Layout-Aware Text Representations Harm Clustering Documents by Type

Catherine Finegan-Dollak and **Ashish Verma**
IBM Research
1101 Kitchawan Rd, Yorktown Heights, NY 10598, USA
cfd@ibm.com, Ashish.Verma1@ibm.com

Abstract

Clustering documents by type—grouping invoices with invoices and articles with articles—is a desirable first step for organizing large collections of document scans. Humans approaching this task use both the semantics of the text and the document layout to assist in grouping like documents. LayoutLM (Xu et al., 2019), a layout-aware transformer built on top of BERT with state-of-the-art performance on document-type *classification*, could reasonably be expected to outperform regular BERT (Devlin et al., 2018) for document-type *clustering*. However, we find experimentally that BERT significantly outperforms LayoutLM on this task ($p < 0.001$). We analyze clusters to show where layout awareness is an asset and where it is a liability.

1 Introduction

Organizations are inundated by paperwork, often in the form of PDFs. Automated processing can help to organize and extract information from these documents, but the right process for a given document depends on its type: invoices are handled differently than contracts, for example. Document classification by type enables such a system; however, it requires training data for all of the desired classes, and finding such data to fit a given business's needs is difficult. There is no one-size-fits-all ontology of document types. While some types, such as invoices, may be common across industries, others, such as loan applications or home-inspection reports, are domain-specific. Users wishing to define their own classes will benefit from a system that enables them to group their own documents. To help with this, the present work addresses the task of clustering documents by type.

Humans grouping documents by type can use both the text and the appearance of documents. For example, we can distinguish a gas bill from an article at a glance, but we need to read at least a few words to determine whether a dense, two-column document is an article or a warranty. We therefore expect that a hybrid document representation that combines layout and text information should outperform a text-only representation when clustering documents by type. LayoutLM (Xu et al., 2019) is such a hybrid system and achieves state-of-the-art performance for document-type *classification*, outperforming text-only baselines. We therefore hypothesized that LayoutLM would also outperform these baselines for document-type *clustering*.

Sections 3 and 4 describe the systems we compared and the experiments we used to try to confirm this hypothesis. However, the main contribution of this work is experimental evidence of the opposite: LayoutLM performed significantly worse than a simple BERT baseline on this task (Section 5). Analysis of output clusters (Section 5.1) helps to explain this unexpected result.

2 Related Work

Hybrid layout/text representations Recent work combines layout with text for information extraction. Chargrid (Katti et al., 2018) assigns each pixel on a page a vector. For pixels inside the bounding box of a character, the vector is a one-hot encoding for that character; otherwise, it is a vector of zeros. This generates a $vocabsize \times height \times width$ tensor representation of the page for input to a CNN encoder-decoder model. BERTgrid (Denk and Reisswig, 2019) is nearly identical, but it replaces the one-hot character encoding with the word's BERT encoding. Liu et al. (2019) represent a document as a fully-connected graph where text boxes are nodes. The edge embedding between two nodes incorporates the distance between them, the text boxes' aspect ratios, and their relative sizes. Similarly, ZeroShotCeres (Lockard et al., 2020) represents semi-structured web pages as

Proceedings of the First Workshop on Insights from Negative Results in NLP, pages 60–65

ISBN 978-1-952148-66-8

graphs, with text-field nodes connected by edges for vertically or horizontally adjacent text fields and siblings or cousins in the DOM tree. Both systems then use graph neural networks over the document graphs.

Document-type classification Classification of documents by type has frequently been treated as an image classification problem. Many works have used varying CNN architectures (Kang et al., 2014; Afzal et al., 2015; Harley et al., 2015; Afzal et al., 2017; Tensmeyer and Martinez, 2017; Das et al., 2018) or other vision-based techniques (Kumar et al., 2014; Sarkhel and Nandi, 2019).

