

Ein modulares Modell, das visuelle und textuelle Merkmale zur Klassifizierung von Dokumentenbildern kombiniert

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Data Science

eingereicht von

Amer Duhan, BSc.

Matrikelnummer 01350859

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Robert Sablatnig

Mitwirkung: Univ.Prof. Dr. Allan Hanbury

Wien, 26. April 2022

Amer Duhan

Robert Sablatnig



Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.



A Modular Model Combining Visual and Textual Features for Document Image Classification

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieur

in

Data Science

by

Amer Duhan, BSc.

Registration Number 01350859

to the Faculty of Informatics

at the TU Wien

Advisor: Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Robert Sablatnig

Assistance: Univ.Prof. Dr. Allan Hanbury

Vienna, 26th April, 2022

Amer Duhan

Robert Sablatnig



Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Erklärung zur Verfassung der Arbeit

Amer Duhan, BSc.

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 26. April 2022

Amer Duhan



Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Acknowledgements

First of foremost, I want to thank my advisor, Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Robert Sablatnig, for giving me the opportunity to write the thesis under his supervision, as well as giving me support and guidance during my writing. I am also really grateful for the support and help from Univ.Prof. Dr. Allan Hanbury.

Finally, I want to thank my family for their support during the writing of the thesis as well as during my study.



Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Kurzfassung

Die Klassifizierung von Dokumentenbildern ist die Klassifizierung von digitalisierten Dokumenten. In der Regel werden diese Dokumente entweder gescannt oder fotografiert. Eine Seite eines solchen Dokuments wird als Dokumentenbild bezeichnet. Die Klassifizierung von Dokumentenbildern ist eine wichtige Aufgabe, da sie ein erster Schritt in nachgelagerten Anwendungen ist. Dieser Schritt wird von Unternehmen derzeit noch manuell durchgeführt, was einen erheblichen Zeit- und Kostenaufwand darstellt. Die Ursache dafür ist, dass Systeme zur Klassifizierung von Dokumentenbildern eine sehr hohe Genauigkeit benötigen (weit über 90%), insbesondere weil dies potentiell der erste Schritt in nachgelagerten Anwendungen ist. Eine hohe Genauigkeit zu erreichen, ohne einen Datensatz mit Millionen von annotierten Dokumenten zu haben, ist nicht trivial. Das derzeit beste Modell zur Klassifizierung von Dokumentenbildern basiert auf einem transformer Netzwerk, welches auf mehr als 11 Millionen gescannten Dokumentenbildern vortrainiert wurde und daher einen enormen Ressourcenaufwand zum Trainieren erfordert. Außerdem haben dieses und andere ähnlich gute Modelle zur Klassifizierung von Dokumentenbildern weit über 100 Millionen Parameter. In dieser Arbeit bewältigen wir beide Herausforderungen. Erstens erstellen wir ein Modell, welches mit dem aktuell besten Modell konkurrieren kann, ohne dass Millionen von gescannten Dokumentenbildern zum Trainieren verwendet werden. Zweitens erstellen wir ein Modell, welches in Bezug auf die Parameter kleiner ist als die aktuell besten Modelle. Um dies zu erreichen, werden die aktuell besten Modelle als Basis verwendet, welche relativ klein in ihrer Größe sind. Ihre optimale Einstellung wird zunächst durch eine Hyperparameteroptimierung auf einer Teilmenge der Daten gefunden. Der Input für diese Modelle basiert auf Bild- und Textmerkmalen. Deren Output wird dann miteinander kombiniert, und ein abschließender Meta-Klassifikator wird auf diese Ergebnisse trainiert, um das finale Ergebnis zu erzeugen. Die Ergebnisse zeigen, dass das entwickelte System, in Bezug auf die Genauigkeit, mit dem aktuell besten Modell mithalten kann. Außerdem benötigt es weniger Parameter zum Trainieren und ist aufgrund seiner modularen Natur leicht parallelisierbar. Ferner zeigen die Ergebnisse, welche spezifischen Teile von Dokumentenbildern für die Klassifizierung wichtig sind. Je nachdem, wie effizient das System sein soll, können weniger Module aus dem System ausgewählt werden.



Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Abstract

Document image classification is the classification of digitized documents. Typically, these documents are either scanned or photographed. One page of such a document is referred to as a document image. Classifying document images is a crucial task since it is an initial step in downstream applications. This step is done manually by companies as of now, which takes considerable time and financial resources. The main reason is the need of a very high accuracy (well beyond 90%) for document image classification systems, specifically because it is potentially the first step in a series of downstream applications. However, achieving a high accuracy without having a dataset with millions of annotated documents is not trivial. The current state-of-the-art document image classification model is based on a transformer network, which is pretrained on more than 11 million scanned document images and thus requires a huge amount of resources to train. Additionally, this and other state-of-the-art document image classification models have well beyond 100 million parameters. In this work, we address both challenges. First, we create a model which is capable to compete with the current state-of-the-art model without pretraining on millions of scanned document images. Second, we create a model which is smaller than current state-of-the-art models, in terms of parameters. To achieve this, current state-of-the-art models are used as base models, which are relatively small in size. Their optimal setting is first found in a hyperparameter tuning on a subset of the data. The input to these models is based on image and text features. Their output is then combined, and a final meta-classifier is trained on these outputs to generate the final result. The results show, that the developed approach can compete with current state-of-the-art models in terms of accuracy. Additionally, it requires fewer parameters to train, and it is easily parallelizable, due to its modular nature. Moreover, the results show, which specific parts of document images are important for classification. Depending on how efficient the system should be, fewer modules from the system can be chosen.



Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Contents

Kurzfassung	ix
Abstract	xi
Contents	xiii
1 Introduction	1
1.1 Motivation and problem statement	1
1.2 Aim of this thesis	4
1.3 Structure	5
2 Datasets	7
2.1 Ryerson Vision Lab Complex Document Information Processing (RVL-CDIP)	7
2.2 Tobacco3482	10
2.3 Summary	10
3 Related work and background	13
3.1 Pre-deep learning approaches	13
3.2 Deep Learning Approaches	15
3.3 Features	17
3.4 Summary	18
4 Methodology	21
4.1 Image stream	21
4.1.1 Convolutional Neural Networks	21
4.1.2 Base classifier	23
4.1.3 Preprocessing steps	24
4.1.4 Hyperparameter tuning	25
4.1.5 Training strategy and architecture	27
4.2 Text stream	28
4.2.1 Transformer	28
4.2.2 Base classifier	31
4.2.3 Text extraction	32
4.2.4 Preprocessing steps	32

4.2.5	Hyperparameter tuning	33
4.2.6	Training strategy and architecture	34
4.3	Stacked generalization	34
4.3.1	Features from the last convolutional and hidden layer	37
4.3.2	Class probabilities	37
4.4	Tobacco3482	37
4.5	Summary	38
5	Results	41
5.1	Hyperparameter tuning	41
5.1.1	Image stream	41
5.1.2	Text stream	43
5.1.3	Meta-classifier results on RVL-CDIP	44
5.2	Results on Tobacco3482	45
5.3	Results on RVL-CDIP	46
5.4	Parameters and runtimes	48
5.5	Summary	51
6	Discussion	53
7	Conclusion	61
7.1	Summary and limitations	61
7.2	Contributions and future work	62
	List of Figures	65
	List of Tables	67
	Acronyms	69
	Appendix	71
A.	Further tuning	71
B.	Tobacco3482 results	73
B1.	Confusion matrix	73
B2.	Softmax output	74
C.	RVL-CDIP results	76
D.	System and Libraries and used	80
	Bibliography	81

Introduction

This master thesis deals with classifying document images, by using a modular model that combines visual and textual features.

1.1 Motivation and problem statement

As early as 1978, the concept of a paperless society was introduced by Lancaster [1]. This type of society would rely on a digital communication, without the need of any paper based communication. Even though paperless offices may exist, that concept has not yet completely caught on in society [2].

The increasing digitalization has led companies to digitize their processes and content [3], as well as to organize their information in order to improve the search and access to relevant data [4]. Thus, paper documents are subject to digitization and the output of this are document images [5]. That is, a "document" in document image refers to a single page, while "image" refers to the document being present in a format, from which the text has to be extracted in order to be able to work with the text directly [6]. The task of document image classification is to categorize a given document image into a set of defined classes [7].

Many different types of document exist, such as receipts, financial reports, police reports, e-mail, accident reports, assessments, etc. They can occur as electronic files, i.e., already digitized from the beginning, or they may be in scanned form, coming either from written or printed paper [8]. Moreover, they can contain, amongst others, figures and tables and be written as a two-column layout, or any other layout. This makes the task of classifying documents challenging, due to different layouts and formats, as well as the potentially poor quality of scanned/photographed documents. In addition, the just mentioned types of documents may come in a different page layout due to human error,

1. INTRODUCTION

i.e., landscape format instead of portrait format or they may be rotated by an arbitrary amount.

With increasing numbers of digitized documents, the variability for each document class increases as well. This adds further complexity to a document classification system [9]. Documents have potentially high intra-class variability and low inter-class variability [10]. For instance, two different reports can look very different, while a memo and a letter can look very similar. This scenario can be seen in the Figures 1.1 and 1.2.

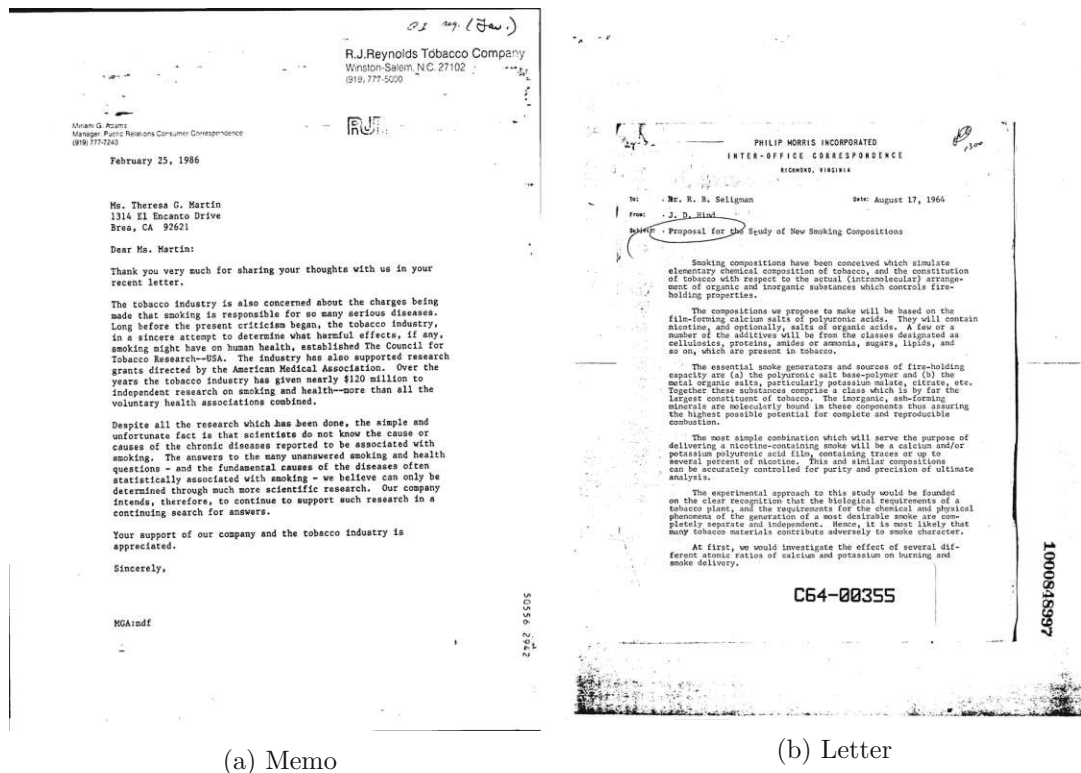


Figure 1.1: The images are from different categories, but show relatively low interclass variability [11].

