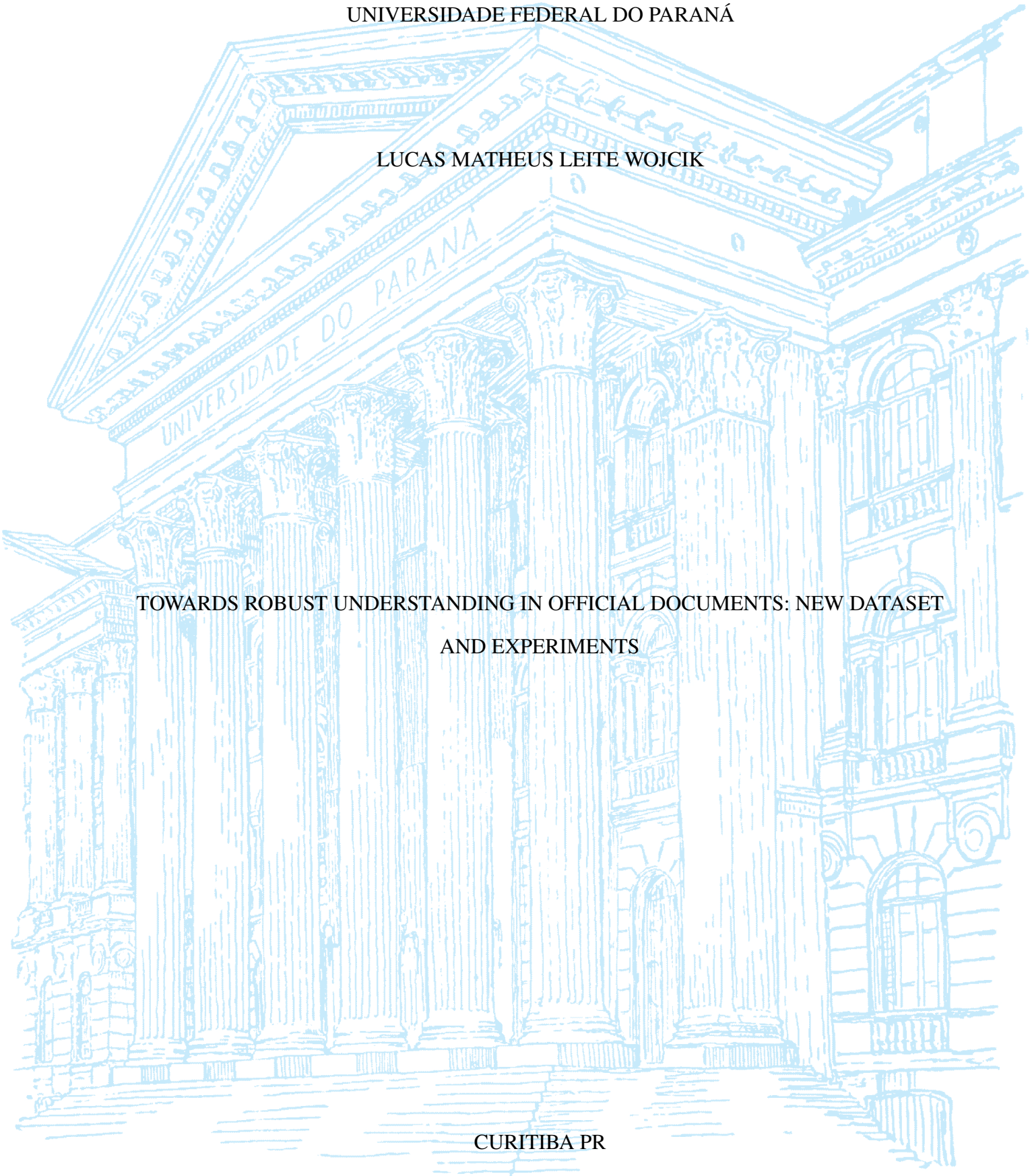
UNIVERSIDADE FEDERAL DO PARANÁ

LUCAS MATHEUS LEITE WOJCIK



TOWARDS ROBUST UNDERSTANDING IN OFFICIAL DOCUMENTS: NEW DATASET AND EXPERIMENTS

CURITIBA PR

2023

LUCAS MATHEUS LEITE WOJCIK

TOWARDS ROBUST UNDERSTANDING IN OFFICIAL DOCUMENTS: NEW DATASET AND EXPERIMENTS

Trabalho apresentado como requisito parcial à conclusão do Curso de Bacharelado em Ciência da Computação, Setor de Ciências Exatas, da Universidade Federal do Paraná.

Área de concentração: *Ciência da Computação*.

Orientador: David Menotti Gomes.

CURITIBA PR

2023

Universidade Federal do Paraná
Setor de Ciências Exatas
Curso de Ciência da Computação

Ata de Apresentação de Trabalho de Graduação II

Título do Trabalho: **TOWARDS ROBUST UNDERSTANDING IN OFFICIAL DOCUMENTS: NEW DATASET AND EXPERIMENTS**

Autor(es):
GRR 20195123 None: LUCAS MATHEUS LEITE WOJCIK

GRR _____ Nome:_____

GRR _____ Nome:_____

Apresentação: Data: 29 / 06 / 2023 Hora: 10:30 Local: https://meet.google.com/off-czef-wct

Orientador: DAVID MENOTTI GOMES _____

Membro 1: LUIZ EDUARDO LIMA COELHO _____

Membro 2: WAGNER M. NUNAN ZOLA _____

(nome) (assinatura)

AVALIAÇÃO – Produto escrito		ORIENTADOR	MEMBRO 1	MEMBRO 2	MÉDIA
Conteúdo	(00-40)				40
Referência Bibliográfica	(00-10)				10
Formato	(00-05)				05
AVALIAÇÃO – Apresentação Oral					
Domínio do Assunto	(00-15)				15
Desenvolvimento do Assunto	(00-05)				05
Técnica de Apresentação	(00-03)				03
Uso do Tempo	(00-02)				02
AVALIAÇÃO – Desenvolvimento					
Nota do Orientador	(00-20)		**********	**********	20
NOTA FINAL		**********	**********	**********	100

Pesos indicados são sugestões.
Conforme decisão do colegiado do curso de Ciência da Computação, a entrega dos documentos comprobatório de trabalho de graduação 2 deve deve respeitar os seguintes procedimentos: Orientador deve abrir um processo no Sistema Eletrônico de Informações (SEI – UFPR); Selecionar o tipo: Graduação: Trabalho Conclusão de Curso; informar os interessados: nome do aluno e o nome do orientador; anexar esta ata escaneada e a versão final do pdf da monografia do aluno.; Tramita o processo para CCOMP (Coordenação Ciência da Computação).

À voz incessante que me impele a prosseguir.

ACKNOWLEDGEMENTS

I'd like to thank my professor, David Menotti, for all the support in this long journey of graduation. I'm also thankful to Unico, the company supporting our research, which also provided the data so that our work could come to fruition. My special thanks to Luiz, Gustavo and Roger for the valuable insights that guided the development of this work.

A huge thanks to my research and graduation colleagues, and to my family for the support in hard times. I thank the support of my close ones, especially those who became far from me in the course of these last years.

RESUMO

A área de *Document Understanding* possui como objeto de estudo documentos, isto é, objetos contendo informações em formato visual e/ou textual, podendo ser arquivos de texto ou imagens. A subárea de *Visual Document Understanding* (VDU) trata especificamente do segundo caso, e é o objeto de estudo deste trabalho. Podemos conceitualizar o problema da área como o de extração automática de informação a partir de imagens de documentos, um problema com várias ramificações que atualmente é dominado por soluções de *machine learning*. De particular interesse é o cenário em que se quer extrair informações de documentos oficiais como cartões de identidade, que não podem ser disponibilizados publicamente por conterem dados sensíveis, e portanto mostra-se como um cenário difícil que ainda é pouco tocado pela pesquisa visto que as técnicas de *machine learning* depende de grandes volumes de dados para treinamento. Este trabalho apresenta um estudo da área de VDU, com uma revisão histórica do campo e do atual estado da arte, bem como dos *datasets* disponíveis e das ferramentas desenvolvidas para criar documentos sinteticamente. Também apresenta melhorias a uma ferramenta utilizada para sintetizar um *dataset* de documentos brasileiros, e um novo *dataset* seguindo o penúltimo formato de RGs físicos, implantado em 2019. O dataset será público, com o objetivo de viabilizar novas pesquisas envolvendo esse tipo de documento. Por fim, será realizado um estudo comparativo entre alguns métodos do estado da arte utilizando o novo *dataset*, tanto o sintético quanto os dados reais usados na sua síntese, e alguns outros disponíveis na literatura. Nosso objetivo é analisar a performance do atual estado da arte, descobrindo quais são as ideias propostas que funcionam melhor em cada contexto, e estudar o impacto da adição de dados sintéticos aos conjuntos de treinamento, decidindo enfim se essa técnica melhora a *performance* e se os dados sintéticos podem substituir os dados reais completamente.

Palavras-chave: Visão computacional. Compreensão de documentos visuais. Documentos oficiais.

ABSTRACT

The area of Document Understanding is the study of documents, that is, objects containing information in visual and/or textual format, in text or image files. The subarea of Visual Document Understanding treats the second case specifically, and is the subject of study in this work. The problem for this area can be conceptualized as the automatic information extraction from document images, a problem with several branching tasks that is currently dominated by machine learning solutions. Of particular interest is the scenario of information extracting from official documents such as identity cards, which cannot be made publicly available since they contain sensitive information, and as such it appears as a difficult scenario that is still mostly untouched by research since the machine learning solutions found in the literature rely on a big volume of data to be trained. This work presents a study of the area of VDU, with a historical review of the field and the current state of the art, as well as the available datasets and the tools developed to create synthetic documents. It also presents some improvements made over an existing synthesizer for brazilian documents, and a novel dataset following the last format of physical identity cards, implanted in 2019. The dataset will be publicly available, with the goal of enabling new research involving this sort of document. Finally, a comparative study between a few state of the art methods will be made, each method being evaluated on the new dataset, both the synthetic and the real data used for synthesis, and some others that are available in the literature. Our goal is to analyze the performance of the current state of the art, unravelling out which ideas work best, and to study the impact of the addition of synthetic data to the training sets, ultimately deciding whether it improves performance and whether it can replace the real data entirely.

Keywords: Computer Vision. Visual Document Understanding. Official Documents.

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

DU	Document Understanding
VDU	Visual Document Understanding
OCR	Optical Character Recognition
BID	Brazilian Identity Document
NBID	New Brazilian Identity Document
NLP	Natural Language Processing
MLP	Multi-Layer Perceptron
CNN	Convolutional Neural Network
RNN	Recurrent Neural Network
GAN	Generative Neural Network
ReLU	Rectified Linear Unit
CE	Cross-Entropy
BCE	Binary Cross-Entropy
MSE	Mean Squared Error
EoI	Entity of Interest
LSTM	Long Short-Term Memory
BiLSTM	Bilateral Long Short-Term Memory
CRF	Conditional Random Fields

CONTENTS

1 INTRODUCTION

The area of Document Understanding deals with tasks involving the extraction of information from documents. The object of study, the "document", is weakly defined and is thought of as many different things, depending on the context. Early OCR research referred to documents as bitmaps containing scanned text to be parsed, while modern NLP research may refer to a document as a sequence of text, and both approaches may be thought of as subareas of DU. This work is focused on the subarea of Visual Document Understanding, which considers a document to be an image, often in RGB format, which contains structured information encoded in the format of text, images and layout.

Many tasks performed within the domain of VDU, such as document classification, are highly relevant for automatized document processing. Document classification consists of assigning one class from a set of known classes to an instance of a document. This may be, for example, the distinction between an identity card and a driver's license. Without a reliable and automatic method for separating these documents, humans would be required to sift through each document and label them manually. This is a very lengthy task that incurs in less productivity and higher labor costs if done by a human.

Another important task is the one of entity labeling. It consists on assigning one class from a set of known classes to an entity within the document, and can be done at segment or token levels. Segment-level entity labeling considers the minimal entity to be a fully coherent block of information within the document, for example the name of the identity card's holder. Token-level entity labeling considers a smaller, if varied granularity. The token may be either each individual word within a segment or a smaller piece, down to the character level, depending on the tokenization process used.

This work focuses on official documents, a scenario that is mostly untouched by current research due to the sensitive nature of the data. The current state of the art in VDU utilizes machine learning technologies, which often requires a certain amount of labelled data to be trained and reach good accuracy levels. Since data as sensitive as personal documents cannot be made publicly available, public datasets must be entirely comprised of anonymized and/or synthesized data, which may affect the model's performance.

Our proposal includes a new, public dataset containing synthesized instances of brazilian identity documents, following the last format issued in 2019. To the best of our knowledge, this is the first public dataset to present this document format. We also propose several improvements to the synthesizer used to create the Brazilian Identity Document dataset, which includes older identity cards, driver's licenses and physical people registries, including a new inpainting method and a more accurate homography. Finally, we perform a state of the art analysis using our new datased, NBID (New BID), as a benchmark.

This work is organized as follows. Chapter 2 presents the basic principles of the field, nomenclatures and techniques used as well as the metrics used for evaluation. Chapter 3 is a review of the current state of the art in the field, complete with the datasets, synthetic data engines and models built for the tasks involved, as well as the key challenges found in the field. Finally, section 3.3 presents the conclusion and the detailed proposals for this work.