Some works have combined vision and NLP for document-type classification, using OCR for text extraction. Noce et al. (2016) assigned the most relevant words unique colors, then filled the bounding boxes of those words with the corresponding color, enabling the CNN processing the image to "see" the word. Asim et al. (2019) provided the most important words as features to a CNN, later combining the output with an image stream that used an InceptionV3 CNN architecture. Dauphinee et al. (2019) concatenated the output of a CNN image classifier with a multilayer perceptron bag-of-words classifier, then fed the concatenation to a meta-classifier. Ferrando et al. (2020) used an ensemble of a BERT classifier and EfficientNets CNNs. Audebert et al. (2020) concatenated image features (from a MobileNet v2 CNN) with text features (generated by passing FastText embeddings for the text through a 1D CNN) to form the input to a multilayer perceptron. Cosma et al. (2020) used text to help pretrain part of their classifier: they performed LDA to determine documents' topics, then trained their CNN to try to predict those topics using only the document image. They ultimately used the CNN as part of a model to predict document type using the image only. All of these systems are supervised, whereas this work addresses unsupervised clustering.

Document-type clustering Csurka et al. (2016) trained models on RVL-CDIP, then used those models to generate representations for clustering other document-type datasets. Abuelwafa et al. (2019) used unsupervised feature learning to improve their representations of document images for clustering. They applied transformations to document images to generate surrogate classes, then trained a CNN to classify them. They used that trained CNN to generate representations of document images for clustering. There is, to our knowledge, no previous work clustering RVL-CDIP.

3 Systems

We compare LayoutLM and BERT, as well as a TF-IDF baseline (sklearn's[1] (Pedregosa et al., 2011) implementation with default hyperparameters). In each case, we use the specified system to generate one vector representation for each document image, then cluster using sklearn's k-means, with k set to the number of gold classes plus one.

BERT (Devlin et al., 2018) is a transformer-based bidirectional model that generates contextualized word embeddings for a sequence of words. The input to a BERT model for the i-th token in the sequence is a sum of (a) its token embedding; (b) a position embedding for position i; and (c) a segment embedding indicating whether the token is in the first or second segment of the input sequence.

LayoutLM (Xu et al., 2019) is a BERT-like transformer model modified to generate layout-aware contextualized word embeddings. In place of BERT's single positional embedding, LayoutLM adds positional embeddings for the x- and y-coordinates of a bounding box around the token. The token's embedding thus incorporates its two-dimensional location on the page and its size. This architecture achieves state-of-the-art performance for supervised classification by document type.

Both BERT and LayoutLM output a vector for each token in the input sequence plus the special `[CLS]` token. However, k-means, like most clustering algorithms, requires a single vector representation of each example. Classifiers use the `[CLS]` embedding as a single-vector representation for the entire sequence. However, prior work (Reimers and Gurevych, 2019; Wang and Kuo, 2020) has shown that, for BERT without fine-tuning, this is not a good representation of the semantics of the entire sequence. Other options include combining all of the vectors in the output sequence by either averaging or max pooling—set the i-th value in the output vector equal to the max i-th value over all of the sequence vectors. For BERT, we use the average as our representation, since Reimers and Gurevych (2019) showed it captured semantic similarity better than the `[CLS]` token. For LayoutLM, we try all three methods.

[1] https://scikit-learn.org/

	F_1	ARI
BERT (average)	**0.23**$_{0.002}$	**0.17**$_{0.002}$
TF-IDF	0.21$_{0.014}$	0.13$_{0.020}$
LayoutLM (average)	0.16*$_{0.002}$	0.09*$_{0.001}$
LayoutLM (`[CLS]`)	0.20*$_{0.003}$	0.14*$_{0.003}$
LayoutLM (max pooled)	0.19*$_{0.001}$	0.13*$_{0.000}$

Table 1: Mean F_1 and ARI over five runs, with standard error of the mean (subscript). Items marked with * are significantly different from BERT average, $p < 0.001$ based on a two-tailed t-test.

