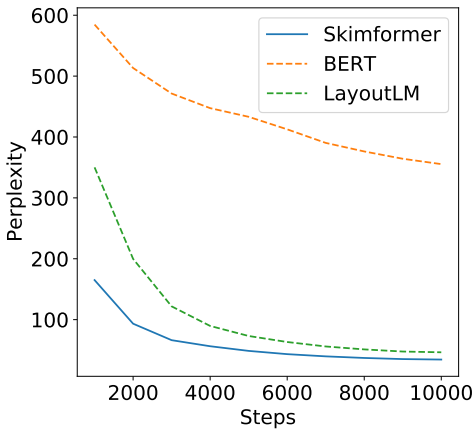


Figure 2: Model perplexity on the MIX validation set with respect to the number of optimization steps. All models are trained from scratch.

Skim-Attention Input	Test Perplexity
Layout	36.41
1D position	54.39
Uniform layout	421.97
Degraded layout	103.39
Contextualized layout	**33.77**

Table 3: Ablation study on the MIX dataset, where perplexity on the test set is reported. All models were trained from scratch. Bold denotes the best score.

LongSkimformer respectively outperform BERT and Longformer by a huge margin, while improving perplexity by more than 10 points over LayoutLM. In addition, Figure 2 demonstrates that Skimformer converges much faster than BERT, and slightly more than LayoutLM.

Ablation Study We further conduct an ablation study about the influence of the Skim-Attention inputs on Skimformer's performance. The results are listed in Table 3. To estimate the impact of the input type, we consider a Skimformer model i) wherein Skim-Attention is based on sequential positions (1D position), ii) the bounding boxes are all set to the same fixed value, preventing the model to gather any information about the true location (Uniform layout), iii) they are replaced by their centers (Degraded layout), and iv) the layout embeddings are contextualized (Contextualized Layout).

We can see that replacing spatial with sequential positions results in an increase in perplexity, indicating that layout information is crucial for the Language Model. It is also observed that assigning the same bounding box to every token leads to a severe drop in performance. Coupled with the perplexity obtained with a degraded layout, this shows

that the model's performance is greatly impacted by the layout input quality. At last, contextualizing the layout inputs through a small Transformer brings slight improvements over computing Skim-Attention directly on the layout embeddings.

Finally, we benchmark Skimformer and LayoutLM on both speed and peak memory usage for training. Results provided in Figure 5 of the appendix show that Skimformer is more time and memory efficient than LayoutLM.

5.3.2 Document Layout Analysis

Table 4 reports the performance on DocBank-LA, the sequence length processed, the number of times attention is computed and the ratio of the total calculation unit ($n^2 \times$ Nb Skim-Attn + Seq. Len$^2 \times$ Nb Standard Attn, where n is the length of the initial sequence on which Skim-Attention is applied; and *Seq. Len* is the length obtained after applying SkimmingMask) to that of BERT/LayoutLM and Longformer. All models were pre-trained from scratch on MIX.

Skimformer is substantially superior to BERT, improving the F1 score by 15% while reducing the number of attentions computed by four. We experimented with plugging the layout embeddings learnt by Skimformer in a BERT model. The resulting model, BERT+SkimEmbeddings, resembles LayoutLM in terms of architecture.[11] Results show that BERT+SkimEmbeddings performs on par with LayoutLM despite simply combining separately pre-trained modalities, as opposed to the latter which requires an extensive joint training.

For the SkimmingMask models (see the last two rows in Table 4), the models attend to only the top-k 128 tokens. Compared to LayoutLM, this reduction to the quadratic factor allows to obtain the same downstream results with only 31.25% of the computational burden. Compared to BERT, it even obtains an absolute improvement of more than 6% in term of F1 score.

LongSkimformer benefits from both Skim-Attention and Longformer's gain in efficiency. It outperforms Longformer by 5% while requiring four times less attention operations, and the use of Longformer's linear attention allows LongSkimformer to process sequences four times larger than Skimformer can.

[11] In BERT+SkimEmbeddings, the layout embeddings are first projected into the same dimensional space as the text embeddings. In this way, we can plug the layout embeddings from any Skimformer model, in particular smaller ones.