2 THEORETICAL BACKGROUND

The current state of the art for VDU is comprised entirely of machine learning models. In this section, we introduce the fundamentals of the field and present a brief overview of common neural networks used as subroutines within state of the art models, as well as the tasks and metrics in which such models are used and evaluated.

2.1 MACHINE LEARNING BASICS

Machine learning is a subfield of artificial intelligence focused on learning from data. Models contain a set of artificial neurons, separated into layers with connections between layers, and the knowledge is encoded within the weights given to each neuron as they are updated when new data is seen.

More specifically, the input is encoded as a vector of length N and passed into the input layer. Each dimension of the input vector is multiplied by the corresponding neuron's weight, and the results are summed and passed through an **activation function** that normalizes the amplitude of the output, usually between 0 and 1 or -1 and 1. Typically, a **neural network** contains **hidden layers** between the input and output, which may acquire a deeper understanding of the data by acting as a feature extractor. The input is passed through the layers sequentially, and the activation function acts as a **residual connection** between two layers, taking the output of the last layer and controlling its amplitude for the next.

The weights of the neurons are updated through the **backpropagation** algorithm, which leverages the gradient descent technique to lower the error. A **loss** function is used, taking as parameters the output of the last layer of the network and the expected outcome, and models how far from the ideal output the model is, that is, the current error. This error is propagated into the network from the last layer to the first, where each neuron has its weight updated: the new weight is the sum of the current weight and the product of the error by the neuron activation value by the **learning rate**, a parameter that encodes how fast the network should converge. Large learning rate values may decrease gradient accuracy (that is, the gradient moves at larger steps, possibly ignoring many points of local minima), while smaller values will cause the network to take a longer time to converge.

This is the basic functionality of a fully connected neural network, where all the neurons in a layer are connected to every neuron in the next and/or prior layers. Usually, we can have network containing sparser connections, or even fully connected networks that use a dropout technique, where some neurons are stochastically "killed" and do not appear in the computation anymore.

2.1.1 Activation and loss functions

The linear activation function is a simple function of the form $ax + b$, that leaves the output mostly unaltered, completely proportional to the input.

The rectified linear unit, ReLU for short, is a refinement of the linear activator that is set to zero if the input is negative. This means only half of the neurons will be triggered, on average.

A gaussian activation function is used for introducing non-linearity, and maps the input into a gaussian distribution. For example, $f(x) = exp(-x^2)$.

As for error modeling, the loss functions used are largely tied to the task to be solved. Classification tasks will often use the cross-entropy loss function. In 2.1, y_i represents the real probability that the input is actually the i-th class, x_i represents the output of the i-th neuron, and c is the number of classes.

$$CE(x, y) = -\frac{1}{c}\sum_{i=1}^{c} y_i \cdot log(x_i) \tag{2.1}$$

The binary cross-entropy loss is a special case of cross-entropy where there are two classes exactly, and is computed as in 2.2.

$$BCE(x, y) = -\frac{1}{c}\sum_{i=1}^{c} (y_i \cdot log(x_i) + (1 - y_i) \cdot (1 - x_i)) \tag{2.2}$$

Some image reconstruction and regression tasks utilize the mean squared error loss presented in 2.3, which encodes the average of the squared differences between the real values and the ones predicted by the network. For image reconstruction, n encodes the number of pixels.

$$MSE(x, y) = \frac{1}{n}\sum_{i=1}^{n} (x_i - y_i)^2 \tag{2.3}$$

Some more specific loss functions are used in other scenarios, tweaked according to the needs of the models, but are mostly variations on the losses presented here. The same holds true for activation functions.

2.1.2 Training Procedures

Training a network with a given dataset consists of forwarding every instance into the model, one batch of instances at a time, and adjusting the weights accordingly to each batch of data. Typically, the choice of a batch size considers a tradeoff between accuracy (larger batch sizes generally increasing the accuracy of the gradient estimation) and memory efficiency (since more instances per batch means more memory being used at a time). One full rotation of the entire dataset is called an **epoch**. The dataset used for training is called the training set. Datasets may be split into training and testing sets, where the testing section is only used for evaluation, and must be disjoint from the training.

Additionally, a third disjoint set may be used for validation, which is essentially a testing set that evaluates the network at regular intervals within training. The validation set is often used for **early stopping**, a technique that is used to find the optimal point to halt training. It consists of evaluating the network in the validation set at regular intervals, and halting the training once a consecutive number of training epochs failed to raise the model's accuracy to a new best. The number of epochs may vary, typical numbers are three and five.

Pre-training and **Fine-tuning** the model is a special training procedure that consists of an early training procedure in one dataset (the pre-training phase) and another, more focused training in another dataset (fine-tuning). Pre-training often utilizes special tasks for representation learning, such as Masked Language Modeling (MLM). MLM is done in NLP with a tokenized text input, and consists of masking 80% of the tokens in the input vector, randomizing 10% and leaving the rest unchanged, and asking the model to reconstruct the original sequence. This task was first presented by (Devlin et al., 2019) and has been widely used since then. Most recent models present their own pre-training procedures with specific goals.

2.2 COMMON NEURAL NETWORKS

2.2.1 Multi-Layer Perceptron

The MLP is one of the earliest examples of neural networks, and implements all the ideas in 2.1. It is a direct improvement of the Perceptron, proposed with a single hidden layer between the input and output. The **hidden size** is a parameter that dictates the dimensionality, that is, the number of neurons, of the hidden layers.

Today, the MLP is used as a submodule of larger models, and can be used as a classifier in a smaller subtask or as a feature extractor. In the first case, the output layer has the number of neurons equal to the number of classes, and in the latter only the output of the last hidden layer is used, as a feature descriptor.

2.2.2 Convolutional Neural Networks

CNNs have been of great relevance for image processing tasks, with the biggest breakthrough on image classification on the large ImageNet dataset being due to a large CNN architecture. These networks implement the convolution operation, which convolves a square matrix with an odd order, called kernel, over each pixel of the input image. The output for every pixel is calculated with the following procedure. The kernel is centered at the pixel and its new value is set as the sum of the products between the kernel elements and the corresponding pixels in the old image. Border pixels are often discarded.

This operation proved to be good at highlighting borders in the image initially, and as such CNNs became popular for segmentation tasks. However, the convolution also proved to be great at general pattern recognition, and so these networks have been largely used in much more varied scenarios.

This operation is used as a layer in the network, each neuron corresponding to one pixel. The residual connections (the neuron connections between subsequent layers in the pipeline) are often implemented through the ReLU activation function. Each convolutional layer is followed by a pooling layer, which halves the dimensionality by reducing each 2x2 square (the reduction operation may vary, the average value is used more often), effectively gathering information at lower resolutions.

The last layers are composed of MLP-like hidden layers and a final fully-connected layer, for classification. CNNs are often used with a CE loss, for the appropriate task, but some lightweight models such as ResNet may also be used as feature extractors.

2.2.3 Recurrent Neural Networks and Transformers

The RNN is a special type of neural network that contains cycles between neurons, which means the network's own output may influence its learning. These networks are often used in NLP tasks, as their recurrent nature makes them naturally better at processing sequential data, where the data at the end of the chain is partially determined by earlier data, for example in the case of natural language.

The Long Short Term Memory model incorporates the ability to forget, fixing the issue of exploding or vanishing gradients due to limited floating point precision. This is implemented through a special gate that multiplies the input by a weight matrix, where the result is closer to zero where the information is deemed less relevant. These weights are dynamically learned from the data. There are two other gates, for input and output. The input gate regulates the input with a sigmoid function, which attributes greater weight to more relevant information, and the

output is a hyperbolic tangent function that acts as a residual connection between the output of the sigmoid and the next layer.

Bi-Directional LSTMs function similarly to LSTMs, except there is a second network running in the opposite direction. The idea is that not only does the earlier data affect the subsequent data, the opposite is also true.

Transformers are a special type of RNN made as an encoder-decoder architecture, where each layer is composed of a multi-headed attention component and a feed-forward component. The attention function is implemented as a mapping between a set of queries and key-value pairs into an output, all of which are vectors of same dimensionality k. The inputs are projected into the attention space for each attention element through learned key (K), query (Q) and value (V) matrices. The output represents the compatibility between the queries and the keys. In the Transformer, a single attention head is a component that performs the transformation described in 2.4.

$$Att(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{2.4}$$

Multi-headed attention consists on computing attention several times in parallel in the same Transformer layer. For each head, the inputs are linearly projected with different learned functions, and the final result of the layer is the concatenation of the output of every head. This makes it possible to learn different aspects from the same data, adapting the network's weights accordingly.

2.3 TASKS AND METRICS

In this section, we present the most common downstream tasks in VDU and the metrics used for evaluation.

2.3.1 Optical Character Recognition

OCR consists of extracting the text from an image, both printed or handwritten. It is the oldest task in Document Understanding, and usually left to established, already existing methods such as Tesseract (Smith, 2007).

2.3.2 Document Classification

The task of classification, in a machine learning context, consists of assigning the correct class to an input. More formally speaking, given a finite set of known classes and an input represented as a numeric vector with an assigned class, the model is asked to predict the correct value from the set of classes. The input representation may be a set of feature vectors as well. For VDU, Document Classification is the particular case of classification where the input vectors represent the image of a document.

2.3.3 Entity Labeling

Entity labeling, or entity recognition, is defined as follows. Given a document containing a set of semantic entities pertaining to a finite set of known classes, assign the correct class to every entity in the document. Essentially, this is the same classification task at a lower level of granularity, where each document instance contains various entities to be classified.

This task can be done at different granularity levels, depending on what is recognized as a semantic entity. It varies from a section of the document to its elements - tables, paragraphs, images, down to each textual token. In this work, we focus on the segment-level entity labeling task, which considers an entity to be an entire entry in the document, not including the headers.

2.3.4 Key Information Extraction

The task of KIE is similar to Entity Labeling. Given a document instance, its entities and a set of classes, the model is asked to predict which entity belongs to each class, where every class is linked to exactly one entity in the document, and some entities may not have a class attributed.

Differently from Entity Labeling, where we wish to assign one label to every entity, possibly having duplicates, here we want to assign exactly one entity to every label. This task is also relevant in our domain, as often we would like to extract specific types of information from the document.

2.3.5 Entity Linking

Given a document containing a set of semantic entities, where each entity may be linked to another entity in a key-value manner, retrieve the keys and values contained in the document and their respective linkings. Entities may either be the key, the value, or neither. It is common for some datasets to include a header class as well, and some may even add another class representing entities that are none of the above, frequently being unimportant to the main document.

2.3.6 Named Entity Recognition

The task of NER is primarily tied to the NLP field, but also appears in some document understanding contexts. It consists of, for a sequence of text, find groupings of tokens representing a key information and assign to them the correct class. The entities to be found are significant elements of the information that are constantly referred to in the text: names, locations, dates, quantities, among others.

2.3.7 Common Metrics

The most common metric used to evaluate these tasks is the accuracy, simply defined as the ratio between the number of correct guesses and the total number of instances predicted. Some variants include the top-3 and top-5 accuracies. Most models assign a probability for each class, and the predicted class is defined as the one with the highest assigned probability. The top-3 metric considers a correct guess to be one in which the correct class is assigned one of the three highest probabilites by the model. The top-5 metric, accordingly, considers a correct guess as the right class being contained in the top 5 probabilities.

Another important metric is the F1-score, or F-measure. For a binary classification problem, where only two classes are possible (which can be arbitrarily called true and false), the F1-score is defined as follows in 2.5. Precision is defined as the ratio between true positives and true positives plus false positives. A true positive means the instance was correctly predicted as true, while a false positive means a false instance was incorrectly predicted as true. Recall is defined as the ratio between true positives and true positives plus false negatives. A false negative means an instance that is true was incorrectly predicted as false. The F1-score is the harmonic mean between the two metrics.

$$F1 = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall} \tag{2.5}$$

This metric can be extended into a multiclass scenario. In this case, the metric is measured separately for each class, that is, for a class a, a true positive is a correct guess, a false positive means the instance does not belong to class a, but was predicted as such, and a false negative means the instance belongs to a, but was predicted as any other class. The F1-score can be macro averaged into a single measure through the arithmetic mean of the scores across all classes.

Another way of computing the F1-score across all classes is by micro-averaging. In this case, the precision and recall scores are calculated jointly considering true positives and false positives and negatives for all classes at the same time, and the final score is computed with equation 2.5. The micro-averaged metrics of precision and recall are sometimes called mean entity precision and mean entity recall respectively, when the presented task deals with entity-level classification.

Accuracy is often used for Document Classification, while the micro-averaged F1-score is more often used for entity labeling, and the mean entity f-score is calculated for NER tasks. The entity linking task mostly uses a particular F1-score variant that is calculated among the possible key-value pairs between every entity. The top-N accuracies are also used for classification, particularly when the number of classes is very high.

3 RELATED WORK

In this chapter, we present the current state of the art overview for VDU and the datasets used in evaluation. We also describe a few data synthesizing methods found in the literature.