4 Experiments

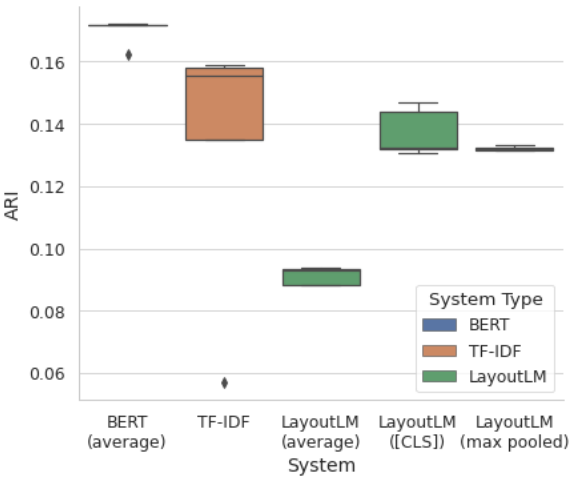
We evaluate on RVL-CDIP[2] (Harley et al., 2015), scanned tobacco-litigation documents from the Illinois Institute of Technology Complex Document Information Processing (IIT-CDIP) collection, labeled with type, such as letter or invoice. The complete class list appears in Table 3. We clustered the validation set (40K pages). Like LayoutLM, we used Tesseract[3] for OCR.

We use LayoutLM's publicly-released code and base model for experiments.[4] This model was pretrained on IIT-CDIP, excluding documents in RVL-CDIP. For BERT, we use the Transformers package[5] with the bert-base-uncased model, pretrained on books and Wikipedia. Because LayoutLM's masked language model pretrained on documents from the same domain, while BERT's did not, the dataset could favor LayoutLM.

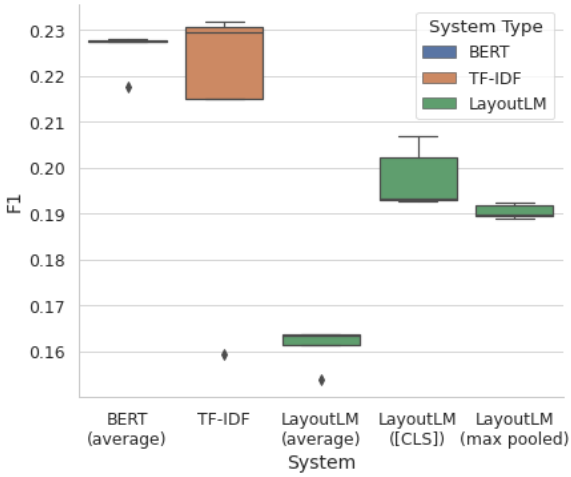
We calculate F_1 and adjusted Rand index (ARI) for each system, using Manning et al. (2008)'s definitions of true and false positives and negatives. We use sklearn (Pedregosa et al., 2011)'s implementation of ARI. We report the mean over 5 runs and use a two-tailed t-test to determine whether systems differ significantly from the BERT baseline.

5 Results

Results are shown in Table 1 and Figure 1. Our experiments show that the performance of a system using LayoutLM vectors is significantly worse ($p < 0.001$) at clustering RVL-CDIP documents by type than a simple BERT baseline. There was no significant difference between the TF-IDF and BERT systems.



(a) Boxplot of ARI



(b) Boxplot of F_1

Figure 1: Boxplots of F_1 and ARI over five runs.

In contrast to prior work on BERT, where the `[CLS]` token was a worse representation than averaging (Reimers and Gurevych, 2019; Wang and Kuo, 2020), the best-performing LayoutLM system used the `[CLS]` token embedding. We suspect this is because averaging or max-pooling LayoutLM vectors blends together bounding box information for all tokens, erasing the benefits of a layout-sensitive transformer. In light of these results, we also tested `[CLS]` token and max-pooling for BERT on this task. Consistent with prior work, averaging outperformed both; see Table 2.

All of these scores are low, especially in comparison to classification results. The comparison is misleading, of course, since classification requires training data, and clustering addresses the case where such data is not available. Neverthe-

[2] https://www.cs.cmu.edu/~aharley/rvl-cdip/

[3] https://github.com/tesseract-ocr/tesseract; we used version 4.1.1.

[4] https://github.com/microsoft/unilm/tree/master/layoutlm. The version as of this writing does not include the optional image embeddings.