As of now, companies classify documents manually [8]. Alternatively, automation through machine learning would save human and financial resources [8]. Automatic classification is an initial step in document processing tasks, such as document retrieval and information extraction, and has a significant role in indexing the documents of a digital library [9]. For example, if the goal is to extract information from a certain category, then document classification as a previous step is necessary. Regarding indexing, classifying document images into a table of contents page or title page narrows the set of pages from which to extract meta data [12].

Document image analysis, which encompasses document image classification [13] and other tasks, such as character recognition, is one of the first topics, where deep

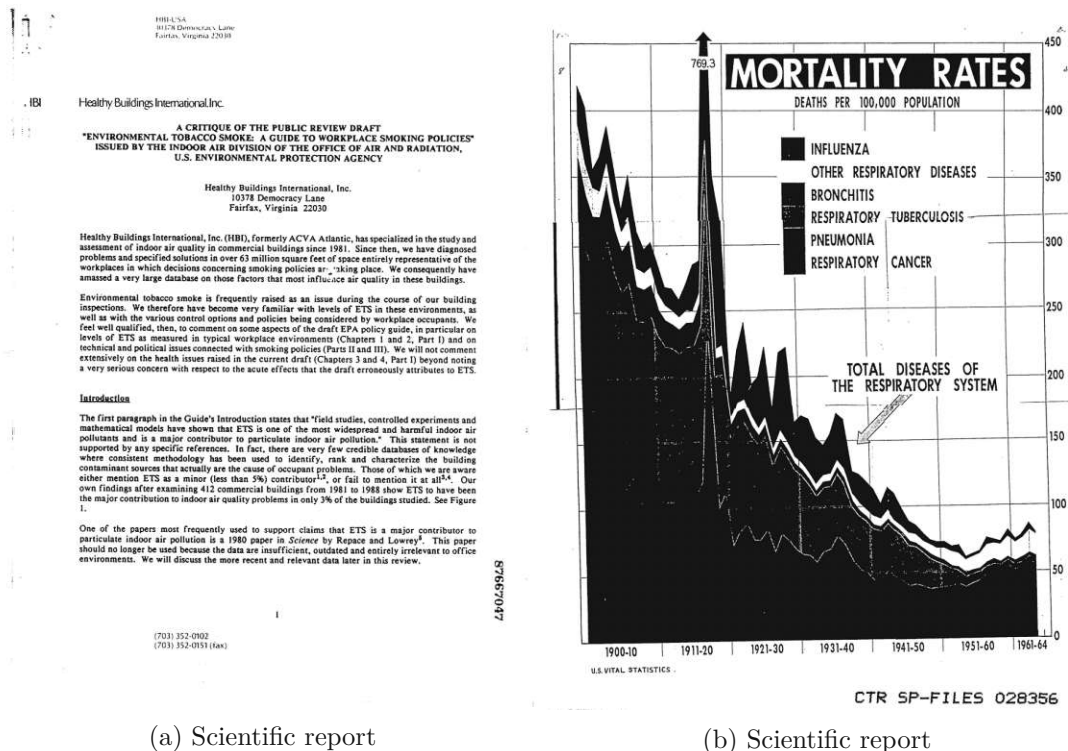


Figure 1.2: The images are from the same category, but show relatively high intraclass variability [11].

learning has been applied to [3]. This topic dates back to the early 1970's [5] and different non deep learning techniques have been tested for document image analysis, which are discussed in Section 3.1. While document classification approaches can generally be grouped into image based and content based approaches, the image based approach is preferred in case of digitized documents [14]. For content based approaches, the text first needs to be extracted using an Optical Character Recognition (OCR) system [9]. Naturally, a third possibility exists, which is combining image and text.

Achieving a high accuracy in document image classification is of importance to any company or government entity that deals with a significant amount of digitized documents. Due to this importance, document image classification has been explored extensively [15]. However, there is relatively little research into how to integrate textual information into Convolutional Neural Network (CNN) approaches [7]. In this work, textual information is used alongside CNN models. Furthermore, the amount of training data is much less, compared to the state-of-the-art (SOTA) model [16], which includes a pretraining on millions of document images. Pretraining a model on a large dataset and then finetuning that model has advantages in terms of accuracy [17]. Thus, achieving a comparable accuracy with the SOTA model without having a dataset with millions of document images is not trivial. This leads to the following question, which this thesis addresses:

How does a modular multimodal model without pretraining on millions of documents, based on state-of-the-art image and language models, compare against the state-of-the-art models, with pretraining, on document image datasets?

1.2 Aim of this thesis

The aim of this thesis is to develop a system, that can compete with current SOTA document image classification systems.

To achieve that aim, visual and textual features are combined. One important aspect is the amount of data used. In this thesis, the training data does not exceed millions of documents, as is done in the current SOTA model [16]. Another important aspect is to have a relatively small system, i.e., a system that does not have parameters in the hundreds of millions. Hyperparameter tuning is carried out, in order to optimize the system in terms of accuracy.

The effect of pretraining and transfer learning is also examined. To test the accuracy on the smaller dataset, first a pretraining is done on the bigger dataset, and then transfer learning is applied on the smaller dataset. The datasets used in this work are described in the next chapter.

Lastly, the individual components, i.e., modules or submodels, of the modular model are examined. In particular, it is tested, how well the submodels perform in comparison to the whole system in terms of accuracy.

Thus, in addition to the main research question presented in the previous section, the following ones are addressed:

- How efficient, in terms of parameters, can the system be made and still compete with SOTA models?
- What are the optimal hyperparameters and their effect on the accuracy?
- How well, in terms of accuracy, does finetuning on the smaller dataset work?
- What are the impacts of the submodels on the overall accuracy?

The main contributions of this work are the following:

- Developing a model which can compete with current SOTA models on the RVL-CDIP dataset, without the need of millions of document images. Moreover, the developed model has over 10 times less parameters than the current SOTA model.
- Improving the SOTA on the Tobacco3428 test dataset by 2.21 percentage points.

1.3 Structure

Chapter 2 introduces the datasets used in this work. It describes the datasets in detail, including their distribution.

Chapter 3 describes the related work and background. The related work is systematically divided into different categories and subcategories. These subcategories are analyzed and summarized, and differences between them are pointed out.

Next, the methodology is described in this work. The methodology is divided into two parts. For each part, the necessary theoretical background is provided and the proposed system is explained.

The results are presented in Chapter 5. A much more detailed presentation of the results is available in the Appendix. Additionally, a comparison with other works is shown as well as the results of other metrics such as runtime and number of parameters.

Chapter 6 discusses and interprets the presented results. Moreover, advantages and disadvantages of the proposed approach are shown.

Chapter 7 concludes the work and points out some limitations. Furthermore, it provides a summary, points out contributions and adds some considerations for future work in the field of document image classification.



Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Datasets

This chapter introduces the datasets which are used in this work. A detailed description, including their distribution and some findings related to preprocessing are presented.

2.1 RVL-CDIP

This work is based on the RVL-CDIP [11] dataset since it was specifically created to test image classification algorithms on document images [18]. RVL-CDIP is a subset of the Illinois Institute of Technology Complex Document Information Processing (IIT-CDIP) Test Collection (11 million documents) [19], which itself is a subset of the Legacy Tobacco Document Library (LTDL) dataset [20] (14 million documents), that was created from public records of lawsuits against American tobacco companies [11]. The RVL-CDIP dataset contains 400,000 grayscale images with 16 classes, split evenly in a 8:1:1 ratio of training, validation and test set, where the images are sized so that their largest dimension does not exceed 1,000 pixels. However, 1 document image from the test set of class scientific publication is corrupted, resulting in a test set size 39,999 document images and thus an overall dataset size of 399,999. That amount is used as an input to CNNs. However, the same amount (399,999) of texts can not be used as an input to transformers, since the text can not be extracted from every document image, or it is lost when performing the preprocessing steps from Section 4.2.4. Figure 2.1 shows the number of document images per category, where no text is left. By far the most represented class without any text is File folder and in total, the text from 16,040 document images is not available after applying OCR and performing preprocessing.

The images are divided evenly into the following 16 classes: letter, form, email, handwritten, advertisement, scientific report, scientific publication, specification, file folder, news article, budget, invoice, presentation, questionnaire, resume, and memo. Figures 1.1 and 1.2 both show example images from this dataset, as well as Figure 2.2, with an example image from each class.

