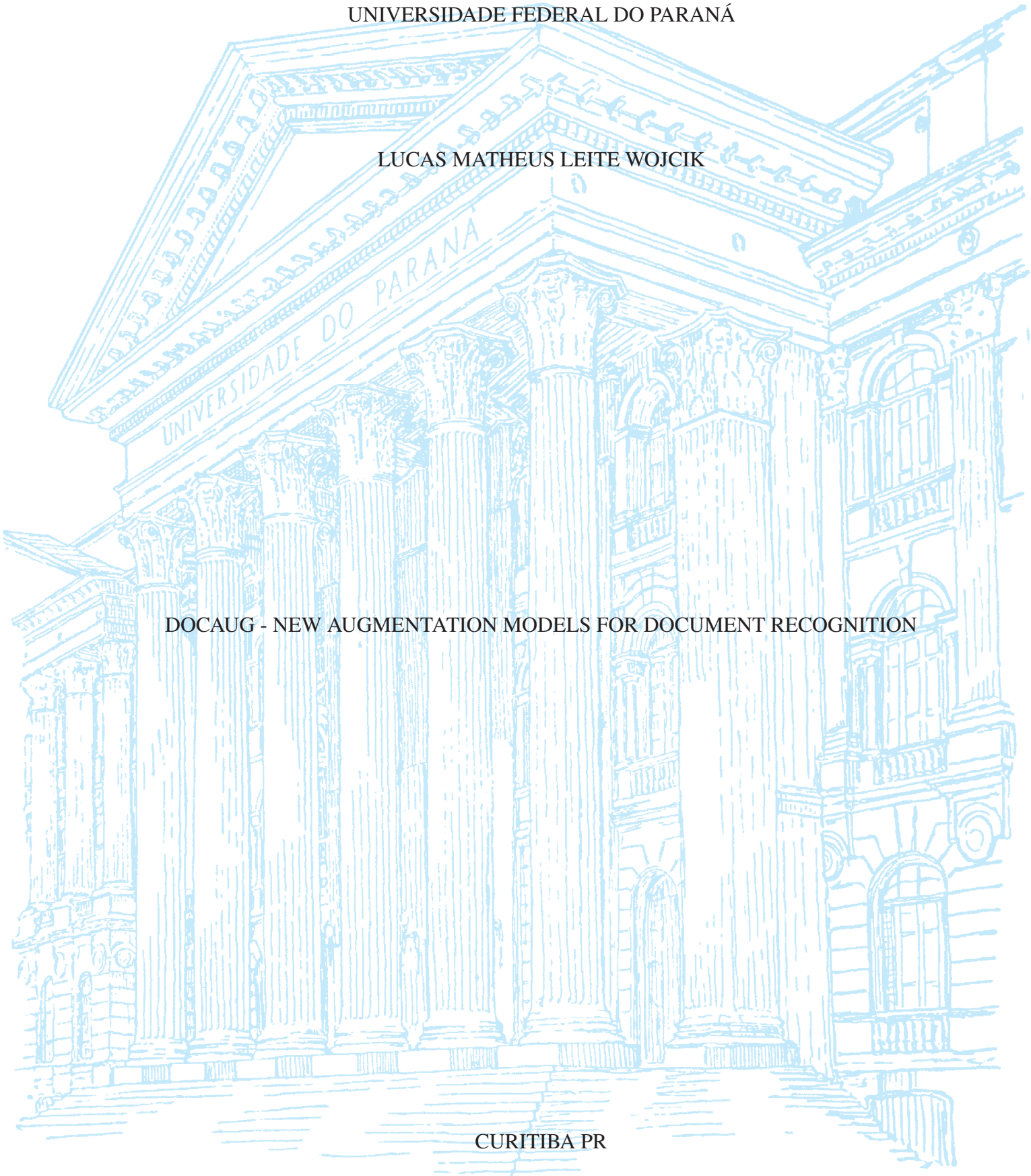
UNIVERSIDADE FEDERAL DO PARANÁ

LUCAS MATHEUS LEITE WOJCIK



DOCAUG - NEW AUGMENTATION MODELS FOR DOCUMENT RECOGNITION

CURITIBA PR

2025

LUCAS MATHEUS LEITE WOJCIK

DOCAUG - NEW AUGMENTATION MODELS FOR DOCUMENT RECOGNITION

Documento apresentado como requisito parcial ao exame de qualificação de Mestrado no Programa de Pós-Graduação em Informática, Setor de Ciências Exatas, da Universidade Federal do Paraná.

Área de concentração: *Computação*.

Orientador: David Menotti Gomes.

Coorientador: Roger Leitzke Granada.

CURITIBA PR

2025

DADOS INTERNACIONAIS DE CATALOGAÇÃO NA PUBLICAÇÃO (CIP)
UNIVERSIDADE FEDERAL DO PARANÁ
SISTEMA DE BIBLIOTECAS – BIBLIOTECA DE CIÊNCIA E TECNOLOGIA

Wojcik, Lucas Matheus Leite
Docaug - new augmentation models for document recognition / Lucas Matheus Leite Wojcik. – Curitiba, 2025.
1 recurso on-line : PDF.

Dissertação (Mestrado) - Universidade Federal do Paraná, Setor de Ciências Exatas, Programa de Pós-Graduação em Informática.

Orientador: David Menotti Gomes
Coorientador: Roger Leitzke Granada

1. Aprendizado do computador. 2. Compressão de dados(Computação). I. Universidade Federal do Paraná. II. Programa de Pós-Graduação em Informática. III. Gomes, David Menotti. IV. Granada, Roger Leitzke. V. Título.

Bibliotecário: Leticia Priscila Azevedo de Sousa CRB-9/2029

TERMO DE APROVAÇÃO

Os membros da Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação INFORMÁTICA da Universidade Federal do Paraná foram convocados para realizar a arguição da Dissertação de Mestrado de **LUCAS MATHEUS LEITE WOJCIK**, intitulada: **DOCAUG - NEW AUGMENTATION MODELS FOR DOCUMENT RECOGNITION**, sob orientação do Prof. Dr. DAVID MENOTTI GOMES, que após terem inquirido o aluno e realizada a avaliação do trabalho, são de parecer pela sua APROVAÇÃO no rito de defesa.

A outorga do título de mestre está sujeita à homologação pelo colegiado, ao atendimento de todas as indicações e correções solicitadas pela banca e ao pleno atendimento das demandas regimentais do Programa de Pós-Graduação.

CURITIBA, 21 de Março de 2025.

Assinatura Eletrônica
23/03/2025 16:59:23.0
DAVID MENOTTI GOMES
Presidente da Banca Examinadora

Assinatura Eletrônica
23/03/2025 19:17:09.0
ROGER LEITZKE GRANADA
Coorientador(a)

Assinatura Eletrônica
24/03/2025 09:42:13.0
PAULO RICARDO LISBOA DE ALMEIDA
Avaliador Interno (UNIVERSIDADE FEDERAL DO PARANÁ)

Assinatura Eletrônica
24/03/2025 09:54:28.0
BYRON LEITE DANTAS BEZERRA
Avaliador Externo (UNIVERSIDADE DE PERNAMBUCO)

À voz incessante que continua me impelindo a prosseguir, apesar de estar mais difícil de ouvir hoje em dia.

ACKNOWLEDGEMENTS

My sincere thanks to my professor and advisor, David Menotti. My work during this master's course was only made possible and fruitful thanks to his enduring support over this period. I am also thankful to Unico, the company that supported the research here conducted and made is possible for our research project to be viable. Furthermore, my special thanks to Roger, Paulo and Byron for the final refinement of the current version of this document.

Finally, I want to thank my colleagues for the companion, my family for the support, and the close ones who are not here anymore for the memories. And a special thanks to my significant other for everything.

RESUMO

A literatura recente em Reconhecimento de Documentos tem visto muitos avanços. Desde a incorporação do modelo BERT ao domínio de documentos, técnicas baseadas em Transformers têm dominado o estado da arte, e assim como acontece no campo de NLP, este estado da arte geralmente é superado através de modelagem de atenção ou ajustes no pré-treino. No entanto, qualidade de dados tem sido um tópico cada vez mais pungente em vários campos de *deep learning*, mas o escopo de documentos ainda não tem visto muitos avanços nesta discussão. Além disso, existem vários domínios no campo de Reconhecimento de Documentos cujos documentos apresentam tarefas *few-shot* onde anotações de qualidade são escassas. Com o objetivo de avançar a pesquisa em documentos neste tópico, apresentamos duas técnicas de aumento de dados que focam em maximizar o conhecimento contido nas instâncias de documentos conhecidas, utilizando técnicas sem imagem. Prosseguimos na linha de utilizar algumas técnicas do campo de NLP com um modelo baseado em LLMs para reescrita de textos, uma técnica de aumento de dados que foi primeiro apresentada em alguns cenários de NLP. Além disso, criamos um algoritmo para extrair *templates* de documentos de acordo com uma estrutura de grafos gerada pelas respectivas entidades (nós). Estes templates possuem a informação de layout de cada documento embutida, e podem ser usados para aumentação de dados numa estratégia simples de preencher o formulário com algum método de geração de texto. Cada uma de nossas técnicas de aumentação são testadas através de um dataset público da literatura, e realizamos a etapa de *fine-tuning* em um modelo pré-treinado para verificar se nossas aumentações auxiliam na melhora da performance. Nossos resultados mostram que estas técnicas conseguem melhorar as métricas de qualidade consistentemente, para ambas as técnicas e em todos os cenários de teste, reduzindo a taxa de erro em cerca de dez por cento para o FUNSD e em até cinquenta para o EPHOIE.

Palavras-chave: Aumento de dados. Reconhecimento de Documentos. Aprendizado de Máquina.

ABSTRACT

The most recent literature in Document Recognition has seen many advances. Ever since the adaptation of BERT, Transformer-based approaches have been dominating and, in line with other NLP improvements, the state of the art is usually broken through attention modeling or pre-training adjustments. However, data quality has been an increasingly relevant topic in various deep learning fields, but the document scope has not seen many advancements in this discussion yet. Furthermore, many areas of Document Recognition involve important few-shot tasks where annotated documents are scarce. To further the document research on this topic, we present two new data augmentation techniques that focus on maximizing the knowledge from the known document instances, using imageless techniques. We continue on the line of employing some techniques from the NLP field with an LLM-based approach for text rewriting, a data augmentation approach pioneered in some NLP scenarios. Furthermore, we also create an algorithm to extract templates from documents based on a graph structure generated by the respective entities (nodes). These templates encode the layout information from each document and can be used for data augmentation by filling the template out with a generator's text. Each of our approaches is validated in a different public dataset taken from the literature, and we fine-tune a pre-trained model to evaluate whether our augmentations help boost its performance. We find out that these techniques consistently improve the quality measure, for both techniques and in all testing scenarios, reducing the error rate in around ten percent for FUNSD and up to fifty for EPHOIE.

Keywords: Data Augmentation. Document Recognition. Machine Learning.

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

NLP	Natural Language Processing
DR	Document Recognizion
LLM	Large Language Model
SoTA	State of the Art
GPU	Graphical Processing Unit
GAN	Generative Adversarial Network
LSTM	Long Short-Term Memory
OCR	Optical Character Recognition
SER	Semantic Entity Recognition
RE	Relation Extraction

CONTENTS

1 INTRODUCTION

Although the Document Recognition field has seen many advances over the last few years that pushed the state of the art to such a degree that most state-of-the-art models can be used in a wide variety of scenarios thanks to the introduction of unsupervised pre-training techniques, there are still quite a few problems regarding the availability of training data in many scenarios. Due to the nature of machine learning systems, some amount of training data is still required for any approach to succeed. In general, more data usually means a better performance, which is evident in the fact that the most challenging datasets often contain few instances in training and testing.

Most of the techniques now used in the document field are borrowed from Natural Language Processing (NLP). The current state of the art consists of transformer models trained using an unsupervised pre-training followed by a supervised fine-tuning pipeline. This approach was pioneered in NLP by BERT (Devlin et al., 2019). The pre-training stage can be understood as the stage at which the model learns the domain's syntax, while the fine-tuning stage is for the semantics. A very large amount of training data is required to learn the syntax only through examples. For documents, the IIT-CDIP dataset (Soboroff, 2022) is the one most commonly used. It is publicly available and most state-of-the-art (SoTA) models use it today.

It is easier to gather data for the pre-training stage because the self-supervised training model does not require a lot of annotations, just the text that can be obtained with OCR systems. However, the supervised training in the fine-tuning stage requires a more specific annotation style according to the task to be learned by the model, and this requirement makes fine-tuning data more expensive. Furthermore, some document domains present an inherent difficulty in gathering document instances due to containing sensitive information. For these reasons, there is usually a lack of quality data for model fine-tuning in commercial applications.

The official document domain is an example, consisting of identity cards and passports. There are few works regarding this domain, something that can be explained by the difficulty both in reproducing experiments and publicizing datasets. This difficulty comes from the sensitive nature of these documents, which makes data augmentation required. Furthermore, recent advances such as the Large Language Model (LLM) technology show promising applications in research, and we are yet to see their full potential in practice. Also, even for documents of public domains, data augmentation may prove to be a powerful tool for helping model performance in document understanding tasks, especially given how we suffer from a lack of annotated data for the fine-tuning stage across various scenarios. We use this as a hypothesis for the presented research, where we perform a few downstream fine-tuning DR tasks.

Our proposed methods aim to resolve the specific issue of document domains where a reduced set of annotated instances is available, but new instances are hard to obtain and/or annotate. With this in mind, our research question is:

Can we use data augmentation to improve performance in downstream tasks with small datasets?

Adjacently, if so, what are the augmentation techniques available for use in a document recognition scenario, and how can they aid in our pursuit of better models? Furthermore, how simple can we make it? Is it possible to perform a virtual, imageless augmentation and still improve baseline results? In this research, we aim to answer these questions to the best of our ability.

1.1 METHODS AND OBJECTIVES

To answer our research questions, our objective is to build two different augmentation pipelines, each one aimed at a different document domain type, and both focusing on generalization of latent information from few document instances. We use two datasets for downstream document recognition tasks: FUNSD (Guillaume Jaume, 2019) and EPHOIE (Wang et al., 2021), both being datasets commonly used for SoTA comparisons, and each belonging to one domain that we tackle. FUNSD features documents with complex text types and document layouts, while EPHOIE features simple texts and layouts.

Our first approach is aimed at tackling complex datasets and is labelled the "LLM" method. It leverages the latent knowledge from the LLM to expand the document corpus through the means of textual augmentation. We employ the LLM together with some other simple augmentation techniques to expand the text variability utilizing rewriting the same document multiple times, creating new instances with new text. To the best of our knowledge, this is the first work of the kind to use LLM textual augmentation specifically for documents.

Our second approach, labelled the "Template" method, focuses on extracting the layout information contained in the known document images and annotations to augment new instances that expand the known corpus while keeping the document's structural integrity. In other words, we create new documents following the same layout from old documents, but with new texts belonging to the same labels. It differs from the first approach in that here, the document's structural semantics are not tied to the text contained in it: a name in a document can be any name.

The idea behind these approaches comes from both the datasets themselves and from correlated research in NLP. Textual augmentation in NLP is done through a number of techniques that include LLM usage, and we follow in these steps when dealing with documents featuring complex natural language sentences. We also use older approaches such as synonym replacement to deal with simpler types of text within FUNSD. The template technique comes from the intuition that some document domains are tied to a reduced number of layouts into which varying texts of the same semantic are written. Form-like document datasets comprised of domains such as identity documents or test headers can be described entirely in terms of the possible layouts, and as such using these layouts as the sole basis for document crafting should yield a representative synthetic dataset.

Furthermore, these techniques perform a virtual augmentation that emulates the creation of a new document without an image representation of it. Here, a document will be represented only by the size of an original document image and by the texts, their labels and coordinates within that theoretical image. Performing imageless augmentation yields a simpler pipeline with no drawback in performance, since models without the image modality remain competitive within the document recognition SoTA, on top of being more light-weight since the image module is not used.

1.2 CONTRIBUTIONS

The contributions of our work are the augmentation procedures and the respective augmented instances, which are examples of the possible new paths for document augmentation. These techniques contribute to the field by making it easier to fine-tune models on more restricted domains as well as opening new paths for using LLMs in the document scenario. Furthermore, we employ an imageless approach (using a bi-modal model for text and layout only) for quick generation of a large number of instances, also circumventing the problem of understanding and

reproducing image bias and noise in the target datasets (which feature scanned, noisy documents). We show that data augmentation techniques can be used to improve performance on downstream tasks with instance-restricted domains, which contributes to the field also by bringing to light the fact that data augmentation is beneficial for DR. This is because the SoTA is often broken by means of improving the architecture of the model or the pre-training procedure.

1.3 PUBLICATIONS

The work presented here has been partially published in two papers: as an initial exploration in the Workshop of Works in Progress of SIBGRAPI 2024 (Wojcik et al., 2024) and as a complete work in VISAPP 2025 (Wojcik et al., 2025). The results presented here represent a consolidation of the concepts presented in these two papers. We perform new experiments with our augmented datasets in a more comprehensive way in order to better answer our research question.