[5] https://github.com/huggingface/transformers

	F_1	ARI
Average	**0.23**	**0.17**
[CLS]	0.21	0.16
Max-pooled	0.20	0.15

Table 2: Comparison of different techniques of combining BERT vectors (mean F_1 and ARI over five runs)

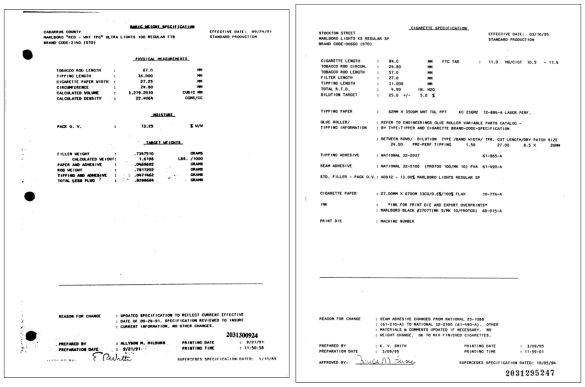


Figure 2: Specifications from a cluster with 0.97 purity.

less, much improvement will be required before document-type clustering is useful for practical applications.

5.1 Analysis

To understand this unexpected result, we reviewed example clusters from one run of the BERT system and one of LayoutLM([CLS]).

Documents in LayoutLM's best clusters had consistent layouts, illustrated in Figure 2. Specifications in the highest-purity cluster seem to have been generated from a few templates. For such documents, the layouts are so consistent that no learning is required to identify which aspects of layout to emphasize in grouping the documents. Not all specifications conform to these templates, though. Figure 3 shows some with different formats, which LayoutLM placed in a different cluster. Document layouts that are common across multiple document types also caused problems for LayoutLM. Figure 4 shows an invoice and resume with similar formats from the cluster with the lowest purity.

Table 3 lists class precision[6] for the sample clustering runs. From this, we see that LayoutLM performed well on scientific publications. A substantial fraction of this class contains two-column documents, like those in Figure 6, which LayoutLM can recognize. In contrast, BERT far outperformed LayoutLM for resumes, where page layout may

[6]Precision of pairs of examples where at least one has the specified gold label.

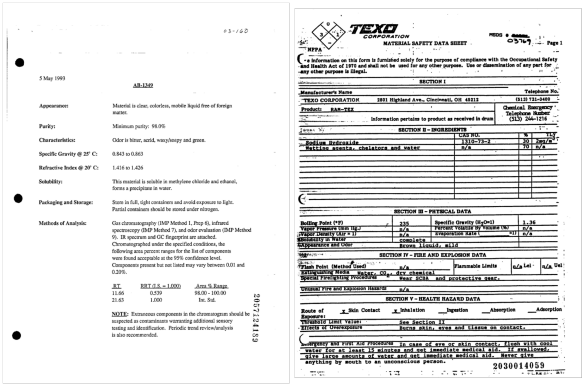
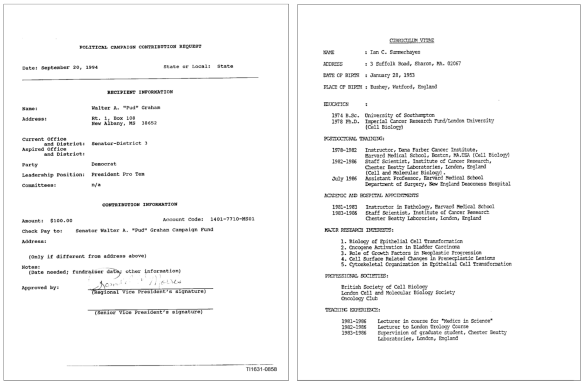


Figure 3: Specifications with different formats, which did not appear in the high-purity specification cluster.



(a) Invoice (b) Resume

Figure 4: Samples from the lowest-purity cluster.

be misleading. BERT correctly clustered the two resume images in Figure 5 together, despite their obvious layout differences. LayoutLM understandably placed them in different clusters.