2. DATASETS

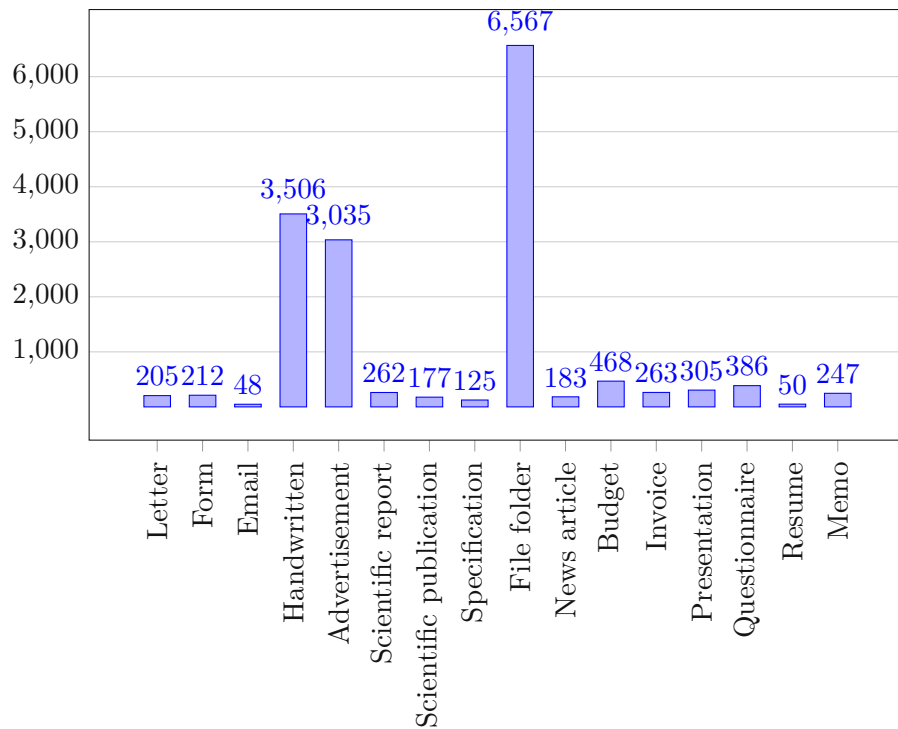
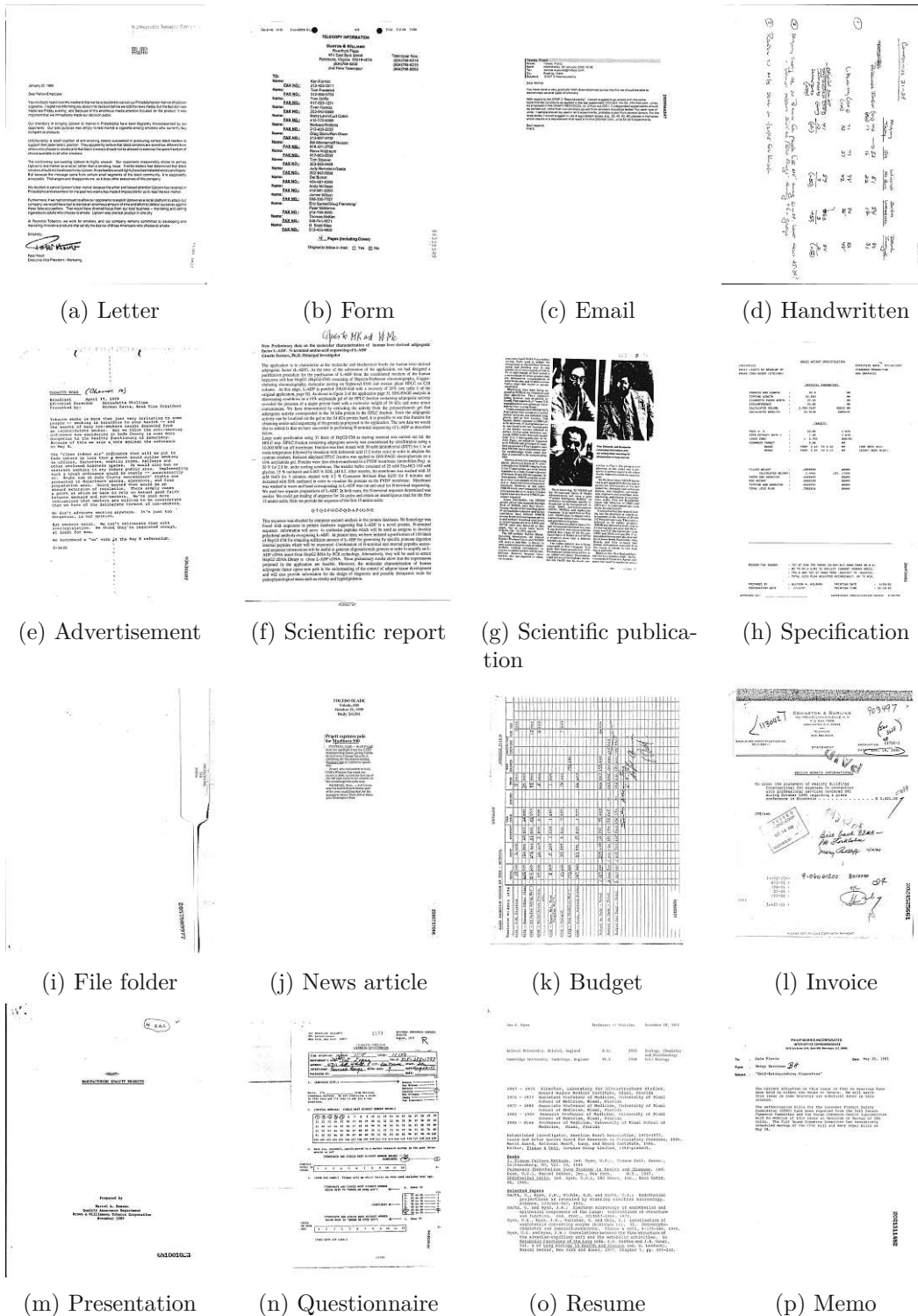


Figure 2.1: Number of files per category, where no text is left, after applying OCR and text preprocessing.

Harley et al. [11] chose the categories based on earlier work on document categorization and on the Tobacco3482 [21] dataset that existed already. Each document in RVL-CDIP belongs to exactly one class. However, since the IIT-CDIP Test Collection has sometimes multiple classes per document, the final categories in RVL-CDIP are not perfectly distinct. See Figure 1.2, for an example of this scenario.



(a) Letter

(b) Form

(c) Email

(d) Handwritten

(e) Advertisement

(f) Scientific report

(g) Scientific publication

(h) Specification

(i) File folder

(j) News article

(k) Budget

(l) Invoice

(m) Presentation

(n) Questionnaire

(o) Resume

(p) Memo

Figure 2.2: An example document image for each class from the RVL-CDIP [11] dataset. Notice the visual similarities, for example in (c) Email and (p) Memo, showing a low interclass variability.

2.2 Tobacco3482

The Tobacco3482 [21] dataset, created from the same dataset as RVL-CDIP, the IIT-CDIP Test Collection, contains 3,482 grayscale document images. These images are split into 10 classes, which are not evenly distributed as in the RVL-CDIP dataset. Figure 2.3 shows the classes and their distribution for this dataset.

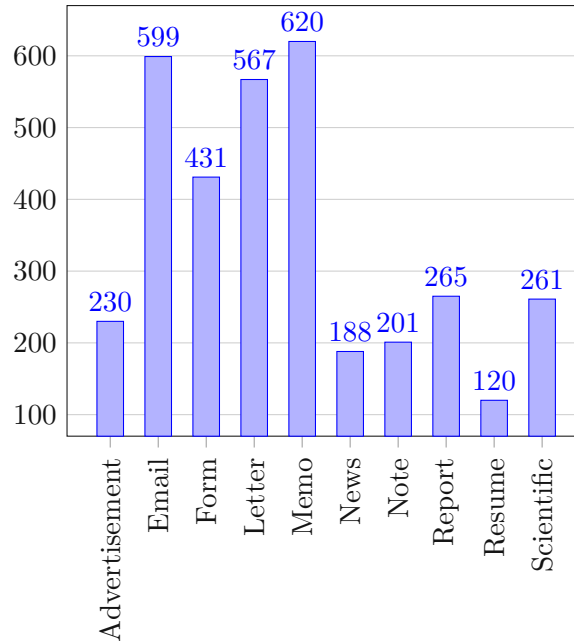


Figure 2.3: Class distribution of the Tobacco3482 [21] dataset.

There is not a predefined training test and validation split, as in RVL-CDIP. To make the results comparable with other works, the same dataset splits are used. The approach to this is described in detail in Section 4.4. Furthermore, unlike in the RVL-CDIP dataset, the document images in this one are not resized in any way.

Similarly as in the RVL-CDIP dataset, the text of some document images is not available after applying OCR and performing preprocessing. In total, the text for 27 document images is not available, with 21 document images being from the class Note and 6 from the class Advertisement.

2.3 Summary

The proposed approach in this work is evaluated on two datasets: RVL-CDIP and Tobacco3482. The RVL-CDIP dataset is more than 100 times larger than the Tobacco3482 dataset, in terms of the number of document images (400,000 vs 3,482) and has 6 more

classes (16 vs 10). RVL-CDIP has an even class distribution, unlike Tobacco3482. Both datasets are based on the IIT-CDIP Test Collection.



Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Related work and background

Generally, the problem of document image classification can be tackled through approaches that differentiate based on the chosen features and the learning algorithms [22].

This chapter gives an overview of both features and learning algorithms. First, an overview of strategies of the pre-deep learning era, as well as an overview of the current SOTA methods. Common strategies, and in particular differences are pointed out. A few strengths and weaknesses of the methods and fundamental concepts are discussed.

Second, the features used in document image classification systems are explained and their connection is summarized as a tree.

3.1 Pre-deep learning approaches

Taylor et al. [23] focus on business documents and design a two-layer hierarchy method. The first layer uses a column detection algorithm, similar to [24], which uses all text and image blocks to detect column boundaries. The second layer is a classifier based on rules. If the classifier finds a certain number of landmarks for a class, it is categorized as that class. A landmark is for instance a salutation, a closing or a date. These landmarks are extracted by an OCR system.

Byun et al. [25] suggest to use a partial matching method in which form structure recognition and form classification are performed for only some areas of the input form. A document is divided into $n \times m$ areas, where horizontal and vertical lines are extracted. The difference of coordinates between two neighboring lines is computed, which serves as the feature. To compute the similarity of an input document and a predefined form model, a weighted graph is calculated using Dynamic Programming (DP) matching. The input form is then classified as one of N known model forms.

In another, mainly rule-based approach, Kochi and Saitoh [26] propose a system, which can handle semi-formatted documents, that is, a format which is not fixed, but

does have recurring textual elements, such as title or authors. This makes it robust against noise in the target document. A model is created by segmenting the first page of a document. Then, textual elements are manually added to those segmented areas, like author and title. A number of such models are created. A document is then classified by computing the minimum distance to those created models in the first step. The main difference of this approach, compared to [23] and [25] is that it is not fixed to particular document types, but it can handle arbitrary different types of documents, depending on how many models are created in the first step.

The following machine learning based approaches mainly emerged after the just described rule-based systems. Shin and Doermann [27] first perform a connected component analysis on foreground and background pixels as a preprocessing step. From those connected components, statistics, such as the count, sum, median or variance are computed. Additional features, such as the percentages of text and non-text (graphics, image, table, and ruling), column structures, variations in the point sizes of fonts or the density of content area, are computed. These features are the input for a decision tree classifier, specifically for the Oblique Classifier 1 (OC1) [28] decision tree.

Shin et al. [29] have a similar approach to [27]. Document features for layout classification, such as content type (text, graphic or image), foreground pixel window density, foreground pixel bounding box density, foreground bounding box density, foreground to background pixel ratio, foreground connected component statistics or foreground connected component bounding box statistics are used for the classification. In total, 21 features are used for the OC1 decision tree classifier.

Hu et al. [30] use a spatial layout representation, which encodes region layout information in fixed-length vectors and thus capturing structural characteristics of the image. These features are used in a Hidden Markov Model (HMM) to classify document images. In addition to classifying document types, the layout is also classified to distinguish between, for instance, a one column letter or a two column letter.

Bagdanov and Worring [31] approach the problem of classifying documents into genres, using Attributed Relational Graphs (ARGs) and First Order Random Graphs (FORGs). They use ARGs to represent the layout structure and FORGs to represent document genres. On their test data set of 130 documents, they report a 24 % higher accuracy, compared to a decision tree.

Reddy and Govindaraju [32] compute features in a pyramidal decomposition. In particular, the black pixel density in each recursive cut is calculated. The first level denotes the black pixel density for the whole image, the second level for 4 regions, the third for 16 regions, etc. For example, a 5 level cut returns a 341 dimensional feature vector ($1 + 4 + 16 + 64 + 256$). The authors then apply Principal Component Analysis (PCA) on these vectors, and feed them into an Adaptive Boosting (AdaBoost) training algorithm, where the weak learner consists of the k-means algorithm. That is, the k-means algorithm is repeatedly called. Like [25], this approach is limited to forms.