1.4 DOCUMENT ORGANIZATION

The remainder of this work is organized as follows. Chapter 2 shows the current state of the field across both models and datasets, showcasing current challenges and their proposed solutions. Chapter 3 presents the detailing of our augmentation techniques, as well as a description of the datasets augmented and the model used for evaluation. Chapter 4 delineates the experiments made to evaluate our approach, as well as a description of the model architecture for fine-tuning each task and the statistics from the dataset, both the real and augmented parts. Chapter 5 shows our results on the tasks used for evaluation on both domains and discusses the results achieved. Finally, Chapter 6 presents the conclusion to our work.

2 RELATED WORK

Since Document Recognition is intrinsically tied to NLP, we will first address NLP as a machine learning problem and its evolution across the years. All recognition models here discussed are based on Deep Learning (DL), here defined as a natural advance of Machine Learning. This advance consists of the enlargement and deepening of previous models, granting the model the possibility of learning a more robust *representation* by themselves.

This advancement is seen in NLP with the introduction of the pre-training plus fine-tuning pipeline used in BERT (Devlin et al., 2019), a transformer-based (Vaswani et al., 2017) architecture. The representation learning occurs at the pre-training stage, and is shown to require large amounts of data, the quantity of which is tied to the model's performance.[1] With this, we also discuss the process of making synthetic data across different document domains, as well as recent NLP techniques that inspired our proposed approaches.

2.1 RECENT LANGUAGE PROCESSING HISTORY

We highlight three landmarks in the recent NLP history as relevant both to the language and document fields. These are the publishing of transformer (Vaswani et al., 2017), BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019). Transformer was proposed as a model for sequence-to-sequence tasks, and has become a staple of both NLP and document solutions. BERT consists of an encoder from the transformer model and proposes the novel training procedure in two stages described previously. Finally, RoBERTa uses the base BERT model with an improved training recipe to improve the model's performance on various downstream tasks. All of these are briefly detailed in this section.

The transformer model consists of an Encoder-Decoder architecture. The basic block of the architecture is called the attention mechanism, which is based on an attention function. This architecture is shown in Figure 2.1 (Vaswani et al., 2017), the Encoder to the left and the Decoder to the right. Each component is composed of a pipeline of N attention blocks.

First, the text is represented via a vector embedding, which is transformed into the Query, Key and Value vectors through the corresponding linear layer. The Q, K and V vectors are the input to the multi-headed attention mechanism, which consists of h parallel attention layers with independent weights. Each attention layer performs a weighted sum between Q and K, and then between this output and V. This mechanism is shown in Figure 2.2 (Vaswani et al., 2017). The scaled dot-product attention function is also represented in Equation 2.1.

$$attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V. \tag{2.1}$$

Intuitively, the attention function creates a filter that highlights the important tokens in the Key sentence for each given token in the Query. An example of this is given in Figure 2.3, which presents a translation problem between Portuguese and English.

As shown in Figure 2.3, the path length between any two elements in the sequence in terms of how much the network signals have to traverse the network, for transformer attention layer, is $O(1)$. This means this operation has a high potential for parallelization, yielding better performance in our current paradigm of GPU processing.

[1] A discussion on data quality is found in Section 2.2

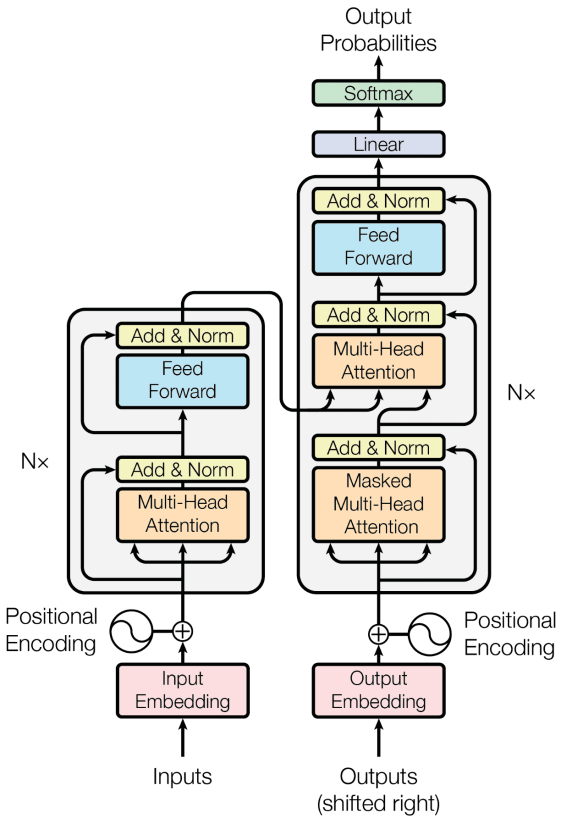


Figure 2.1: Transformer architecture (Vaswani et al., 2017)

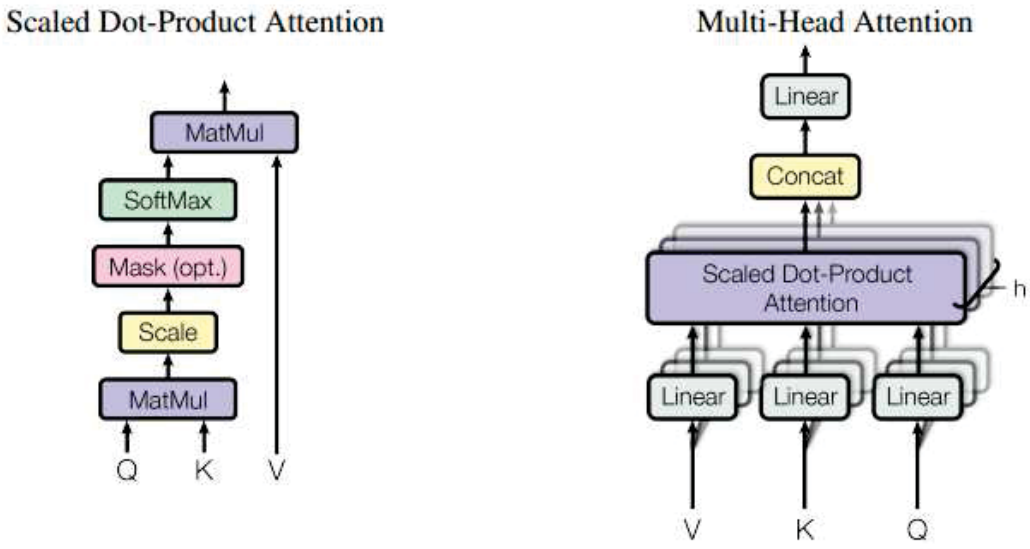


Figure 2.2: Attention modeling (Vaswani et al., 2017)

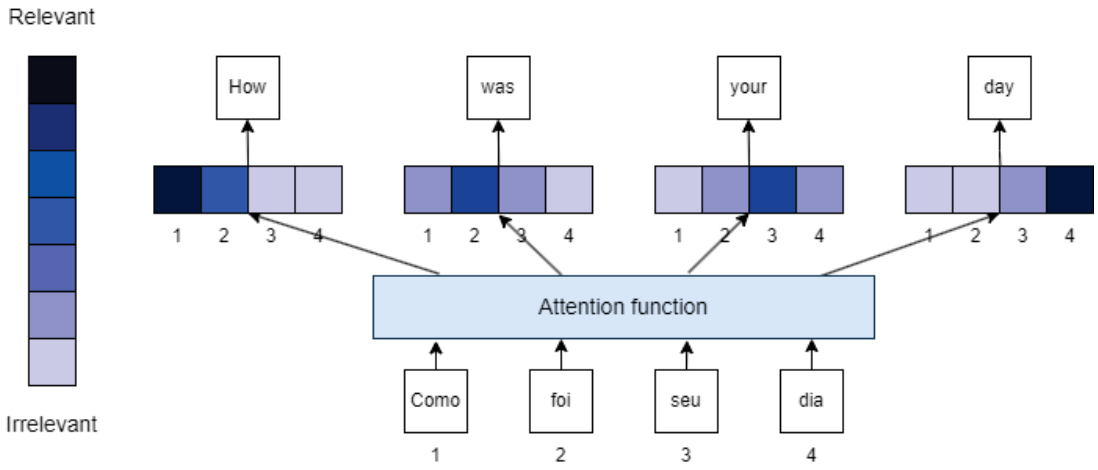


Figure 2.3: Attention example

This model has served as a baseline for most subsequent SoTA advances in NLP. BERT (Devlin et al., 2019), or "Bidirectional Encoder Representations from transformers", set a new SoTA by using the transformer encoder together with the pre-training strategy that is omnipresent to this day.

Intuitively, this model's learning is split into two different parts: *syntax* and *semantics*, wherein the syntax is learned in pre-training and the semantics in fine-tuning. The pre-training stage in BERT consists of two self-supervised tasks[2]. These are Masked Language Modeling (MLM) and Next Sentence Prediction (NSP).

MLM trains the model through a simple Cross-Entropy (CE) loss. From the input sentence, a set of 15% tokens (corresponding to random slots in the input vector) are randomly chosen. From these 15%, 80% tokens are replaced with a reserved `[MASK]` token, 10% tokens are replaced with a randomly chosen token and the remaining 10% tokens are left unchanged. The model is then asked to reconstruct the original, unmasked sentence, and the performance is evaluated using the CE loss function.

At every step of training, the model receives as input two sentences (over which the MLM task is performed) and a single binary token named `isNext` that corresponds to whether the first sentence is followed by the second sentence in the text corpus they were taken from. In NSP, the model is asked to predict whether this relationship is true, the Binary Cross-Entropy (BCE) loss function is used to compare the model prediction with the ground truth from the `isNext` token. These tasks model two things, respectively: the syntax of natural language sentences, such that the model intrinsically understands what well-formed sentences look like, and the continuation between sentences, something that is directly transferable to some downstream fine-tuning tasks.

This pre-training technique has advantages and disadvantages. The main advantage, as seen in the BERT results, is the fact that the model can leverage the availability of a large corpus of natural language texts on the internet to learn a robust representation. However, two disadvantages are: the requirement of a large amount of data (although it does not need to be annotated, just grammatically correct), and the requirement of intense training, consuming a lot of time and GPU resources. Regardless of these limitations, these tasks are still a staple in the SoTA, especially MLM, which has been adapted to document learning as well.

[2]Some of the literature uses the word unsupervised since this training does not have a separate label. We prefer to call it self-supervised since the ground truth is the text itself.

Finally, RoBERTa authors improve on BERT by fine-tuning some training parameters from the original BERT, using the same pre-training techniques. These experiments correspond the fine-tuning of parameters such as learning rate and batch size. For us, the most important set of experiments is the incremental addition of training data and a longer pre-training stage. The results showcase that the most significant improvements on performance on downstream tasks from the GLUE dataset are the addition of more data and longer pre-training stage. Table 2.1 displays these improved results.

The RoBERTa architecture is identical to BERT, which shows that the most determining factor for transformer performance is the quantity and quality of data. This is seen in a comparison with similar papers (You et al., 2019) where the improvement with architectural changes is less than the improvement gained from more intensive training.

A comparison of the NLP field in a few of the GLUE tasks at the time of the release of RoBERTa is shown in Table 2.1. MNLI stands for MultiNLI Matched / Mismatched (wherein NLI means Natural Language Inference). QQP stands for Quora Question Pairs. QNLI stands for Question NLI. SST is the Stanford Sentiment Treebank.

Model	MNLI-(m/mm)	QQP	QNLI	SST-2
BiLSTM+ELMo+Attn (2018)	76.4/76.1	66.1	82.3	93.2
GPT (2018)	82.1/81.4	70.3	87.4	91.3
BERT$_{large}$ (2018)	86.7/85.9	72.1	92.7	94.9
RoBERTa (2019)	**90.2/90.2**	**92.2**	**94.7**	**96.4**

Table 2.1: NLP state of the art at the release of RoBERTa

These are the NLP fundamentals that have established the current document SoTA, which will be described in Section 2.3. As we will show, the current document SoTA follows a similar route, using these models and strategies fine-tuned for a document scope. However, we can identify a fourth key moment in NLP, which is the birth of the Large Language Model (LLM). This kind of model is a radicalization of the presented approaches, with larger models and quantities of data, following the same steps shown in this section. We will not present a discussion on these models as their basis has been fully covered in this section.

2.2 DATA AND QUALITY

It is evident that the data used by machine learning models dictates its learning, given how these models work. We structure our data quality discussions with some concepts from (Belgoumri et al., 2024). That paper surveys the use of the term “Data Quality” across the machine learning literature. This concept entails two different observations. First, it is not only dependent on the dataset itself but also on its context. Second, it can be described on a variety of attributes.

The survey presents a taxonomy of data quality attributes (dimensions) where the first-order dimensions are six. Statistical independence, attribute skew, label noise, fairness, privacy protection and reputation, trust, and security. The best data can produce a Bayes-optimal model. It will be large enough and drawn from the real distribution. In this sense, the authors highlight that the second dimension, the one of attribute skew, is the most important for making sure that the assumption that the sample data follows the real distribution is true. Essentially, the data may be biased according to how it was gathered. For image datasets based on real pictures, the location in which the pictures were taken and the hardware used may produce an innate bias in the data that the model may capture unwillingly.

To address this issue, the literature has produced a number of tools. One such tool is data augmentation. In a scenario where there is a global distribution and several local datasets with different biases, these local datasets can be augmented to bring them closer to the global distribution. Two such examples are in (Abay et al., 2020) and (Jeong et al., 2018), the latter of which is based on GANs. We further discuss some examples of data augmentation in a document scenario in Section 2.2.2.

Recently, this discussion has become an issue regarding the LLM technology, as it has been already began to see commercial usage. A phenomenon known as AI hallucination has been observed in commercial LLMs, where the model produces fake information (hallucinations). In (Rejeleene et al., 2024), there is a hypothesis that this has been an issue regarding the pre-training stage, due to information quality issues. This is very concerning as human curation of these datasets can be very expensive due to their enormous size.

However, these techniques for data understanding in the NLP scenario are still in a preliminary state of research. Some other domains of data science have produced an extensive body of work regarding data quality for various domains. For example, in (Liu et al., 2023), a systematic classification and analysis of existing studies on data quality in health information systems is presented. It is not only necessary to understand the possible dimensions of data quality in each domain, but also to effectively leverage it in practice.

Since our current SoTA has focused on training and modeling improvements, we believe looking at data issues will soon become a very important practice not just for pushing the SoTA forward but also for ensuring reliability across various domains. In Section 2.2.1, we present the main domains of research for documents and some of the datasets the current SoTA uses for evaluation.

2.2.1 Document Datasets

Recent document research has seen a lot of focus on a few domains. For example, we have the FUNSD dataset (Guillaume Jaume, 2019). This dataset is composed of noisy scanned documents in a form-like format. This is one of the datasets we use for the evaluation of our methods, and we detail it in Section 3.1. There is also XFUND (Xu et al., 2022), which features a similar format as FUNSD and functions as an extension to it. XFUND is composed of seven partitions, each for documents of a different language: Chinese, Japanese, Spanish, French, Italian, German, and Portuguese. An usual practice is to treat FUNSD as an eighth partition of XFUND, corresponding to the English language.

FUNSD is composed of some samples from a larger dataset, RVL-CDIP (Harley et al., 2015), which is itself a subset of the even larger IIT-CDIP (Soboroff, 2022). RVL-CDIP is composed of 400 thousand noisy pictures of scanned documents across training and testing, all of which are labelled as one of sixteen classes such as memo, letter, and scientific paper (25 thousand images per class). IIT-CDIP comes from the Legacy Tobacco Document Library and is a dataset of documents from the litigation between several US states and tobacco companies from the nineties. There are almost seven million records describing lawsuit documents. Due to the large size, this is the dataset usually chosen for pre-training of large document models.

Another dataset used in this work for evaluation is EPHOIE (Wang et al., 2021). This dataset is composed of scanned examination test headers from various chinese schools. It is also described in more detail in Section 3.1. The EPHOIE paper also presents a novel visual information extraction method that consists of a few branches for detection, recognition, and information extraction. Their evaluation procedure has been used as a staple for comparison in this dataset for later models.

Another common domain is the one of scientific papers. One reason for this is that documents of this kind are easily available in the web, and very large in number. In this scenario, there are datasets such as PubLayNet (Zhong et al., 2019b), PubTabNet (Zhong et al., 2019a) and DocBank (Li et al., 2020).

PubLayNet focuses on the layout analysis problem. It comes from the PubMed Central™ Open Access repository of journal article preprints. This repository contains over a million PDF and XML document instances, from which PubLayNet authors scraped the documents that make up the dataset. These were automatically annotated by matching the XML annotation to each respective PDF document. This dataset has been used by DocSynth (Biswas et al., 2021), a document generation GAN that we describe in Section 2.2.2. It features over 360 thousand document instances, which feature five types of entities: titles, text blocks, tables, lists, and figures. The sheer size of this dataset makes it comparable to other computer vision datasets of other domains such as ImageNet (Krizhevsky et al., 2012).