5.4 Attention Visualization

Figure 3 shows the attention maps produced by Skimformer on a sample document.[12] Given a semantic unit (either title or abstract in our example), we select the corresponding tokens and compute their average attention over the whole document. We observe, both qualitatively and quantitatively, that tokens attend mainly to other elements in the same semantic unit, thus creating clusters of tokens that are relevant to each other. This shows that the model has grasped the concept of semantic unit with only self-supervision, enabling the emergence of a document structure representation. We argue that these structure-aware clusters could pave the way for long text encoding and unsupervised document segmentation.

6 Conclusion

We present Skim-Attention, a new structure-aware attention mechanism. We conduct extensive experiments to show the effectiveness of Skim-Attention, both as an end-to-end model (Skimformer) and as a mask for any language model (Skim-Attention). We hope this work will pave the way towards a new research direction for efficient attentions. For future works, we will investigate how to integrate image features, and explore tasks that require capturing longer-range dependencies.

7 Acknowledgments

We thank the reviewers for their insightful comments. This work is supported by the Association Nationale de la Recherche et de la Technologie (ANRT) under CIFRE grant N2020/0916. It was partially performed using HPC resources from GENCI-IDRIS (Grant 2021-AD011011841).

References

Adnan Amin and Ricky Shiu. 2001. Page segmentation and classification utilizing bottom-up approach. *International Journal of Image and Graphics*, 1(02):345–361.

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Bruce K Britton, Shawn M Glynn, Bonnie J Meyer, and MJ Penland. 1982. Effects of text structure on use of cognitive capacity during reading. *Journal of Educational Psychology*, 74(1):51.

[12] Randomly picked. Similar results can be observed across different samples (see Section 6 in the appendix).

Model	Skimming Mask	Seq. Len	Nb Attentions Original*	Nb Attentions Skim-Attn	Total Compute	Rec.	Prec.	F1
BERT (Devlin et al., 2019)	✗	512	12	0	100.00%	67.21	59.28	60.98
LayoutLM (Xu et al., 2020b)	✗	512	12	0	100.00%	81.60	77.96	**79.28**
Skimformer	✗	512	0	3**	25.00%	78.80	74.35	75.86
BERT+SkimEmbeddings	✗	512	12	0	100.00%	**82.42**	77.06	**79.16**
BERT+SkimmingMask	✓	128	12	3**	31.25%	72.32	64.39	67.36
LayoutLM+SkimmingMask	✓	128	12	3**	31.25%	81.15	**78.30**	**79.26**
Longformer (Beltagy et al., 2020)	✗	2,048	12	0	100%	74.88	69.29	71.17
LongSkimformer	✗	2,048	0	3**	25%	81.22	73.45	76.61

* Standard self-attention for Skimformer, BERT-based and LayoutLM-based models. Longformer self-attention for Longformer and LongSkimformer.

** Attention is computed twice (by a 2-layer Transformer) during layout contextualization, then once by Skim-Attention.

Table 4: Model performance (in %) on the DocBank-LA dataset. *Seq. Len* indicates the number of tokens attended with either standard attention (for Skimformer, BERT-based and LayoutLM-based models), or Longformer attention (for Longformer and LongSkimformer). *Nb Attention* represents the number of times attention (original and Skim-Attention) is computed and stored. *Total Compute* specifies the ratio of the final computational cost (# operations needed to compute attention) w.r.t. BERT/LayoutLM or Longformer. Each model was pre-trained from scratch on MIX, then fine-tuned on DocBank-LA.

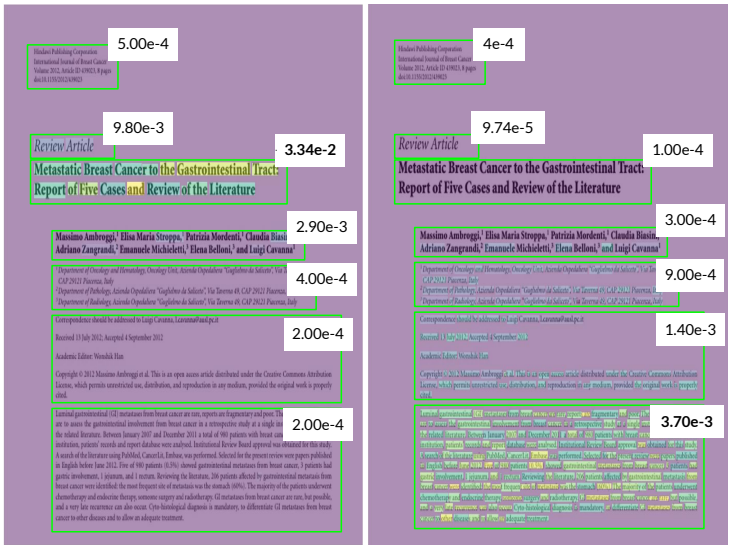


Figure 3: Skim-attention maps corresponding to the title (left) and the abstract (right), along with average attention score (in white) per text block (in green). We consider the skim-attention matrix averaged over all the attention heads. Given a semantic unit (title or abstract), we plot the average attention score for each token.

Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, et al. 2020. Rethinking attention with performers. *arXiv preprint arXiv:2009.14794*.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988, Florence, Italy. Association for Computational Linguistics.

Timo I Denk and Christian Reisswig. 2019. Bertgrid: Contextualized embedding for 2d document representation and understanding. *arXiv preprint arXiv:1909.04948*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. 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)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Łukasz Garncarek, Rafał Powalski, Tomasz Stanisławek, Bartosz Topolski, Piotr Halama, and Filip Graliński. 2020. Lambert: Layout-aware language modeling using bert for information extraction. *arXiv preprint arXiv:2002.08087*.

Adam W Harley, Alex Ufkes, and Konstantinos G Derpanis. 2015. Evaluation of deep convolutional nets for document image classification and retrieval. In *2015 13th International Conference on Document Analysis and Recognition (ICDAR)*, pages 991–995. IEEE.

Anoop Raveendra Katti, Christian Reisswig, Cordula Guder, Sebastian Brarda, Steffen Bickel, Johannes Höhne, and Jean Baptiste Faddoul. 2018. Chargrid: Towards understanding 2d documents. *arXiv preprint arXiv:1809.08799*.

Panayiota Kendeou and Paul Van Den Broek. 2007. The effects of prior knowledge and text structure on comprehension processes during reading of scientific texts. *Memory & cognition*, 35(7):1567–1577.

Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. 2020. Reformer: The efficient transformer. *arXiv preprint arXiv:2001.04451*.

Gunther R Kress, Theo Van Leeuwen, et al. 1996. *Reading images: The grammar of visual design*. Psychology Press.

Frank Lebourgeois, Zbigniew Bublinski, and Hubert Emptoz. 1992. A fast and efficient method for extracting text paragraphs and graphics from unconstrained documents. In *11th IAPR International Conference on Pattern Recognition. Vol. II. Conference B: Pattern Recognition Methodology and Systems*, volume 1, pages 272–273. IEEE Computer Society.

Sara Leckner. 2012. Presentation factors affecting reading behaviour in readers of newspaper media: an eye-tracking perspective. *Visual Communication*, 11(2):163–184.

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

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Thierry Olive and Marie-Laure Barbier. 2017. Processing time and cognitive effort of longhand note taking when reading and summarizing a structured or linear text. *Written Communication*, 34(2):224–246.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *arXiv preprint arXiv:1912.01703*.

Subhojeet Pramanik, Shashank Mujumdar, and Hima Patel. 2020. Towards a multi-modal, multi-task learning based pre-training framework for document representation learning. *arXiv preprint arXiv:2009.14457*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Patricia Wright. 1999. The psychology of layout: Consequences of the visual structure of documents. *American Association for Artificial Intelligence Technical Report FS-99-04*, pages 1–9.

Te-Lin Wu, Cheng Li, Mingyang Zhang, Tao Chen, Spurthi Amba Hombaiah, and Michael Bendersky.

2021. Lampret: Layout-aware multimodal pretraining for document understanding. *arXiv preprint arXiv:2104.08405*.

Yang Xu, Yiheng Xu, Tengchao Lv, Lei Cui, Furu Wei, Guoxin Wang, Yijuan Lu, Dinei Florencio, Cha Zhang, Wanxiang Che, et al. 2020a. Layoutlmv2: Multi-modal pre-training for visually-rich document understanding. *arXiv preprint arXiv:2012.14740*.

Yiheng Xu, Minghao Li, Lei Cui, Shaohan Huang, Furu Wei, and Ming Zhou. 2020b. 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*, pages 1192–1200.

Xu Zhong, Jianbin Tang, and Antonio Jimeno Yepes. 2019. Publaynet: largest dataset ever for document layout analysis. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 1015–1022. IEEE.