Kumar et al. [33] use a Bag-of-words (BOW) model to classify document images. A

Support Vector Machine (SVM) is used in their work for each segmented zone, with Triple Adjacent Segment (TAS) features as input, a special type of the k-Adjacent-Segment features [34], where $k=3$. The zones are obtained by applying the Voronoi++ [35] method. For the final page classification, a zone-wise voting is conducted. However, their classification is restricted to classifying whether an image is machine printed or handwritten.

In another BOW model, Kumar et al. [36] propose a method based on statistics of patch-codewords over different regions of an image. They start by extracting patches from unlabeled images to learn a codebook. To model the spatial relationships between patches, the image is recursively partitioned horizontally and vertically, and a histogram of patch-codewords is computed in each partition. A Random Forest (RF) model is trained with the extracted features.

Kumar et al. [37] extend the approach of [36]. The downside of the approach of Kumar et al. [36] is, that raw image-patch based features, albeit fast to compute, are not invariant to scale or robust to noise. Thus, in the extended approach, Kumar et al. [37] use Speeded up Robust Features (SURF), which are several times faster and more robust to noise than Scale-Invariant Feature Transform (SIFT) features. The features are the input for a RF and SVM.

Table 3.1 provides an overview of the main methods of each described pre-deep learning work in this section, with their corresponding used features.

Paper	Method	Feature(s)
[23]	2 layer classifier	predefined Landmarks
[25]	Similarity to N form templates with DP matching	Lines
[26]	Similarity to N templates with a matching distance	Image segments
[27]	OC1 classifier	Image features
[29]	OC1 classifier	Image features
[30]	HMM	Interval encoding
[31]	Graphs (ARGs and FORGs)	Layout
[32]	AdaBoost with k-means as weak learner	Black pixel density
[33]	SVM on Voronoi++ segments and zone-wise voting	TAS
[36]	RF	Image-patch based
[37]	RF and SVM	SURF

Table 3.1: Overview of different methods and their used features.

3.2 Deep Learning Approaches

The methods in all of the following works are tested on the RVL-CDIP test set. The most used class of neural networks present in all deep learning approaches is the CNN. The CNN is either being used as a whole document image classification system, or as a part of such system.

Harley et al. [11], who have created the RVL-CDIP dataset, stack 5 CNNs, one of which is trained on the whole document image, and the others are trained over the header, footer, left body and right body. These CNNs are either trained from scratch or transfer-learned from AlexNet [38]. The feature vectors of each model are then compressed via PCA [39], which is a dimensionality reduction technique. Das et al. [9] use a similar technique. However, their CNNs are transfer-learned from Visual Geometry Group (VGG)-16 [40]. A Multi layer perceptron (MLP), a class of artificial neural networks, is then found to perform as the best ensemble technique.

Transfer learning refers to applying the knowledge of a neural network from one domain to another, related domain by transferring information [41]. Generally, transfer learning on a pretrained network, such as the VGG-16 works well due to effectiveness and generality of the learned representations [42]. However, note that the domain, on which the VGG-16 is trained, is the ImageNet dataset [43], and thus not related to the RVL-CDIP dataset. Afzal et al. [15] show that even though these two datasets have different domains, a pretrained network on ImageNet has a better accuracy score on the RVL-CDIP test set than no pretraining. In particular, AlexNet, GoogLeNet [44], VGG-16 and Residual Network (ResNet)-50 [45] show a better accuracy score when pretraining, with VGG-16 having the best test set accuracy on RVL-CDIP.

Tensmeyer and Martinez [13] train CNNs from scratch, i.e., randomly initialized. Various modifications are performed, such as changing the network depth, i.e., removing or adding convolutional layers, changing the width, i.e., changing the number of neurons in each layer or changing the input size. The input size has a significant impact on the performance, and it is found that a CNN, which can handle varying input sizes from 320 to 512 pixels, achieves a higher accuracy than smaller input sizes.

Sarkhel and Nandi [46] utilize a spatial pyramid model to extract highly discriminative multi-scale feature descriptors from a visually rich document by leveraging the inherent hierarchy of its layout. The spatial pyramid model in this work is a hierarchical construct of components at various layout level abstractions of the document, where each level of the pyramid corresponds to an anonymized equivalent image at that level of abstraction.

Ferrando et al. [10], Jain and Wigington [7] and Audebert et al. [3] combine image and text features by utilizing a CNN for image and an embedding for text. Jain and Wigington [7] use the VGG-16 to get image features, and 3 different methods to extract text features. These 3 text features represent text at the sequence, word and character level, which are combined using an ensemble method, which in turn is combined with the image features. Audebert et al. [3] utilize the MobileNetV2 [47] for image feature extraction, which has a similar performance in terms of accuracy, compared to VGG-16, while being significantly faster. As in [7], word level text features are generated with FastText [48], a word embedding technique. Ferrando et al. [10] combine EfficientNet [49] for image features and a reduced version of Bidirectional Encoder Representations from Transformers (BERT) [50], a transformer model, for text features.

A transformer [51] architecture for document image classification is used in the work

of Xu et al. [8]. More specifically, this architecture is an extended version of BERT [50]. However, the model is pretrained on the IIT-CDIP Test Collection, which contains more than 11 million scanned document images. Another major difference, compared to all other approaches, is that this method is not only suitable for classifying document images, but also for example for form understanding, where the goal is to extract key-value pairs from document images.

Finally, Xu et al. [16] extend [8]. It integrates visual information in the pre-training stage and uses 2-D relative position representation for token pairs, instead of absolute 2-D position embeddings, which Xu et al. [8] use to model the page layout. This model is the current SOTA model and is also pretrained on the IIT-CDIP Test Collection. Just as its predecessor, this model is also suitable for other tasks outside of classifying document images.

Table 3.2 provides an overview of the main methods of each described deep learning work in this section, with their corresponding used features.

Paper	Main method(s)	Feature(s) ¹
[11]	5 stacked CNNs based on AlexNet	I
[9]	5 stacked CNNs based on VGG-16	I
[15]	CNN based on AlexNet, GoogleNet, VGG-16 and ResNet-50	I
[13]	CNN without pretraining	I
[46]	CNN with ImageNet weights initialized and spatial pyramid	I
[10]	CNN based on EfficientNet + transformer based on Bert	I + T
[8]	BERT with additional features, pretrained on IIT-CDIP	I + T
[16]	Extension to [8] using visual information in pre-training	I + T
[7]	CNN based on VGG-16 + 3 textual features	I + T
[3]	CNN based on MobileNetV2 + FastText	I + T

Table 3.2: Overview of different methods and their used features on the RVL-CDIP dataset.

¹ I = Image, T = Text

3.3 Features

The features used for document image classification can either be image based, text based or a combination of both [22]. Some approaches use only image based features, like [15], [9] or [11], while other approaches use a combination of image and text based features, such as [10], [8], [16], [7], [3] or [52]. However, not a single SOTA paper on the RVL-CDIP dataset uses only text based features, which suggests that text based features alone are not as promising and do not perform as well as image based features.

Every paper from those mentioned in the previous paragraph, which use textual features, use the tesseract OCR system [53] to extract the text, except the current SOTA [16], which uses the OCR system from Microsoft. The free and open source tesseract

OCR System in its ≥ 4.0 version is based on the Long Short Term Memory (LSTM) network [54].

According to Audebert et al. [3], one reason why using only text based features does not perform well is because the extracted text in all state-of-the-art papers mentioned above is based on the tesseract OCR system [53], which outputs noisy text. Salt-and-pepper noise may be one reason for that, which can be seen in Chapter 5, Figure 5.1. This, however, is a conjecture and is not verified. The term salt-and-pepper noise refers to a wide variety of processes, resulting in image degradation, where a few pixels are very noisy; similar to sprinkling black and white (salt and pepper) pixels on an image [55]. Moreover, the quality of the OCR output depends on the quality of the original image, on the device used to digitize documents and on the nature of the document, e.g., text generated from recent or historical newspapers do not usually have the same quality [56].

Text based features, e.g., the presence of keywords, may be computed from the OCR output, or directly from document images [6]. For embeddings, however, only the textual output of an OCR system can be used [7]. Word embeddings, such as word2vec, are vector representations of words, which are trained on very large data sets (1+ billion words) [57]. Other text features, such as sentence level or character level embeddings, may also be used.

Image based features can further be divided into structural/layout features and visual features [6], [8]. Structural features define the relationships between objects on one page. For example, given a key in a document, e.g., "Birthday:", the corresponding value is more likely to be on the right or below, instead of on the left or above. Visual features typically contain some features, which signal the importance of some segments in a document. They can either be extracted at a global level or a local level. Global level visual features are extracted from the whole image, and can represent e.g., the density of black pixels, while local level visual features are extracted from regions in an image, and can represent e.g., the number of horizontal lines in a segmented block, or even word-level features, such as bold or underline [6]. Figure 3.1 summarizes the different features that can be used for a document image classification system.

3.4 Summary

Rule-based approaches for document image classification systems work with techniques, such as template matching [23], [32]. This technique allows to match an input document image to a set of N predefined templates. The matching process is carried out by computing the similarity of an input document image to each model or template. It is commonly used in cases where document images are formatted or at least semi-formatted, such as forms or business letters [6], which also depicts the disadvantage of this approach. Another disadvantage is that designing templates is a cumbersome task and requires one to have knowledge in the document layout of a certain class [58].

Machine learning based approaches overcome the former disadvantage of rule-based

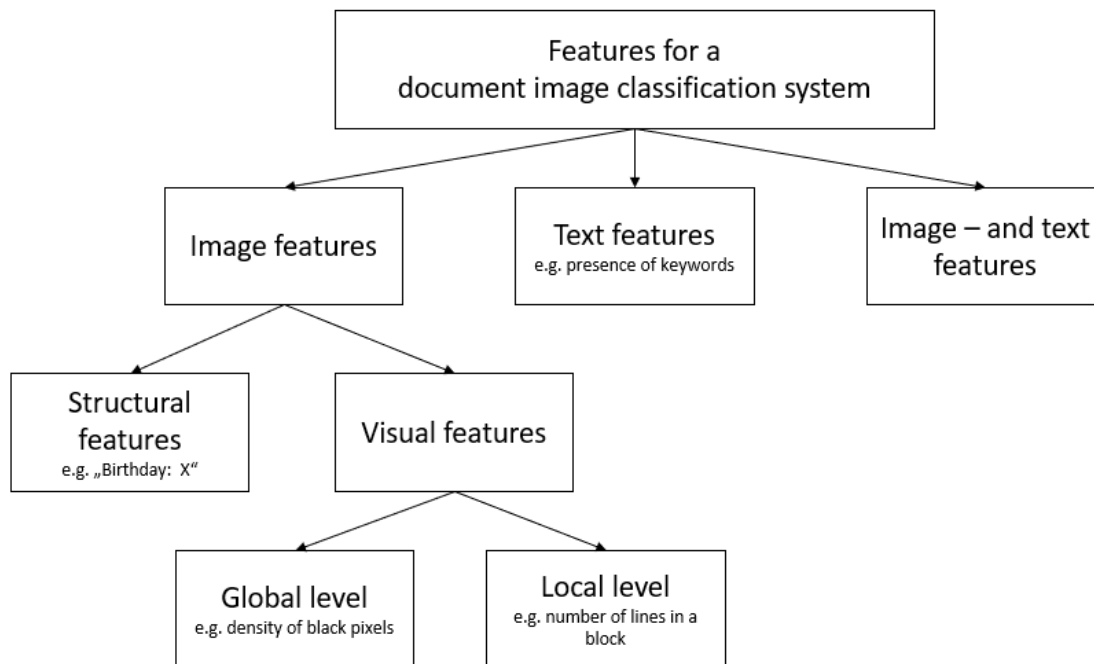


Figure 3.1: Overview of different features for a document image classification system, adapted from [6].

approaches. However, similar to designing a template, features need to be designed, or engineered, first. As for example in [27] or [29], various features, such as pixel densities in a certain window size or connected component statistics are first computed, which serve as input for a classifier. BOW models are another machine learning approach, which have shown promising results in document image classification [33], [36], [37]. A decision tree based classifier is a common choice for document image classification, such as the OC1 classifier [27], [29] or the RF [36], [37].