PubTabNet follows the same instance gathering and annotation procedure as PubLayNet, using PMCOA. However, it narrows the scope for dealing with only tabular data, featuring over 500 thousand tables in the HTML format of annotation. Also, that paper presents an encoder-dual-decoder model for table recognition, evaluated in the PubTabNet dataset itself.

DocBank also features over 500 thousand instances and features LaTeX documents that were annotated automatically. These annotations are generated by creating a colored outline in the output PDF for every entity, which is used to generate the bounding box in the end annotation. This dataset uses the same five types of entities from PubLayNet plus another seven: abstract, author, caption, equation, footer, reference and section. This dataset also features an extremely fine-grained annotation, down to the token level.

We also highlight the domain of invoices, for which at least two datasets exist as representatives: SROIE (Huang et al., 2021) and CORD (Park et al., 2019). SROIE stands for Scanned Receipts OCR and Information Extraction, and was presented in ICDAR 2019 for a competition on information extraction from scanned receipts. The dataset presents challenges for OCR (text localization and extraction) and key information extraction. This dataset contains a thousand annotated images of receipts across training and testing.

CORD stands for Consolidated Receipt Dataset, an invoice dataset that features high quality fine-grained annotations at a token level, and like EPHOIE, features two levels of entity class annotation, for individual segments (subtotal price, discount, service charge) and segment groups (such as subtotal, where all the previous classes would be subclasses of this super-segment). CORD has over 11 thousand receipts collected from Indonesian shops.

More recently, a number of different datasets were presented at ICDAR 2023. These datasets feature very challenging task across a large number of domains. We present a few of these that are closer to the domains we already presented in this section.

The competition on structured text extraction from visually-rich document images (Yu et al., 2023) features two novel datasets: HUST-CELL and Baidu-FEST. HUST-CELL contains 30 document classes across over 4 thousand document instances. This dataset also contains over four hundred key and value entity types, with fine-trained key-value pairs, nested keys and multi-line keys and values. These documents were collected from public websites and cover a large number of domains: receipts, certificates and industry licenses. Baidu-FEST uses a more commercial set of scenarios, for example from finance, insurance, logistics and customs inspection. It features 11 document classes in training and 10 in testing, and around 60 instances for each class.

DocILE (Šimsa et al., 2023) is used in the competition on document information localization and extraction. DocILE is a large-scale benchmark built for this task, featuring

almost 7 thousand business documents and another 100 thousand synthetic documents, as well as almost 1 million unlabelled documents for transformer pre-training. This dataset features a large amount of domain-specific knowledge in the business document scenario.

The FinTabNet (Zheng et al., 2021) datasets is used in the visual question answering on business documents challenge, with a focus on table recognition. This dataset features almost 90 thousand document pages with over 100 thousand annotated tables across training and testing partitions.

Lastly, the Document UnderstanDing of Everything (DUDE) (Landeghem et al., 2023) challenge features multi-domain, multi-purpose and multi-page documents. The challenge consists of a series of question-answer pairs related to information contained inside a given document instances. The DUDE dataset contains 5 thousand documents across training and testing, with almost 19 thousand question-answer pairs in the training set. This dataset presents a novel problem that's related to multi-page documents, a challenge that has not been presented by past datasets.

Finally, the next section deals with some data augmentation techniques in the document scenarios. There are many other document datasets generated using these augmentations, and they are described following the data augmentation techniques.

2.2.2 Document Data Augmentation

Most research in document augmentation focuses on creating new datasets. In the official document scenario, there is MIDV-2020 (Arlazarov et al., 2018), BID (Álysson Soares et al., 2020) and NBID (Wojcik et al., 2023). MIDV authors follow a simple augmentation pipeline consisting of pasting generated text onto background samples from Wikimedia Commons, names stemming from some name datasets available on the web, and other fields such as gender and birth being generated randomly following legislation from each country. These documents were then scanned, taken photos of, and filmed, generating the MIDV-2020 dataset, which contains the corresponding partitions: scans, photos and videos.

Both BID and NBID use roughly the same approach. The augmentation pipeline uses the image, text, and bounding box annotations to inpaint the sensitive text parts and paste new, synthetic info on top of the erased image. The inpainting is necessary since the input image is a real document, with sensitive data from real people. This also means experiments done using the real data cannot be reproduced, since the real partition cannot be made public since it contains sensitive information.

Donut (Kim et al., 2022) presents both an innovative OCR-free transformer approach and SynthDoG, an augmentation engine that is used to create instances to be used in the Donut pre-training stage alongside IIT-CDIP. SynthDoG stands for Synthetic Document Generator, and works with a rather simple approach where text is projected into a blank slate which is projected linearly into a background image. The backgrounds are sampled from the ImageNet dataset (Krizhevsky et al., 2012), the texts are sampled from Wikipedia, and the text layouts are created using a simple rule-based algorithm for grid stacking. The images are further processed to mimic image noise from real documents. For more details on the algorithm, refer to the Donut paper.

A simpler approach similar to Donut in the sense that it uses a repository of background images and a set of texts that are pasted into the images is DocCreator (Journet et al., 2017). This model couples this pasting process with some usual data augmentation techniques to create weathering and noise, simulating real noise from scanned documents. One of the main features from this model is the fact that the implementation features an annotation extractor with extremely

fine-grained annotation. This makes the augmentation tool useful for essentially any downstream task in DR.

On the other side of the spectrum, in (Raman et al., 2021) we find an augmentation engine that aims towards an annotation-free approach for the layout recognition problem. The proposed method uses a Bayesian network, which is structured as a set of rules to define the concept of a document. The authors define a document as an arrangement of a set of primitive elements such as paragraphs, titles and lists. The structure of the document is defined by that set of rules. The proposed network generates new layouts that are used by a set of image manipulation routines to extract the augmented document image.

GANs have been the staple for data generation for a long time, and they have been adapted for document augmentation as well. DocSynth (Biswas et al., 2021) is a GAN that creates new layouts using five structural elements from scientific papers: tables, figures, text blocks, titles, and lists. These five classes have been chosen for the baseline experiments from DocSynth because the model was trained and evaluated based on PubLayNet (Zhong et al., 2019b), which is a dataset that features these entity types.

Another use of GANs is in (Pondenkandath et al., 2019). In this paper, the authors tackle the historical document domain, augmenting images using a few of the SoTA GAN models (at the time): CycleGAN (Zhu et al., 2017) and VGG-19 (Simonyan and Zisserman, 2014). The authors train the model in a style-based augmentation technique where the model must create an older version of the document image, that is, a direct image-to-image translation task. Each GAN is trained separately using a different approach. CycleGAN is trained with the usual approach from its original paper, while the VGG-19 model in the Neural Style Transfer training style.

Lastly, in (Márk and Orosz, 2021) there is an overview of possible data augmentation methods. That paper focuses on text-centric approaches and goes through a few methods already used in NLP. It does not present any experiments, only an evaluation of each approach in terms of whether it can or cannot be used for the legal document scenario. The paper hypothesizes the use of ML-based text generators using LSTMs and LLMs (GPT-1 and 2, which were the ones available at the time). For LLMs, the authors conclude that they cannot be used because there is no way to protect certain keywords from the generation, which is a requirement for their scope.

The use of LLMs for data augmentation is already being put into practice in the NLP field. For example, in (Ye et al., 2024) and (Guo et al., 2023), some LLMs are employed for text rewriting. The idea of rewriting text using LLMs is proven to consistently improve performance in both cases, while the first paper shows that only *some* LLMs can generate new instances *ex nihilo*. We go into detail over these two papers in Section 3.2 to substantiate our first augmentation approach.

2.3 DOCUMENT RECOGNITION STATE OF THE ART

The current SoTA in DR follows a research line pioneered by LayoutLM (Xu et al., 2019). This model uses the baseline BERT architecture with a new embedding style and adapted pre-training objectives, effectively treating DR as a task of NLP with new dimensions (layout and image). To tackle these new dimensions, a new embedding type is presented. LayoutLM adds two extra embeddings that correspond to the 2-D positional embedding and an image embedding. The image features are extracted by using an F-RCNN, which works as an image backbone for this model. The text and positional embeddings are merged using an element-wise addition operation, while the visual (image) embeddings are added at the end of the pre-training procedure to aid in the fine-tuning stage.

LayoutLM also adapts the pre-training procedure by developing two new pre-training objectives. The Masked Visual-Language Model (MVLM) task corresponds to BERT's MLM, which is adapted to the new embedding style by feeding the model with the corresponding unmasked 2-D positional embeddings for the masked text embeddings, making sure that the model can correlate the positional and textual information. The Multi-label Document Classification (MDC) is designed to use the IIT-CDIP dataset (Soboroff, 2022) since this dataset features documents with multiple labels per document. In this task, the model must predict every label for the document independently, and the authors define a novel loss function for it. Since the multi-label classification is a specificity of IIT-CDIP that is not shared with most other document datasets, the authors define this second task as optional. It is of note that MDC is a supervised task. While the first model to introduce the BERT paradigm for DR uses a supervised task, this style of learning was abandoned in favor of other self-supervised tasks.

While LayoutLM is outdated, we present its contributions since we believe it is essential to understand how the current DR SoTA came to be, since the use of the NLP technique of self-supervised pre-training has become ubiquitous in said SoTA, with MVLM being widely used still today. LayoutLM served as the basis for LayoutLMv2 and 3 (Xu et al., 2020; Huang et al., 2022), the latter of which is still used in SoTA competitions of DR at the time of writing[3]. LayoutLMv3 innovated by placing the image and text embeddings side by side in a single embedding vector to represent the document at both training phases. The positional embeddings are again merged by element-wise addition for both image and text embeddings. The authors also dismember the MVLM task into the original MLM task and a brand new Masked Image Modeling objective, which functions the same as MLM, but masking the image tokens instead. The LayoutLM models were evaluated, among others, on the FUNSD, CORD and RVL-CDIP datasets.

Most of the recent advances in the DR SoTA come from three sources: adapting the pre-training stage, improving representation through more complex embedding modeling, and changes to the vanilla attention mechanism from the transformer. The architectures are usually the same, drawing all the way back to BERT. The LayoutLM series of models serve as examples of adaptations to the training and modeling that made it possible to apply the BERT approach to DR. Now, we discuss a few SoTA models that show how attention modeling can be performed.

First, we discuss MGDoc (Wang et al., 2022b), wherein "MG" stands for Multi-Granular. This model leverages the hierarchy that structures most documents for effective recognition. MGDoc authors use text, layout, and image features extracted at three different levels of abstraction from the document: page, region, and word. This multi-modal and multi-granular embedding uses a multi-granular attention mechanism that encodes the hierarchical relationships between regions and words. In short terms, two new bias terms are added to the vanilla attention function, respective to the hierarchy and relative distance between embeddings. These are trainable weights, where the hierarchy bias encodes the inside or outside relation that models the spatial hierarchy in the document image, and the relative distance bias encodes the relative Euclidean distance between each embedding's bounding boxes. MGDoc was evaluated on FUNSD, CORD, and RVL-CDIP.

ERNIE-Layout (Peng et al., 2022) continues the trend of incorporating NLP advances into DR through the use of the disentangled attention mechanism presented by DeBERTa (He et al., 2020). In DeBERTa, each word is represented by two vectors: the textual content itself and its positional encoding. The attention weights between words are then calculated through the use of disentangled matrices over contents and positions. This is in opposition to the vanilla attention, where the positional encoding is directly merged with the textual embedding, such that

[3]In particular, we highlight the 2023 ICDAR competitions.

both modes are represented through a single vector. In ERNIE-Layout, the attention calculation corresponds to a sum of four elements, which are themselves attention products between text and text, text and relative position (1-D distance within the sentences), text and x-coordinate, and text and y-coordinate, such that all of these scores are calculated independently and joined together at the end, as opposed to having all of them be entangled within a single embedding.

The ERNIE-Layout model also experiments with the *reading order* of the document, an approach that consists in running the list of entities through an off-the-shelf layout analyzer that serializes the entities according to the left-to-right, up-to-down order that corresponds to the western reading order. These entities are reordered on both text and visual embeddings, in a rather simple approach. The paper also presents four pre-training objectives, one of which uses this reading order idea to model a reading order prediction task, in which the model must indicate in a matrix corresponding to the relationship between input tokens which tokens are followed by the other in the input text. While the input is already serialized, this task makes sense because there is no explicit boundary between text segments in the transformer input. Among others, ERNIE-Layout was evaluated on FUNSD, SROIE, CORD and RVL-CDIP.

The idea of leveraging the reading order to enhance prediction is also explored in (Zhang et al., 2023). In this paper, the authors present the Token Path Prediction (TPP) task, a simple prediction head that works by predicting entity mentions as token sequences in a document. TPP models the document's layout as a complete directed graph, and as such the head is used to predict token paths between entities in the document graph. This leverages the fact that the entity recognition task in DR is often treated as a simple sequence tagging task, just like the Named Entity Recognition task in NLP. This model was evaluated on FUNSD and CORD, but it uses the LayoutLMv3 model as a baseline to raise its performance in some scenarios.

RORE (Zhang et al., 2024) is a Transformer model that changes the reading order prediction task by modeling it as an ordering of relations between layout elements, such that the model is trained on relation extractions instead of sequence predictions, which is the usual state-of-the-art approach for Transformers. The model first produces a reading order matrix that is then used to calculate the final attention score, as illustrated in Equation 2.2, where $\{p_{ij}^{i}\}_{i \leq i,j \leq n}$ represents the reading order relation matrix and λ^l is a learnable scalar that represents the weight of the reading order information at layer l in the Transformer. The reading order score can be understood as a bias factor that influences the relation between entities, meaning they are more likely to be related if one is read after the other. RORE was evaluated on FUNSD, CORD and SROIE, among others.

$$a_{ij}^{l} = \frac{exp((q_i^{l^T} + \lambda^l \rho_{ij})/\sqrt{d_k})}{\Sigma_{j=1}^{n} exp((q_i^{l^T} + \lambda^l \rho_{ij})/\sqrt{d_k})} \tag{2.2}$$

Following this line, treating visual documents as graphs is quite a common concept. While we also follow this idea for one of our proposed augmentation methods (see Section 3.3). Even models that pre-date the BERT approach such as PICK (Yu et al., 2020) may feature a graph module to leverage the inter-entity relationships in an explicit way. Some more recent models from the SoTA also use this idea. In particular, we discuss GraphDoc (Zhang et al., 2022) and Doc2Graph (Gemelli et al., 2023).

GraphDoc also imports an NLP approach in the document field. In particular, this model uses an idea pioneered by Star-transformer (Guo et al., 2019), where the attention mechanism is only performed between neighboring tokens (adjacent in the text) and a global entity representing the entire text. GraphDoc presents the graph attention layer using a similar idea, leveraging the spatial relationships between entities inside the document image. In GraphDoc, the entities

correspond to the nodes of the graph, and it has been evaluated on FUNSD, SROIE, CORD and RVL-CDIP.

Doc2Graph uses a Graph Neural Network (GNN) and a special graph representation module for documents. It is, as a novelty, not based on transformers. This model uses a fully connected undirected graph, where the nodes correspond to the semantic entities in the document. The authors experiment with two granularity levels: considering the minimal entity to be a text block and each singular word. The model designs the features differently for nodes and edges. Nodes are represented in the typical multi-modal style, the text being encoded using the spaCy (Honnibal and Montani, 2017) large English model to extract the text vector encoding, and the image encoded by a U-Net (Ronneberger et al., 2015) that was pre-trained on FUNSD for entity segmentation. Edges use the Euclidean distance between nodes and the polar coordinates for relative positioning. This model was evaluated on FUNSD, on the entity recognition task and a few other tasks including another subset of RVL-CDIP for entity recognition and segmentation.

LayoutMask (Tu et al., 2023) is another model that does not use image features for training. It introduces novel text-layout pre-training objectives in the form of Masked Language Modeling and Masked Position Modeling, two tasks that follow in the same line as the traditional MLM modes with the twist that image features are not used for prediction. LayoutMask also introduces word-level masking at pre-training level for the given objectives. LayoutMask was evaluated on FUNSD, CORD, SROIE and RVL-CDIP.

Table 2.2 presents an overview of the models presented in their evaluation scenarios. We report the micro-averaged entity-level F1 score on all cases, with the exception of RVL-CDIP, where the reported metric is the Accuracy. The number of parameters of LiLT depends on the text model that is coupled with the layout module that is also called, by metonymy, LiLT. The number of parameters reported here correspond to the total size of the layout module. A full explanation of this architecture can be found in Section 4.2.