3.1 OCR AND VDU DATASETS

Name	Year of release	Nature of data	Number of instances
PubLayNet (Zhong et al., 2019b)	2019	PDFs of scientific articles	360k+
PubTabNet (Zhong et al., 2019a)	2019	Tables from scientific articles	568k
NIST special database 2 (Garris, 2008)	2016	Black and white synthesized tax forms	5590
NIST special database 6 (Dimmick et al., 1992)	2016	Black and white synthesized tax forms	5595
SROIE (Huang et al., 2021)	2019	Scanned invoices	1000
DocBank (Li et al., 2020)	2020	Scientific papers	500k+
EPHOIE (Wang et al., 2021)	2021	Admission exams	1494
EATEN (Guo et al., 2019)	2017	Tickets, passports and business cards	600k+
BID (Álysson Soares et al., 2020)	2020	Official brazilian documents	28k+
MIDV (Arlazarov et al., 2018)	2018	Passports and ID cards	3k images, 69k frames
Kleister (Stanislawek et al., 2021)	2021	Confidentiality agreements and charity reports	64k pages
CORD (Park et al., 2019)	2019	Pictures of invoices	11k+
RVL-CDIP (Harley et al., 2015)	2015	Scanned document images	400k
FUNSD (Guillaume Jaume, 2019)	2019	Scanned document images	199

Table 3.1: Document vision datasets

PubLayNet (Zhong et al., 2019b) is a very large and fairly recent dataset for document layout analysis. It is derived from PubMed Central™ Open Access repository of journal articles pre-prints, which contains over a million document instances in PDF and XML formats. The PubLayNet dataset is composed of a subset of these documents, which were automatically annotated by matching the XML document to the PDF. The dataset describes five types of semantic entities: titles, text, figures, tables and lists. It is comprised of just over 360 thousand instances, making it comparable to other computer vision datasets such as ImageNet (Deng et al., 2009).

Likewise, PubTabNet (Zhong et al., 2019a) is also derived from PMCOA, and contains an even larger number of tables extracted from it. The annotation follows the HTML format, which was also automatically extracted through matching the XML document to the PDF, and the dataset reaches a total of 568 thousand instances of tabular data. The PubTabNet paper also proposes an encoder-dual-decoder model for table recognition, evaluated in the proposed dataset.

NIST (Garris, 2008; Dimmick et al., 1992) provides a variety of special databases, which contain many types of digital data. Of special interest are the datasets 2 and 6, called "Structured Forms Reference Set of Binary Images". They consist of just above 11k pages of binary images of tax forms, filled with hand-printed data. These datasets are suitable for classical OCR tasks such as character recognition and segmentation, and can also be used in downstream VDU tasks such as entity-linking and form identification.

The ICDAR 2019 Robust Reading Challenge on Scanned Receipts OCR and Information Extraction (SROIE) (Huang et al., 2021) presented the homonym dataset, used for training and evaluation on the tasks of text localization and recognition and key information extraction. Across training and testing, the dataset contains 1k annotated images of scanned receipts. Apart from the aforementioned tasks, this dataset can also be used for the task of entity labeling, so long as only the annotated entities are considered.

DocBank (Li et al., 2020) is another large document dataset built for layout recognition, containing half a million instances of automatically labeled LaTeX-compiled documents. The instances are annotated with a weakly supervised approach based on segmenting the end document by adding a colored outline to the LaTeXsource file. The dataset defined the five elements from PubLayNet plus seven new entity types: abstract, author, caption, equation, footer, reference, and section. Differently from PubLayNet however, the annotations are fine-grained down to the token level, allowing not only NLP tasks to be performed with it, but also for more refined VDU models to use the textual information as an embedding.

EPHOIE (Wang et al., 2021) is a collection of 1,494 images of examination tests from various chinese schools. The images are scanned and cropped to the head, which contains most key information about the school, student and test itself. These informations are anonymized and synthesized to protect privacy. The paper also proposes a new visual information extraction model, divided into detection, recognition and information extraction branches, the last of which is based on a multi-headed attention layer for learning.

Entity-aware Attention, or EATEN (Guo et al., 2019) is the name of both a model and a dataset in VDU. The model is an encoder-decoder RNN that uses a CNN as backbone for visual feature extraction. The dataset is composed of over 600k instances, split across train tickets, passports and business cards. All of the data is synthetic, except for 1.9k real train tickets. The synthesizing is done by a proposed engine in three stages: text preparation by web scraping, font rendering on prepared backgrounds and pure images without text, and some data augmentation techniques, namely rotation and resizing, and application of several types of noise. This dataset includes segment labels for all types of data, and as such may be used for segment-based downstream tasks, even though the paper focuses on simple OCR for evaluation.

The BID, or Brazilian Identity Document dataset (Álysson Soares et al., 2020), is composed of three types of documents: identity cards, driver's licenses and physical person registries. The data is synthesized via a publicly available engine, which will be described in detail later. Unfortunately, although the segments are properly annotated with their respective transcripts, there is no set of classes for each segment, which limits the usefulness of the dataset.

MIDV-2020 (Arlazarov et al., 2018) is a dataset composed of 2k scanned images, 1k photos of mock identity documents and 1k video clips, with unique text field values and artificially generated faces. The documents are identity cards and passports, the background images of which were obtained from Wikimedia Commons, with the sensitive text erased and re-synthesized, and faces generated through the StyleGAN (Karras et al., 2018) model. Videos have individual annotated frames. With over 72k total annotated images, this is the largest official document dataset to date, to the best of our knowledge, and features documents from many european and asian countries.

Kleister (Stanislawek et al., 2021) is composed of two datasets of long, visually complex documents spanning many pages each. The NDA, or Non-Disclosure Agreement dataset, is composed of just over 600 documents spanning over 3.2k pages and four entity classes. The charity dataset has over 2.7k documents on 61k pages, and is composed of annual financial reports by various charity organizations in England and Wales. This dataset has eight entity types. Kleister is fit for downstream entity-level tasks, such as key information extraction.

CORD (Park et al., 2019), or Consolidated Receipt Dataset, is a fine-grained invoice dataset, with individual word position annotations and two levels of entity classes, one for each individual segment (subtotal prica, discount, service charge) and for groups of segments (all the previously mentioned classes belong to the subtotal superclass, for instance). As such, CORD is a fairly large dataset fit for various NLP and VDU tasks, and is one of the most commonly used benchmarks in recent literature.

RVL-CDIP (Harley et al., 2015) is a large-scale dataset composed of 400k noisy pictures of scanned documents pertaining to 16 different classes, such as memo, letter and scientific publication. These images have low quality and resolution, and do not have annotations apart from the document class. It is a very challenging dataset, commonly used for evaluation in the document classification task.

FUNSD (Guillaume Jaume, 2019) is a subset of RVL-CDIP that was annotated using a mix of automatic and manual annotation procedures. It features a set of four generic classes: header, question, answer and other. Having clear key-value annotations, this dataset is very commonly used for entity linking evaluation. It is also a very challenging dataset, as it contains only 199 instances in total, for training and testing.

3.1.1 Document Synthesizing

Surprisingly, none of the models studied perform any downstream task in any official document dataset, not even MIDV. However, there is research in document synthesizing.

Donut (Kim et al., 2022) authors propose SynthDoG, which samples document backgrounds from ImageNet (Deng et al., 2009) and text from Wikipedia articles. The document itself is assembled from a heuristic rule-based generator for the layout, which treats the segments of text as squares and uses such perspective to assemble the layout before adding in the text.

DocSynth (Biswas et al., 2021) is a GAN for automatic document creation that uses the adversarial strategy of training to learn how to mimic a document dataset. The GAN is trained on PubLayNet (Zhong et al., 2019b) and, although this dataset is already quite large, is proposed as a means of data augmentation that simply creates new data that is convincingly real.

Yang et al (Yang et al., 2017) also present a method for creating synthetic documents. The proposed engine has two different methods for generation. The first involves data scraped from the web arranged into a randomized layout on a LaTeX source file. The second is a more simple approach, where a few hundreds of real, manually-annotated documents with complex layouts have their elements randomly replaced with web-scraped data.

Pondenkandath et al (Pondenkandath et al., 2019) propose a GAN-based synthesis for historical documents. Essentially, the network works in the image to image task by taking as input the image of a document and transforming it into a seemingly older, weathered manuscript. The authors utilize two different architectures for this task, cycleGAN (Zhu et al., 2017) and VGG-19 (Simonyan and Zisserman, 2014), the latter of which proved to yield more faithful results.

DocCreator (Journet et al., 2017) is a framework that performs a rather simple, but powerful document synthesis process. It works by pasting text with selected fonts onto pre-defined background images, and applying some data augmentation techniques to create weathering and noise. The generated images have extremely finegrained XML annotations, being suited for almost any downstream VDU task. However, the synthesized documents lack variety, as the backgrounds and texts are preset. It also functions as a document data augmentation engine.

Raman, Shah and Veloso (Raman et al., 2021) define a document generator that is generated as a Bayesian Network. This network is built as a structured set of rules that defines what a document is. The authors treat a visual document as a set of primitive elements, such as paragraphs, images, titles and tables, logically grouped together according to a set of rules of structure and layout. The bayesian network stochastically generates the composing elements and their layout, and an image manipulation library is used to decode the end result as a real picture of a document. The authors also perform a layout analysis task on the synthetic data and prove it is just as good as real data in terms of training quality.

BID authors have also released their implementation of the synthesizer used for the creation of the dataset. The process implemented follows a simple routine of inpainting sensitive data, randomly generating new fake data, and pasting it back into the image. The synthesizer does not require the annotations to include the field class apart from the information type (date, name, city/state), but also does not include it in the end annotation.

3.1.2 Key Challenges

Most challenged in VDU for official documents lie in the sensitive nature of this data. From the two synthetic datasets of official documents available in the literature, one is not fit for downstream tasks, and the other does not seem to have been used for evaluation in recent state of the art comparisons. Also, MIDV-2020 contains few high quality instances: apart from the video frames, there are only about three thousand instances of documents. As such, a few challenges identified are:

A large scale dataset for entity-based downstream tasks. Since all the models in the state of the art today are based on machine learning techniques, they require a lot of data to reach high levels of precision. It is vital that a large dataset is available, particularly for brazilian documents, which currently do not integrate any dataset with entity-level class annotations.

A synthesis engine for official documents. Proposed BID dataset features an open source synthesizer, which is focused on the three kinds of documents presented in the dataset. Currently, since it does not work with entity classes, it cannot be used to generate a dataset to be used in entity-level downstream tasks.

An analysis of the impact of synthetic data. Although a few authors have made a quantitative analysis for document layout recognition, training the models on real versus synthetic data, there are no experiments on mixed training procedures, with both real and synthetic. There is also no experimentation on entity-based downstream tasks, neither on official document datasets.

3.2 INFORMATION EXTRACTION METHODS

Earlier automatic information retrieval relied on instance-specific knowledge of the document layout and regular expression matching. One of the examples is Intellix (Schuster et al., 2013), which defines a list of possible layouts for each known document class and matches the input against each one of them. The inherent problem with this approach is that portability becomes a problem, as the model has to be tweaked for every new layout instance.

An approach that proved to be more robust is presented by Lample et al (Lample et al., 2016), which proposes that a document can be viewed as a sequence of text, and as such the information retrieval task can be mapped into the NER problem, which is a task in the domain of NLP. A recurrent technique for such problems is the use of RNN. In (Lample et al., 2016), the authors employ a simple RNN as a textual feature extractor, and a LSTM together with a CRF for classification, beating most of the state of the art. However, such approach fails to consider bi-dimensional layout information, and as such has its own limitations.

(Yang et al., 2017) uses a fully convolutional neural network for semantic segmentation in the task of text recognition. In this context, the output of the CNN is binarized into a pixel-wise mask that indicates the presence of text at each location (segmentation task). The pixels with recognized text then receive the value of the text embedding in the corresponding position, and remain at zero where there is no text. The proposed network is an encoder-decoder fully convolutional architecture that also reconstructs the original document during training, with an additional decoder.

Another use of CNNs in document understanding is in (Harley et al., 2015), the same paper that proposes the RVL-CDIP dataset. Their approach consists of cropping five different sections out of the document, one consisting of the entire document, and four overlapping regions from the top, bottom and sides. These regions are resized, processed by a CNN, compressed by a PCA and concatenated into a single feature vector. These features are then classified by another trained fully-connected network.

We highlight these three sources of information - layout, text and visual - as the main features used in most state of the art models today. Also notorious is the predominance of RNN-based architectures, particularly Transformers (Vaswani et al., 2017).

One of the many examples is the LayoutLM (Xu et al., 2019, 2020; Huang et al., 2022) family of Transformer-based architectures, which ties together these three ideas in the form of a multi-modal feature that encodes the text, the position and the visual embeddings as the input for the Transformer encoder. The models are also trained with a pre-training strategy that seeks to make the model learn a stronger correlation between each modal feature. The pre-training goals include word-patch matching (corresponding words to sections of the document image) and predicting real values to masked parts of the document.

The LayoutLM paper also describes a new technique for generating visual embeddings from the image. While most methods utilize a generic CNN, most frequently a ResNet (He et al., 2015) model, the authors sliced the image into several patches and performed a linear projection onto them, effectively encoding a new visual embedding. This process borrows from a similar technique used in the Vision Transformer (Dosovitskiy et al., 2020), which treats the image as a sequence of square patches, most commonly used for image classification tasks.