Skim-Attention: Learning to Focus via Document Layout – Appendix

A Preliminary Experiments: Human Evaluation

To evaluate the impact of layout on readers' understanding, we conduct an experiment in which we measure the amount of time required for human annotators to answer questions from both formatted and non-formatted documents. We hand-pick four document pages from the DocBank dataset (Li et al., 2020), and create a plain-text version out of each of these documents by flattening them. This results in eight pages: the four original document pages, and their serialized versions with no layout nor formatting. The original, formatted document pages are displayed in figure 4. We create two basic questions for each document (answers are provided in italic):

- Document (a) :
  - Who are the authors of this paper ? *E.C Merkle, D. Furr, S. Rabe-Hesketh.*
  - What are the keywords ? *Bayesian information criteria, conditional likelihood, cross-validation, DIC, IRT, leave-one-cluster out, marginal likelihood, MCMC, SEM, WAIC.*
- Document (b) :
  - What paper did N.D. Tracas and P.M. Zerwas write ? *e + e – Colliders: The Window To Z's Beyond The Total Energy.*
  - Who does the author thank ? *The organizers, those who contributed to the content of the discussion (J. Bagger, M. Berggren, J. Kanlinowski, W. Kilian, J. List, J. Mnich, M. Peskin, F. Richard, G. Wilson), P. Zerwas.*
- Document (c) :
  - What is proposed in this paper ? *A Reinforced Neural Extractive Summarization model to extract a coherent and informative summary from a single document.*
  - What is compared in table 3 ? *Human evaluation in terms of informativeness(Inf), coherence(Coh) and overall ranking.*
- Document (d) :
  - When was this paper submitted ? *May 2028, 2020.*
  - What are the keywords of this paper ? *Touchscreen keyboards, gesture input, model-based design, Monte Carlo simulation.*

Four annotators are asked to answer these questions. Each of them alternates between fully formatted contents (i.e. the original document page) and plain text. We decide that annotators 1 and 3 have access to the document layout for documents 1 and 3, while annotators 2 and 4, for documents 2 and 4.

Given a document, the instructions are as follows:

1. Read the entire document, then the questions;
2. Start the timer;
3. Find the answer to the first question (without writing it down);
4. Stop the timer and check if the answer is correct;
   - If this is the case, write down the time indicated by the timer, then reset it and answer the second question by re-iterating steps 2 to 4.
   - If not, resume timer until you find the correct answer.
5. Proceed to the next document.

	Formatted	Plain-text
Doc 1	2.95 ± 0.43	10.85 ± 5.67
Doc 2	4.29 ± 1.54	14.44 ± 11.25
Doc 3	7.75 ± 0.38	20.81 ± 10.26
Doc 4	9.20 ± 4.57	14.65 ± 9.06

Table 5: Time (in seconds) required to retrieve information per document and document type (formatted/non-formatted). Standard deviation is also reported.

Results per document and document type (i.e., formatted or non-formatted) are given in table 5. The entirety of the results is reported in table 6.

B Implementation Details

Pre-training For BERT, LayoutLM and Longformer, we use the PyTorch implementation from Hugging Face's Transformers library (Wolf et al., 2020).

from the left- and right-handed couplings extracted from forward-backward asymmetries and charge asymmetries in two-fermion processes, different high-scale models can be discriminated (cf. e.g. [25]).

One final remark: if something similar like the 2 TeV anomaly in $WW/WZ/ZZ$ at the end of the 8 TeV run or the 750 GeV anomaly in diphotons will remain at the end of run II or the high-lumi run, then the ILC is the only option in the near future to comfirm or refute such a signal.

3 Summary

In this talk I tried to collect the facts in favor of a future high-energy lepton collider (that is capable to reach at least 500 GeV) with the focus lying on new physics beyond the SM. Both the two main SM pillars, the Higgs boson and top quark measurements serve as indirect tools for new physics searches, but there is also a plethora of direct search opportunities at such a machine. Most prominent examples are dark matter searches, searches for other light weakly coupling particles, and a scan over all weakly interacting particles. The interplay of the ILC with the LHC, but more importantly with future hadron machines is elucidated. Conditions, or better, scenarios for possible BSM discoveries at the ILC have been given. Several prime examples for the BSM potential of the ILC have been highlighted.