Model	Modes	Year	# Params	FUNSD	SROIE	CORD	RVL-CDIP
LayoutLM	T+L	2019	343M	78.66	94.38	94.72	94.42
LayoutLMv2	T+L+I	2020	426M	82.76	96.25	94.95	95.25
LayoutLMv3	T+L+I	2022	368M	92.08	-	97.46	95.93
MGDoc	T+L+I	2022	203M	89.44	-	97.11	93.64
ERNIE-Layout	T+L+I	2022	355M	93.12	97.55	97.21	**96.27**
GraphDoc	T+L+I	2022	265M	87.77	**98.45**	96.93	96.02
Doc2Graph	T+L+I	2022	2.68M	82.25	-	-	-
LiLT	T+L	2022	11M	88.41	-	96.07	-
LayoutMask	T+L	2023	404M	**93.2**	97.27	97.19	93.8
RORE	T+L+I	2024	399M	91.84	96.97	**98.52**	93.8

Table 2.2: Document Recognition SoTA

2.4 POSSIBILITIES OF IMPROVEMENT

Recent advancements in the document SoTA, as we have seen in the previous sections, mostly work around the pre-training objectives and attention modeling. As seen in the NLP approaches, fine-tuning training parameters and focusing on data quality and quantity may be valid approaches for pushing the current SoTA. In this sense, we believe there is a lot to gain by leveraging the massive body of knowledge contained in LLMs, as well as its capacities for rapid learning.

Furthermore, there is a lack of research regarding document data augmentation. This might be explained by the complexity contained in a document instance, especially for domains with non-trivial layouts and complex image features. Usually, document models use embeddings that encode three modes of knowledge: image, layout, and text. So, ideally, Augmentation techniques should deal with all three cases or use a simpler model that requires less information.

Lastly, research has not managed to deal effectively with a few challenges such as documents with sensitive data. One such example is ID cards and passports. One such example is inpainting the images and generating fake texts to be pasted back (such as in (Wojcik et al., 2023)). While this approach creates safe data, since most models use image cues, the performance might be harmed since these embeddings are probably corrupted in the inpainting phase.

The approaches we design in Chapter 3 are an attempt to close the gap and present a few directions for future research in document data augmentation. We present an exploratory work that improves the baseline using novel data augmentation approaches, inching closer to solving the mentioned problems.

3 PROPOSAL

In this chapter, we present our augmentation proposals and the scopes in which they can be used. We also present the datasets we choose to augment with each proposal. Our LLM approach is used for FUNSD, which features varied text types, including some complex natural language sentences. For the template approach, we use EPHOIE, a simpler dataset with simple entity types. We present our definitions of a document, a template, and how we can use these definitions to create new approaches for document data augmentation.

Our methods are best fit for a specific working scope where gathering and annotating new document instances is hard or not possible at all. Typically, these scopes usually deal with documents containing sensitive or private information. A typical example is the one of official documents such as identity cards, where the documents are not easily available in the web and cannot be publicized.

This is also the case for the datasets we experiment with: FUNSD and EPHOIE. The instances in the FUNSD dataset come from lawsuits held against the tobacco industry in the nineties by the United States. In a legal scenario where documents are often unable to be made public, gathering more instances and annotating them is posed as a significant challenge, and augmenting already known instances serves as an alternative. EPHOIE is comprised of examination paper headers that contain sensitive information about the students and schools they come from, and as such the text pertaining to these labels had to be erased and re-synthesized for the dataset to be made public. Given this difficulty, this is another scope where gathering new instances is a hard task and therefore data augmentation becomes a viable alternative.

3.1 SCOPES OF WORK

We first define a document as of a list of entities, where each entity is an object that contains, at least, a textual string that corresponds to its OCR annotation, a bounding box corresponding to the entity's placement in the document, and a label, indicating the entity's class. Additionally, a document may define a set of key-value relations between entities. The list of entities is presented in an annotation file. The base documents have images from which these annotations are extracted, but our augmentation approaches do not take the image into account. Also, for datasets where the document image contains text that is not annotated, this text is treated as part of the background and ignored.

We define our two augmentation strategies based on this definition. We leverage two of the defining features of the document: the entities' text and placement from the document. The text augmentation is tackled using an LLM to provide rewritten versions of each entity's text. The LLMs in the literature contain a vast knowledge of the English language, the language in which these models are mostly trained in. Our approach is inspired by recent NLP developments (Ye et al., 2024; Guo et al., 2023) that perform a similar operation. This is our first approach, detailed in Section 3.2.

To validate the LLM approach, we picked the FUNSD dataset (Guillaume Jaume, 2019). FUNSD is an acronym for Form Understanding in Noisy Scanned Documents, and the dataset is composed of Form-like documents that were sampled from the larger RVL-CDIP dataset (Harley et al., 2015), meaning FUNSD is a subset of RVL-CDIP. It contains 199 documents, 149 for training and 50 for testing. As the dataset name suggests, FUNSD is composed of noisy scanned documents and stands out from RVL-CDIP due to its more fine-grained annotations. Unlike its

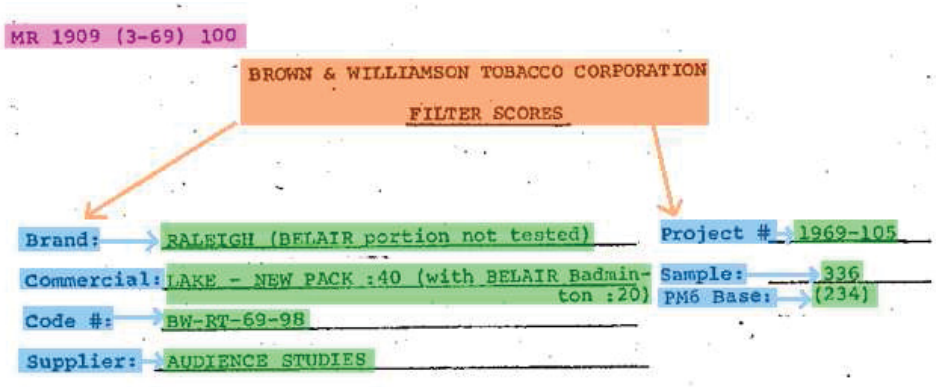


Figure 3.1: Example from a FUNSD instance

superset, FUNSD can be used for downstream tasks such as Semantic Entity Recognition (SER) and Relation Extraction (RE), thanks to the more detailed annotations. SER consists of the task of labeling every entity in the document with its correct label, while RE consists of extracting the inter-entity relationships of parent and child, using the annotation file as the ground truth in both cases.

Figure 3.1 features an excerpt from a document from the FUNSD dataset. Magenta represents an entity assigned the "other" label. Orange is the header, blue is the question and green is the answer, corresponding to the four entity classes defined by FUNSD. Arrows represent relations, such that they come from the parent entity and are destined towards the child entity.

The form of this dataset, as explained here, follows our definition of a document as described at the start of this section. Each document in FUNSD corresponds to a list of entities, every entity has a bounding box representing the position, the text written in said position, and a class. On top of this, there may or may not exist a key-value relationship between any two entities in the document. For FUNSD, this relation is structured such that this relationship may exist between headers (key) and questions (value), and questions (key) and answers (value).

Figure 3.1 also illustrates the SER and RE tasks: SER requiring the labeling of each entity and RE requiring the recognition of which relationships between entities exist. We highlight that some of the relations are not shown in Figure 3.1, for example between the header and the entity represented by the texts "Code #:" and "Supplier:". We have omitted these arrows for readability, but in this sample, the header is a parent to every question.

FUNSD has entity-level annotations for text and labels, gathered using a semi-supervised OCR mechanism detailed in their paper. We have found there are a few OCR errors, but have opted not to fix these manually as we directly compare our training protocols with the ones from the baseline model using the same dataset. Furthermore, the OCR errors do not impact our LLM augmentation neither positively nor negatively, since the LLM is able to distinguish the semantics from the text regardless of a few misplaced characters. The one exception to this is the empty string that some entities have for the text annotation. Usually, these entities correspond to graphical elements such as logotypes and signatures, which the OCR model failed to recognize. These entities are left without a text so that this is not a factor to consider when discussing the results of our augmentation procedure since the baseline model uses the faulty OCR transcriptions.

We also use the placement of the entities in the document to generate *templates* that encode information from the document's layout to generate new documents. These *templates* are extracted from the known documents, with identical templates being merged. The templates are iteratively chosen at random and their entities are filled with text according to their class. This

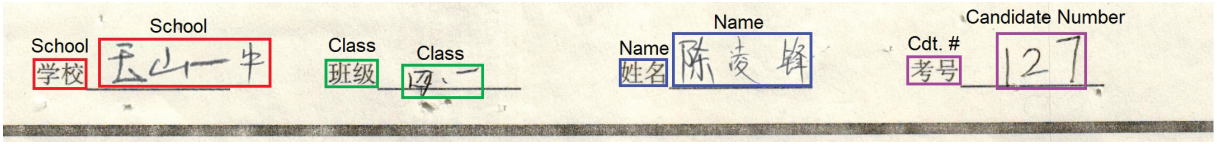


Figure 3.2: An instance from the EPHOIE dataset

approach is fit for simple documents with repeated layouts. We detail this approach in Section 3.3 and further detailed in a pseudo-code presented in Appendix A.

To validate the template augmentation approach, we choose the EPHOIE dataset (Wang et al., 2021). EPHOIE stands for Examination Paper Head Dataset for OCR and Information Extraction. It is a Chinese dataset composed of real examination paper headers from many schools across China and features a total of 1494 samples across training (1183) and testing (311). These documents feature scanned images, and the text is both printed and handwritten.

Figure 3.2 presents an instance from the EPHOIE dataset. This dataset features both an entity-level and a token-level label annotation, apart from the textual transcriptions and bounding boxes. In every entity, every token has a separate label corresponding to the token-level class, which may be one out of ten entity types, according to the paper. Each entity may also contain a label that corresponds to the Form-like annotation between key, value and none. As such, this dataset is fit for both SER and RE tasks, depending on which set of labels is used for training. However, since the baseline model does not perform RE on this dataset, we do not use this task for comparison.

Since our augmentation approaches do not easily carry over to images, that is, we virtually augment the text and layout without changing the images, we use the LiLT (Wang et al., 2022a) model for our experiments. LiLT is a pre-trained bi-modal transformer encoder developed for Document Recognition whose extracted features correspond only to layout and text. To the best of our knowledge, LiLT is currently one of the state of the art for this kind of bi-modal models where both the code and some of the pre-trained models are publicly available, making it possible to replicate the results. We explain the model's architecture and the design of the downstream tasks we perform as well as the training protocols in Section 4.2.

The LiLT paper evaluates the model on both of our selected datasets and reports the result for the SER and RE tasks on FUNSD and only SER on EPHOIE. Therefore, we perform our baseline comparison using the same fine-tuning setting. We compare our results to the ones reported in their paper as a baseline and also to our reproductions of their fine-tuning.

3.2 LLM APPROACH

We develop our LLM-based approach for FUNSD taking into account two features of this dataset. These are the presence of various complex natural language texts, and the complex layouts from each document. Since altering the layouts is a complex task given these forms are complicated and intricately designed, we employ a simple rewriting technique using an LLM for augmentation. This keeps the document's internal coherency intact while increasing textual variability.

This approach is inspired by a few papers from the NLP field. The idea of using LLMs to augment text by creating new, rewritten versions of it, has been presented in at least two papers (Ye et al., 2024; Guo et al., 2023). We discuss these papers from the text-only domain and their results briefly to explain our choices for the development of our technique.

LLM-DA (Ye et al., 2024) approaches the problem of Few-Shot Named Entity Recognition in textual data by using LLM-based data augmentation at Context and Entity levels. Context-level augmentations feature strategies for sentence length (longer, shorter), vocabulary

usage (advance words, adverbs, adjectives, prepositions, conjunctions), subordinate clause incorporation, and presentation styles (news, spoken language, magazines, fiction, Wikipedia, and movie reviews). The Entity-level augmentation features a single strategy of substituting entities in a given sentence by other entities with the same label, keeping the grammatical structure of the original text intact. These other entities can be generated by the LLM, beyond the known entities in the training dataset. This rewriting strategy was shown to be successful in the paper.

Furthermore, in (Guo et al., 2023), a simple rewriting of Question-Answer pairs is introduced using GPT-3.5 and GPT-4 to improve performance on a Q&A dataset. An important finding of this paper is that both evaluated LLMs were able to rewrite the existing Q&A pairs in a semantically meaningful way, boosting the baseline performance. However, the pairs that were *created* by the model were heavily dependent on the model's domain-specific knowledge. In their paper, GPT-4 was able to boost performance by creating the Q&A pairs, while GPT-3.5 wasn't. This is very limiting - since we do not have access to the entire training dataset of most LLMs, we cannot verify whether the model has the domain-specific knowledge for a given application in a way that is not experimental.

Given the results from the papers presented, we have opted to perform augmentation by only rewriting the entity texts using the LLM. The results from the NLP research show that augmentation by altering the presentation of data is sufficient to boost performance in downstream tasks. We extend this idea for the document scenario.

The rewriting technique developed is tailored for the FUNSD entities. A close inspection on the FUNSD documents reveals that their entities are very diverse in nature, and there is no one-size-fits-all strategy for augmentation. In particular, we identify four types of entities according to the nature of their texts. The first type corresponds to complex natural language sentences. The second type corresponds to simple questions, entities with generic nouns that expect a second entity such as "Name:", "Supplier:" and "Code Number:". The third type corresponds to simple answers, which would be the names, supplying companies and codes in question. We note that the simple question type does not necessarily correspond to the question label from the dataset, and the simple answer also does not correspond to the answer label from the dataset. These classes are created on top of the labels defined by the dataset itself. The remaining entities that cannot be classified in these three bins are the fourth class, which is "none".

In order to perform our augmentation, every entity is manually classified among these four bins. The first three bins are augmented, but not the fourth one, which is composed of texts that could not be augmented. For the complex sentences, we ask the LLM to rewrite the given text up to five times. The LLM provides both alternative syntax styles and vocabulary. This augmentation style is largely unsupervised in the sense that the model does not receive instructions on how the augmentation procedure must be carried out, only that it must rewrite the text a given number of times.

The LLM is also used for the simple question class. In this case, the model is asked to provide a list of synonyms for the text. For the simple questions, the model is not oriented to reach a certain number of augmentations, so the number of rewritten versions varies. For these augmentations, we use an off-the-shelf pre-trained Llama2 model (Touvron et al., 2023) with no fine-tuning. Llama-2 was shown to be inconsistent with output formatting, sometimes inserting unwanted extra text padding that had to be filtered out. [1] The LLM generations are revised manually in order to remove the filler and format the output in the manner of a list of strings, as well as to make sure that the generations make sense.

[1] These are textual replies such as "Certainly! Here's a list of synonyms:", "Thank you for asking! Here's some synonyms:", etc.

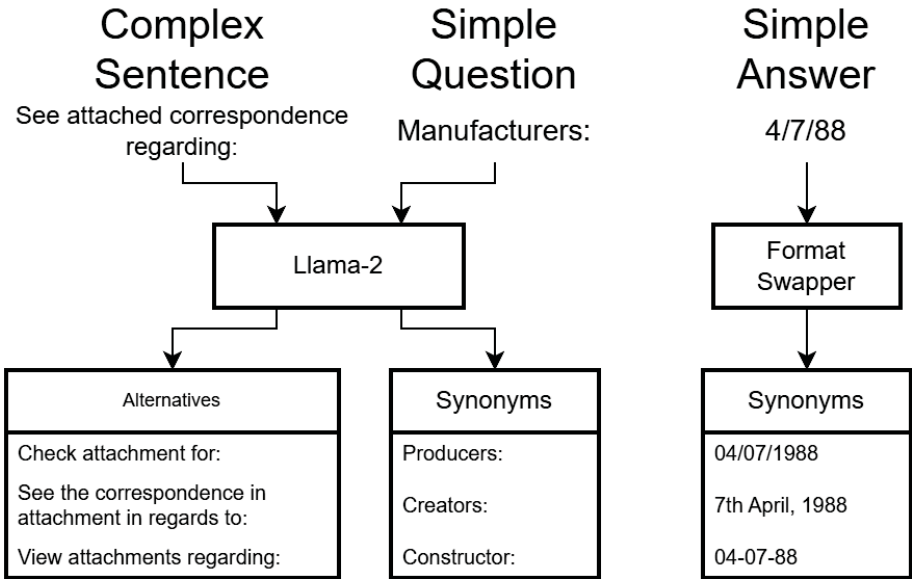


Figure 3.3: LLM Entity Augmentation Example

The third class, the simple answers, is augmented using a set of simple rules for augmentation. Names with initials are expanded via a dictionary, such that the initial becomes a random name from a dictionary of common english names, and full names are retracted into initials as well. They also change format, from “Surname, Name” into “Name Surname” and vice-versa. Dates are augmented by changing the format they are written in. The “MM-DD-YY” format can be changed to “Month DD, YYYY” and vice versa. Measures and numbers receive an OCR noise such that some of the digits are randomly replaced with different digits. For every entity in this class, a list of rewritten versions is created according to the rules presented and shuffled.

Figure 3.3 presents an entity example for each of the three classes. The last class is left untouched. Among these entities are texts with three characters or less, complex codes and unknown acronyms as well as empty strings which may be due to an OCR annotation failure. These are accounted for in Section 4.1.

Figure 3.4 presents a representation of our LLM augmentation approach. Entities on each document are manually annotated into each bin and forwarded towards their respective augmentation generators. As such, for every entity text, a number of new texts is generated, and a repository is created for the augmentations. For every document, each entity is assigned a list of possible entity texts that also contains the original text from the base document annotation. The un-augmented entities are assigned to a single-item list that corresponds to the entity’s original text.