In GraphDoc (Zhang et al., 2022), a gate fusion layer is proposed, where multi-modal feature fusion is done explicitly via residual connections between layers. The model also presents a graph attention mechanism, which computes a hidden representation for each semantic entity in the image (entity to node and document to graph being the correspondence) based on its neighbors. The paper also utilizes a pre-training technique, where random sentences are masked and the model must predict the corresponding text embeddings based on the known text and visual embeddings.

For end-to-end VDU, there is also research in OCR-free solutions. Apart from the synthesizing engine, Donut (Kim et al., 2022) authors also propose a new VDU model. This model employs the Swin Transformer (Liu et al., 2021) for visual feature extraction, the result of which is passed onto a BART (Lewis et al., 2020) textual decoder, which is also Transformer-based. Finally, as a pre-training approach, the model must generate the tokens present in the input image in reading order.

EATEN (Guo et al., 2019)'s homonym model, Entity-aware Attention Text Extraction Network, is proposed to extract Entities of Interest using a single, simple attention-based network. The work is an entity-level text recognition model, and uses a CNN-based backbone for image feature extraction, and an encoder-decoder attention network into which the CNN feature maps are passed. Interestingly, the network has no sense of text recognition, and instead only learns EoIs from the visual features alone. It features a LSTM model into which the decoder's embeddings are passed for classification.

EPHOIE (Wang et al., 2021) authors also propose VIES, Visual Information Extraction System, composed of detection, recognition and information extraction modules. The detection module takes the region of interest as input, pools it, performs 2D convolution and pools it again, and finally passes it through a linear projection layer at the end. The recognition module takes the box as input, and runs it through an attention-based encoder-decoder architecture for feature

extraction. Finally, the information extraction branch uses a multi-headed attention network for sequence labeling, taking the output of the other two modules as input.

Three other models are considered for our experiments. These are PICK (Yu et al., 2020), StrucTexT (Li et al., 2021b) and DocFormer (Appalaraju et al., 2021).

PICK is an encoder-decoder network that features a third module for graph convolution. The graph is built between entities, the connections being weighted according to the entities' relation between one another. The idea is to leverage the spatial relationship between entities and learn from it. The encoder is built as a combination of a CNN and a Transformer, and the decoder is composed of a BiLSTM and a CRF for classification.

StrucTexT is a Transformer that relies on cross-modal feature fusion between a WordPiece tokenization of the text, a ResNet-based visual embedding, and other features to encode relative and absolute positioning of the entity and text. Its state of the art performance is achieved through a few special pre-training procedures. These are self-supervised: Masked Visual Language Modeling, Segment Length Prediction and Paired Box Detection. MVLM asks the model to reconstruct the original textual embedding, where part of it has been masked by a MASK token and another part was randomized. SLP asks the model to estimate the length of a sequence, given its visual embeddings. PBD requires the model to estimate the angle (discretized into 12 buckets) between two entities given the subtraction of their visual embeddings.

DocFormer is an end-to-end trainable Transformer model that proposes a novel self-attention mechanism, tying together visual and textual embeddings. The embeddings are extracted via a ResNet50 model and the WordPiece tokenizer into a pre-trained LayoutLM model, respectively. The attention mechanism is modelled as an expanded version of the vanilla self-attention in the Transformer, with the (element-wise) addition of three other attention results. These are the query ID, key ID and feature modal attentions. Equation 3.1 features the entire self-attention mechanism for visual features.

$$a_{ij}^{v} = (x_i^v W_v^Q)(x_i^v W_v^K)^T + (x_i^v W_v^Q a_{ij}) + (x_j^v W_v^K a_{ij}) + (V_s W_s^Q)(V_s W_s^K) \tag{3.1}$$

In equation 3.1, the first term represents vanilla self-attention, the second term represents Query ID attention, the third represents Key ID attention and the last term is calculated based on the visual embeddings of the corresponding layer. The equation is analogous for textual features. The attention weights are shared across both features in the same layer, which helps the model to learn feature correlation.

DocFormer is also pre-trained with multi-modal tasks, MVLM is performed here using the fused features from the encoder, as well as other two tasks: Learn To Reconstruct which asks the network to reconstruct the original image from its embeddings, and Text Describes Image, which tries to predict whether the textual and visual information belong to the same document.

3.2.1 Pre-training Frameworks

Following the line of research based on conditioning the model into learning specific correlations explicitly through some pre-training tasks, some pre-training frameworks were developed for generic document learning.

SelfDoc (Li et al., 2021a) is a task-agnostic and coarse-grained framework that presents a novel self-attention mechanism for feature fusion. It is able to work with unlabeled data, as it uses a document object detector (a Faster R-CNN (Ren et al., 2015) in the original implementation) and a generic OCR engine (Tesseract (Smith, 2007), in the original implementation) for parsing.

It uses the last layer of the CNN as a visual embedding and Sentence-BERT for textual embeddings (Reimers and Gurevych, 2019).

Its self-attention mechanism is split into two blocks: a single modality encoder, which is akin to the vanilla Transformer self-attention, and a cross-modal encoder, which uses two novel attention functions. The cross-modal attention is modeled by exchanging the embedding inputs from each modality. The first function swaps the keys between visual and textual attention flows, modeling the agreement between the modalities, and the second swaps the queries, which models new cross-modal relationships. At the time of its release, it beat the state of the art in entity recognition on FUNSD with 83.36% F1-score.

VLCDoC (Bakkali et al., 2022) proposes a Vision-Language Contrastive Pre-Training Model for the document classification task. It utilizes a pre-trained visual transformer (Dosovitskiy et al., 2020) architecture as a visual feature extractor, and a standard OCR mechanism into a pre-trained BERT (Devlin et al., 2019) model for textual features. Its main proposal is the self-attention mechanism and architecture, the latter of which can be thought of as two separate Transformer flows that interact with one another through the cross-modal attention module.

The framework proposes two Cross-Modal Alignment attention modules, IntraMCA and InterMCA. InterMCA enhances cross-modal features by swapping the queries between language and vision attention flows and computing vanilla self-attention, and fusing the mixed result via an additive operation with the pre-attention visual embedding. IntraMCA performs element-wise product between the features and the attention flow, and fuses the two via a linear additive layer at the end. Since the two modal features each have their own Transformer module, and as such, independent weights, only one of them are used for classification at a time.

Using only visuals for evaluation, the model beat the state of the art models that only rely on two of the three modalities: vision, language and layout, in the classification task on RVL-CDIP, at 92.64% top-1 accuracy. A cross-dataset experiment is made, with pre-training on one dataset and fine-tuning on another, also beating the state of the art.

XDoc (Chen et al., 2022) is optimized into being a light-weight model, trained with only textual embeddings. The paper uses a vanilla Transformer as architecture, and focuses on performing the masked language modeling pre-training task and extracting three types of textual features from different documents. These are the plain text, acquired through adding the WordPiece tokenization to the 1D positional embedding, and the document and web text embeddings, acquired through adding the plain text embedding with the result of the adaptive layers for each case.

The adaptive layer is a simple Linear-ReLU-Linear pipeline, and there is one for each of document and web text types. the document adaptive layer takes as input the 2D embedding for the text, encoding the corners plus the width and height of the bounding box, while the web layer takes the XPath embedding as input, encoding information about the HTML tags of the text. In the document text tests, XDoc reaches state of the art performance on entity recognition on FUNSD, with an F1-score of 89.4%.

MGDoc (Wang et al., 2022) aims to leverage multi-granular information by modeling features at different levels: the entire page, its entitites, and its words. For pre-training, the tasks used are the masked text and masked vision modeling, where the model is asked to predict the original value of the masked features on each modality. The multi-modal encoding works by extracting the features at the three levels of granularity: a ResNet run on the Regions of Interest is used for visual features, Sentence-BERT for textual, and the bounding boxes for the layout.

The attention mechanism is modelled into a multi-granular setting, aiming to learn interaction between granularity levels intead of fusing features. To the vanilla self-attention equation are added the hierarchical bias and relative bias. The hierarchical bias encodes the

inside or outside layout relationship between entities, encoding whether the words belong to a region or a region contains words. The relative bias encodes the relative distance between bounding boxes, at both word-level and entity-level. There is also a second attention mechanism for feature fusion, following the practice in VLCDoC's InterMCA swapping the keys between modalities. This is done from text to visual and vice versa. Some special loss functions are also designed for each module.

MGDoc also manages to reach 89.4% F1-Score on FUNSD, and 97.1% on CORD, beating the state of the art on both datasets. Accuracy on document classification on the RVL-CDIP dataset is not as high, losing to most of the models compared, including SelfDoc, at 93.64%.

3.2.2 Key Challenges

The main challenges in the field, besides the data availability, are listed below. These are observed from the state of the art review presented, and based on comparisons between the study, the state of the art in other fields and the current direction of research.

Official document-focused models. Most of the datasets used for evaluation feature a variety of scanned documents, which feature many different layouts, but are inherently different from other structured documents such as ID cards. Purely layout-based classifiers cannot generalize their knowledge to new layouts, and machine learning solutions are developed with other types of document in mind.

Cross-dataset models. Although some experiments have been made in cross-dataset scenarios, these seem to be limited to document classification tasks. It is hard to map entity-based recognition between datasets as their class sets may differ greatly. Furthermore, cross-dataset protocols often involve a fine-tuning phase on the destiny dataset, instead of just testing the model that was trained on another dataset.

Zero-shot classification. Zero-shot learning as a concept has appeared in classification tasks in the fields of image (Yi et al., 2022) and text (Gera et al., 2022) recognition, but is a challenge that has not yet been touched in VDU. It is of particular relevance for official documents, where fields are usually accompanied by a header, and the fields of interest may vary from country to country, and even from document to document under the same nationality.

Performance in small datasets. While state of the art performance goes well above the mark of 90% accuracy/F1-Score in large datasets such as RVL-CDIP, smaller datasets such as FUNSD do not reach that mark. Most models fail to generalize the knowledge to the point of reaching such a high standard on small datasets.

3.3 CONCLUSION

The sensitivity of the data in official documents such as ID cards seems to have, so far, impeded the development of official document understanding in the current state of the art. Although there are a few datasets available, there is no research focused on extracting information from this kind of document specifically. Moreover, the impact of synthetic data in the training has not yet been properly analyzed in VDU as a field. Some data augmentation techniques may prove to be better than others in this context, but currently a significant quantitative analysis is missing.

This work will be focused on developing a robust document synthesizer engine, as well as analysing the impact of adding the synthetic data to the training. A novel synthetic document dataset will be proposed, and the three chosen state of the art models will be evaluated in both the synthetic and real data. These models will also be trained with a mix of real and synthetic data at

varying ratios in order to analyze the impact of synthetic data in training. Finally, a comparative analysis in famous VDU benchmarks will be made between the three, with a focus on standard training procedures to verify the impact of the pre-training tasks done by their authors.

4 PROPOSAL

In this chapter, we present the structure of the dataset synthesizer, as well as describing the dataset generated from real document images provided by Unico, our sponsoring company. We also describe the models used for evaluation on the new dataset, and discuss the approaches used.

4.1 THE SYNTHESIZER

Our approach is divided into two independent processes, anonymization and synthesizing, where we erase the sensitive data in the document and replace it with new, synthetic data, respectively. The following two sections are dedicated to explaining each process.

4.1.1 Anonymization

The input to the anonymization module consists of the RGB image representing the document, its orientation in the image (in 0, 90, 180 or 270 degrees, counted in clockwise rotation), and a set of semantic entities to be erased. These entities represent sensitive information in the document: names, dates, document codes, etc., as well as containing one entity for the document itself. Essentially, every entity that is not standardized within the document (meaning headers and the like) must be annotated for anonymization. The entities are represented by the bounding box surrounding the text, the field label, and the text transcripts, if any.

The field labels must be contained in a set of known entity types, such as name, date of birth and expedition, parents' names, etc. These entities vary according to the type of document. In our case, the documents used are the front and back of the national identity card implanted in 2019. The fields for each are described in section 4.2.

Currently, there are several OCR systems available, so why shouldn't we skip the annotation process by using an off-the-shelf OCR for automatic text extraction? The reason is that these systems are specialized in different kinds of document, and in our experiments they have performed very poorly. We utilized Tesseract (Smith, 2007)[1] and EasyOCR[2] for evaluation on a few real instances gathered by the author himself[3]. Another reason for this is that these systems, even if robust, are not perfect, especially given that the documents in our datasets are often small, rotated, bent or out of focus, making text recognition especially challenging. As such, using OCR could imply a need for manual revisions of the output, particularly of the bounding boxes, as these are the information used for the anonymization process.

Anonymizing starts with rotating the document according to the annotated orientation. Then, the image is resized into a canonical width, keeping the aspect ratio, and the bounding boxes are also transformed accordingly. The default width is 1920. Then, we perform a warping and cropping transformation in the document. The warped and cropped image is then masked in a few special regions: the annotated face, signature and fingerprint. The masked image is then inpainted to erase the sensitive information. Figure 4.1 presents the image as it comes as the input (top left), after 90-degrees rotation (top right), after warping (bottom left) and inpainted (bottom right).

[1] Publicly available at `https://github.com/tesseract-ocr/tesseract`.

[2] Publicly available at `https://github.com/JaidedAI/EasyOCR`.

[3] We encountered the same issue when trying to detect the text orientation, which justifies the need for this annotation as well.

Figure 4.1: Example of an image as it passes through the anonymization pipeline.