6 Conclusion

LayoutLM captures textual and layout information about documents. When training data is available,

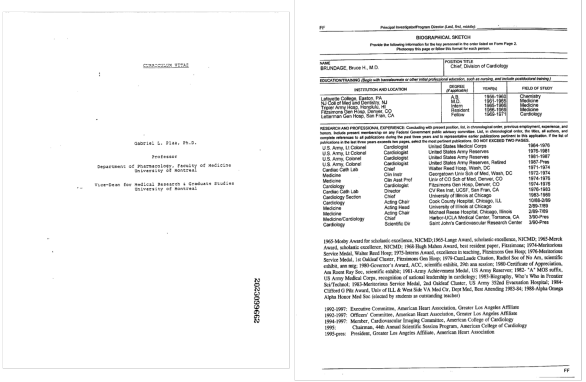


Figure 5: BERT correctly clustered these two resume pages together despite their very different layouts; LayoutLM put them in different clusters.

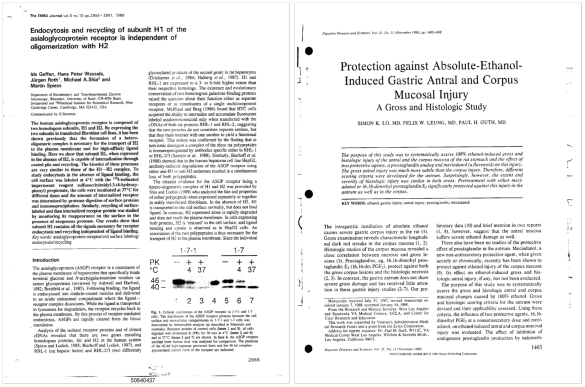


Figure 6: LayoutLM correctly clustered these two scientific documents together.

Class	BERT	LayoutLM
scientific publication	0.29	**0.37**
file folder	**0.32**	0.30
email	**0.39**	0.29
questionnaire	0.11	**0.21**
handwritten	**0.24**	0.19
specification	**0.27**	0.19
resume	**0.60**	0.16
news article	**0.15**	**0.15**
advertisement	0.09	**0.14**
memo	0.13	**0.14**
letter	**0.15**	0.12
budget	**0.13**	0.11
invoice	**0.18**	0.10
presentation	**0.16**	0.10
scientific report	**0.12**	0.09
form	**0.12**	0.09

Table 3: Class precisions for the sample clustering.

a model can learn when to leverage each. Thus, LayoutLM performed quite well at classifying documents by type. But when clustering, there is no model to indicate how to weight features in determining document similarities. In this context, layout information significantly harms performance. Future work should explore ways to incorporate benefits of layout information into a representation while limiting its harm, as well as how layout information affects tasks that fall between classification and clustering, such as semi-supervised learning. Such questions must be answered for document-type clustering to become practical.

Acknowledgments

We would like to thank the anonymous reviewers for their helpful comments, as well as Anik Saha for many discussions on LayoutLM's strengths and weaknesses for supervised tasks.

References

Sherif Abuelwafa, Marco Pedersoli, and Mohamed Cheriet. 2019. Unsupervised Exemplar-Based Learning for Improved Document Image Classification. *IEEE Access*, 7:133738–133748.

Muhammad Zeshan Afzal, Samuele Capobianco, Muhammad Imran Malik, Simone Marinai, Thomas M. Breuel, Andreas Dengel, and Marcus Liwicki. 2015. DeepDocClassifier: Document classification with deep convolutional neural network. In *2015 13th International Conference on Document Analysis and Recognition (ICDAR)*, pages 1111–1115.

Muhammad Zeshan Afzal, Andreas Kolsch, Sheraz Ahmed, and Marcus Liwicki. 2017. Cutting the error by half: Investigation of very deep cnn and advanced training strategies for document image classification. In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, pages 883–888.

Muhammad Nabeel Asim, Muhammad Usman Ghani Khan, Muhammad Imran Malik, Khizar Razzaque, Andreas Dengel, and Sheraz Ahmed. 2019. Two Stream Deep Network for Document Image Classification. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 1410–1416.