Deep learning based approaches overcome the requirement to perform feature engineering. Kumar et al. [37] achieve the highest accuracy on the Tobacco-3482 dataset, a subset of IIT-CDIP, using a machine learning approach. In contrast, Kang et al. [59] use a CNN, to automatically learn the necessary features to classify document images. In that paper, a significant increase of more than 22 percentage points in classification accuracy is reported by the authors. Subsequent approaches use a CNN, together with additional techniques, such as transfer learning and/or an ensemble for at least 2 models. Approaches, utilizing image and text features are also introduced. The current SOTA approach [16] uses a transformer instead of a CNN. However, in that work, the IIT-CDIP Test Collection dataset is used as a pretraining step. The metric for each of the described works is the overall accuracy. The overall accuracy is the number of correctly predicted document images divided by the total amount of document images.



Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Methodology

This chapter elaborates the approach developed to classify document images. First, the methodology for the image stream is discussed, covering the basic building blocks of a CNN, the selected architecture and the training strategy. Next, in a similar way, the methodology for the text stream is explained. Finally, the method to combine the image and text stream to form the final piece of the whole document image classification system is covered. For all three major parts, i.e., the image stream, text stream and meta classifier, a hyperparameter tuning is conducted to find the optimal hyperparameter setting.

4.1 Image stream

Compared to textual features, image features are preferred for the problem of document image classification [14]. This can also be concluded from Table 3.2., since every SOTA method based on the RVL-CDIP dataset uses image features. Based on the related work section, the current SOTA CNN architecture is used for the image stream. The image stream and text stream are two independent parts of the whole model, which are combined in a later stage. In the following, first the general CNN architecture is reviewed. Thereafter, the base classifier of the image stream is discussed, followed by the preprocessing steps, the hyperparameter tuning and the training strategy and architecture itself.

4.1.1 Convolutional Neural Networks

CNNs are a type of a feedforward neural network and are able to extract features without any human intervention. The architecture of a CNN was inspired by the visual cortex [60]. In 1998, Lecun et al. [61] were one of the first who developed a CNN and successfully trained it using the backpropagation algorithm, the central algorithm in deep learning,

which was first described in a neural network setting in 1970 [62]. It was applied to handwritten character recognition, and its architecture is depicted in Figure 4.1. However, at that time it could not exceed the SVM and as a result, it did not get much attention [63].

The breakthrough of CNNs and deep learning in general arguably happened in 2012, when Krizhevsky et al. [38] proposed a CNN architecture, AlexNet, which won the ImageNet 2012 competition [43]. It was demonstrated that a GPU can be used to accelerate the training process. Furthermore, it was shown that data augmentation can be employed, to improve the generalization ability and reduce overfitting. Since AlexNet, a variety of CNN models have been proposed, both deeper and wider [63].

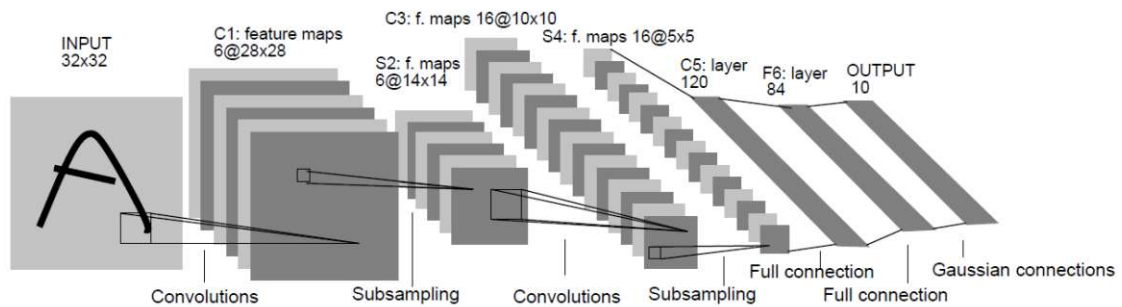


Figure 4.1: The architecture of LeNet-5 [61].

To build a CNN, four components are typically needed [63]:

Convolution Layer

A convolutional layer consists of kernels. A CNN kernel (also known as filter) represents different receptors that can respond to various features, such as lines and edges [63]. The kernel is slid over the input image and other subsequent layers in a CNN, starting from the top left and ending at the bottom right. The extracted features of a kernel represent a feature map. The depth of the feature map depends on the number of kernels used in a 1:1 ratio. That is, if 10 kernels are used, the feature map has a depth of 10. The convolution operation provides an advantage, compared to a general MLP. Due to the convolutional layer, weights are shared. Thus, a group of weights within one filter stays fixed, when sliding across the whole input image (or any other subsequent layer), reducing the amount of parameters and thus speeding up the training process.

Padding

A kernel has a certain size, with which to slide over a certain input, for example, a 3×3 kernel. When applying the sliding window technique, information on the border gets lost. With a kernel of size k and an input of size $(m \times n)$, the resulting feature map is of size $((m - k + 1) \times (n - k + 1))$, where m is the number of rows and n the number of columns. For CNNs in document image classification, it is common to have m equal to n [13]. To

prevent losing information from the borders, images are padded with zeros. Additionally, this keeps the height and width of the input size the same as the resulting feature map.

Stride

Stride is another way to control the size of the convolved input. The stride is the step size of the kernel. It determines, how many pixels between the applied kernels on an input should exist. Thus, the larger the stride, the smaller the resulting feature map in terms of height and width.

Pooling layer

The pooling layer of a CNN consists of pooling operations and is applied on a feature map, the same as a kernel on an input. It has two basic functions: reducing the size of a feature map, in terms of height and width, and reducing the amount of overfitting. This is because a pooling layer omits unnecessary information, while retaining information that is useful for a CNN. What is useful for a CNN to perform well on a classification task is determined automatically. Moreover, padding and stride can be applied on the pooling layer as well. Figure 4.2 summarizes the described components. First, an input image is padded, followed by deciding for a stride. Then, it is convolved with a kernel, and finally a pooling operation (in this case: max pooling) is applied. Max pooling takes the largest activation in a specified window.

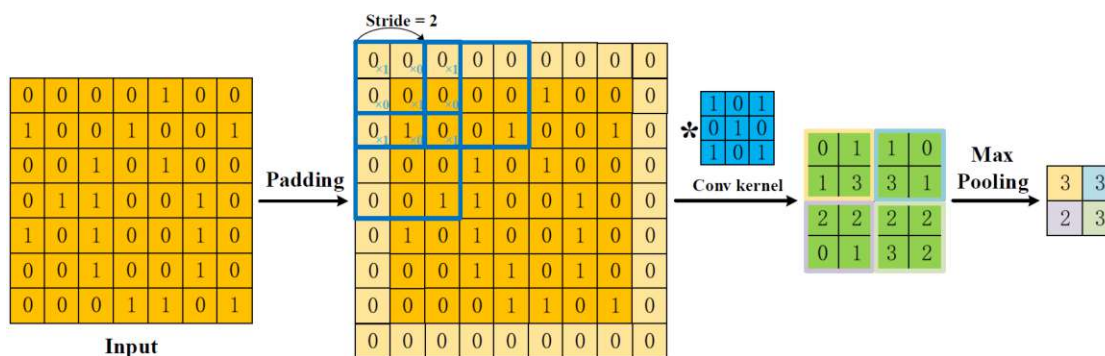


Figure 4.2: The four components described and their general procedure, for a 2D input. [63].

4.1.2 Base classifier

The architecture of the image stream is based on the EfficientNet [49], which is a family of CNN models. EfficientNets have shown to achieve SOTA results [49] on the ImageNet dataset. First introduced in 2019 by Tan and Le [49], there have been more than a dozen different variants proposed since then for the EfficientNet, especially in the training strategy of EfficientNets. As of April 2022, an EfficientNet trained with Meta Pseudo Labels [64] is the best CNN, in terms of the test set accuracy on the ImageNet dataset.

The main contribution of the EfficientNet model family is the compound scaling method, which uses a compound coefficient ϕ to scale the depth (d), width (w) and resolution (r) in the following way: $d = \alpha^\phi$, $w = \beta^\phi$ and $r = \gamma^\phi$, subject to the constraints $\alpha \cdot \beta \cdot \gamma \approx 2$, and $\alpha \geq 1$, $\beta \geq 1$, $\gamma \geq 1$. The constants α , β and γ are found in a small grid search first, which determine the baseline model EfficientNet-B0. These constants are then taken and scaled up by ϕ , to obtain the EfficientNet models B1 to B7. The baseline model, EfficientNet-B0, has about 5 million parameters, while the largest model, EfficientNet-B7, has about 67 million parameters, which is still notably smaller than the 138 million or 144 million parameters in the VGG-16 or VGG-19, respectively.

4.1.3 Preprocessing steps

In this work, the image stream consists of five EfficientNets, each focusing on a different part of the input. The preprocessing steps partly follow the work of [11]. However, one simple and key difference in the preprocessing steps is the size of the image. As is found by [13], using a larger input size significantly impacts the performance and leads to the biggest performance gain, compared to other preprocessing techniques, such as mirroring the image, rotating the image, applying shear transformation, etc. Unlike other studies in document image classification, such as [11], [9], [15], [7], [18], [14], [46], [65] or [21], which use an input image size of either 227×227 or 224×224 , Tensmeyer and Martinez [13] have found that an image size of 384×384 leads to the best results on the RVL-CDIP dataset. The accuracy with different image sizes constantly increases up to 384×384 , the second largest size tested. 512×512 , the largest size tested, yields worse results.

The preprocessing steps are as follows. First, all images are resized to 936×720 . Then, 5 regions are defined for an image; holistic, header, footer, left body and right body. The holistic region is simply the whole image itself. The header is defined as the first 307 pixel rows. Similarly, the footer is defined as the 307 pixel rows. The left body is defined as the 480 central pixel rows and the first 360 pixel columns and similarly, the right body is defined as the 480 central pixel rows and the last 360 pixel columns. Therefore, a slight intersection exists between the left and right body areas with the header and footer. Finally, each image is resized to 384×384 . These resized image parts are scaled by a factor of 1.2 compared to the resized image parts in [11]. This results in less resizing when performing the final resize to 384×384 , which in turn retains more image quality, since resizing an image is degrading the image quality [66].