The anonymization pipeline may run on its own, or it may serve as well as a pre-processing stage for the synthesizer. If it does run alone, the annotation is anonymized and the image is rewarped back to the original stage. Transcript anonymization works by replacing any letter with the letter "a", and any number with the number "0". Graphic signs such as hiphens and slashes are kept as-is.

The warping is done in the following way. Given the image, its size, and the document contained within the image with its bounding box, the warped image's new height is defined as the maximum euclidean distance between the top-left and bottom-left, and the top-right and bottom-right coordinates in the y-axis, and the width is defined accordingly for the top-left and top-right coordinates, and the bottom-left and bottom-right ones. This is done because the document may be twisted or inclined in the image, meaning the left height may differ from the right height, and the same for the top and bottom width.

Then, the document within the image is linearly projected into a rectangle of $(H^n x W^n)$ dimensions, where H^n is the new height and W^n is the new width. This projection is done via a transformation matrix calculated from the original document bounding box within the image and the new bounding box calculated from its original heights and widths. The result is a rectified image containing only the document, cropped out of the original image and projected. The transformation matrix is used to warp the coordinates of every entity in the original image to the same plane as the warped image.

Masking works by replacing every pixel contained within the annotated polygon for the entity with the white value. This is done for the face, signature and fingerprint in order to keep some image cleanliness, and to ensure that the GAN does not attempt to recreate these kinds of detail.

Finally, our inpainting method uses a specially trained GAN to learn the background of the documents. To this end, we use a slightly modified GLCIC (Iizuka et al., 2017) model with a new training technique. The experiments are detailed in section 4.3.

4.1.2 Synthesizing

The synthesizer module takes as input the warped document image and the entity labels. The labels must include, at least, the field type and the warped bounding box. The field names must be part of the set of known entities. If there are unknown field types, the process skips the entity. The synthesizing process is executed for every entity with sensitive information stored.

It works as follows. First, a "person" object is generated. This object will contain all the information required for any document instance, i.e. transcripts for all of the known field types. Then, for every entity, the polygonal bounding box is reduced into a rectangle by taking the minimum and maximum between the points in the x and y coordinates as the new bounding box. The new text is extracted from the person object based on the entity type, and the height of the image is used to estimate the height of the new text in the image according to a set of known ratios for all entity types.

The new text is then added into a temporary white text on black background mask, in order to estimate a rectangular bounding box for the new entity, and so such mask is created separately for every entity and then discarded. Then, it is also pasted in a black text on white background mask, which will be used for pasting the whole text into the image itself.

After this process, the black text mask is pasted into the original image in the following way. For every pixel in the mask, if the R, G and B values are all lower than a certain parameter (set by default to 150), the pixel is replaced with the black pixel in the mask. The intuition is that the painted text in the mask is not completely black, but corresponds to some shades of grey. This process of checking the "clarity" of each pixel prevents that pixels close to, but different to

255 in value stay in the synthesized image, which would break the text against the background, making it look fake.

At the end of this process, the image and annotations are re-warped using the inverse of the transformation matrix applied in the anonymization process. This produces the final result of the pipeline, which is a document image with the sensitive text replaced by synthetic data. The rewarped and synthesize image and annotations are then saved to disk.

This process is executed repeteadly for every real instance provided, creating a new synthetic instance for every iteration. By default, the argument of number of iterations is five, the value at which we found the performance of most models to reach a plateau.

By default, we use the Arial MT Bold truetype font. We have not found official sources confirming that this is, indeed, the real font used in the real documents, but it has produced results that are indistinguishable from the font used in the real instances gathered by the author. The font height is dynamically set according to the entity type, and the ratios between the text height to the total document height we found are: $\frac{16}{425}$ for the card holder name, $\frac{13}{425}$ for the holder's parents' names and the general registry number, and $\frac{12}{425}$ for all other fields.

Also, the mask used for pasting the text and then merging it with the real image is originally thrice the size. This is due to limitations in the library used for text to image manipulation. In particular, there is no anti-aliasing function available, so our workaround was to paste the text into a mask three times larger and then resizing it down to the correct size using an anti-aliasing method, which is set by default to the Lanczos resampling in the library.

Furthermore, we would like to take a look at the text generating with more detail. As stated, a "person" object is created at the time of synthesizing, and the text is generated all at once upon class initialization. This allows us to set the order for generation, which also allows us to create conditional generation based on already defined fields.

For example, the card holder's name may contain up to two surnames taken from the parents' names, the expedition date will always be after the birth date, and the voter registration card will only be generated for individuals of legal age for voting (16, in Brazil). If the correct conditions are not met, the generated text is set to the anonymous string "*****". In section 4.2, all the rules for generation, as well as the fields considered, are described in detail.

Part of the code used for text synthesizing is borrowed from the original BID authors, and is publicly available on github[4]. We also use the name, surname and city/state dictionaries provided, with a few modifications, also described in section 4.2.

4.2 DATASET DESCRIPTION

The proposed dataset is crafted from 123 real instances for the front of the document and 141 from the back. Out of these, 4 front images and 9 back images were removed due to excessive bending of the document, which caused the linear projection to not result in a rectified image. Therefore, since we use a default of 5 new instances from every real document, we generate 595 front images from 119 real documents and 660 back images from 132 back images, for a total of 1255 synthetic instances. All of the real images were annotated and the anonymized versions were provided by Unico, and none of the instances gathered by the author make up the definitive version of the dataset.

We define 25 field types contained in entities across all instances. These do not include the fields that are anonymized but not synthesized, such as the signature, fingerprint and face. There are 10 entity types in the front image, and 15 in the back. Table 4.1 presents all the entity types in the front images, while table 4.2 presents the entity types in the back images, as annotated

[4]BID generator is available at `https://github.com/AlyssonDSS/GeradorBaseSintetica`.

Front		
Tag	Meaning	Count
nome	Identity holder's name	595
filiacao1	First parent name	590
filiacao2	Second parent name	580
datanasc	Date of birth	595
naturalidade	Place of birth	595
orgaoexp	Issuing authority	495
serial	5-digit serial code	375
codsec	4-digit hex code	380
rh	Blood type and Rhesus factor	90
obs	Observation	100

Table 4.1: All field types present in the front section of the dataset.

Back		
Tag	Meaning	Count
rg	General registry number	660
dataexp	Date of expedition	660
regcivil	Civil registry of issue	900
cpf	Physical person registry number	655
dni	National identification document number	40
te	Voter's registry number	330
ctps	Worker's record and social security number	355
serie	Serial general registry number	260
uf	State of expedition	260
pis	Social integration program number	310
profissional	ID number for some professions	55
cnh	Driver's license number	210
cns	National health card registry	200
militar	Military identification number	175

Table 4.2: All field types present in the back section of the dataset.

in the label files, their meaning and the number of times they appear in the entire dataset. For the front images, every entity may appear at most once, while the civil registry may appear twice in the back images, since it is often composed of two lines instead of one like the other fields.

We also define a fifteenth entity deemed "other" in the back documents for spurious information that may be annotated but does not belong to any of the listed classes. This class is not present in our dataset.

Figure 4.2 presents some of the easiest instances for front and back in the dataset. Figure 4.3 presents some more challenging instances, again for front and back.

We now go into detail on the syntax and generation of each field. As described in section 4.1.2, the text generation is a sequential process with dependencies. There is no correlation between the front and back images, i.e., there is no such information for two separate images representing the two faces of the same document. However, for every document we generate all of the fields, regardless of whether it may be used or not.

Figure 4.2: Easy front and back documents.

Figure 4.3: Some more challenging front and back documents.

Upon initialization, the person object is represented by an empty dictionary. Then, the dictionary is filled in the following order. First, the parents' names. Then, the card holder name and the cpf and rg fields. Then we generate cnh, pis, dni, naturalidade, uf and regcivil. Then, the date of birth and date of expedition. Then, orgaoexp, obs, cns and rh. Then we generate te, militar, profissional, ctps and serie.

The dependencies are as follows. The card holder name depends on the parents' names. Date of expedition depends on date of birth (and vice-versa: in the implementation, if one is defined, then the second must go before or after it). The voter's registry depends on the date of birth. The ctps field depends on the date of birth and of cpf. The serie field depends on ctps and cpf. The fields profissional, militar and cnh all depend on the date of birth.

The name dependencies exist to ensure the holder's surnames are based on the parents' names. For the fields te, profissional, militar, cnh and ctps, the date of birth is a dependency because a person too young is not legally allowed to vote, work or drive, and as such the document must not contain these fields. For militar, cnh and ctps, the restriction defined in our code is the age of 18, while for te and profissional it is the age of 16.

For the name generation, we use a weighted name dictionary, that contains male, female and undifferentiated names alike. There is a grand total of 29511 names, where the vast majority (29537, precisely) have a weight of one. The remaining, more popular names (exactly 154), have a higher weight in order to reflect their higher frequency in the population. Our base dictionary is the one provided in the original BID synthesizer, and we add 38 new names to it.

Out of the 154 weighted names, the 10 most frequent are assigned a weight of 4, the following 11-50 names have a weight of 3, and the rest are assigned a weight of 2. We leave the surname dictionary unchanged, and it is the same from the original BID generator. The parents' names are generated as two or three random names from the dictionary (the probability is one half for each) plus one random surname. The holder name is generated as two or three random names (at a probability of 2/3 to be two names) plus one surname.

If the holder name is set at three names, one surname between the two parents' surnames is chosen randomly (at a probability of one half for each). If the holder name is set at two names, there is a probability of 1/4 that the holder will gain one random surname between the two parents' names, and a 3/4 probability that the holder will gain both surnames.

Both cpf and rg fields are generated with random numbers: cpf takes 9 random digits and adds 2 verification digits, while rg takes 9 random digits and 2 verification digits. The syntax for each field is "000.000.000-00" and "00.000.000-00", respectively. The cnh field is identical to rg. The cns, militar, te and pis fields are all based on random digits as well. For cns, the text generated consists of a string of 15 random digits, while for militar it is 7 digits. For te, it is a string of one zero followed by 12 random digits. The pis field is composed of a string 11 random digits in the format "0000.00000.0-0".

The rh field is a combination of the blood type (A, B, AB, O) and the Rhesus factor (+ or -). The date syntax is "dd/mm/yyyy". All age-conditioned fields are set to "*****" in case the condition is not met, and dni is always set to the anonymous string.

The fields for ctps and serie are both derived from the cpf field. For ctps, the generated text consists of the concatenation of the first 8 digits in cpf, without formatting. For serie, the last four digits of cpf are concatenated, again without formatting.

The uf, orgaoexp, obs and naturalidade fields are all based on dictionaries. The uf field is chosen randomly from all 27 federative units in Brazil, and corresponds to their 2-letter code. For naturalidade, the syntax is "city-state", where state is a 2-letter code for a federative unit accordingly, and the cities are chosen uniformly from a dictionary (we also use the cities dictionary provided in the original BID generator repository). The orgaoexp field corresponds to

the acronym for the responsible legal units with authority of identity card issuing. There are ten of them in our implementation. The obs field may be any random letter from the latin alphabet followed by a semicolon, or the text "EXERCE ATIVIDADE REMUNERADA;".

The profissional field has the following syntax: "<ORG>/FU 00.000.000", where <ORG> is one of "CREA" or "CONFEA", which are national councils responsible for labor surveillance, FU corresponds to the 2-letter code for some brazilian state, and the first digit is always set to zero. Both codsec and serial are implemented by random codes: codsec is a string of 5 random digits with syntax "0000-0", and serial is a 4-digit random hex code.

Finally, regcivil is the most complex field, as not only is it usually comprised of two lines, but its syntax also varies wildly in real documents. To represent the complexity found in the wild, we define three different templates. The first template is of syntax "CERT.NAS=000 LV=000 FL=000" for the first line and "CART. <ZONA>-CITY/FU". In the first line, the numbers are randomly and uniformly sampled between 1 and 500. In the second line, <ZONA> may be any between "SEDE" and "Xa ZONA" with X varying between 1 and 5, and CITY/FU corresponds to a random city/state entry in the dictionary. All of the sampled numbers are padded with zeros to the right if required to fit the template.

The second template has syntax "CERT. NASCIMENTO CARTÓRIO: <ZONA> TERMO:0000000 FOLHA:0000" for the first line and "LIVRO: 00000 CITY-FU" for the second. In the first line, the number accompanying TERMO is sampled between 1 and 1000000, and the number for FOLHA is sampled between 1 and 8000. In the second line, the number for LIVRO is sampled between 1 and 10000, and CITY-FU is another entry in the cities dictionary.

The third and final template has syntax "CITY-FU CN:LV.a0000/FLS.0000/N.0000" for the first line and "MATRÍCULA: 000000 00 00 0000 0 00000 000 0000000 00" for the second. The numbers sampled for the first line are, in order of appearance: 1 and 1000, 1 and 1000 and 1 and 195000. The third number may vary in length, but always has a minimum of 4 digits with zeroes to the left if smaller than 1000. Every digit in the second line is randomly chosen from 0 to 9.

We also employ a probabilistic generation for te, ctps, cns, profissional and militar fields. The intuition is that not every adult possesses these credentials and, if they do, they may have been acquired posterior to the issuing of the document. As such, if the generated age obeys the criteria, we also check a probability for non generation of the fields. For militar, this probability is 5 out of 6. For profissional and cns, it is 2 out of 3. For ctps, it is 1 out of 4 and for te it is 1 out of 2.