Acknowledgments

JRR wants to thank the organizers for a fantastic conference in the Canadian wilderness. Many thanks who contributed the content of this discussion, among them are J. Bagger, M. Berggren, J. Kanlinowski, W. Kilian, J. List, J. Mnich, M. Peskin, F. Richard, G. Wilson. Special thanks go to P. Zerwas for guiding all of us in the field of lepton collider physics over decades.

References

[1] K. Hagiwara and K. Hidaka, *Physics at TeV Energy Scale. Proceedings, Meeting, Tsukuba, Japan, May 28-30, 1987*, KEK-87-20; K. Hagiwara and S. Komamiya, *Search For New Particles at e^+e Colliders*, Adv. Ser. Direct. High Energy Phys. **1**, 785 (1988).

[2] N. D. Tracas and P. M. Zerwas, *e^+e^- Colliders: The Window To Z's Beyond The Total Energy*, In *La Thuile/Geneva 1987, Proceedings, Physics at future accelerators, vol. 2* 214-219.

[3] M. E. Peskin, *Theory of e^+e^- collisions at very high energy*, Conf. Proc. C **8708101**, 1 (1987).

[4] F. Richard, J. R. Schneider, D. Trines and A. Wagner, hep-ph/0106314.

[5] J. A. Aguilar-Saavedra *et al.* [ECFA/DESY LC Physics Working Group Collaboration], hep-ph/0106315.

(b)

Bayesian model assessment: Use of conditional vs marginal likelihoods

E. C. Merkle
University of Missouri

D. Furr and S. Rabe-Hesketh
University of California, Berkeley

Abstract

Typical Bayesian methods for models with latent variables (or random effects) involve directly sampling the latent variables along with the model parameters. In high-level software code for model definitions (using, e.g., BUGS, JAGS, Stan), the likelihood is therefore specified as conditional on the latent variables. This can lead researchers to perform model comparisons via conditional likelihoods, where the latent variables are considered model parameters. In other settings, typical model comparisons involve marginal likelihoods where the latent variables are integrated out. This distinction is often overlooked despite the fact that it can have a large impact on the comparisons of interest. In this paper, we clarify and illustrate these issues, focusing on the comparison of conditional and marginal Deviance Information Criteria (DICs) and Watanabe-Akaike Information Criteria (WAICs) in psychometric modeling. The conditional/marginal distinction corresponds to whether the model should be predictive for the clusters that are in the data or for new clusters (where "clusters" typically correspond to higher-level units like people or schools). Correspondingly, we show that marginal WAIC corresponds to leave-one-cluster out (LOcO) cross-validation, whereas conditional WAIC corresponds to leave-one-unit (LOuO). These results lead to recommendations on the general application of these criteria to models with latent variables.

Keywords: Bayesian information criteria, conditional likelihood, cross-validation, DIC, IRT, leave-one-cluster out, marginal likelihood, MCMC, SEM, WAIC.

The research reported here was supported by NSF grant 1460719 and by the Institute of Education Sciences, U.S. Department of Education, through Grant R305D140037. Code to replicate the results from this paper can be found at `http://semtools.r-forge.r-project.org/`. Correspondence to Edgar Merkle, email: merklee@missouri.edu.

(a)

maries extracted by RNES are of higher quality than summaries produced by previous works.

Table 2: Performance comparison on CNN/Daily Mail test set, evaluated with full-length F1 ROUGE scores (%). All scores of RNES are statistically significant using 95% confidence interval with respect to previous best models.

Model	R-1	R-2	R-L
Lead-3	39.2	15.7	35.5
(Nallapati et al. 2016)	35.4	13.3	32.6
(Nallapati et al. 2017)	39.6	16.2	35.3
(See et al. 2017)	39.53	17.28	35.38
NES	37.75	17.04	33.92
RNES w/o coherence	**41.25**	**18.87**	**37.75**
RNES w/ coherence	40.95	18.63	37.41

Though RNES with the coherence reward achieves higher ROUGE scores than baselines, there is a small gap between its score and that of RNES trained without coherence model. This is because that the coherence objective and ROUGE score do not always agree with each other. Since ROUGE is simply computed based on n-grams or longest common subsequence, it is ignorant of the coherence between sentences. Therefore, enhancing coherence may lead to a drop of ROUGE. However, the 95% confidence intervals of the two RNES models overlap heavily, indicating that their difference in ROUGE is insignificant.