Given these text lists, our approach for generating new documents is as follows. For each document, we create up to five new documents, such that the entities in each one of these have their texts are randomly chosen from their respective set of possible texts. This choice is uniformly random, meaning the n-th entity in the first augmented document does not necessarily receive the first augmented text in its list. The original entity text is part of the entity’s text list and can also be chosen.

Also, since the new text may have a different length from the original one, we use the top-left coordinates from the entity to simulate the writing of the new text in the document image. The text size is sampled from the document itself by analyzing the text and its bounding box. We leverage the word-level bounding box annotations to extract the character size, taking the

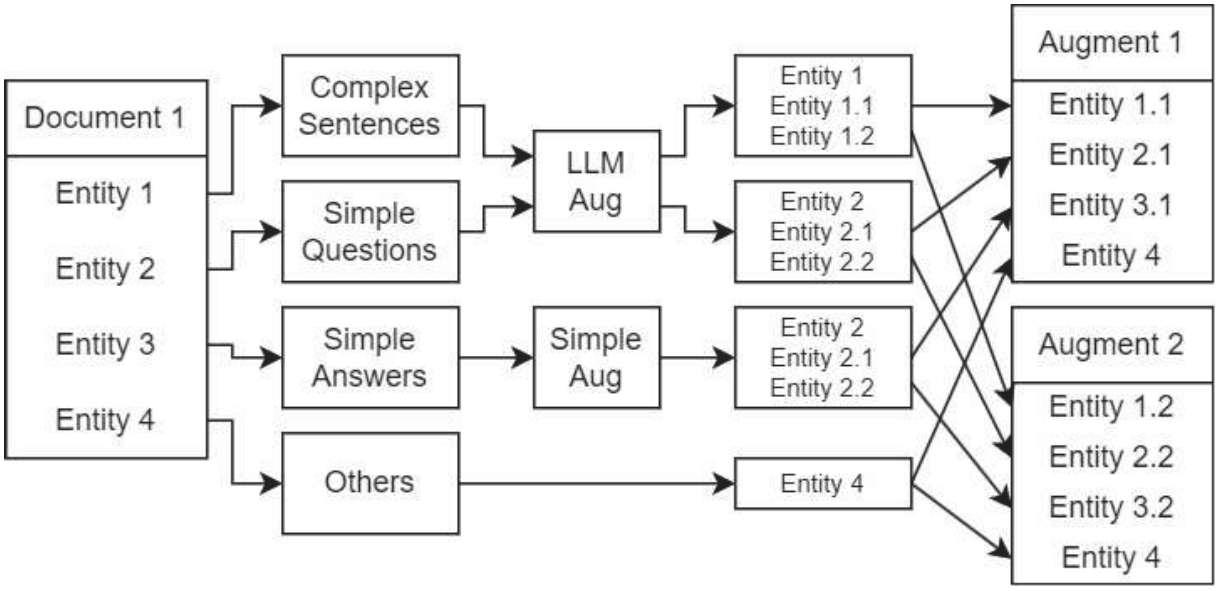


Figure 3.4: Diagrammatic representation of the LLM augmentation technique

absolute difference between the vertical values, and to find where the text wraps around to a new line, wherein the next word's top coordinate value is greater or equal to the last word's bottom coordinate value. With this, we can create a new bounding box that more accurately represents the new text. Also, since we keep the semantic structure of the document's entities intact, the key-value relationships remain the same.

Lastly, we have tried to use the template approach, detailed in Section 3.3, to augment FUNSD too, but it did not work well. We detail the text generation method used for EPHOIE, which was applied to FUNSD too. In summary, this method consists of swapping an entity's text with the text of a random entity from the dataset. The results were poor, which can be explained by the fact that for FUNSD, the document loses internal coherency, as there is no correlation between random entities when matching together complex texts from different documents. We have also refined this approach by picking only header-question-answer trees to fill the templates in such a way that the trees were composed by related entities. However, this approach was shown to be inefficient as the resulting augments harmed the model's performance. For this reason, this idea was scrapped.

3.3 TEMPLATE APPROACH

Our template-based approach is destined for a particular domain where the documents are pre-defined in a reduced set of *templates*. This means most documents share the same arrangement of entities within the image. An example of a scope such as this is the one of official documents. Usually, each country defines a set of rules for the layout of an identity document, for example, and all identity cards from that country, or from the same issuing organization, will follow the same layout, with every entity in roughly the same spot with a given background image.

It is with this in mind that we define the concept of a template. We define a template as a fully-connected directed graph that represents a document. The semantic entities contained in the document correspond to the vertices, each vertex containing a property corresponding to the entity type. Each edge contains a property called the direction. The direction can be one of eight - up, down, left, right and the four diagonals. The direction of an edge represents the spatial relation between the two entities it connects. Since the graph is directed and fully-connected,

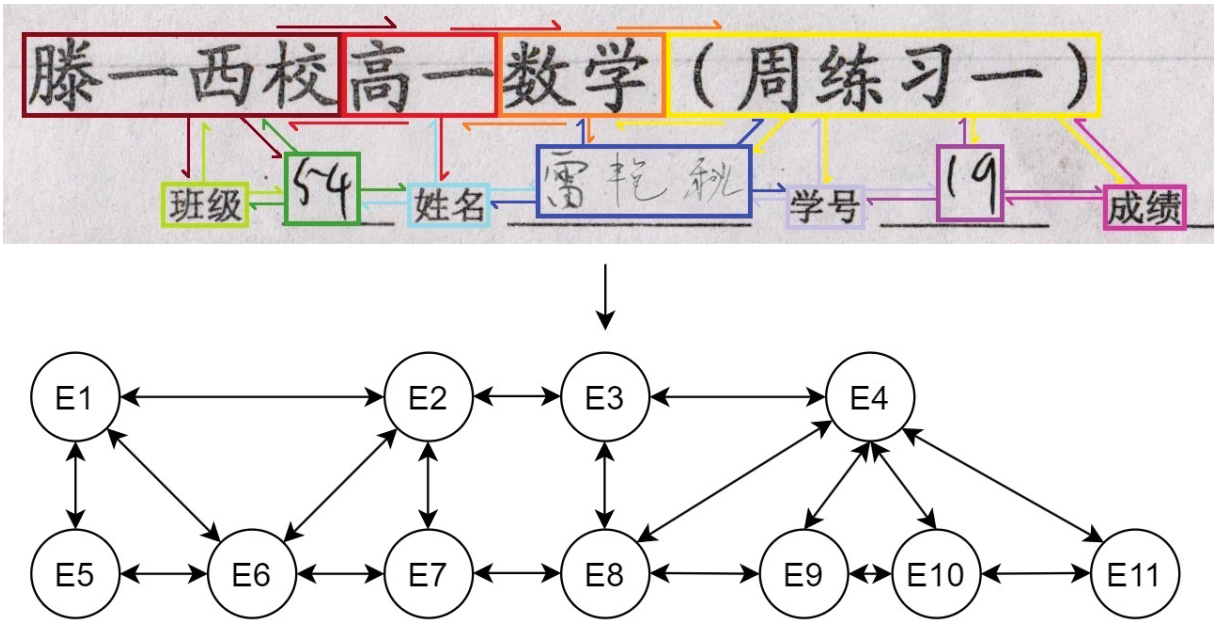


Figure 3.5: Template example from EPHOIE

there is one pair of edges between every entity. the edge that goes from edge *A* to *B* contains the direction towards B in relation to A, and the edge that goes from *B* to *A* contains the direction towards A in relation to B.

An example of a template is given in Figure 3.5. This figure is an excerpt from the EPHOIE dataset, which we chose for augmentation using the template method. The graph shown in Figure 3.5 is not represented as complete only for the sake of readability. The template object extracted from the EPHOIE documents is always fully-connected.

Since a single template can be the root of hundreds of document instances, we create our template augmentation approach by extracting templates from the known instances and using them to generate new, augmented document instances. If the templates are known, a simple augmentation procedure consisting of picking a template randomly and filling it out with new texts can be used to generate very representative datasets. This approach is similar to the way that new official documents are issued, where the holder's information is printed on a background image that contains a fixed layout. Our approach works by reverse-engineering the dataset to acquire the templates from the known instances.

The only issue with defining this approach is the matter of text generation. When issuing a new document, the details are provided by its holder. When creating synthetic data, there are a few possible approaches. Since the EPHOIE dataset has hundreds to thousands of instances per class, we create a text repository for each entity label, where this repository will contain all the text attributes across all entities of the same class in the dataset. Datasets with smaller distributions can rely on automatic generation via LLMs or use entity dictionaries available in the web for person names, organization names and the like.

Therefore, to generate a new document instance, a given template has its entities filled out with random texts from the repositories, according to the entity labels. The upside of using this method is that it is very simple, and works for any entity type. The downside is that it requires a significant amount of data for the dictionaries to be relevant, and it is not scalable. Since EPHOIE is fairly significant in size, with just over a thousand instances, this downside has not been a big problem in our experiments.

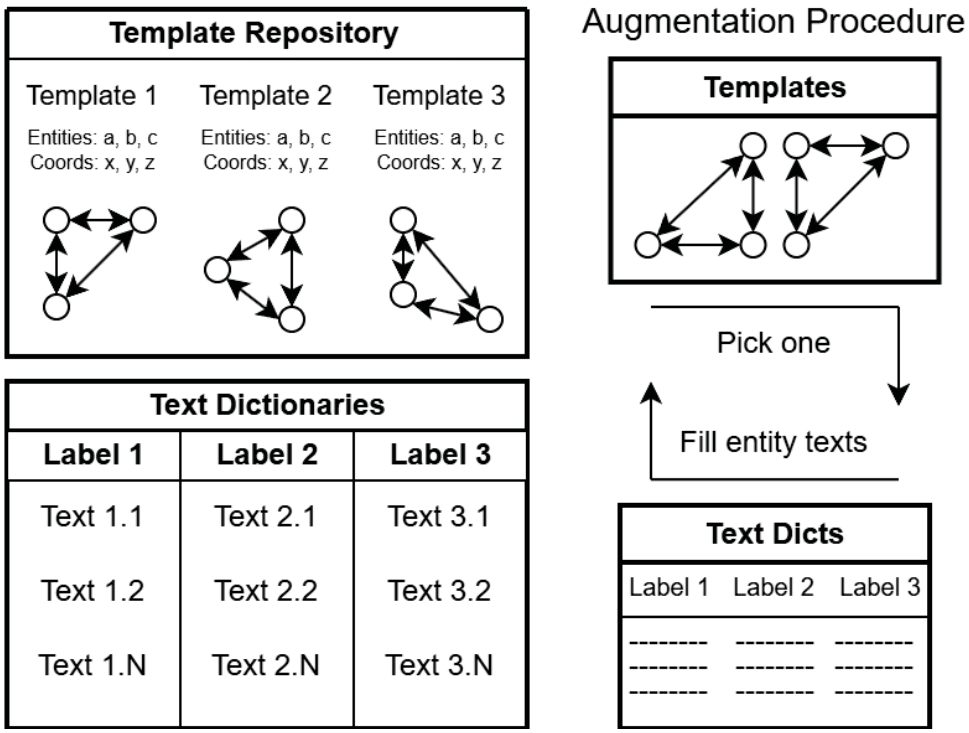


Figure 3.6: Schematization of the template augmentation

Our template augmentation approach is built using the presented building blocks. In this work, we use the training partition from the EPHOIE dataset as a basis for our augmentations, but the algorithm can be used for any document scope that falls into the definition we presented at the start of this section. The outline of our augmentation method is as follows.

- First, extract the template from every document instance in the original repository. Equal templates (rigorously either: the exact same graph or one template being a subgraph of another) are merged together.
- Then, compile a dictionary for every entity type, such that we assign to every entity type a list of entity texts corresponding to all possible texts in the known document entities that pertain to that given entity type.
- Finally, pick a template at random and fill its entities with corresponding samples from the entity type dictionaries, recalculating the bounding boxes with the new text's length and the entity's bounding box saved in the template. Repeat this step N times, where N is the total number of desired augmentation samples.

This approach is illustrated in Figure 3.6. Also, we provide a detailed pseudo-code for the main steps in this pipeline in Appendix A. The code for this augmentation process is fairly verbose, and the template object might need different extra information depending on the dataset used. For EPHOIE, we save the x and y coordinates for each entity, as well as the original entity label sequences. EPHOIE has multi-label entities, where every character is assigned a separate label. These entities are split according to the clusters of same-label characters, and this information is stored in the template object for reassembling at the end of the augmentation process.

Also, the main reason for using a graph representation for the templates is so that the newly generated dataset can have a balanced representation of the root templates. The source

dataset can have a variable number of documents pertaining to each given template, with an unbalanced distribution. If we list all the available templates, it is possible to generate a new dataset with an uniformly random distribution of these templates.

Our template augmentation approach, apart from improving the performance of the model (as per Chapter 5), also makes it possible to generate a large amount of instances using few examples, provided that the available examples contemplate a good amount of the possible templates. It is also possible to generate template-level augmentations for further variability. This would be done, for example, by randomly deleting or adding entities. Also, since this approach has an immense few-shot potential, it is possible to find known layouts in the internet to manually generate some root document instances for the template repository.

Finally, we remark that we cannot use the LLM approach to augment EPHOIE. This is because we would not be able to verify the LLM's generations due to our lack of knowledge of the Chinese language.

4 EXPERIMENTAL PROTOCOL

In this chapter, we present the datasets used for the baseline augmentation and describe them in terms of their entities. We also present the result of our augmentation procedures. Finally, we present and detail the model we used for validation, its innovations in terms of architecture and training, and how it was adapted for each downstream task.

4.1 DATASETS AND AUGMENTATIONS

We refer to the original datasets as presented in the literature, and the ones used as a basis for our augmentation, as the "real" datasets that contain real instances, and to the results of our augmentation processes as the "augmented" datasets, which contain augmented instances. In Table 4.1 we present the final number of instances and entities in the real and augmented datasets. It is important to note that only the training partitions are considered in the augmentation process, while the test partition remains always the same, only containing real instances. Also, the entities considered on both FUNSD and EPHOIE are only the ones that are annotated in the respective datasets. Other texts are considered part of the background and ignored. The model will be trained on partitions that contain both the real training set and the corresponding augmented partitions, such that the experiments with 1 augment of FUNSD are done by fine-tuning the model on 298 instances - the 149 real instances and their augmented counterparts. The same is true for EPHOIE.

For FUNSD, our augmentation model produces up to five instances for each one of the 149 documents in the real training set. A given document is augmented by replacing the entity texts with the augmented versions. This is done by choosing one of the possible augmentations at random, including the original entity text (which is required for the entities for which no augmentation was produced, for example). The augmented training sets are built incrementally from the previous partitions, such that the 2 augments partition contains the entire 1 augment partition plus another new 149 instances.

For FUNSD, we generate a number of different augmentations for each entity according to our approach detailed in the last chapter. Table 4.2 presents a relation of the number of entities and augmentations by type. In this table, we are considering only the *training* instances, such that the number of 7411 entities corresponds to the number of entities across the entire real training partition of FUNSD. Our augmentations on FUNSD use the pre-trained Llama-2-7b-hf (Touvron et al., 2023) for generation.

Table 4.1: Number of instances and entities for the EPHOIE and FUNSD datasets

	FUNSD		EPHOIE	
Partition	Documents	Entities	Documents	Entities
Real Train	149	7411	1183	12411
1 Augment	149	7411	1200	12921
2 Augments	298	14822	2400	25850
3 Augments	447	22233	3588	38656
4 Augments	696	29644	-	-
5 Augments	745	37055	-	-
Test	50	2332	311	3343

Table 4.2: Number of entities in FUNSD

Entity Amounts		Number of Augs		By Type	
Header	441	1	978	Complex	647
Question	3266	2	1580	Synonym	4106
Answer	2802	3	1560	Simple	375
Other	902	4 or More	1010	None	2283
Total	7411	Augments	14611	Augmented	5128

We experimented with a few prompts for Llama2, the most robust ones being rather simple. The prompts used for complex sentences and simple questions are, respectively:

```
Please rewrite the following text, maintaining the
semantics with different vocabulary or syntax:
```

```
Please provide a list of synonyms for the following
text:
```

As seen in Table 4.2, there is a significant imbalance in the number of entities pertaining to each label. Most of the entities in this dataset are composed of simple texts, which explains why more than half of them are augmented using the Synonym augmentation. The variation in the number of augmentations per entity is explained by the fact that the LLM does not always manage to produce the same number of semantically relevant augmentations for every entity, in both augmentation cases where the LLM was used.

In some cases, it was not possible to augment the entity at all. These cases are comprised of entities that contain texts composed of three characters or less, such as tickboxes, as well as entities with no text annotated. We assume that the latter is due to a failure of the OCR mechanism used in the annotation of the original dataset. A minority of these no augmentation cases is also comprised of entities to which the LLM did not produce any meaningful generations, such as entities belonging to the "other" label whose text consists of long codes of letters and numbers, acronyms the model did not understand, and chemical substances.