4.3 GAN INPAINTING

Previous research in document synthesizing (Álysson Soares et al., 2020) used a simple filter for pixel value replacement. Given an entity and its bounding box, for every pixel contained within the bounding box, the dominant and average colors are calculated, and the pixel value is replaced by the higher of the two. The dominant color is calculated using the median cut algorithm (Heckbert, 1998), while the average color is defined as the arithmetic mean between the value of the pixel itself and the pixel directly above and below.

While this approach does safely dilute the information contained by the region to the point of no recovery, the result looks artificial, and the fine-grained detail contained in the background image is lost. We improve this approach by using a GAN model to learn the the document's background and, using this knowledge, inpaint the text in such a way that captures context from the document itself.

Figure 4.4: Double bounding box example for GAN training

Since the background pattern for the ID cards used is known, it should be possible to train a model to learn and reconstruct it, taking into account the neighboring regions such that there is a smooth transition in images of real documents, which often display visual signs of use. There is significant research in the area of image reconstruction, but the scope in the field is more broad and, again, research for document image reconstruction is very limited, if not non-existent. Our approach is based on an existing image inpainting GAN, which we modify and train using a novel procedure for background reconstruction.

One problem we immediately stumble on is the unavailability of existing raw document images, with no text. Although the background pattern is known, without significant instances of it, it is hard to train a machine learning model to reconstruct the same pattern in images that feature problems such as partial occlusion (a picture taken by a person whose thumb is directly upon the document), non-trivial perspective shifts (i.e. the document is not a rectified rectangle) and documents that have varying colors due to erosion over time.

Our approach to circumvent this problem relies on double bounding boxes annotated in the following manner. For every entity we want to erase from the document, two bounding boxes are annotated. The first one is tightly wrapped around the text, while the second one contains the first one completely and spans part of the background surrounding the text (but does not include any text from other parts of the document). The GAN is asked to reconstruct the entire area on the second bounding box, including the text, but the loss function is only applied to the area of the second box that is not comprised by the first box. The idea is to condition the network into learning based only on the background surrounding the entity, and subsequently erase the text with the same background.

Figure 4.4 illustrates our approach. The green box represents the outer bounding box that covers the background, while the orange box is the inner box fit tightly around the text. The loss is calculated based on the green area that does not contain the orange area. The presented document is an earlier sample and is not contained in our dataset.

The GAN we chose is the GLCIC (Iizuka et al., 2017) model. GLCIC is composed of a CNN-based completion network and a two-stage discriminator for local and global image patches

pooled together by a linear layer. In the existing implementation of GLCIC[5], these components are trained, first separately in a two-stage process, where the completion network is trained with a MSE loss in the first stage, and the discriminator is then trained based on the results from the completion network vs the real images with a BCE loss. The third phase is comprised of the adversarial training.

In particular, we highlight two details about the components. The completion network, which is a CNN, features four dilated convolution layers in the middle, sandwiched by conventional convolution layers. This is done in order to allow for taking a bigger area into account in the model. The discriminator is also based on convolution layers. The global module takes the entire 512x512 image into account, while the local discriminator takes the 256x256 square that may be real or artificially generated. The output of the two is concatenated and passes through a linear layer, into a binary decision between fake or real.

We use the provided implementation of GLCIC, and modify it for training with annotated bounding boxes instead of randomly generated squares. We train it with an early stopping of three epochs on the loss function, every epoch spanning 100k steps as per in the original implementation. We manually annotate 50 synthetic images from the dataset, and split these into 42/4/4 images for training, validation and testing respectively. These images are generated from early instances of our synthesizer, and all of them are of the front of the document.

Furthermore, we modify the base model in two ways. The first of them is a change in the discriminator. As stated, it is composed of a global and a local module, which take as input the image itself and a fixed-size patch, convolved consecutively in six and five layers respectively. In particular, the GAN is trained by reconstructing 512x512 patches chosen randomly within the image. Since the annotated bounding boxes are of varying size, the CNN-based local component cannot be used as-is anymore, and we opted for discarding it.

The second modification is the replacement of the loss function. We develop a new loss function from the Structural Similarity measure (SSIM), calculated as in equation 4.1, where x is the GAN output, with all of its bounding boxes reconstructed, and p is the original image. SSIM is a better similarity measure for our case than MSE because the reconstruction goal is to understand and replicate a texture, and as such basing the learning on a structural measure makes more sense than using the pure pixel value as in the MSE loss. The values of the inner bounding box are set to zero in order to not interfere with the loss function at all.

$$Loss_{ssim} = 1 - SSIM(p, x) \tag{4.1}$$

In our experiments, we've trained the network separately with both losses, and compared the results qualitatively. In particular, we notice that the result for the MSE loss presents some artifacts in the inner bounding box that are not present for the SSIM loss. We also compare the GAN results with the previous inpainting method, and notice that the GAN method manages to capture some fine-grained detail from the background, allowing for a smoother transition between the region itself and the background. Although not all details from the background pattern are learned, the model managed to create a fairly convincing and clean inpainting capturing the context from the neighboring regions.

Our results are illustrated in Figure 4.5. The first image is the original (synthetic) document, anonymized with our chosen approach. The second image represents the bounding box of inpainting. The third image is the result of anonymization with the previous method. The fourth image is the GAN inpainting with the MSE loss, and the last image is our final approach, the GAN trained with a SSIM loss.

[5]Available at https://github.com/otenim/GLCIC-PyTorch

Figure 4.5: Inpainting results.

Figure 4.6: Some anonymized documents.

Finally, we have also considered a few texture-specific measures, namely STSIM-1 and STSIM-2 (Zujovic et al., 2013), for the loss function modeling. However, these measures have displayed a big calculation overhead that has made training unfeasible. We showcase some results of our inpainting in Figure 4.6.

4.4 MODELS FOR EVALUATION

The dataset is used for training and testing of three document vision models. These are PICK (Yu et al., 2020), StrucTexT (Li et al., 2021b) and DocFormer (Appalaraju et al., 2021). The reason behind the choices is the availability of code. Both PICK and StrucTexT have official open source repositories[6], PICK making the training and testing routines directly available. The DocFormer implementation is also available, although it is an unofficial repository with authors unaffiliated with the writers of the paper[7].

For StrucTexT and DocFormer, the training routines were not available at the time of writing, and as such have been developed by us in accordance to the fine-tuning stage of the entity labeling task. Both of these models, which are based on multi-modal Transformers, rely on a heavy pre-training stage with task-agnostic training goals (further detailed below) on huge datasets as a means of understanding the structure of documents before specializing it on various downstream tasks. This approach is commonly used in NLP problems, and we interpret it as learning the syntax of the medium first, in the pre-training, and then learning the semantics in the fine-tuning phase.

We skip this stage for a few reasons. First, it is very computationally expensive, to the point of being unfeasible with the given hardware, and the pre-trained weights for both models were not available at the time of writing. Second, the implementation of the pre-training tasks were also not available Finally, as our goal is simply to evaluate the impact of artificial data on training, and not to perform a comparative analysis between the models, we judge that such a procedure is not necessary.

[6]PICK is available at: `https://github.com/wenwenyu/PICK-pytorch`, StrucTexT at `https://github.com/PaddlePaddle/VIMER/tree/main/StrucTexT/v1`.

[7]Available at `https://github.com/shabie/docformer`

4.4.1 PICK

PICK uses an Encoder-Decoder architecture together with a graph module for the task of Key Information Extraction. It is composed of the three modules, where the encoder crafts a hidden representation for text embeddings and image segments, which are combined into a local representation for each entity in the document. This representation is called a "node", and the set of nodes is input into the Graph module, which aims to learn the inter-segment relationships between nodes through a graph convolution operation. Finally, the decoder takes both the graph embeddings and the set of nodes from the encoder, and fuses the information. The decoder treats the information extraction task as a task of sequence tagging. As the other models, it also uses multi-modal text, vision and layout features for document learning.

The encoder module is composed of a Transformer, a CNN, and a fusion operation. The text segments are encoded by the Word2Vec (Mikolov et al., 2013a) embedding and input into a pre-trained Transformer for encoding. The corresponding image segments are encoded by a CNN, the implementation of which corresponds to a pre-trained ResNet50. The two features are fused by a simple element-wise adding operation, the output of which corresponds to the set of nodes, one for each entity in the document.

The nodes output by the encoder layer are then fed into the Graph module. This module aims to capture the relationship between entities (nodes) through the graph modeling. The paper expands upon previous research on graph learning convolutional neural networks. Their method computes a soft adjacency matrix in various steps.

First, the graph learning component takes the visual input of the nodes, and generates the adjacency matrix where each element represents the pairwise relationship weight between two nodes, similar to the attention matrix in Transformers. This component consists of a single layer, with a learnable set of weights. The matrix will be composed of the weighted difference between two nodes, as in equation 4.2. The LeakRelu is used in order to prevent the vanishing gradient issue. The loss function for the graph component is calculated based on this matrix and on the proximity of the nodes in the embedding space.

$$\begin{cases} A_i = softmax(e_i), i = 1, ..., N, j = 1, ..., N \\ e_{ij} = LeakRelu(w_i^T |v_i - v_j|) \end{cases} \tag{4.2}$$

Then, the visual and textual features are fed into a MLP network for feature extraction. The features output from the MLP are convolved, which spreads the information between the nodes and generates a new representation. This is convolved together with the adjacency matrix in the graph convolution component, which aims to capture global visual information and layout nodes. The convolution is done in the node-edge-node triplets (which means the learned embeddings of two nodes together with their relationship score) and aims to generate the relation embedding between the two nodes.

Initially, it is set as the concatenation between the distances in the x and y axes, the aspect ratio of the first node, the ratio between the height of the second and the height of the first node, the ratio between the width of the second node and the first, and the ratio between the text length of the second and the first. This embedding is multiplied by a learnable matrix of dimensions hidden size by six.

The hidden features between two nodes are then extracted by the graph convolution operation, which is defined for the l-th layer in equation 4.3. Here, W represents the learnable weight matrices, b is a bias parameter, α_{ij} is the relation embedding generated from a layer-specific trainable matrix for each layer and the hidden node-to-node representation in the same layer, and

σ is a non-linear activation function. The paper does not disclose the number of layers in the graph convolution operation, but the implementation uses a default of 2.

$$h_{ij}^{l} = \sigma(W_{v_i h}^{l} v_i^{l} + W_{v_j h}^{l} v_j^{l} + \alpha_{ij}^{l} + b^{l}) \tag{4.3}$$

The decoder takes as input the concatenation between the set of nodes output by the encoder and the hidden representation learned by the graph module, and is composed of a union layer, a BiLSTM layer and a CRF layer. The union layer simply concatenates the features and adds a padding value at the end. The BiLSTM layer features a standard model for generating scores that will be classified by the CRF, which generates a family of conditional probabilities via a softmax for each sentence. The idea is that each token is tagged independently, but the entire sentence must have only one label.

Training is made jointly by combining the loss of the graph learning module and the loss of the CRF layer by multiplying the graph loss by a constant λ and summing the two. By default, $\lambda = 0.01$. PICK does not feature any pre-training task, instead it is used for the KIE downstream task directly. It uses a hidden size of 512, 2 BiLSTM layers, 3 layers in the encoder, and a learning dimension of 128 for the graph module. These numbers are gathered from the model configuration in the official implementation, and not reported in the paper.

The optimizer used is the Adam algorithm, with a learning rate of 10^{-4}, with no weight decay. The model is trained for at most 100 epochs with an early stop tolerance of 40. We use these parameters, saved as the standard configuration in the git repository, as our training procedures for the proposed dataset.

4.4.2 StrucTexT

StrucTexT is a Transformer-based architecture that relies on a cross-modal fusion of multi-modal feature embeddings and self-supervised pre-training techniques to reach state of the art performance in entity labeling. The features used include a layout embedding, a textual embedding, and a visual embedding, as well as a few supporting embedding pieces. StrucTexT is designed for various downstream tasks, including token and segment (entity) level labeling. In our experiments, we consider the segment-level model.

The layout embedding is created from the four coordinates in the 2D axes encoding the top left and bottom right corners of the entire entity bounding box, the width and the height. For the token-level embeddings, the bounding box is estimated from the entity bounding box. It is important to note that a linear layer is used to project each embedding modality separately, although this is only implied in the paper.

The textual embedding is crafted from the text sentence as tokenized by the WordPiece (Song et al., 2020) algorithm. The text from the entire document is tokenized and serialized according to the top to bottom, left to right order according to the bounding box coordinates, resulting in a language sequence where each slot in the array corresponds to the text of one entity in the document. The final embedding is created from the element-wise sum of the encoded textual segment with the layout embedding. It is not mentioned in the paper, but the text is encoded with an ERNIE (Zhang et al., 2019) model in the official implementation.

The visual embedding is based on each entity's bounding box, which is encoded via a pre-trained ResNet50 (He et al., 2015) + FPN (Lin et al., 2016) model. The embedding is extracted from the encoding via a ROIAlign (He et al., 2017), and then summed with the corresponding layout embedding. The feature maps for the entire document image are also embedded as the first visual embedding, enabling the interaction with the global information in the document in each segment.