Nicolas Audebert, Catherine Herold, Kuider Slimani, and Cédric Vidal. 2020. Multimodal Deep Networks for Text and Image-based Document Classification. *Communications in Computer and Information Science*, 1167:427–443.

Adrian Cosma, Mihai Ghidoveanu, Michael Panaitescu-Liess, and Marius Popescu. 2020. Self-Supervised Representation Learning on Document Images.

Gabriela Csurka, Diane Larlus, Albert Gordo, and Jon Almazán. 2016. What is the right way to represent document images?

Arindam Das, Saikat Roy, and Ujjwal Bhattacharya. 2018. Document Image Classification with Intra-Domain Transfer Learning and Stacked Generalization of Deep Convolutional Neural Networks.

Tyler Dauphinee, Nikunj Patel, and Mohammad Rashidi. 2019. Modular Multimodal Architecture for Document Classification.

Timo I. Denk and Christian Reisswig. 2019. BERTgrid: Contextualized Embedding for 2D Document Representation and Understanding. In *Workshop on Document Intelligence at NeurIPS 2019*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of

Deep Bidirectional Transformers for Language Understanding. *arXiv preprint arXiv:1810.04805*.

Javier Ferrando, Juan Luis Domínguez, Jordi Torres, Raúl García, David García, Daniel Garrido, Jordi Cortada, and Mateo Valero. 2020. Improving Accuracy and Speeding Up Document Image Classification Through Parallel Systems. In *Computational Science – ICCS 2020*, pages 387–400, Cham. Springer International Publishing.

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

Le Kang, Jayant Kumar, Peng Ye, Yi Li, and David Doermann. 2014. Convolutional Neural Networks for Document Image Classification. In *2014 22nd International Conference on Pattern Recognition*, pages 3168–3172.

Anoop R. Katti, Christian Reisswig, Cordula Guder, Sebastian Brarda, Steffen Bickel, Johannes Höhne, and Jean Baptiste Faddoul. 2018. Chargrid: Towards Understanding 2D Documents. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4459–4469, Brussels, Belgium. Association for Computational Linguistics.

Jayant Kumar, Peng Ye, and David Doermann. 2014. Structural similarity for document image classification and retrieval. *Pattern Recognition Letters*, 43:119–126.

Xiaojing Liu, Feiyu Gao, Qiong Zhang, and Huasha Zhao. 2019. Graph Convolution for Multimodal Information Extraction from Visually Rich Documents. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Industry Papers)*, pages 32–39, Minneapolis, Minnesota. Association for Computational Linguistics.

Colin Lockard, Prashant Shiralkar, Xin Luna Dong, and Hannaneh Hajishirzi. 2020. ZeroShotCeres: Zero-Shot Relation Extraction from Semi-Structured Webpages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8105–8117, Online. Association for Computational Linguistics.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press, USA.

Lucia Noce, Ignazio Gallo, Alessandro Zamberletti, and Alessandro Calefati. 2016. Embedded Textual Content for Document Image Classification with Convolutional Neural Networks. In *Proceedings of the 2016 ACM Symposium on Document Engineering*, DocEng '16, pages 165–173, New York, NY, USA. Association for Computing Machinery.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Ritesh Sarkhel and Arnab Nandi. 2019. Deterministic routing between layout abstractions for multi-scale classification of visually rich documents. In *IJCAI International Joint Conference on Artificial Intelligence*, pages 3360–3366.

Chris Tensmeyer and Tony Martinez. 2017. Analysis of Convolutional Neural Networks for Document Image Classification. In *Proceedings of the International Conference on Document Analysis and Recognition, ICDAR*, volume 1, pages 388–393.

Bin Wang and C.-C. Jay Kuo. 2020. SBERT-WK: A Sentence Embedding Method by Dissecting BERT-based Word Models.

Yiheng Xu, Minghao Li, Lei Cui, Shaohan Huang, Furu Wei, and Ming Zhou. 2019. LayoutLM: Pre-training of Text and Layout for Document Image Understanding.