Table 3: Comparison of human evaluation in terms of informativeness(Inf), coherence(Coh) and overall ranking. Lower is better.

Model	Inf	Coh	Overall
RNES w/o coherence	1.183	1.325	1.492
RNES w/ coherence	**1.125**	**1.092**	**1.209**

We also conduct a qualitative evaluation to find out whether the introduction of coherence reward improves the coherence of the output summaries. We randomly sample 50 documents from the test set and ask three volunteers to evaluate the summaries extracted by RNES trained with or without coherence as the reward. They are asked to compare and rank the outputs of two models regarding three aspects: informativeness, coherence and overall quality. The better one will be given rank 1, while the other will be given rank 2 if it is worse. In some cases, if the two outputs are identical or have the same quality, the ranks could be tied, i.e., both of them are given rank 1. Table 3 shows the results of human evaluation. RNES model trained with coherence reward is better than RNES model without coherence reward in all three aspects, especially in the coherence. The result indicates that the introduction of coherence effectively improves the coherence of extracted summaries, as well as the overall quality. It is surprising that summaries produced by RNES with coherence are also more informative than RNES without coherence, indicating that ROUGE might not be the gold standard to evaluate informativeness as well.

Table 4 shows a pair of summary produced by RNES with or without coherence. The summary produced by RNES without coherence starts with pronoun 'That' which is referring to a previously mentioned fact, and hence it may lead to confusion. In contrast, the output of RNES trained with coherence reward includes the sentence "The earthquake disaster . . . " before referring to this fact in the second sentence, and therefore is more coherent and readable. This is because the coherence model gives a higher score to the second sentence if it can form a coherent sentence pair with the first sentence. In REINFORCE training, if the second sentence receives a high coherence score, the action of extracting the first sentence before the second one will be strengthened. This example shows that coherence model is indeed effective in changing the behavior of RNES towards extracting summaries that are more coherent.

Table 4: Examples of extracted summary.

Reference: Peter Spiaks from the Sydney Morning Herald reported on Amasia. Within 200 million years, he said the new supercontinent will form. One researcher recently travelled to Nepal to gather further information. He spotted that India, Eurasia and other plates are slowly moving together.
RNES w/o coherence: That's according to one researcher who travelled to the country to study how the Indian and Eurasian plates are moving together. And using new techniques, researchers can now start examining the changes due to take place over the next tens of millions of years like never before. Earth's continents are slowly moving together, and in 50 to 200 million years they are expected to form a new supercontinent called Amasia. In 2012 a study suggested this may be centered on the North Pole. The idea that Earth is set to form a new supercontinent-dubbed Amasia - is not new.
RNES w/ coherence: **The earthquake disaster in Nepal has highlighted how Earth's land masses are already in the process of forming a new supercontinent. That**'s according to one researcher who travelled to the country to study how the Indian and Eurasian plates are moving together. And using new techniques, researchers can now start examining the changes due to take place over the next tens of millions of years like never before. Earth's continents are slowly moving together, and in 50 to 200 million years they are expected to form a new supercontinent called Amasia.

Conclusion

In this paper, we proposed a Reinforced Neural Extractive Summarization model to extract a coherent and informative summary from a single document. Empirical results show that the proposed RNES model can balance between the cross-sentence coherence and importance of the sentences effectively, and achieve state-of-the-art performance on the benchmark dataset. For future work, we will focus on improving the performance of our neural coherence model and introducing human knowledge into the RNES.

Acknowledgments

This work is supported by grants from WeChat-HKUST Joint Lab on Artificial Intelligence Technology (WHAT Lab). Baotian Hu acknowledges partial support from the University of Massachusetts Medical School.

(c)

A Monte Carlo Simulation Approach for Quantitatively Evaluating Keyboard Layouts for Gesture Input

Rylan T. Conway[a], Evan W. Sangaline[b]

[a]*Physics Department, University of California, Davis, CA*
[b]*National Superconducting Cyclotron Laboratory, Michigan State University, East Lansing, MI*