There are also three other embeddings: a segment id, which assigns an unique identifier to each text segment's feature maps, a position embedding that identifies the token's position within its text segment, and a segment embedding, which denotes the modality of each feature. This segment embedding is an ID that is the same for visual and textual embeddings stemming from the same entity. The full feature maps are constructed by concatenating the textual and visual features, and then adding the three other embeddings to the result. If necessary, a shorter input will padded with a special token, while longer inputs will be truncated, to ensure a length of 512 tokens.

These features are then encoded via a Transformer model, fusing them into a multi-modal output that is used in three self-supervised tasks: Masked Visual Language Modeling (MVLM), Segment Length Prediction (SLP) and Paired Box Detection (PBD). There are no changes to the standard Transformer architecture, and the attention mechanism utilized is also standard. As such, we skip to the pre-training task descriptions.

In MVLM, 15% of the textual tokens are randomly selected. Out of these, 80% are replaced with a special mask token, and 10% are replaced with a random token from the corpus. The other 10% are left unchanged. The model must then reconstruct the original sequence based on the textual and visual context in the encoded features. This task is trained using a Cross-Entropy loss, as it may be interpreted as a classification task. This is a variant of the Masked Language Modeling (MLM) introduced by BERT (Devlin et al., 2019), and is originally proposed in LayoutLM (Xu et al., 2019)[8]. The goal for MVLM is to learn the syntactic structure in text, as well as aligning the shared knowledge between visual and textual features.

SLP is a novel task proposed by the authors. From the visual features only, the model is asked to find the length of the corresponding text segment. This is done to learn more fine-grained information in the image-side, and the authors argue that it aids the model into learning more robust cross-modal feature fusions. The loss used is not described, but it may be a CE loss as well for the same reason as MVLM. An important detail is that WordPiece tokenizing may split words into subwords and thus increase the token count for an entity. The authors consider only the first sub-word for these split words, in order to keep a consistent length.

The authors also propose a second novel task. PBD is designed to exploit global layout information, and consists of a classification task for the relationship between two entities. Given two entities and their representation after the Transformer encoding, the model must predict the angle (discretized into 12 buckets of 30 degrees) between the two according to the subtraction of the two features.

Finally, these tasks are used for pre-training on the DocBank (Li et al., 2020) and RVL-CDIP datasets (Harley et al., 2015) for 10 epochs with a batch size of 64. With the pre-trained weights, the original paper describes three ways to utilize the features for three different downstream tasks, and the model is evaluated on these. They are segment and token level labeling and entity linking.

For both labeling tasks, the Cross-granularity Labeling Module replaces the pre-training tasks in the fine-tuning phase. The process is described as such: the tokens with the same segment ID in the language embedding are aggregated by an arithmetic average. For token-level, each token is averaged from the subwords composed by WordPiece, and in segment-level, all subwords belonging to the entity are averaged. The average is then fused with the visual feature through a Hadamard product, and a fully connected layer is used for prediction. Both tasks are classification tasks, and as such both use the Cross-Entropy loss.

[8]The original MVLM in LayoutLM proposes that the image regions also be masked. StrucTexT authors have opted for not following the practice.

In entity linking, the Segment Relationship Extraction Module is devised for classification. Similar to the attention matrix in Transformer layers, this module calculates a square matrix that encodes the probability that a token is linked to another. This is done by multiplying the output features with a matrix of learnable parameters, and using a sigmoid activation equation at the end.

This is an expensive operation, and the authors use the Negative Sampling technique introduced by Word2Vec (Mikolov et al., 2013b). It consists on stochastically ignoring part of the embedding that may not be relevant each iteration, and only updating the weights partially. This introduces instability on training, and as such the Binary Cross-Entropy loss used for this task is combined with a Margin Ranking loss.

Our training is based on the existing implementation of the model, and we perform fine-tuning on entity labeling directly on our dataset with randomly initialized weights. The model is trained with an early stopping tolerance of three epochs, a learning rate of $5 * 10^{-5}$ and the Adamax optimizer. The architecture utilizes the same parameters as that of the base Transformer encoder, with 12 layers and 12 attention heads, and a hidden size of 768.

4.4.3 DocFormer

DocFormer (Appalaraju et al., 2021) is an end-to-end trainable transformer that experiments with a new way of fusing multi-modal features and of modeling self-attention. It also draws information from the three modes: layout, visual and textual, and fuses them explicitly and implicitly using a Transformer architecture. The training also follows the unsupervised pre-training and supervised fine-tuning stages.

The layout features are gathered at both word and segment level. In particular, for every token in the text, the bounding box coordinates are extracted and used for encoding of various features. The width and the x-axis coordinates, and height and the y-axis coordinates, are encoded separately by two linear layers, together with the distance between the centroids and four corner of the given entity and the corresponding coordinates in the next entity in the left-to-right order.

The output for the x and y axis features are summed, together with an absolute 1D positional encoding that contains the index of each entity. This encoding process happens twice, as the layout features for the text and visual embeddings are separate, meaning there is a total of four linear layers for this embedding. This is done primarily because the features may be modality-specific.

The textual features are also acquired from the tokenized sentence (again, using the WordPiece algorithm), with the same padding and truncating techniques to ensure a length of 512. The tokenized sequence is fed through an embedding layer, initialized with pre-trained LayoutLM weights.

Visual features are obtained using a ResNet50, from which the convolution map at layer 4 is extracted. This map is then convoluted with a $1x1$ kernel, flattened and passed through a linear transformation layer. The result is a $(768x512)$ feature embedding, which is the same dimensionality and the textual embedding, which also has a hidden size of 768. There is also a global representation for the entire document, as in StrucTexT.

The Transformer architecture used as backbone for the model is different for DocFormer. The model presents two separate flows for the Transformer, one for each modality between visual and textual, with separate weights in the matrizes of each flow. The authors also experiment with novel methods for feature fusion that utilize an altered version of the self-attention mechanism. The standard Transformer's attention distribution is based on the output of the previous layer and three learnable matrices for key, query and value. DocFormer expands the equation by adding in three other results: the query and key ID relative attention and the feature modal attention.

The query ID relative attention is the multiplication of the query matrix by the incoming features from the Transformer flow. the result is then multiplied by the 1D relative position embedding between each pair of tokens. This result is "clipped", so that the model attends to local features more heavily. The key ID relative attention is analogous, but utilizing the key matrix instead. It is important to note that the matrices involved, including the vanilla self-attention that is also calculated with the same key-query matrices, are modality-specific and not shared.

The feature modal attention is calculated with two separate key and query matrices, which are shared across both modalities. This is the point of cross-modal interaction, and the attention scores are calculated from the initial embeddings, not from the Transformer flow. The output of the four attention scores, vanilla, relative key and query and cross-modal, are summed to yield the final attention score.

The Transformer encoder architecture used is modified with the new attention mechanism, and "residual" connections from the input visual features and bounding boxes are added in the odd and even layers, respectively. This is simply for computing the cross-attention. Other than these changes, the architecture used is a simple 12 layers with 12 heads encoder, with a hidden size of 768.

The paper also describes three novel pre-training tasks: Multi-Modal Masked Language Modeling (MM-MLM), which is a twist on the MLM task introduced by BERT Learn To Reconstruct (LTR) and Text Describes Image (TDI). The pre-training goal is the weighted sum of the losses for the three tasks, and the weights are set as 5, 1, and 5 for MM-MLM, LTR and TDI respectively. MM-MLM is described as identical to MVLM from StrucTexT, including the fact that DocFormer does not mask the visual features. The loss function used is also the Cross-Entropy.

LTR is a similar task to MM-MLM, but on the image side. The entire image is masked with the same proportions, and the model is asked to reconstruct the original image using multi-modal features. For this task, a shallow decoder is attached to the output of DocFormer, presumably a one-layer Transformer decoder. Unlike MM-MLM, the loss used is a smooth-L1 loss.

For TDI, the output of the multi-modal encoding is pooled by a linear layer and fed into the model. The model must then predict a binary label for the features, representing whether the image corresponds to the right text. For each batch of images, 20% have their textual features replaced with ones from another image. Since the negative pairing interferes with the LTR task, the LTR loss is ignored in these cases.

DocFormer is presumably pre-trained on the IIT-CDIP dataset, as the paper performs a comparative analysis of pre-training with other models (such as LayoutLM) which have been pre-trained on the same corpus. For the downstream tasks, a trainable linear head is attached to the end of the model for class prediction on each dataset used for evaluation. Our implementation for the entity labeling task follows the practice used in StrucTexT, described above. We train the model on our dataset using the AdamW optimizer with a learning rate of $2e-5$, with an early stopping parameter of three epochs.

5 EXPERIMENTS

In this section, we detail the experiments run on the proposed dataset. The hyper parameters, optimizers and learning rate configurations are all described in section 4.4, and we reiterate these here.

For PICK, we use the standard model configuration found in the official implementation, including learning rate schedulers and other hyper parameters. PICK uses the Adam optimizer with a learning rate of 10^{-4} with no weight decay. The model is trained for at most 100 epochs with an early stopping tolerance of 40 epochs. The graph loss weighting parameter is set to 0.01. Hidden size is set to 512. Decoder features 2 BiLSTM layers, encoder uses 3 layers. Graph module has a hidden size of 128. All these parameters are gathered from the official implementation.

For StrucTexT, the architecture is a standard Transformer encoder with 12 layers and 12 attention heads, and a hidden size of 768. We train the model for up to 50 epochs with an early stopping tolerance of three epochs and a learning rate of $5 * 10^{-5}$, using the Adamax optimizer. We do not perform pre-training, nor use any pre-configured weight initialization.

For DocFormer, the same Transformer architecture is used, with 12 encoder layers, 12 attention heads and a hidden size of 768. The model is also trained for up to 50 epochs with an early stopping tolerance of three epochs. The optimizer used is AdamW with a learning rate of $2 * 10^{-2}$.

Although it may seem unfair to allow a lower early stopping tolerance on the Transformer-based models, our experiments have shown that these models are more prone to overfitting, and also have a much faster convergence speed than PICK. We have found that there is not much to be gained from extensively training the models further. Table 5.1 contains the number of trainable parameters for each model.

5.1 TRAINING PROTOCOLS

Our goal is to understand the impact of adding more synthetic data to our dataset. To this end, we simulate the training with larger and larger datasets by reducing the number of instances in the training set in two ways. The first of them is to remove every instance created from some real images from the dataset, and the second is to remove some instances crated from all real images from the dataset.

These protocols are each carried in two parts. The first is to perform said removal to reduce to 50% the size of the dataset, and the second is to do it to reduce the dataset to 25% of its original size. As such, we have five training protocols, the fifth being the training with the entire dataset.

We would also like to understand whether training with more than one type of document will aid in understanding both. To this end, we perform three separate training stages, for the

Model	Number of trainable parameters
PICK	68.580.056
StrucTexT	178.336.532
DocFormer	120.561.392

Table 5.1: Number of trainable parameters for each model

Partition	Full set	Inst50	Inst25	Dup50	Dup25	Validation	Testing
Front	355	180	90	142	71	130	110
Back	390	195	100	156	78	150	120
Full	745	375	190	298	149	265	245

Table 5.2: Number of synthetic instances in each set for each protocol.

Model	Full set	Instance 50	Instance 25	Duplicate 50	Duplicate 25
PICK	68.98 (90.22)	**73.89** (88.75)	**62.9** (80.65)	**69.98** (89.98)	**69.51** (83.08)
StrucTexT	55.63 (71.37)	50.37 (76.42)	37.75 (70.14)	46.63 (65.06)	42.75 (75.89)
DocFormer	**84.87** (84.15)	73.38 (83.48)	44.5 (81.01)	69.13 (70.9)	48.38 (46.47)

Table 5.3: Micro-averaged F1-score for the front partition.

front, back and both types of documents. With this, every model is evaluated in 15 training rounds.

The dataset is partitioned into the train, validation and testing sets with a ratio of 60%, 20% and 20% respectively. We reduce the training set size in each protocol, keeping the validation and testing sets fixed. We label the first reduction as "instance" reduction, and the second reduction as "duplicate" reduction. The duplicate reduction makes it so the total of instances per real image is 2 and 1 for the 50% and 25% reduction respectively, for both front and back. Table 5.2 presents the total number of synthetic instances in the training, validation and testing sets on each protocol.

5.2 RESULTS

We report the results of the 15 protocols for the three models. Table 5.3 presents the results for the front section of the dataset. We report the F1-score micro-averaged across all entities as the metric. In parenthesis is the highest F1-score reached during the training phase. From these results, we can gather that the Transformer-based methods seem to overfit more quickly. DocFormer manages to reach the best F1-score for this experiment in the full set, but falls behind when the training data becomes scarce.

In particular, we also note that PICK reaches a higher F1-score with a dataset reduction. This may be due to some entities being relatively more scarce in the full set than compared to the instance-reduced set, which makes the training more diluted in the second case.

In table 5.4, we can see a continuation of the trend observed in the front partition for the back partition of the dataset. Again, there is a drastic drop in performance from the training to the testing sets. The biggest example is the Instance 25 protocol on DocFormer, which features a drop of 61.45 in the F1-score. Again, PICK is the leader when dealing with less data, across all datasets. In particular, we note that there does not seem to be a significant change in performance between the reduction types, but it is clear that the addition of more data benefits the training.