For resizing an image, different methods and implementations are available, such as nearest neighbor, bilinear, bicubic, bicubic b-spline or lanczos. The nearest neighbor is the simplest of all these mentioned interpolation algorithms and the fastest in terms of processing time. This method, however, would not be an appropriate option for document image classification, especially for this dataset, due to the low interclass variability, as explained in Section 1.1. To distinguish between documents with a low interclass variability, the details should be preserved as much as possible, when resizing. For that reason, the lanczos interpolation is chosen, since it has the best properties in terms of detail preservation [67]. The other interpolation methods, except the nearest

neighbor, would be a legitimate choice, but are not considered for this work, except for the bilinear interpolation.

The focus on specific regions of a document follows from the fact that certain categories show a low interclass variability. Figure 1.1 provides an example for that. It shows two documents from different classes, memo and letter. They typically differ on the header of a document. While memos often have a full address section, letters typically have a "To:", "From:", often also either "Subject:" or "RE:". Even though a holistic CNN will probably learn these differences, having a CNN to classify documents using only this region will much more likely learn those differences [11]. Similarly to the header region, different CNNs are implemented to each region described in the previous paragraph.

Due to the size of the dataset, data augmentation is not performed. On the contrary, data augmentation techniques, such as resizing and cropping, would likely decrease the performance. This is because the most discriminative parts of document images may reside in the outer part of an image, for instance the header or the footer of a document, as can be seen in examples in Figure 1.3. Moreover, this justifies the reason to focus on different parts of the images with different CNNs.

Normalization of input images is widely used as a data preprocessing step [68], such as centering or scaling the data. However, since the EfficientNet expects the pixel values in the range $[0 - 255]$, no normalization is performed. Instead, the normalization is performed internally in the range $[0 - 1]$.

The last preprocessing step is to transform the input into images with three channels. Since the data is present in a single channel with grayscale values, it is copied two times and stacked depth wise, along the third axis.

4.1.4 Hyperparameter tuning

Three restrictions are set for the hyperparameter tuning in order to reduce the computational complexity. First, the tuning is carried out only on the classifier that is added on top of the base classifier, i.e., on the classification head. Second, a smaller subset of the training data, as well as the validation data is taken. The training data consists of 10,016 files, and the validation data of 5,024 files, following the class distribution of the whole training data set. Third, the input size of the document images is reduced to 224×224 . This, however, is increased to 384×384 in the second part of the tuning process. After first experiments, the hyperparameters that are considered in this work are the following:

- Batch Normalization (BN): True, False
- Base model: EfficientNet-B0, EfficientNet-B4, EfficientNet-B7
- Dropout: 0.1, 0.3
- Fully connected (FC) layer: True, False
- Number of neurons in the FC layer: 50

- Optimizer: Adaptive moment estimation (Adam), Stochastic gradient descent (SGD)
- Learning rate: 0.0001, 0.001

The classification head consists of the following layers:

GlobalAveragePooling2D → BN → *Dropout* → *FC layer (50 neurons)* → *Output layer*

The *GlobalAveragePooling2D* layer receives an input of size (*height* × *width* × *channels*) and computes the average of the values across each channel, such that the output of this layer would be a one dimensional vector of size (*channels*). This layer greatly reduces the number of parameters and thus the computational complexity. For instance, if instead of the *GlobalAveragePooling2D* the input is flattened after the BN layer as depicted above, the number of parameters increases from around 6.5 million to almost 16 million, assuming the *EfficientNet-B1* model as the base classifier. The BN layer normalizes each mini-batch by subtracting the mini-batch mean and dividing by the mini-batch variance. After the normalization step, the input is scaled by a factor γ and shifted by an offset β [69]. Both γ and β are learnable parameters. The dropout layer addresses the problem of overfitting by randomly dropping neurons, as well as their connections, from the neural network during training [70].

The hyperparameter values "True" and "False" mean that either that hyperparameter is considered or not. In particular, it is considered, whether adding an additional FC layer with 50 neurons increases the validation accuracy or not. The same set of hyperparameters is considered on all five image models, and the best hyperparameters on average is taken for all. As an example, if for four models, the best dropout rate is 0.3, but for the fifth it is 0.1, then 0.3 is taken for all models. This is because the results between 0.1 and 0.3 on the fifth model are too close, so that it would be more stable and beneficial to have a general set of hyperparameters for all models. The same applies to the other sets of hyperparameters. The tuning is done with an early stopping setting, and a patience of 10 on the validation loss, i.e., if after 10 epochs the validation loss does not decrease, the training is stopped. The loss in this tuning, and in fact in all models in this work, is the categorical cross entropy loss, which is defined as

$$Loss = - \sum_{i=1}^C y_i \log \hat{y}_i, \quad (4.1)$$

$$\hat{y}_i = \text{softmax}(s), \quad (4.2)$$

where s denotes the output scores from the last layer, y_i the label for class i , \hat{y}_i the prediction for class i and C the number of classes.

In addition to finding the best base classifier, a different set of weights for initialization is considered. The model architecture for those different weights¹ is exactly the same,

¹<https://github.com/tensorflow/tpu/tree/master/models/official/efficientnet>

which makes it possible to use them. In particular, the noisy student version [71] of EfficientNet is tested. The noisy student training requires a set of labeled and a set of unlabeled images. First, a teacher model on the labeled data set is trained. The teacher model then generates pseudo-labels on the unlabeled dataset. A student model is subsequently trained on the labeled and pseudo-labeled data. The student then becomes the teacher and makes again predictions on an unlabeled dataset, which are again taken as pseudo-labels, etc. This procedure achieves a test set accuracy on the ImageNet dataset of 86.9% with an EfficientNet-B7 model, an improvement of 2.5 percentage points, compared to the original EfficientNet-B7 model.

Further tuning

During this work, a few other hyperparameters and considerations are taken into account. These are BN (whether or not to update its weights), the interpolation method for resizing document images (bilinear vs lanczos) and the evaluation metric for training neural networks. That is, whether it is better to use the model with the highest validation accuracy, or the model with the lowest validation loss, in terms of accuracy.

However, to reduce the computational complexity, the impact of BN and the interpolation method have only been tested on the holistic model. Regarding the evaluation metric, the region based image models with the highest validation accuracy and the lowest validation loss are each tested, to find out, how it impacts the test set accuracy. The results lead to the decision to choose the models with the highest validation accuracy.

4.1.5 Training strategy and architecture

The training strategy and architecture on the full dataset is inspired by [9]. The main benefit of the following training strategy is to reduce the computational complexity. This is achieved by a three level transfer learning and can be seen in Figure 4.3.

The first level of transfer learning is initializing the weights of the holistic model from the corresponding EfficientNet-B1 model, trained on the ImageNet dataset. The EfficientNet-B1 is chosen because of the highest validation accuracy, as shown in Section 5.1. To train the holistic model, only the classifier added on top of the EfficientNet-B1 model is trained first, and all the other weights of the model are frozen, such that they are not updated during backpropagation. This model's weights are then used for initialization of the same model, but with all layers unfrozen, including the batch normalization layers. Now, all weights can be updated to further increase the prediction accuracy.

The L1 holistic model is trained with the optimal hyperparameter setting as described in Section 5.1. The weights of this model are then taken to initialize the remaining four models, i.e., the models for the header, footer, left body and right body region. Similar to the holistic model, these four models are trained with early stopping on the validation loss and a patience of 10 and the optimal hyperparameter setting as described in Section 5.1. The Rectified Linear Unit (ReLU) [72] is used as the activation function. In each epoch, the model is evaluated on the validation set and the model is saved after each

epoch for further analysis. Specifically, it is analyzed whether using the model with the lowest validation loss or the model with the highest validation accuracy has a noticeable impact on the test accuracy.

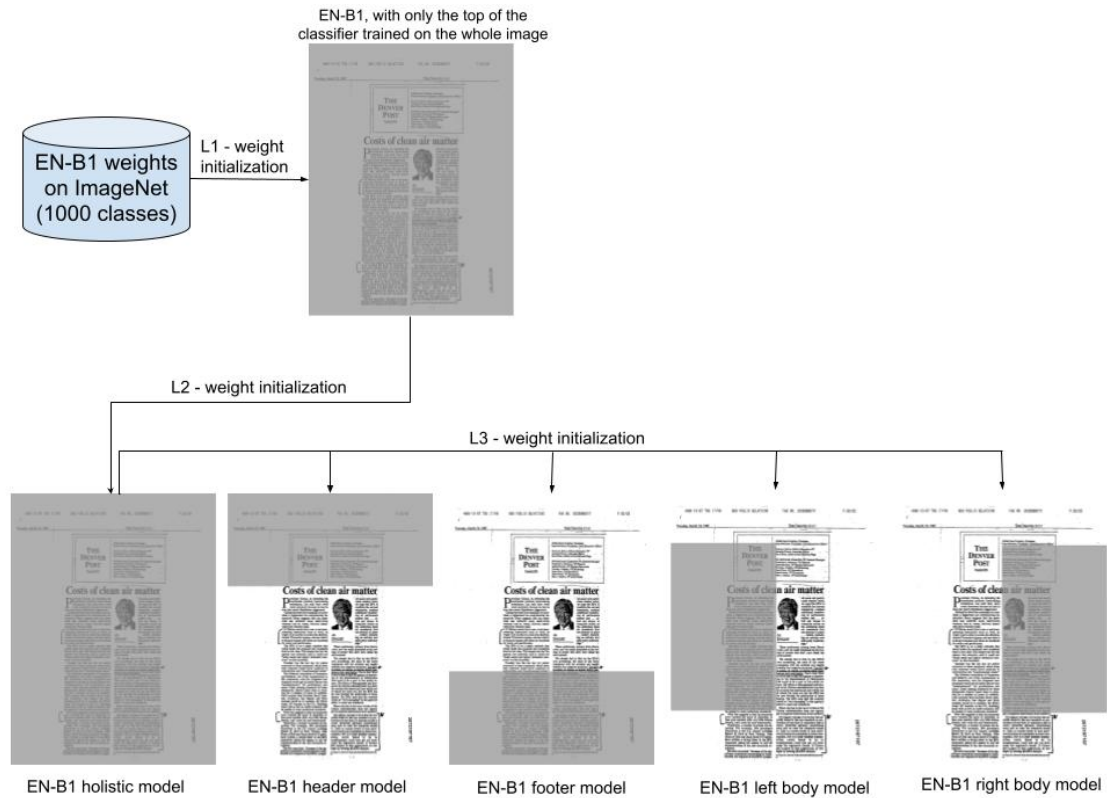


Figure 4.3: The training strategy for the image stream ¹.

¹ EN = EfficientNet

4.2 Text stream

Even though image features are preferred compared to textual features when it comes to document image classification [14], the recent development in this field suggests that textual features are necessary to achieve SOTA results. In the following sections, the transformer architecture is first reviewed. Then, the base classifier of the text stream is discussed, as well as the text extraction, preprocessing steps and the hyperparameter tuning.