For FUNSD, which has 149 training documents, we create up to five augmentations per document, numbered one to five. Then, the n-th training set is constructed as the original 149 training instances from FUNSD plus the 1 to n augmentations of each training document. This means the base training set (the real train) contains only the original 149 training instances, the training set with one augment (the 1 augment) contains the real train set plus the first augmentation of each document, totalling 248. The training set with two augments contains the 1 augment training set plus the second augmentation of each document, totalling 447 instances, and so on. For EPHOIE, we create 3600 documents by choosing templates at random from the repository, and build the partitions 1 to 3 using subsets of 1200 synthetic documents. A curation of the synthetic samples revealed that 12 documents were malformed, with too many overlapping entities, and as such these documents were removed.

Table 4.3 presents the number of entities in the EPHOIE dataset. This table also only considers the real training partition. We can see that the amount of entities per label is also highly imbalanced in EPHOIE. However, EPHOIE is different from FUNSD in that each annotated semantic entity has two sets of labels. One of them indicates the form-like label between key, value, and none, and is applied to the entire entity. The second one consists of the twelve entity types presented in Table 4.3, and every token in the entity's text has a separate label, that may

Table 4.3: Number of entities in the real training partition of EPHOIE

Entity type	Real Train	1 Augs	2 Augs	3 Augs	Test
Other	5679	5930	11867	17769	1601
Exam Number	128	144	257	380	36
Score	377	398	827	1236	92
Name	2365	2400	4797	7172	620
Student Number	422	382	847	1245	108
School	1358	1396	2671	3989	335
Grade	441	458	958	1415	103
Seat Number	184	181	383	555	43
Class	1625	1686	3421	5088	438
Subject	376	395	836	1241	93
Candidate Number	467	476	940	1434	120
Test Time	79	72	154	240	13

differ from the label of other tokens in the same entity. This means that a line from the document that features both a student's name and number may be annotated as the same entity, but the characters corresponding to the name and number are labeled differently inside the same entity.

To cope with this peculiarity in the template repository building stage from our template augmentation method, we first split every entity in the training documents, clustering together same-label character groups. The corresponding bounding boxes are also split, such that the horizontal coordinates are interpolated according to the number of characters in the string relative to its size. This is equivalent to generating an annotation to the bounding boxes shown in Figure 3.5. The relation of which entities were split into which new entities is stored in the template object and used to reassemble these entities after the new documents are generated.

Furthermore, we note that there is a mismatch between the reported number of entity labels in the paper (10) from the number of entity labels in the dataset itself (12). The two entities that the EPHOIE paper seems not to consider are "other" and "candidate number", which are absent from the evaluation table in their paper. In the Appendix, LiLT authors follow the 10 entity remarks from EPHOIE's paper. We do not consider these entities for the experiments in order to accurately compare our results to the LiLT baseline.

4.2 MODEL AND TRAINING

To validate our approach, we use the pre-trained LiLT model (Wang et al., 2022a) for fine-tuning on both FUNSD and EPHOIE datasets. We reproduce a similar training setting as described in the LiLT paper for a fair comparison. LiLT is a double transformer model, consisting of two separate, independent transformer Encoder models, as represented in Figure 4.1 (Wang et al., 2022a). Each model is responsible for one mode between language (text) and layout. Each one of these models has its own attention flow, and they exchange information via a specially designed attention mechanism.

The LiLT attention mechanism is called a "Bi-directional attention complementation mechanism" (BiACM). In this model, each embedding (text and layout) is inputted into each sub-model, and the attention function (see Equation 2.1 for the vanilla version from the original transformer) is defined as,

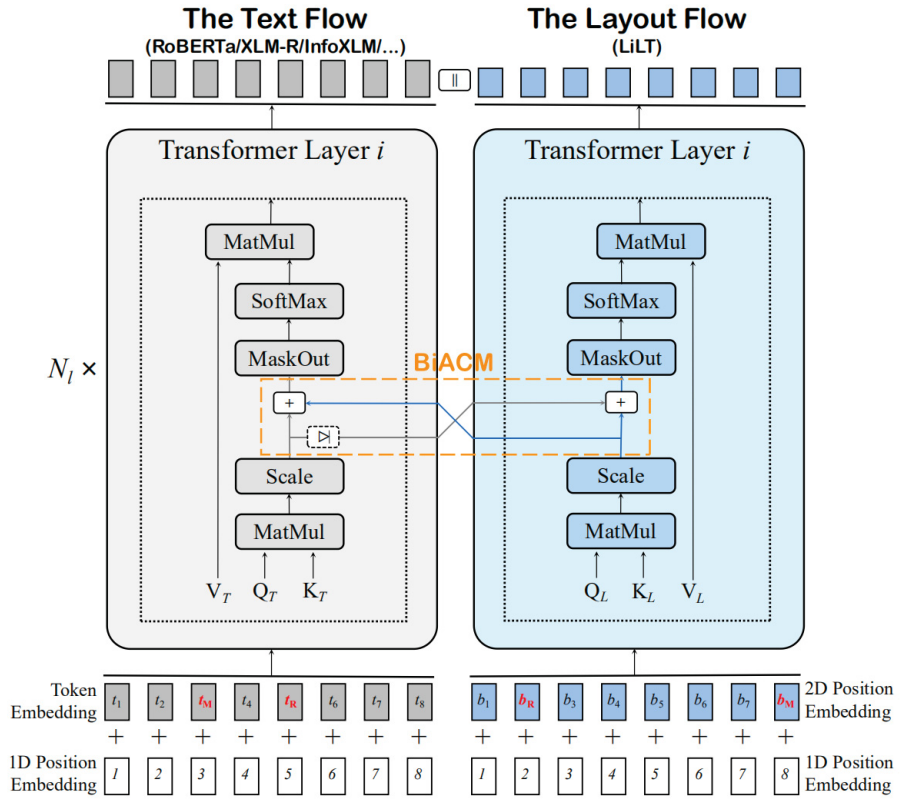


Figure 4.1: LiLT architecture

$$\widetilde{a_{ij}^{T}} = a_{ij}^{L} + a_{ij}^{T}$$
$$\widetilde{a_{ij}^{L}} = \begin{cases} a_{ij}^{L} + \text{DETACH}(a_{ij}^{T}) & \text{in pre-training} \\ a_{ij}^{L} + a_{ij}^{T} & \text{in fine-tuning} \end{cases} \tag{4.1}$$

where a_{ij}^{T} means the output of the vanilla attention mechanism from the textual flow, and a_{ij}^{L} is the output from the layout flow. Therefore, each attention mechanism takes as final output the element-wise sum between each vector from text and layout. However, the layout flow uses the detached version of the textual output in the pre-training stage. LiLT authors have opted to not let the textual gradient affect the layout model so that it can work with other off-the-shelf textual modules from the NLP SoTA.

This design of a dual transformer as done by LiLT gives the model an advantage: the layout model can be detached from the textual model and be coupled with other text models. The original LiLT paper uses this to fine-tune the model in many different datasets across various languages. For our purposes, we use LiLT-RoBERTa-EN, which is LiLT (the baseline layout model, dubbed LiLT by metonymy) plus the original RoBERTa model, trained in the English language. We also use LiLT-InfoXLM. InfoXLM is pre-trained in 94 languages, including the seven languages of XFUND and English.

Both hybrid LiLT models are publicly available, and we use them for evaluation. For FUNSD, we use LiLT-RoBERTa-EN for the SER task, and InfoXLM for the RE task. This is because LiLT authors report their best result on SER using the English model, but this model is not used for evaluation on the RE task, only InfoXLM. Therefore, we follow this line to accurately compare our results.

Pre-training on LiLT works with three objectives: Masked Visual-Language Modeling (MVLM), Key Point Location (KPL) and Cross-modal Alignment Identification (CAI). MVLM works just like MLM in BERT, but in this case the layout information remains unchanged. KPL consists of dividing the input into 49 squares in a 7×7 grid, masking some bounding boxes, and asking the model to predict the box edges into each square. CAI uses the encoded features of masked text and box pairs and asks the model whether these pairs are aligned or not (if they were replaced by MVLM or KPL).

The LiLT paper performs fine-tuning on three downstream tasks: Document Classification, which consists of assigning a label to a given document, Semantic Entity Recognition (SER), and Relation Extraction (RE). We perform SER and RE on FUNSD and only SER on EPHOIE. This is in accordance with the experiments done by LiLT, in order to accurately compare our results. As a task, SER consists of the problem of assigning a label to each entity inside a given document. For FUNSD, the labels correspond directly to the form-like labels (header, question/key, answer/value, and none), while EPHOIE defines twelve other semantic entities, as described in Table 4.3. RE is the problem of linking semantically related entities in a formulary. This means extracting the correct relations between headers and questions, and questions and answers.

Fine-tuning occurs differently for the SER and RE tasks. For SER, an additional layer is built on top of the output of the transformer. It predicts a token-level label for every token in each entity field. For RE, an additional head is added in a more complex scenario. This head is a pipeline, which begins by incrementally constructing the set of relation candidates by enumerating all possible entity pairs, the representations of which are then projected by two Feed-Forward Network layers, concatenated, and fed into a bi-affine classifier.

4.3 EXPERIMENTS

To validate our approach, we perform three sets of experiments on both FUNSD and EPHOIE using the new augmented instances: one to five augmentations per real image for FUNSD and one to three for EPHOIE. As usual for these datasets, these are fine-tuning experiments performed on pre-trained LiLT models. Our first experiment follows the same line used in the literature and consists in exhaustively training the model until the test set performance reaches a plateau, such that the testing set acts as a validation set. This experiment mirrors LiLT's original training and will be performed in order to check whether it is possible to surpass the performance achieved by the original model without augmentations. We perform it for both SER and RE tasks.

The second experiment is a n-fold protocol with cross-validation over 1000 epochs and an early stopping parameter of 30 epochs, experimentally found to yield a model that reached the performance plateau. We use 5 folds for FUNSD and 10 for EPHOIE. This difference is due to the size of the training set: 149 images for FUNSD and 1183 for EPHOIE. We aim at reaching a balance between the validation set size and the number of experiments. If the validation set is too small, the out-of-sample estimates will not be as accurate. If the number of folds is too small, it is harder to draw conclusions with statistical significance. The goal of this experiment is to ascertain, with statistical significance, that training with our data augmentation improves the performance in downstream tasks for the chosen datasets, and whether there is a specific number of augmentations that is ideal. For this experiment, we also perform both SER and RE tasks.

Finally, we perform a leave-one-out set of experiments, where the testing set is reduced to a single instance and all the other instances are added to the training set. We train for 100 epochs in FUNSD and 50 for EPHOIE, which are close to the average number of epochs reached

by the model in the fine-tuning of the n-fold experiments.[1] The goal of this experiment is to better examine the strengths and shortcomings of our augmentation methods through its results in the best possible scenario. We perform this experiment only with the SER task.

We use the base LiLT model, with maximum sequence length of 512, a 12-layer encoder of 192 hidden size, 768 feed-forward filter size and 12 attention heads. The model is fine-tuned on a NVIDIA Titan V GPU with an Adam (Kingma and Ba, 2014) optimizer. The first and second training protocols use a max number of epochs of 1000 and an early stopping parameter of 30. The third protocol trains the model for 100 epochs. For SER, the batch size is 16 with a learning rate of $1e-5$ and a warm-up ratio of 0.1, and the loss function used is the Cross-Entropy. For RE, the batch size is 8 with a learning rate of $6e-6$, also with a warm-up ratio of 0.1, and a Binary Cross-Entropy loss.

[1] There is no statistical significance in the difference between the number of epochs with or without augmentations.

5 RESULTS AND DISCUSSION

In this chapter, we present the results of the experiments we discussed in the last chapter and discuss these results, presenting the advantages and limitations of our approaches. Chapter 5 presents the results on both tasks on FUNSD and the SER task on EPHOIE. Section 5.3 presents a detailed discussion of the results and explains both fortes and shortcomings of our approaches.

5.1 FUNSD RESULTS

Partition	Test F1-Score
Real only (Reported) - RoBERTa-EN	88.41
Real only (Reported) - InfoXLM	85.86
Real + 1 Augment	88.82
Real + 2 Augments	**89.76**
Real + 3 Augments	89.04
Real + 4 Augments	<u>89.72</u>
Real + 5 Augments	89.02

Table 5.1: Results for the SER task on FUNSD

For FUNSD, we present our results for the first experiment on the SER task in Table 5.1. On this task, LiLT presents its best result using the LiLT-RoBERTa-EN model. However, they also evaluate using InfoXLM, yielding a slightly worse result. We fine-tune the pre-trained LiLT-RoBERTa-EN model for a fair comparison against their best result. We report the micro-averaged F1-score as calculated over the testing set. As seen in the results, our augmentations managed to beat the baseline in every scenario.

Still regarding FUNSD, we also evaluate our approach in the RE task, which is more challenging as seen by how the model's performance is worse in this scenario, in line with the rest of the state of the art. Unlike the SER task, in RE LiLT authors only use LiLT-InfoXLM for fine-tuning and evaluation, and no result using the RoBERTa-EN model is provided. In order to compare our results, we follow the same line of experiments and fine-tune the pre-trained LiLT-InfoXLM model using our augmented datasets.

LiLT presents various training scenarios for FUNSD, treating it as an extension of XFUND, a dataset similar to FUNSD that contains seven partitions in seven different languages. LiLT authors fine-tune the LiLT-InfoXLM model on all XFUND partitions (including FUNSD as an eighth partition) in three scenarios. Mono-lingual, Cross-lingual and Zero-shot Cross-lingual. Mono-lingual training consists of fine-tuning the model separately on each partition, once for each language. The Cross-lingual scenario consists of fine-tuning on all partitions at once, and the Zero-shot Cross-lingual is a simple evaluation of the fine-tuned model on FUNSD on the other partitions, with no further training. Our approach follows the *monolingual* scenario, since our augmentation technique is destined only towards FUNSD, and as such we only use this dataset in our experiments.

The results in the RE task, including the result reported by LiLT in the mono-lingual scenario, are shown in Table 5.2. Again, we report the micro-averaged F1-Score over the testing set, as evaluated by the LiLT-InfoXLM model. Here, we also manage to beat the baseline in every scenario, by a larger margin than in SER.

Partition	Test F1-score
Real only (Reported) - InfoXLM	62.76
Real + 1 Augments	64.4
Real + 2 Augments	68.44
Real + 3 Augments	**70.52**
Real + 4 Augments	70.35
Real + 5 Augments	69.55

Table 5.2: Results for the RE task on FUNSD

Partition	Mean Valid F1	Mean Test F1	Test F1 Stdev	Paired T-test p-value				
				0	1	2	3	4
0 Augs	82.68	85.44	0.53	-	-	-	-	-
1 Augs	84.21	86.78	0.78	0.025	-	-	-	-
2 Augs	84.15	86.67	0.42	0.004	0.618	-	-	-
3 Augs	84.02	86.98	0.51	0.015	0.276	0.159	-	-
4 Augs	84.05	86.44	0.19	0.02	0.759	0.793	0.955	-
5 Augs	83.54	86.6	0.78	0.023	0.762	0.584	0.804	0.374

Table 5.3: 5-Fold Fine-tuning SER on the FUNSD Dataset

Partition	Mean Valid F1	Mean Test F1	Test F1 Stdev	Paired T-test p-value				
				0	1	2	3	4
0 Augs	57.64	61.27	1.69	-	-	-	-	-
1 Augs	60.32	64.59	2.44	0.016	-	-	-	-
2 Augs	62.77	64.28	1.78	0.01	0.654	-	-	-
3 Augs	61.82	64.91	2.74	0.017	0.451	0.387	-	-
4 Augs	64.02	67.18	2.49	0.007	0.123	0.109	0.111	-
5 Augs	65.05	67.1	2.44	0.007	0.178	0.109	0.124	0.517

Table 5.4: 5-Fold Fine-tuning RE on the FUNSD Dataset

Partition	Test F1	Perfect Instances	Performed Better	Equal
0	89.82	5	16	12
1	90.56	6	22	12

Table 5.5: Overall Results for FUNSD Leave-one-out

Partition	Header F1	Question F1	Answer F1	Other F1
0	70.39	93.27	91.47	73.53
1	66.39	93.36	93.33	70.59
Number of Entities	119	1070	809	272

Table 5.6: Entity-wise F1 for FUNSD Leave-one-out

Our second set of experiments is the 5-fold with cross-validation. Tables 5.3 and 5.4 present our results in the SER and RE tasks, respectively. This protocol involves partitioning the training set into five folds of (more or less) the same number of documents. The model is trained five times, each time trained on all folds except one fold that is used for validation. The validation fold changes according to the training run, each fold being used for validation once. As usual, we use LiLT-RoBERTa-EN for SER and LiLT-InfoXLM for RE, and report the micro-averaged F1-score on the test and validation sets, averaged across the five runs. As usual, the test set for each fold remains the same across all augmented partitions: the real test set with no augmented instances. We can see the same trend suggested in the first experiments, where the improvement in RE is larger than the improvement in SER, and the augmented partitions manage to improve the results in every case.