Abstract

Gesture typing is a method of text entry that is ergonomically well-suited to the form factor of touchscreen devices and allows for much faster input than tapping each letter individually. The QWERTY keyboard was, however, not designed with gesture input in mind and its particular layout results in a high frequency of gesture recognition errors. In this paper, we describe a new approach to quantifying the frequency of gesture input recognition errors through the use of modeling and simulating realistically imperfect user input. We introduce new methodologies for modeling randomized gesture inputs, efficiently reconstructing words from gestures on arbitrary keyboard layouts, and using these in conjunction with a frequency weighted lexicon to perform Monte Carlo evaluations of keyboard error rates or any other arbitrary metric. An open source framework, Dodona, is also provided that allows for these techniques to be easily employed and customized in the evaluation of a wide spectrum of possible keyboards and input methods. Finally, we perform an optimization procedure over permutations of the QWERTY keyboard to demonstrate the effectiveness of this approach and describe ways that future analyses can build upon these results.

Keywords: touchscreen keyboards, gesture input, model-based design, Monte Carlo simulation

1. Introduction

The advent of smartphones and tablets has made the use of touchscreen keyboards pervasive in modern society. However, the ubiquitous QWERTY keyboard was not designed with the needs of a touchscreen keyboard in mind, namely accuracy and speed. The introduction of gesture or stroke-based input methods significantly increased the speed that text could be entered on touchscreens [Montgomery (1982); Zhai and Kristensson (2003); Zhai et al. (2009); Kushler and Marsden (2006)]. However, this method introduces some new problems that can occur when the gesture input patterns for two words are too similar, or sometimes completely ambiguous, leading to input errors. An example gesture input error is illustrated in Figure 1. A recent study showed that gesture input has an error rate that is about 5-10% higher compared to touch typing [Bi et al. (2013)]. With the fast and inherently imprecise nature of gesture input the prevalence of errors is unavoidable and the need to correct these errors significantly slows down the rate of text entry. The QWERTY keyboard in particular is poorly suited as a medium for swipe input. Characteristics such as the "u", "i", and "o" keys being adjacent lead to numerous gesture ambiguities and potential input errors. It is clearly not the optimal layout for gesture input.

(d)

Figure 4: Documents selected for our preliminary cognitive experiment.

	Doc 1			Doc 2			Doc 3			Doc 4		
	Q1	**Q2**	**AVG**	**Q1**	**Q2**	**AVG**	**Q1**	**Q2**	**AVG**	**Q1**	**Q2**	**AVG**
A 1	1.48	3.81	2.65	19.52	25.27	22.40	11.91	4.12	8.02	12.98	29.12	21.05
A 2	9.23	4.44	6.84	5.12	5.63	5.38	24.5	31.62	28.06	7.02	4.92	5.97
A 3	3.76	2.76	3.26	7.19	5.77	6.48	9.19	5.77	7.48	3.93	12.55	8.24
A 4	8.66	21.05	14.86	3.49	2.9	3.20	15.9	11.2	13.55	6.9	17.96	12.43

Table 6: Time (in seconds) taken by each annotator to answer each question. Average per document is also reported. Yellow cells indicate that the document layout was provided for corresponding documents and annotators.

Each model is trained from scratch on the MIX dataset for 10k steps with a batch size of 8, except for Longformer which was trained with a smaller batch size of 4 due to memory limitations. We use the Adam optimizer with weight decay fix (Loshchilov and Hutter, 2017), a weight decay of 0.01 and $(\beta_1, \beta_2) = (0.9, 0.999)$. The learning rate is set to $1e^{-4}$ and linearly warmed up over the first 100 steps. The maximum sequence length is set to $n = 512$, with the exception of Longformer and LongSkimformer, for which $n = 2,048$. Following BERT, we mask 15% of the text tokens in MVLM, among which 80% are replaced by a special token [MASK], 10% are replaced by a random token, and 10% remains the same.

Document Layout Analysis Each model pre-trained on MIX is fine-tuned on DocBank's document layout analysis task for 10 epochs, with a learning rate of $5e^{-5}$ and a batch size of 8 (except for Longformer which was fine-tuned with a batch size of 4). The models are extended with a token-classification head on top, consisting of a linear layer followed by a softmax layer, and are trained using cross-entropy.

For SkimmingMask, we select the Skim-Attention module from the Skimformer model pre-trained from scratch on MIX. We then plug it into a BERT model, also pre-trained from scratch on MIX. The resulting model is fine-tuned with the same settings as described previously.