4.2.1 Transformer

Transformers have achieved a great success in many areas, such as Natural Language Processing (NLP) and computer vision [73]. The transformer architecture originated

from [51] and is depicted in Figure 4.4.

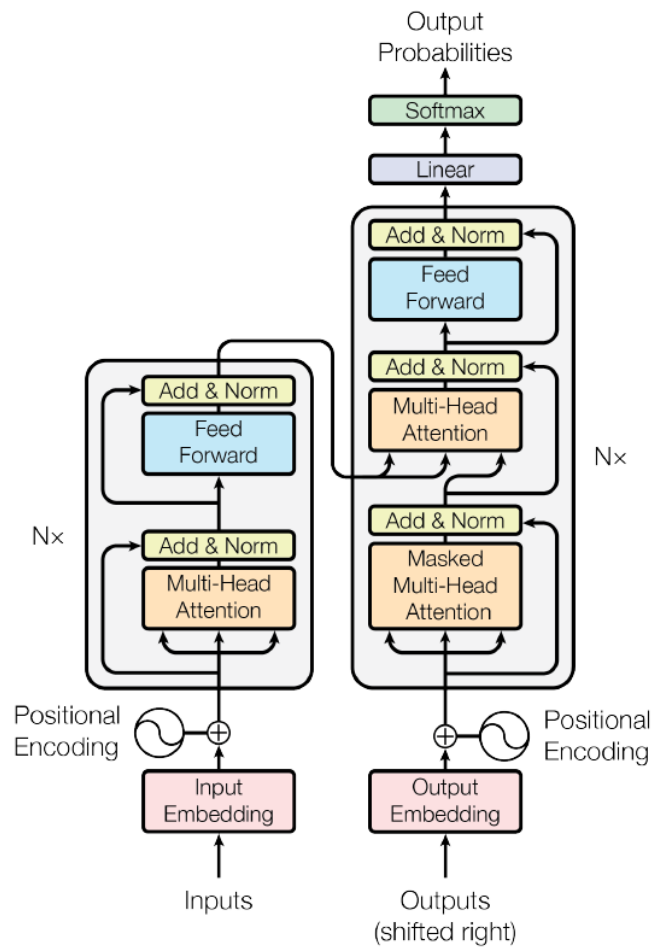


Figure 4.4: The vanilla transformer architecture [51].

The vanilla transformer architecture [51] is a language model, consisting of N encoder and N decoder blocks, where $N = 6$ in the original implementation. Recurrent neural networks (RNN) based models, such as the LSTM [54], were once SOTA approaches. Their arguably biggest disadvantages are computational complexity when dealing with long sequences, and the lack of parallelization due to its sequential nature in the architecture [73]. In the following, the most important modules of the vanilla transformer architecture are described.

Attention module

Unlike the RNNs, the transformer architecture, however, is able to perform computations in parallel. It relies mainly on the attention mechanism, which occurs in the attention module (see the orange boxes in Figure 4.4). The attention is defined as

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (4.3)$$

where $Q \in \mathbb{R}^{N \times d_k}$ are the queries, $K \in \mathbb{R}^{N \times d_k}$ are the keys and $V \in \mathbb{R}^{N \times d_v}$ are the values, with N denoting the sequence length, and d_k and d_v denoting the dimensionality of the queries, keys and values. QK^T is divided by $\sqrt{d_k}$ to counteract the vanishing gradient problem due to large values of d_k and thus a large dot product QK^T [51].

The Q , K and V matrices are the result of applying weight matrices $W^Q \in \mathbb{R}^{d_{model} \times d_k}$, $W^K \in \mathbb{R}^{d_{model} \times d_k}$ and $W^V \in \mathbb{R}^{d_{model} \times d_v}$ to the input data matrix $X \in \mathbb{R}^{N \times d_{model}}$, where d_{model} denotes the embedding size. The input data matrix X refers to any of the three inputs to the orange boxes, as seen in Figure 4.4. In the Encoder, X is the same in each multiplication with the weight matrices, i.e., $Q = K = V$. This is referred to as self-attention. In the Decoder, the upper orange Box in Figure 4.4, X refers to the values and keys from the Encoder, which are the same, and queries from the Decoder, which are different to the values and keys, i.e., $K = V$ but $K, V \neq Q$. This is referred to as cross-attention. In the lower orange Box in the Decoder in Figure 4.4, X also refers to the same for all three inputs, but a mask is applied to the self-attention, which masks the input that appears later in the sequence, i.e., it sets it to $-\infty$.

Multi-head attention

Typically, not just one such computation, as described in (4.3), is performed per attention module, but rather h such computations. In this case it is referred to as multi-head attention, with h heads. The queries, keys and values get linearly projected h times with h different sets of learned projections. That means, h heads are employed. On each such projection, the attention is computed according to (4.3). The outputs of the attention mechanism are then concatenated and projected once again with a weight matrix $W^O \in \mathbb{R}^{d_{model} \times d_k}$ back to the original dimension:

$$MultiHeadAttention(Q, K, V) = Concat(head_1, \dots, head_H)W^O, \quad (4.4)$$

$$head_i = Attention(QW_i^Q, KW_i^K, VW_i^V) \quad (4.5)$$

where $W_i^Q \in \mathbb{R}^{d_{model} \times \frac{d_k}{h}}$, $W_i^K \in \mathbb{R}^{d_{model} \times \frac{d_k}{h}}$, $W_i^V \in \mathbb{R}^{d_{model} \times \frac{d_v}{h}}$ and $W^O \in \mathbb{R}^{d_k \times d_{model}}$.

Multi-head attention allows the model to learn richer representations, because different parts of the embeddings can learn different meanings of the words, and each head can focus on particular aspects of the language [51]. The h different matrices for Q , K and V are only logically separated. Thus, a single matrix multiplication can compute the attention score for all h heads.

In addition to the multi-head attention, the encoder and decoder contain a second sublayer, the FC feed forward neural network. The exact same feed forward neural

network, with a ReLU activation function, is independently applied to each position. In other words, the weights of this network are shared, because it is applied to each word embedding. After each sublayer, a residual connection and layer normalization is applied. That is, the output of each sublayer is first added with the output of the previous sublayer, and then normalized across all channels and spatial dimensions [74].

The original transformer, as depicted in Figure 4.4, uses an encoder and a decoder, making it a sequence-to-sequence model. Other types of transformers include autoencoding and autoregression. Autoencoding models, such as BERT, use the encoder part of the original transformer. These models mask some portion of the input, which the model tries to predict. Autoregressive models, such as the Generative Pre-trained Transformer (GPT) model family (e.g., [75]), use the decoder part of the original transformer. These models try to predict the next token based on all the previous tokens, while masking the future tokens.

4.2.2 Base classifier

A distilled version of BERT is used in this work, called DistilBERT [76], whose architecture is based on BERT [50]. BERT, at the time it was published, has achieved eleven SOTA results in NLP tasks, and still is SOTA in tasks, such as common sense reasoning and question answering. However, it requires more computational resources for training compared to DistilBERT. The pre-training of the BERT model takes 4 days on 16 TPU chips [50], while DistilBERT was trained on 8 16GB V100 GPUs for approximately 90 hours [76].

DistilBERT is 40% smaller in size, compared to the BERT model, while retaining 97% of its language understanding capabilities. This is achieved by knowledge distillation [77], which tries to compress a large model or an ensemble of models into a smaller model, such that the smaller model is not significantly worse in terms of a certain metric, e.g., accuracy. To achieve distillation, the DistilBERT model is trained on three different objectives:

- Distillation loss: DistilBERT learns to output the same class probabilities as BERT.
- Masked language modeling loss: To learn a bidirectional representation of the input, a percentage of the input tokens, in this case 15, are masked randomly. These masked tokens are then predicted.
- Cosine embedding loss: This loss makes the DistilBERT model generate hidden state vectors as close as possible to the ones from BERT.

However, the next sentence prediction loss, as it is used in BERT, is not implemented in DistilBERT. Apart from that, DistilBERT has the same general architecture as BERT, but with half as many layers (6). It has 66 million parameters, compared to 110 million

in the case of the base model of BERT, and 340 million parameters in the case of the large variant of BERT.

Both models are pretrained on English Wikipedia² (2,500M words) and on BooksCorpus (800M words) [78]. For this work, the uncased version of DistilBERT³ is used, i.e., it sets all text to lowercase.

4.2.3 Text extraction

When performing a NLP task, some text to train a model is generally needed⁴. This is not a difficult task. For example, GPT-3 was trained on more than 570GB of data, scraped from the internet.

Since this work is based on document images, the text has to be extracted first, in order to use it for classification. This is done by applying an OCR system on those document images. Many commercial OCR systems exist, such as those from Google⁵, Amazon⁶ or Microsoft⁷. In this work, the open-source tesseract OCR System⁸ is used, like in other papers in the field of document image classification, as mentioned in Section 2.3.

First developed at HP between 1984 and 1994, the tesseract OCR system supports over 100 languages. In this work, version 4.1.1 of tesseract is used.

4.2.4 Preprocessing steps

After performing OCR on document images, the second step is to preprocess the extracted text before feeding it into a neural network. This is even more important, when the text is extracted from document images, instead of, for instance, scraping the text from the web. Noise in the extracted text from the web will generally be much higher due to reasons mentioned in Section 2.3. The goal of this step is not only to remove noise, but also to remove non-discriminative characters, such as punctuation. Non-discriminative elements depend on the task [80].

Vijayarani et al. [81] describe key steps of text preprocessing, which are tokenization, stop words removal and stemming. Tokenization refers to splitting the text into separate tokens, as words or numbers. The idea of stop word removal, is to remove non-discriminative characters, i.e., characters which appear frequently over different sets of documents. There are different collections of stop words available⁹. The best one depends on the task [80]. Other text preprocessing techniques include lower casing,

²Only text passages are extracted; lists, tables and headers are ignored

³<https://huggingface.co/distilbert-base-uncased>

⁴with exceptions, such as [79], moving closer to building a 'textless NLP' application

⁵<https://cloud.google.com/vision/>

⁶<https://aws.amazon.com/textract/>

⁷<https://docs.microsoft.com/en-us/azure/cognitive-services/computer-vision/overview-ocr>

⁸<https://tesseract-ocr.github.io/tessdoc/Home.html>

⁹for example, <https://www.ranks.nl/stopwords> has different stop word lists for english

stemming and lemmatization [82]. Lower casing puts all words in lower case. Stemming and lemmatization are both techniques, which produce different normalized forms of a word. Lemmatization normalizes a word to its basic form, while stemming produces an approximated basic form. For example, lemmatization transforms the word "was" into "be", while stemming would return the word "fast" for the words "fast", "faster" and "fastest".

Uysal and Gunal [80] investigate the impact of different preprocessing steps on the text classification. They report that removing stop words can have a negative effect on the accuracy and suggest that stop words have a higher importance in contrary to what most of the text classification studies suggest.