This time, we use the results to perform a paired t-test, using the F1 scores as the sample measurements. As per the literature's usual practices, we set a p-value of 0.05 as the significance threshold. For this test, the null hypothesis states that the difference between the distributions is zero, meaning they are statistically the same. In Tables 5.3 and 5.4, we present all possible test p-values in the five last columns, testing how the column-wise partition relates to the row-wise partition in the statistical test. We use the alternative hypothesis that the first sample (indicated by the column) is smaller than the second sample (indicated by the row).

Our results indicate that fine-tuning with any number of augmentations yields a more effective model than training without augmentations, since the p-values in the first column indicate that we can reject the null hypothesis with a confidence of 0.05 for all five training sessions, for both SER and RE tasks. However, there is no way to discriminate between any given number of augmentations, since the p-value for all cases but one is not within the 0.95 or 0.05 thresholds. The only exception is in the SER task, between two and four augmentations, where the p-value is greater than 0.95, indicating the null hypothesis can be accepted, suggesting that training with two or four augmentations is statistically equivalent.

Our third and final protocol for FUNSD is the leave-one-out protocol. We train the model 50 times, each time leaving one of the testing instances out of the training set. The collected results across all training sessions are presented in Table 5.5. Here, we can see a marginal increase in the overall performance, where the augmented model performed equal to or better than the un-augmented one 34 times out of 50, yielding a perfect result in one extra sample.

But the question remains: where is the new model improving on performance? To answer this, we bring the detailed F1 result across each individual label in Table 5.6, as well as the confusion matrices in Figure 5.1. Here, we can see that performance gets worse for the classes with less instances in the testing set (a distribution that is reflected on training, as seen in

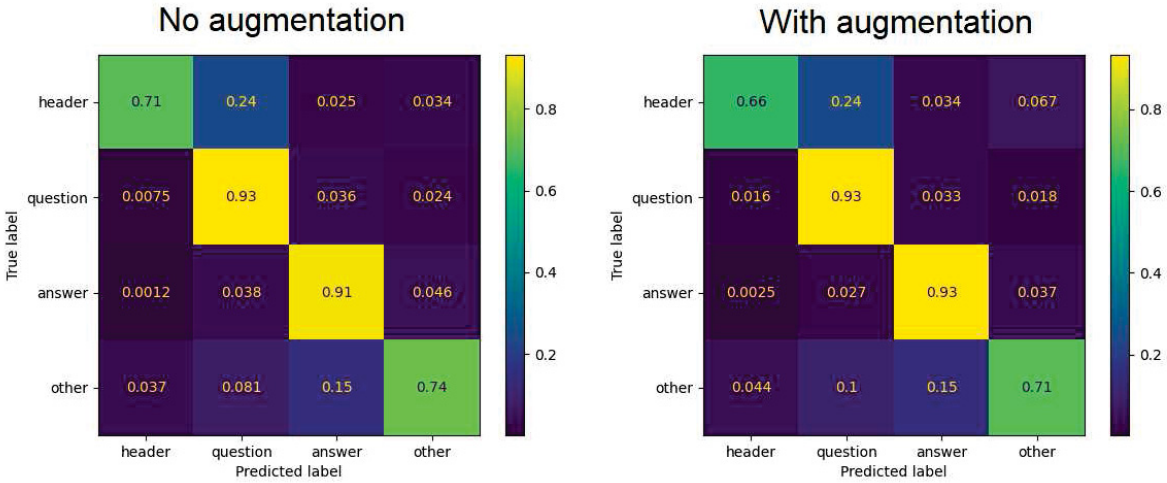


Figure 5.1: Confusion Matrices for FUNSD Leave-one-out

Partition	Test F1-score
Real only (Reported) - RoBERTa-ZH	97.97
Real only (Reported) - InfoXLM	97.59
Real + 1 Augment	**99.2**
Real + 2 Augments	99.19
Real + 3 Augments	99.13

Table 5.7: Results for the SER task on EPHOIE

Table 4.2), but marginally better for the more represented classes. In particular, the header-other confusion increases for the augmented version of the model.

5.2 EPHOIE RESULTS

Finally, we present the results on the SER task on EPHOIE in Table 5.7. The LiLT paper presents results using both LiLT-InfoXLM and LiLT-RoBERTa-ZH (Cui et al., 2020), the latter of which consists of a specially trained RoBERTa model for the Chinese language. Since this second model is not available in the official LiLT GitHub[1], we only fine-tune using the LiLT-InfoXLM model. As expected, LiLT reports a better performance when using the specially trained Chinese RoBERTa than the more general InfoXLM model. However, our fine-tuning of LiLT-InfoXLM still beats both baselines. Again, we report the micro-averaged F1-Score over the testing set.

[1]Available at `https://github.com/jpWang/LiLT/tree/main`

Partition	Mean Valid F1	Mean Test F1	Stdev	Paired T-test p-value		
				0	1	2
0 Augs	96.56	97.65	0.32	-	-	-
1 Augs	97.45	98.02	0.28	0.023	-	-
2 Augs	96.95	97.96	0.24	0.003	0.648	-
3 Augs	97.21	97.71	0.3	0.342	0.991	0.943

Table 5.8: 10-Fold Fine-tuning SER on the EPHOIE Dataset

Partition	Test F1	Perfect Instances	Performed Better	Equal
0	99.3	273	1	306
1	99.33	275	4	306

Table 5.9: Overall Results for EPHOIE Leave-one-out

Partition	Other F1	Grade F1	Subject F1	School F1
0	99.06	90.03	98.92	99.1
1	**99.88**	90.03	98.92	99.1
Number of Entities	1601	103	93	335
Partition	Test Time F1	Class F1	Name F1	Candidate Number F1
0	92.31	98.63	99.35	99.17
1	92.31	98.63	99.35	99.17
Number of Entities	13	438	620	120
Partition	Score F1	Seat # F1	Student # F1	Examination # F1
0	97.83	93.02	99.07	1
1	97.83	**95.35**	99.07	1
Number of Entities	92	43	108	36

Table 5.10: Entity-wise F1 for EPHOIE Leave-one-out

For EPHOIE, we also perform a 10-fold cross-validation training, an increase in the number of folds that is made possible by the larger training set. We follow the same line used for FUNSD, except this time our experiments are done with up to three augmentations instead of five, following the augmented datasets generated, and with ten folds instead of five. The results are reported in Table 5.8. We use the same testing scenario for the paired t-test.

As we can see, there is a statistical significance ($p < 0.05$) between no augmentations and up to two augmentations, but the performance decreases for three augmentations, and the p-value is no longer significant. However, it is also possible to ascertain that using three augmentations yields similar results to using one, as the large p-value suggests we can affirm the null hypothesis that the distributions are similar.

Finally, we report the leave-one-out results for EPHOIE in the same fashion as FUNSD, with the overall results reported in Table 5.9 and the entity-wise metrics reported in Table 5.10, split into three sub-tables so that all entities can fit into the page. Again, we see a marginal increase in performance, with two more perfect instances for the augmented model and four instances with improved performance versus one in the un-augmented case.

The performance between each individual entity is very similar, as seen in Table 5.10. For EPHOIE, we omit the confusion matrices as there is only one type of error: a switching of the Other and Seat number labels that the un-augmented model performs.

5.3 DISCUSSION

As we have seen in the results section, both of our augmentation methods manage to beat the baseline consistently in every evaluation scenario. For FUNSD, we have trained with up to five augmentations per instance for the same reason as to why we do only three augmentations for EPHOIE. These numbers were chosen experimentally, since our experiments have shown that this is where the model's performance reaches a plateau for each dataset and approach. For

the LLM approach, there are only so many meaningful synonyms, and paraphrases also have limited potential for improving the model's invariability towards vocabulary. For the template approach, we use a much larger dataset (149 vs. over a thousand training instances), and there is also limited variability since we do not expand the text corpus in this dataset.

Therefore, one of the limitations of our approach is the one of scaling. However, in the template approach this may only be a limitation because we are dealing with a dataset that is already significant in size. We hypothesize that scaling might be exactly one of the template approach's main advantages when dealing with limited amounts of data, provided a few significantly different samples are available. Text generation in this approach can be done via dictionaries and by simple rules such as the ones used in MIDV (Arlazarov et al., 2018), for example.

In general, we highlight that each method works best in distinctly separate cases. The LLM method is better for datasets with many varied and complex texts that span many contexts. LLMs are good at dealing with human language and this kind of text is very suited towards this augmentation technique.

The template method is better for datasets with simple text types. While it is designed for scenarios with a reduced set of templates, it worked well for EPHOIE, a dataset where almost every document is its own template. Nonetheless, this method still takes advantage of the fact that EPHOIE text types are specific and simple, which allows for indiscriminate text swapping.

Our LLM method is a first step towards the application of LLMs in the document scenario. We use it for data augmentation because we find that these models have a very large knowledge of the languages they were trained in, given the large amounts of data they were provided with during training. It shows to be very promising, given the results. We highlight that this method reaches a higher level of performance improvement when used to augment the corpus of the LiLT-InfoXLM training set.

InfoXLM is a cross-lingual NLP model that was trained in several languages, and the English examples in its pre-training are less numerous than those of RoBERTa-EN. We believe there is a bigger gap to be crossed in InfoXLM because of this. Since InfoXLM has less knowledge of the English language, and since our augmentation leverages the LLM's knowledge to provide the model with richer and more numerous instances of the English language, we improve the model's performance more than we would for RoBERTa-EN, since this model already has a larger knowledge of English. This would explain why the improvement for InfoXLM is bigger than for RoBERTa-EN. However, even for the English model we still see a meaningful increase in the evaluation metric.

The performance improvement seen in most cases is a reflex of what we call Syntax Invariability, which is how we call how we try to skew the model's learning through our augmentation approaches. In the LLM approach, we replace sentences by modifying their syntax, changing some words for equivalent synonyms, and changing some format conventions for dates, names, etc. All of this is done while keeping the semantics intact. We aim to force the model to learn only the meaning, making sure that an entity is classified the same regardless of its exact textual content.

The same is true for the template approach, but we go a little further in this case. In EPHOIE, by directly replacing texts using random samples from the dataset, we are also creating a template-wide invariability. This is because some schools might use the same template, but by allowing information from other templates to be placed in a given template, we force the model to become invariable to learning only templates of specific schools. This helps the model not reach an overfit state so quickly. However, this also limits the scaling of this approach in datasets such as EPHOIE, for similar reasons as the ones previously stated.

6 CONCLUSION

In this work, we have presented two new data augmentation methods for visual documents. We pioneer the usage of LLMs in this scenario with our first method, drawing inspiration from NLP research. We also introduce a new way of using layout information for document recognition at data level in the second method. Neither of our approaches rely on image features and augmentations, and as such we use LiLT, an imageless model, for validation.

Our validation experiments are split into three protocols: the one used for comparison in the literature, a n-fold cross-validation protocol and a leave-one-out cross-validation protocol. The goals for each experiment set were, respectively, to see whether the augmentation process manages to beat the baseline set by the model, to ascertain this improvement with statistical significance, and to examine the strengths and shortcomings of each method. The results show that augmented training manages to improve the baseline in almost all scenarios. Furthermore, our fine-tuning code, and augmentation results and source code can be found in GitHub.[1]

The present research opens up new possibilities for data augmentation in documents. We have shown there is potential for improvement when leveraging LLM technology and layout information for data augmentation, but whether directly using these for digesting the data during training, via a new language module in document Transformers will yield a more robust improvement is a question that will be answered with future research.

Another question is whether different, newer LLMs can yield better augmentation results than the one used in this work. New LLMs with increasing knowledge are being released constantly, and these new models may prove to be better suited for the data augmentation task proposed in this work.

Also in future research, it is possible to extend the ideas presented here for new datasets as well. Each augmentation technique is versatile enough within its scope of documents to be used in new scenarios, and there are several datasets in the literature, old and new, that could benefit from augmentation. This can also be done in order to further test each method's scalability, dealing with small and large datasets alike. A possible ablation study may involve further partitioning the training sets into smaller chunks in order to check for the impact of the amount of training data in performance.

Furthermore, bi-modal Transformers have shown to be both effective in document augmentation and lighter than multi-modal models, since there is no image module. Another direction in future research is to create a new state-of-the-art model using only the text and layout modalities for document recognition that can fully leverage our augmentation approaches.

[1] Fine-tuning code at `https://github.com/BOVIFOCR/DocAug-Finetune-Code`, datasets at `https://github.com/BOVIFOCR/DocAug/tree/main` and augmentation code at `https://github.com/BOVIFOCR/sampler`.

REFERENCES

Abay, A., Zhou, Y., Baracaldo, N., Rajamoni, S., Chuba, E., and Ludwig, H. (2020). Mitigating bias in federated learning. *ArXiv*, abs/2012.02447.

Arlazarov, V. V., Bulatov, K., Chernov, T. S., and Arlazarov, V. L. (2018). MIDV-500: A dataset for identity documents analysis and recognition on mobile devices in video stream. *CoRR*, abs/1807.05786.

Belgoumri, M. D., Bouadjenek, M. R., Aryal, S., and Hacid, H. (2024). Data quality in edge machine learning: A state-of-the-art survey.

Biswas, S., Riba, P., Lladós, J., and Pal, U. (2021). Docsynth: A layout guided approach for controllable document image synthesis. In *Int. Conf. on Document Analysis and Recognition (ICDAR)*.

Cui, Y., Che, W., Liu, T., Qin, B., Wang, S., and Hu, G. (2020). Revisiting pre-trained models for Chinese natural language processing. In Cohn, T., He, Y., and Liu, Y., editors, *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 657–668, Online. Association for Computational Linguistics.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Conf. of the North American Chapter of the Association for Computational Linguistics*, pages 4171–4186.

Gemelli, A., Biswas, S., Civitelli, E., Lladós, J., and Marinai, S. (2023). Doc2graph: A task agnostic document understanding framework based on graph neural networks. In Karlinsky, L., Michaeli, T., and Nishino, K., editors, *Computer Vision – ECCV 2022 Workshops*, pages 329–344, Cham. Springer Nature Switzerland.

Guillaume Jaume, Hazim Kemal Ekenel, J.-P. T. (2019). Funsd: A dataset for form understanding in noisy scanned documents. In *Accepted to ICDAR-OST*.

Guo, Q., Qiu, X., Liu, P., Shao, Y., Xue, X., and Zhang, Z. (2019). Star-transformer. In *Conf. of the North American Chapter of the Association for Computational Linguistics*.

Guo, Z., Wang, P., Wang, Y., and Yu, S. (2023). Improving small language models on pubmedqa via generative data augmentation.

Harley, A. W., Ufkes, A., and Derpanis, K. G. (2015). Evaluation of deep convolutional nets for document image classification and retrieval. In *International Conference on Document Analysis and Recognition (ICDAR)*.

He, P., Liu, X., Gao, J., and Chen, W. (2020). Deberta: Decoding-enhanced bert with disentangled attention. *CoRR*, abs/2006.03654.

Honnibal, M. and Montani, I. (2017). spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

Huang, Y., Lv, T., Cui, L., Lu, Y., and Wei, F. (2022). Layoutlmv3: Pre-training for document ai with unified text and image masking. *CoRR/arXiv*, abs/2204.08387.

Huang, Z., Chen, K., He, J., Bai, X., Karatzas, D., Lu, S., and Jawahar, C. V. (2021). ICDAR2019 competition on scanned receipt OCR and information extraction. *CoRR*, abs/2103.10213.

Jeong, E., Oh, S., Kim, H., Park, J., Bennis, M., and Kim, S. (2018). Communication-efficient on-device machine learning: Federated distillation and augmentation under non-iid private data. *CoRR*, abs/1811.11479.

Journet, N., Visani, M., Mansencal, B., Van-Cuong, K., and Billy, A. (2017). Doccreator: A new software for creating synthetic ground-truthed document images. *Journal of Imaging*, 3(4).

Kim, G., Hong, T., Yim, M., Nam, J., Park, J., Yim, J., Hwang, W., Yun, S., Han, D., and Park, S. (2022). Ocr-free document understanding transformer. In *European Conference on Computer Vision (ECCV)*.

Kingma, D. and Ba, J. (2014). Adam: A method for stochastic optimization. *International Conference on Learning Representations*.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. In *Int. Conf. on Neural Information Processing Systems (NeurIPS)*, pages 1097–1105.

Landeghem, J. V., Tito, R., Łukasz Borchmann, Pietruszka, M., Józiak, P., Powalski, R., Jurkiewicz, D., Coustaty, M., Ackaert, B., Valveny, E., Blaschko, M., Moens, S., and Stanisławek, T. (2023). Icdar 2023 competition on document understanding of everything (dude). In *Proceedings of the ICDAR 2023*.

Li, M., Xu, Y., Cui, L., Huang, S., Wei, F., Li, Z., and Zhou, M. (2020). Docbank: A benchmark dataset for document layout analysis.

Liu, C., Talaei-Khoei, A., Storey, V., and Peng, G. (2023). A review of the state of the art of data quality in healthcare. *Journal of Global Information Management*, 31:1–18.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019). Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Márk, C. and Orosz, T. (2021). Comparison of data augmentation methods for legal document classification. *Acta Technica Jaurinensis*, 15.