Model	Full set	Instance 50	Instance 25	Duplicate 50	Duplicate 25
PICK	**84.73** (95.01)	**73.05** (94.25)	**61.23** (66.13)	**74.38** (91.7)	**72.61** (81.91)
StrucTexT	44.92 (81.43)	40.2 (91.22)	32.94 (93.38)	40.9 (79.5)	43.22 (80.71)
DocFormer	62.51 (94.54)	49.04 (95.77)	30.75 (92.2)	49.55 (93.5)	42.81 (90.76)

Table 5.4: Micro-averaged F1-score for the back partition.

Model	Full set	Instance 50	Instance 25	Duplicate 50	Duplicate 25
PICK	**67.2** (90.72)	**65.41** (85.19)	**48.8** (75.18)	**67.18** (84.13)	**60.5** (72.2)
StrucTexT	54.84 (89.62)	43.28 (88.36)	30.7 (84.52)	50.86 (89.15)	38.87 (82.9)
DocFormer	62.85 (92.54)	48.01 (93.59)	37.04 (86.99)	47.99 (85.33)	27.47 (86.44)

Table 5.5: Micro-averaged F1-score for the entire dataset.

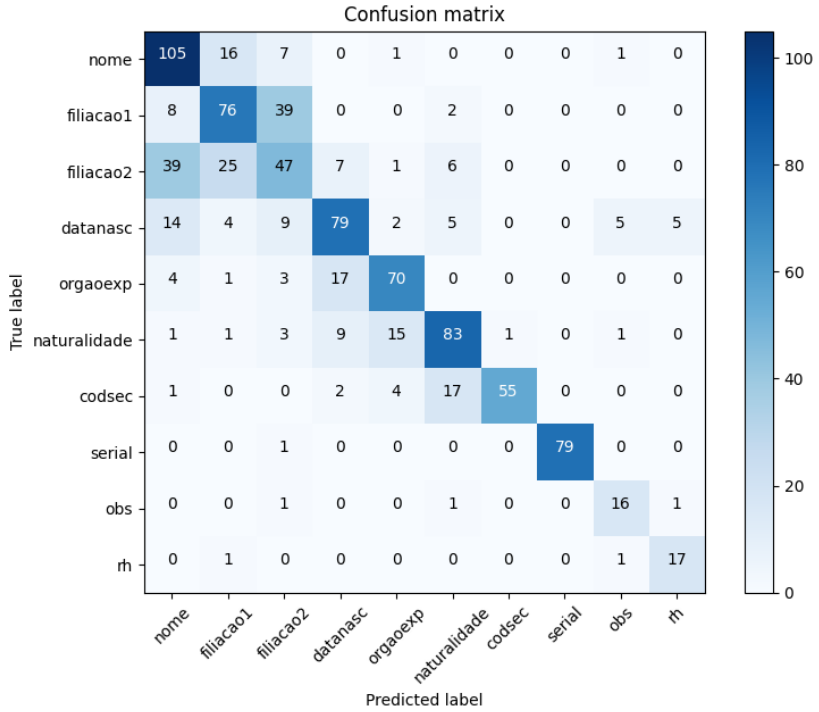


Figure 5.1: Confusion matrix for the front partition

As for the full partition, table 5.5 presents the results for training on the entire dataset. The results do not differ much from what was already seen in the other two partitions. In particular, we note that the models do not experience a rise in performance when compared to the single-mode training, which leads us to believe that multi-modal training does not bring a lot of benefit for performance. On the other hand, we do have a drop in performance, and as such is a potentially harmful practice, even though it could be more practical to have a single model work for both types of documents.

Finally, for a deeper insight into the difficulty of our dataset, we present some of the confusion matrices generated. We focus on PICK, which exhibited the most consistent performance across all models in our training, and present the confusion matrix for the testing set of the full set protocol for the front and back partitions.

Figure 5.1 presents the confusion matrix for the evaluation of PICK when training with the complete partition of the front set. Of particular interest is the cluster on the name fields: nome and filiacao1 and 2. These represent the card holder name and the two parents' names. The semantic similarity is not well discriminated by the layout features, and confusion occurs.

Figure 5.2 presents the confusion matrix for the evaluation of PICK when training with the complete partition of the back set. We highlight that the entities with fewer instances in the set have poorer relative performance. For cns, profissional and militar, the model failed more often than succeeded. This may be due to the scarcity of these fields in our dataset. We note that the dni field was not present in the testing set.

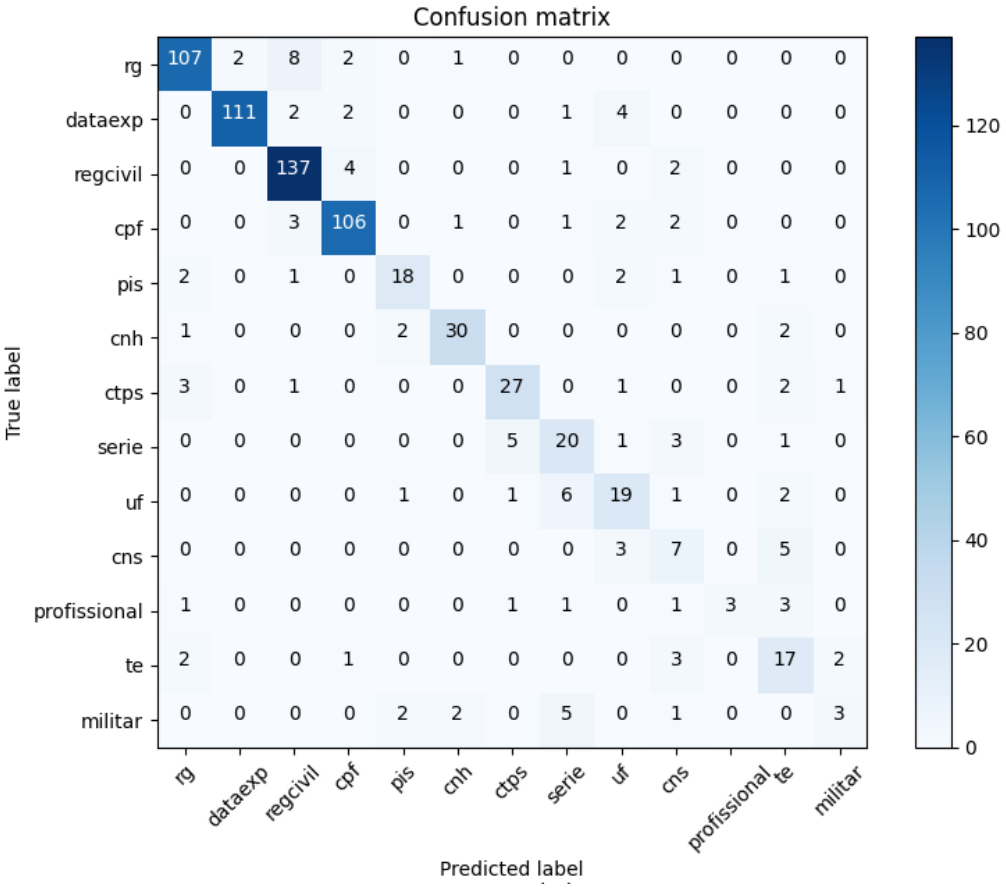


Figure 5.2: Confusion matrix for the back partition

5.2.1 Discussion and Conclusion

Our results have shown a constant trend of an increase in performance when adding more data to the dataset. We also note that the dataset is fairly challenging due to the variations in data, especially for fields that are relatively scarce in the document. Our studies with real data are left as a future work.

6 CONCLUSION

In this work, we focused on building a dataset for document vision tasks. We built a synthesizer for document anonymization and text generation, expanding upon previous research for broader territories in the field of Document Recognition. We presented a novel inpainting method using a specially trained GAN and conditional rule-based entity generation to achieve maximum fidelity to the real instances.

Finally, we evaluated the dataset using three different VDU models. Our experiments focused on learning the impact of adding more data to the training set, and we show that annotating more instances, even though it may be more expensive in time and resources, boosts performance more than adding more synthetic instances per real image. We also show that the back partition proves a bigger challenge and that joint training does not benefit the models.

We conclude that adding more synthetic data to the training sets largely improve model performance, but we still note some difficulty in generalization. For future directions, we aim at recreating the experiments with real data, in order to analyze the impact of adding synthetic data in performance. We may also expand the synthesizer by adding support to different kinds of document.

REFERENCES

Appalaraju, S., Jasani, B., Kota, B. U., Xie, Y., and Manmatha, R. (2021). Docformer: End-to-end transformer for document understanding. *CoRR/arXiv*, abs/2106.11539.

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.

Bakkali, S., Ming, Z., Coustaty, M., Rusiñol, M., and Terrades, O. R. (2022). Vlcdoc: Vision-language contrastive pre-training model for cross-modal document classification.

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

Chen, J., Lv, T., Cui, L., Zhang, C., and Wei, F. (2022). Xdoc: Unified pre-training for cross-format document understanding.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Dimmick, D., Garris, M., and Wilson, C. (1992). Nist special database 6 - structured forms database ii user's guide.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. *CoRR*, abs/2010.11929.

Garris, M. (2008). Nist special database 2. nist structured forms reference set of binary images (sfrs).

Gera, A., Halfon, A., Shnarch, E., Perlitz, Y., Ein-Dor, L., and Slonim, N. (2022). Zero-shot text classification with self-training.

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, H., Qin, X., Liu, J., Han, J., Liu, J., and Ding, E. (2019). EATEN: entity-aware attention for single shot visual text extraction. *CoRR*, abs/1909.09380.

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, K., Gkioxari, G., Dollár, P., and Girshick, R. B. (2017). Mask R-CNN. *CoRR*, abs/1703.06870.

He, K., Zhang, X., Ren, S., and Sun, J. (2015). Deep residual learning for image recognition. *CoRR*, abs/1512.03385.

Heckbert, P. (1998). Color image quantization for frame buffer display. *ACM SIGGRAPH Computer Graphics*, 16.

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.

Iizuka, S., Simo-Serra, E., and Ishikawa, H. (2017). Globally and locally consistent image completion. *ACM Transactions on Graphics*, 36:1–14.

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).

Karras, T., Laine, S., and Aila, T. (2018). A style-based generator architecture for generative adversarial networks. *CoRR*, abs/1812.04948.

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)*.

Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., and Dyer, C. (2016). Neural architectures for named entity recognition. *CoRR/arXiv*, abs/1603.01360.

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., and Zettlemoyer, L. (2020). BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

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

Li, P., Gu, J., Kuen, J., Morariu, V. I., Zhao, H., Jain, R., Manjunatha, V., and Liu, H. (2021a). Selfdoc: Self-supervised document representation learning. *CoRR*, abs/2106.03331.

Li, Y., Qian, Y., Yu, Y., Qin, X., Zhang, C., Liu, Y., Yao, K., Han, J., Liu, J., and Ding, E. (2021b). Structext: Structured text understanding with multi-modal transformers. *CoRR/arXiv*, abs/2108.02923.

Lin, T., Dollár, P., Girshick, R. B., He, K., Hariharan, B., and Belongie, S. J. (2016). Feature pyramid networks for object detection. *CoRR*, abs/1612.03144.

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., and Guo, B. (2021). Swin transformer: Hierarchical vision transformer using shifted windows. *CoRR*, abs/2103.14030.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013a). Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013b). Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

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.

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.

Reimers, N. and Gurevych, I. (2019). Sentence-bert: Sentence embeddings using siamese bert-networks. *CoRR*, abs/1908.10084.

Ren, S., He, K., Girshick, R., and Sun, J. (2015). Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems (NIPS)*.

Schuster, D., Muthmann, K., Esser, D., Schill, A., Berger, M., Weidling, C., Aliyev, K., and Hofmeier, A. (2013). Intellix – end-user trained information extraction for document archiving. In *2013 12th International Conference on Document Analysis and Recognition*, pages 101–105.

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

Smith, R. (2007). An overview of the tesseract ocr engine. In *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*, volume 2, pages 629–633.

Song, X., Salcianu, A., Song, Y., Dopson, D., and Zhou, D. (2020). Linear-time wordpiece tokenization. *CoRR*, abs/2012.15524.

Stanislawek, T., Gralinski, F., Wróblewska, A., Lipinski, D., Kaliska, A., Rosalska, P., Topolski, B., and Biecek, P. (2021). Kleister: Key information extraction datasets involving long documents with complex layouts. *CoRR*, abs/2105.05796.

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., 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. I. (2022). Mgdoc: Pre-training with multi-granular hierarchy for document image understanding.

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., 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.

Yang, X., Yumer, E., Asente, P., Kraley, M., Kifer, D., and Giles, C. L. (2017). Learning to extract semantic structure from documents using multimodal fully convolutional neural network. *CoRR/arXiv*, abs/1706.02337.

Yi, K., Shen, X., Gou, Y., and Elhoseiny, M. (2022). Exploring hierarchical graph representation for large-scale zero-shot image classification. *ECCV*.

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.

Zhang, Z., Han, X., Liu, Z., Jiang, X., Sun, M., and Liu, Q. (2019). ERNIE: Enhanced language representation with informative entities. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1441–1451, Florence, Italy. 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.

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 International Conference 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*.

Zujovic, J., Pappas, T. N., and Neuhoff, D. L. (2013). Structural texture similarity metrics for image analysis and retrieval. *IEEE Transactions on Image Processing*, 22(7):2545–2558.

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.