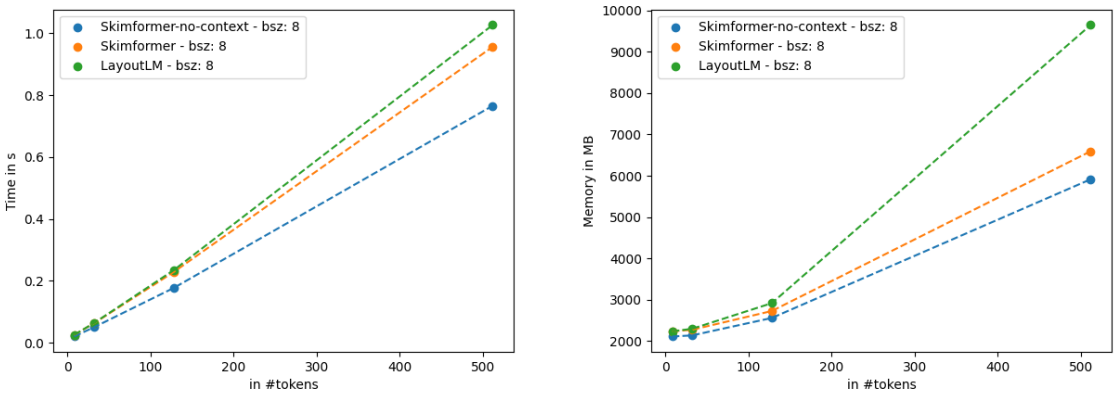
C Benchmark

Using Hugging Face's Transformers benchmarking tools (Wolf et al., 2020), we benchmark Skimformer and LayoutLM on both speed and required memory for pre-training. We consider the base variant of LayoutLM, and use the implementation from the Transformers library. In addition to the full Skimformer, we evaluate a variant in which the small Transformer contextualizing layout embeddings is removed (Skimformer-no-context). The batch size is fixed to 8, and memory and time performance is evaluated for the following sequence lengths: 8, 32, 128 and 512. We use Python 3.7.10, PyTorch 1.8.1+cu101 (Paszke et al., 2019), and Transformers 4.6.0.dev0. All experiments were conducted on one Tesla T4 with 15GB of RAM.

Figure 5b reports the time (figure 5a) and peak memory consumption (figure 5b) with respect to the sequence length.

D Attention Visualization

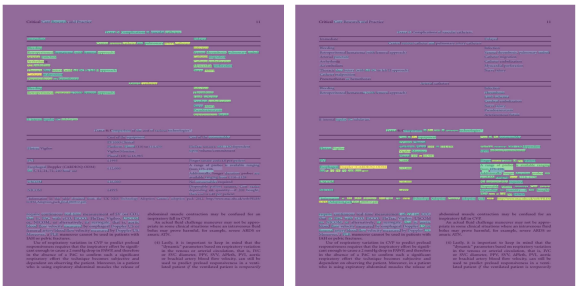
Figure 6 contains the attention maps obtained by Skimformer on two documents sampled from PubLayNet (Zhong et al., 2019). For each sample, we average the attention scores of tokens belonging to a given semantic unit, and map the result to the document image. In the first document (figure 6a), we focus on the top table (left) and the bottom one (right). In the second sample (figure 6b), we investigate the title (left), the authors (center) and the abstract (right).



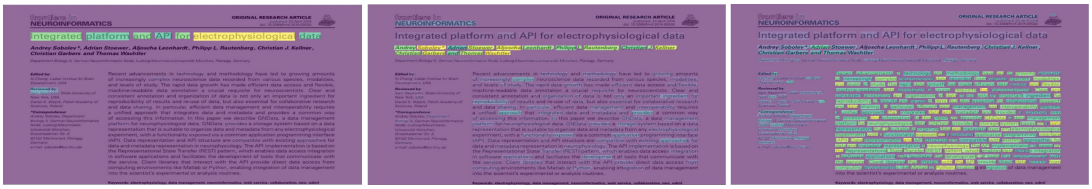
(a) Time usage for pre-training.

(b) Memory usage for pre-training.

Figure 5: Comparison of time and memory usage for LayoutLM (green), Skimformer with layout contextualizer (orange) and without (blue). Results are plotted against sequence length.



(a) Skim-attention maps corresponding to the top table (left) and the bottom table (right).



(b) Skim-attention maps corresponding to the title (left), the authors (center) and the abstract (right).

Figure 6: Skim-Attention maps on two sample documents.