Park, S., Shin, S., Lee, B., Lee, J., Surh, J., Seo, M., and Lee, H. (2019). Cord: A consolidated receipt dataset for post-ocr parsing.

Peng, Q., Pan, Y., Wang, W., Luo, B., Zhang, Z., Huang, Z., Cao, Y., Yin, W., Chen, Y., Zhang, Y., Feng, S., Sun, Y., Tian, H., Wu, H., and Wang, H. (2022). ERNIE-layout: Layout knowledge enhanced pre-training for visually-rich document understanding. In Goldberg, Y., Kozareva, Z., and Zhang, Y., editors, *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 3744–3756, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Pondenkandath, V., Alberti, M., Diatta, M., Ingold, R., and Liwicki, M. (2019). Historical document synthesis with generative adversarial networks. In *2019 International Conference on Document Analysis and Recognition Workshops (ICDARW)*, volume 5, pages 146–151.

Raman, N., Shah, S., and Veloso, M. (2021). Synthetic document generator for annotation-free layout recognition. *CoRR*, abs/2111.06016.

Rejeleene, R., Xu, X., and Talburt, J. (2024). Towards trustable language models: Investigating information quality of large language models. *ArXiv*, abs/2401.13086.

Ronneberger, O., Fischer, P., and Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In Navab, N., Hornegger, J., Wells, W. M., and Frangi, A. F., editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pages 234–241, Cham. Springer International Publishing.

Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Šimsa, Š., Šulc, M., Uřičář, M., Patel, Y., Hamdi, A., Kocián, M., Skalickỳ, M., Matas, J., Doucet, A., Coustaty, M., and Karatzas, D. (2023). DocILE benchmark for document information localization and extraction.

Soboroff, I. (2022). Complex document information processing (CDIP) dataset. `https://data.nist.gov/od/id/mds2-2531`.

Touvron, H., Martin, L., Stone, K. R., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., Bikel, D., Blecher, L., Ferrer, C. C., Chen, M., Cucurull, G., Esiobu, D., Fernandes, J., Fu, J., Fu, W., Fuller, B., Gao, C., Goswami, V., Goyal, N., Hartshorn, A., Hosseini, S., Hou, R., Inan, H., Kardas, M., Kerkez, V., Khabsa, M., Kloumann, I. M., Korenev, A., Koura, P. S., Lachaux, M.-A., Lavril, T., Lee, J., Liskovich, D., Lu, Y., Mao, Y., Martinet, X., Mihaylov, T., Mishra, P., Molybog, I., Nie, Y., Poulton, A., Reizenstein, J., Rungta, R., Saladi, K., Schelten, A., Silva, R., Smith, E. M., Subramanian, R., Tan, X., Tang, B., Taylor, R., Williams, A., Kuan, J. X., Xu, P., Yan, Z., Zarov, I., Zhang, Y., Fan, A., Kambadur, M., Narang, S., Rodriguez, A., Stojnic, R., Edunov, S., and Scialom, T. (2023). Llama 2: Open foundation and fine-tuned chat models. *ArXiv*, abs/2307.09288.

Tu, Y., Guo, Y., Chen, H., and Tang, J. (2023). Layoutmask: Enhance text-layout interaction in multi-modal pre-training for document understanding. *ArXiv*, abs/2305.18721.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. *CoRR/arXiv*, abs/1706.03762.

Wang, J., Jin, L., and Ding, K. (2022a). LiLT: A simple yet effective language-independent layout transformer for structured document understanding. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7747–7757. Association for Computational Linguistics.

Wang, J., Liu, C., Jin, L., Tang, G., Zhang, J., Zhang, S., Wang, Q., Wu, Y., and Cai, M. (2021). Towards robust visual information extraction in real world: New dataset and novel solution. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Wang, Z., Gu, J., Tensmeyer, C., Barmpalios, N., Nenkova, A., Sun, T., Shang, J., and Morariu, V. (2022b). MGDoc: Pre-training with multi-granular hierarchy for document image understanding. In Goldberg, Y., Kozareva, Z., and Zhang, Y., editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 3984–3993, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Wojcik, L., Coelho, L., Granada, R., Führ, G., and Menotti, D. (2023). NBID dataset: Towards robust information extraction in official documents. In *Anais da XXXVI Conference on Graphics, Patterns and Images*, pages 145–150, Porto Alegre, RS, Brasil. SBC.

Wojcik, L., Coelho, L., Granada, R., and Menotti, D. (2024). Novos caminhos para aumento de documentos com templates e modelos de linguagem. In *Anais Estendidos da XXXVII Conference on Graphics, Patterns and Images*, pages 99–104, Porto Alegre, RS, Brasil. SBC.

Wojcik, L., Coelho, L., Granada, R., and Menotti, D. (2025). New paths in document data augmentation using templates and language models. In *Proceedings of the 20th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - Volume 2: VISAPP*, pages 356–366. INSTICC, SciTePress.

Xu, Y., Li, M., Cui, L., Huang, S., Wei, F., and Zhou, M. (2019). Layoutlm: Pre-training of text and layout for document image understanding. *CoRR/arXiv*, abs/1912.13318.

Xu, Y., Lv, T., Cui, L., Wang, G., Lu, Y., Florencio, D., Zhang, C., and Wei, F. (2022). XFUND: A benchmark dataset for multilingual visually rich form understanding. In Muresan, S., Nakov, P., and Villavicencio, A., editors, *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3214–3224, Dublin, Ireland. Association for Computational Linguistics.

Xu, Y., Xu, Y., Lv, T., Cui, L., Wei, F., Wang, G., Lu, Y., Florêncio, D. A. F., Zhang, C., Che, W., Zhang, M., and Zhou, L. (2020). Layoutlmv2: Multi-modal pre-training for visually-rich document understanding. *CoRR*, abs/2012.14740.

Ye, J., Xu, N., Wang, Y., Zhou, J., Zhang, Q., Gui, T., and Huang, X. (2024). LLM-DA: Data augmentation via large language models for few-shot named entity recognition.

You, Y., Li, J., Hseu, J., Song, X., and Hsieh, C.-J. (2019). Reducing bert pre-training time from 3 days to 76 minutes.

Yu, W., Lu, N., Qi, X., Gong, P., and Xiao, R. (2020). PICK: processing key information extraction from documents using improved graph learning-convolutional networks. *CoRR/arXiv*, abs/2004.07464.

Yu, W., Zhang, C., Cao, H., Hua, W., Li, B., Chen, H., Liu, M., Chen, M., Kuang, J., Cheng, M., Du, Y., Feng, S., Hu, X., Lyu, P., Yao, K., Yu, Y., Liu, Y., Che, W., Ding, E., Liu, C.-L., Luo, J., Yan, S., Zhang, M., Karatzas, D., Sun, X., Wang, J., and Bai, X. (2023). Icdar 2023 competition on structured text extraction from visually-rich document images. In Fink, G. A., Jain, R., Kise, K., and Zanibbi, R., editors, *Document Analysis and Recognition - ICDAR 2023*, pages 536–552, Cham. Springer Nature Switzerland.

Zhang, C., Guo, Y., Tu, Y., Chen, H., Tang, J., Zhu, H., Zhang, Q., and Gui, T. (2023). Reading order matters: Information extraction from visually-rich documents by token path prediction. In Bouamor, H., Pino, J., and Bali, K., editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 13716–13730, Singapore. Association for Computational Linguistics.

Zhang, C., Tu, Y., Zhao, Y., Yuan, C., Chen, H., Zhang, Y., Chai, M., Guo, Y., Zhu, H., Zhang, Q., and Gui, T. (2024). Modeling layout reading order as ordering relations for visually-rich document understanding. In Al-Onaizan, Y., Bansal, M., and Chen, Y.-N., editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 9658–9678, Miami, Florida, USA. Association for Computational Linguistics.

Zhang, Z., Ma, J., Du, J., Wang, L., and Zhang, J. (2022). Multimodal pre-training based on graph attention network for document understanding. *CoRR/arXiv*, abs/2203.13530.

Zheng, X., Burdick, D., Popa, L., Zhong, P., and Wang, N. X. R. (2021). Global table extractor (gte): A framework for joint table identification and cell structure recognition using visual context. *Winter Conference for Applications in Computer Vision (WACV)*.

Zhong, X., ShafieiBavani, E., and Yepes, A. J. (2019a). Image-based table recognition: data, model, and evaluation. *arXiv preprint arXiv:1911.10683*.

Zhong, X., Tang, J., and Yepes, A. J. (2019b). Publaynet: largest dataset ever for document layout analysis. In *2019 Int. Conf. on Document Analysis and Recognition (ICDAR)*, pages 1015–1022. IEEE.

Zhu, J.-Y., Park, T., Isola, P., and Efros, A. A. (2017). Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Computer Vision (ICCV), 2017 IEEE International Conference on*.

Álysson Soares, das Neves Junior, R., and Bezerra, B. (2020). BID dataset: a challenge dataset for document processing tasks. In *Anais Estendidos do XXXIII Conference on Graphics, Patterns and Images*, pages 143–146, Porto Alegre, RS, Brasil. SBC.

APPENDIX A – PSEUDOCODE FOR LLM AUGMENTATION

Here, we present the full algorithms for extracting templates from a document instance as well as augmenting new document instances using this method. The template object can be exported as a JSON file, while the documents are represented by an annotation file that contains a list of entities, each entity having, at least, a bounding box and a class label. The `extract_template` function below creates a graph (the `template` dictionary returned) that corresponds to the fully-connected graph we explained in Section 3.3. Its keys correspond to its nodes, which are the document's entities. The value of each key corresponds to a list of all remaining entities and its corresponding directions relative to the key entity. The input document corresponds to a dictionary where its keys correspond to the entities and its values are 2D coordinates for the top-left corner of the entity bounding box.

Algorithm 1 Template extraction routines

function GET-ANGLE($x1, y1, x2, y2$)
    $vec \leftarrow (x1 - y1, x2 - y2)$
    $orig \leftarrow (1, 0)$
    $angle\_radians \leftarrow$ ARCTAN2($orig[1], orig[0]$) − ARCTAN2($vec[1], vec[0]$)
    $angle\_degrees \leftarrow$ RADIANS-TO-DEGREES($angle\_radians$)
    if $angle\_degrees < 0$ **then**
        $angle\_degrees \leftarrow 360 -$ ABS($angle\_degrees$)
    end if
    return $angle\_degrees$
end function
function GET-RELATION($x1, y1, x2, y2$)
    $angle \leftarrow$ GET-ANGLE($x1, y1, x2, y2$)
    if $angle < 12.5$ **or** $angle > 347.4$ **then**
        return $RIGHT$
    else if $angle \geq 12.5$ **or** $angle < 77.5$ **then**
        return UP_RIGHT
    else if $angle \geq 77.5$ **or** $angle < 102.5$ **then**
        return UP
    else if $angle \geq 102.5$ **or** $angle < 167.5$ **then**
        return UP_LEFT
    else if $angle \geq 167.5$ **or** $angle < 192.5$ **then**
        return $LEFT$

```
    else if angle ≥ 192.5 or angle < 257.5 then
        return DOWN_LEFT
    else if angle ≥ 257.5 or angle < 282.5 then
        return DOWN
    else if angle ≥ 282.5 or angle < 347.5 then
        return DOWN_RIGHT
    end if
end function
function EXTRACT_TEMPLATE(document)
    n_entities ← document['entities'].length
    index_entities = INDICES(document.entity_types)
    all_coordinates ← Array(n_entities)
    for all i ∈ index_entities do
        all_coordinates[i] ← document.entities[i]['coordinates']
    end for
    knn ← KNN(all_coordinates, n = n_entities)
    all_neighbors ← Array(n_entities, n_entities − 1)
    for all i ∈ index_entities do
        neighbors ← knn.get_nearest(all_coordinates)[1 : n_entities]
        all_neighbors[i] ← neighbors
    end for
    template ← Dictionary()
    for all i ∈ index_entities do
        template[document['entities'][i]] ← Dictionary()
        for all j ∈ all_neighbors[i] do
            template[document['entities'][i]][document['entities'][j]] ←
                GET-RELATION(all_coordinates[i], all_coordinates[j])
        end for
    end for
    return template
end function
```

The following is an example of a template extracted from a document. Figure A.1 presents a graphical representation of the same template.

```
"template": {
   "e1": {
      "e4": "down_right",
      "e2": "right",
      "e5": "down_right",
      "e3": "right"
   },
   "e2": {
      "e1": "left",
      "e4": "down_left",
      "e5": "down_right",
      "e3": "right"
   },
   "e3": {
      "e1": "left",
      "e2": "left",
```

```
        "e4": "down_left",
        "e5": "down_left"
    },
    "e4": {
        "e1": "up_left",
        "e2": "up_right",
        "e5": "right",
        "e3": "up_right"
    },
    "e5": {
        "e1": "up_left",
        "e4": "left",
        "e2": "up_left",
        "e3": "up_right"
    }
}
```

Our template extraction algorithm also considers that the document dataset may feature identical templates that should be merged together. Also, the templates need to contain information about the entities' placement in the document for augmentation to be possible later down the line. The following algorithm illustrates our approach for creating the full template repository. For simplicity, we only consider the top-left coordinates in this code, and assume all entities are pasted in a single line.

Finally, the algorithm below illustrates our augmentation approach. In this algorithm N corresponds to the number of desired augmentations, which is taken as an user-given parameter for the generator. This algorithm is abstracting two elements: the text generation tool (in the `text_gen` function, which for our EPHOIE example corresponds to picking a random text sample from all entities from the training set with the same label) and the method of extracting the resulting bounding box. The bounding box can be extracted both by pasting the text in an empty canvas image and extracting the bounding box from the image itself (maximum and minimum non-blank x, y coordinates) or by calculating the final box as, for the x coordinates, the stored x-coordinate plus a product between the number of characters in the text and the average character length (which can be sampled from the dataset itself), and, for the y coordinates, the stored y-coordinate plus the average text height. In our approach for EPHOIE, we store the average character height and width for every entity pertaining to the documents in the same template.

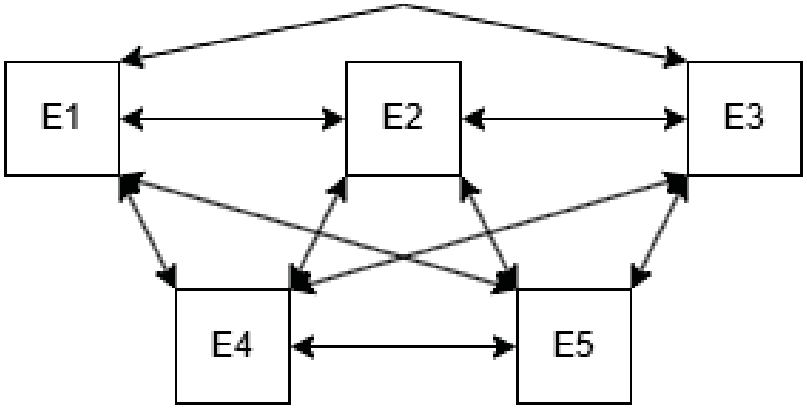


Figure A.1: Template example

```
function MERGE-TEMPLATES(templates, template, document)
    for all cur_template ∈ INDICES(templates) do
        if COMPARE-TEMPLATES(cur_template['graph'], template) = TRUE then
            templates[i]['documents'].append(document)
            return
        end if
    end for
    templates.append(Dictionary())
    templates[templates.length − 1]['graph'] ← template
    templates[templates.length − 1]['documents'] ← List()
    templates[templates.length − 1]['documents'].append(document)
end function
function COMPUTE-STATISTICS(template_full)
    template['entities'] ← Dict()
    for all entity ∈ template['graph'].keys do
        all_x_coords ← Array(templates['documents'].length)
        all_y_coords ← Array(templates['documents'].length)
        for all i ∈ INDICES(template['documents']) do
            all_x_coords[i] ← templates['documents'][i][entity][0]
            all_y_coords[i] ← templates['documents'][i][entity][1]
        end for
        x_mean ← MEAN(all_x_coords)
        y_mean ← MEAN(all_y_coords)
        template['entities'][entity] ← Dict()
        template['entities'][entity][x] ← x_mean
        template['entities'][entity][y] ← y_mean
    end for
end function
function CREATE-REPOSITORY(documents)
    templates ← List()
    for all document ∈ documents do
        template ← EXTRACT-TEMPLATE(document)
        MERGE-TEMPLATES(templates, template, document)
    end for
    for all template_full ∈ templates do
        COMPUTE-STATISTICS(template_full)
    end for
end function
```

Algorithm 2 Template augmentation routine

```
function AUGMENT-DOCUMENT(templates, text_dicts)
    template_full ← RANDOM(templates)
    augmented ← Dictionary()
    for all label, coordinates ∈ template_full do
        new_text ← RANDOM(text_dicts[label])
        new_box ← DRAW-BOX(new_text, coordinates)
        augmented[label] ← Dictionary()
        augmented[label]['text'] ← new_text
        augmented[label]['box'] ← new_box
    end for
    return augmented
end function
```