Following that result, and the analysis of the extracted text of this dataset, the decision not to remove stop words is made. Other preprocessing steps, that are undertaken in this work, are the following:

- Removing "NaN". These are larger regions of the document, where tesseract outputs a bounding box, but not the corresponding text.
- Remove everything that is not a letter or a digit, since tesseract outputs a lot of noise, i.e., characters like &," },!,? ,*, etc., which are not supposed to be output.
- Perform lower casing.
- Ignore text, which consists only of whitespace. This is another noisy output.
- Ignore text, which is zero or one characters long, but keep single digit numbers, since they have a much better discriminative power than one character long strings.

There are some pages where no text can be extracted by tesseract. In this case, or in cases where the whole extracted text of a document image is removed due to the preprocessing steps, the extracted text is set to "", i.e., a string of length zero.

4.2.5 Hyperparameter tuning

Due to the relatively large size of the transformer model, the tuning is reduced to a minimum in order to prevent a large computational complexity. Like in the image stream, the same subset is taken for the training and validation set, since tuning on the whole training set is not feasible. The tuning is done in two steps. First, the maximum sequence length is taken as a hyperparameter. Based on initial experiments, a tuning over the following set of hyperparameters is conducted, with the bold values being set as fixed in the first step of the tuning:

- Dropout: **0.3**, 0.5
- Number of neurons in the FC layer: **256**,512

- Learning rate: 0.00003, **0.00001**

The maximum lengths 128, 256 and 512 are considered. The maximum length is the number of tokens one document image is represented with. If, for example, the maximum sequence length is set to 256, then document images with more than 256 tokens are truncated, while documents with less than 256 tokens are padded. Once the optimal maximum length is found, the hyperparameters mentioned above (dropout, number of neurons in the FC layer and learning rate) are tuned.

4.2.6 Training strategy and architecture

Following the results from the image stream, the training strategy in the text stream takes a similar approach. Here, the DistilBERT model is taken as the base model. Only the classification head added on top is trained first, with the features extracted from the base model.

DistilBERT, as well as the original BERT model, has two special tokens: [CLS], a classification token, and [SEP], a separator token. The [CLS] token is used for classification tasks and is added in front of every sequence, i.e., in front of each extracted text from a document image in this work. Specifically, the last hidden state representation of the [CLS] token is used, which in this case is a 768 dimensional vector. This hidden state representation is then used as an input to the classification head.

Like in the image stream, first the classification head is trained. To train the classification head, a hyperparameter tuning is conducted with an early stopping strategy and a patience of 10. Following the tuning procedure, all layers of the DistilBERT model are unfrozen in order to change the weights of these layers. The final model is trained with early stopping and a patience of 10, using the hyperparameters found in the tuning step and the Adam optimizer. ReLU is used as the activation function for all models. As in the image stream, the model is evaluated in each epoch on the validation set and a model is saved after each epoch for further analysis.

4.3 Stacked generalization

The last part of the document image classification system is to train a meta-classifier, which outputs the final predictions. This technique is called stacked generalization and was first introduced in 1992 [83]. It is adopted in document image classification models, such as in [9], [3], [10], [7], [52], and works by combining the (intermediary) output of one or more classifiers and feeding that as an input to a meta-classifier. To reduce overfitting, the meta-classifier is trained on the validation set. Thus, the input for the meta-classifier consists of the (intermediary) outputs of base classifiers from the validation set. The goal of stacked generalization is to provide a lower generalization error than the base models, which in this work are the region based CNN models and the transformer. It is demonstrated by Das et al. [9], that stacked generalization achieves a better test

set accuracy than a single model. However, Harley et al. [11] report, that the stacked generalization achieves a worse test set accuracy than a single model. Both results are based on the RVL-CDIP dataset. The meta-classifier is the last module of the document image classification system, and the full architecture is shown in Figure 4.5.

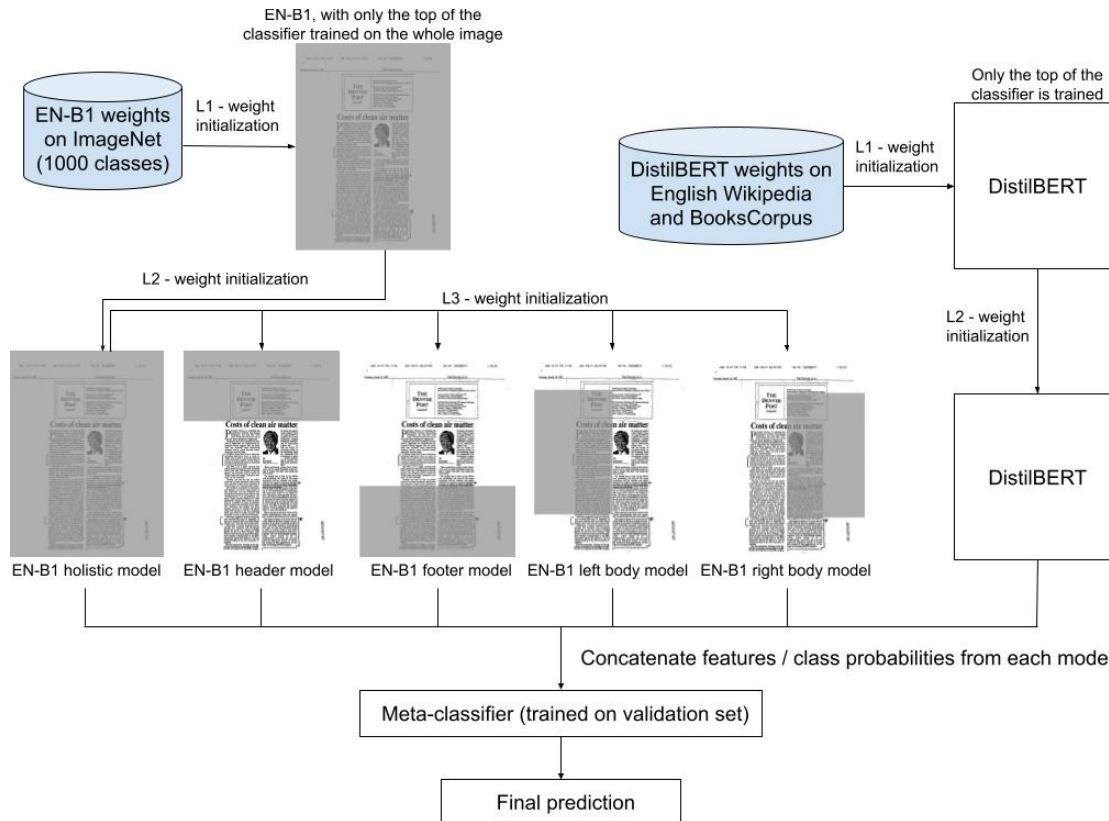


Figure 4.5: Proposed architecture for document image classification based on SOTA architectures, with an image - and text stream and utilizing different levels of transfer learning.

The input for the meta-classifier could either be the class probabilities (i.e. the softmax output) or some extracted features on the validation set from the base classifiers. The final class predictions could also be taken (i.e. the predicted class with the highest probability or $\text{argmax}(\text{softmax output})$). However, this results in a worse overall accuracy, as shown by Ting and Witten [84]. In this work, the meta-classifier combines visual and textual features, and produces the final output of the document image classification system. This is done by trying both the class probabilities and the extracted features as an input to the meta-classifier.

Das et al. [9] explore different machine learning models for feature combination on the RVL-CDIP dataset, such as linear regression, ridge regression, k-nearest neighbours, SVM, bootstrap aggregating with SVM, extreme learning machine and a multilayer

neural network. They report that a neural network achieves the highest validation accuracy, followed by bootstrap aggregating with SVM, and SVM. Thus, in this work, only a neural network as the architecture for the meta-classifier is considered.

The dataset to train the meta-classifier is based on the validation set, containing 40,000 document images, with even class distribution. Instead of doing a train and validation split, a k-fold cross-validation is performed, where $k = 5$. Based on initial experimentation, the set of hyperparameters to tune on are the following:

- Dropout: 0.1, 0.3, 0.5
- Number of neurons in the FC layer: 128, 256, 512
- Number of FC layers: 0, 1, 2, 3
- Learning rate: 0.0001, 0.0005, 0.00005

Adam is chosen as the optimizer. To get more stable predictions, the same hyperparameter set is evaluated on all 5 folds and the average validation accuracy is reported. Moreover, not only different hyperparameters are tested, but also different combinations of features, and thus different region based models are taken into account. Since this is a modular model, this task can be done by leaving out certain vectors. This ablation study investigates the influence of the text features, as well as the region based CNNs. The following model combinations are tested

- Full model
- Holistic + text
- Holistic + header
- Holistic + header + text
- Holistic + text
- Holistic + header
- Holistic + header + text
- Holistic + header + footer
- Holistic + header + footer + text
- Holistic + header + left body
- Holistic + header + left body + text
- Holistic + header + left body + right body
- Holistic + header + left body + right body + text
- Holistic + header + footer + left body + right body

where text refers to the textual features extracted from DistilBERT. Due to having a much smaller dataset to train on and a much smaller model in terms of parameters, this extensive tuning is feasible. The following two sections describe the two distinct approaches for generating the input data for the meta-classifier in more detail.

4.3.1 Features from the last convolutional and hidden layer

This approach, which is also used in [11], uses features extracted from the last convolutional layer. In this case, the last convolutional layer from each region based CNN model has the shape $7 \times 7 \times 1,280$. In order to transform that into a vector, each of these 1,280 channels is averaged, generating a 1,280 dimensional vector. This is done for all five CNN models, and each vector is then concatenated, creating a $5 \times 1,280 = 6,400$ dimensional vector. Additionally, a feature vector from the transformer model has to be generated. The last hidden layer of the text model has a dimension of 256×768 , where 256 is the sequence length, which is found in the hyperparameter tuning. As mentioned in Section 4.2.6, the last hidden state representation of the [CLS] token is used, which is a 768 dimensional vector. This vector is concatenated with the features from the CNN models and thus the final input vector with dimension 7,168 is generated.

4.3.2 Class probabilities

Using class probabilities as an input for the meta-classifier is a simpler approach, since it does not require any preprocessing of the data. Because this proposed document image classification system contains 6 different classifiers and this dataset has 16 classes, the input vector has a maximum dimension of $6 \times 16 = 96$. The 16 dimensional vectors from each model are concatenated and used as an input for the meta-classifier.

4.4 Tobacco3482

The document image classification model is also trained, or rather finetuned, and evaluated on the Tobacco3482 dataset. To make results comparable with other works, such as [11], [59], [21], [37] or [10], the dataset is split as follows. From 3,482 images, 100 images per class are randomly selected. This constitutes the training set, and the remaining 3,482 images are the test set. This process is repeated 10 times, such that there are 10 different training and test sets, from which the median test set accuracy is reported. From the 1,000 training images, 200 are used for the validation set.

The training approach first uses the pretrained models on the RVL-CDIP dataset and then finetunes on the Tobacco3482 dataset. No hyperparameter tuning is performed. Instead, a new classification head is added on top of the pretrained models since this dataset has 10 classes instead of 16. Moreover, only the added classification head on top is trained.

The classification head for the image stream consists of the following layers:

GlobalAveragePooling2D \rightarrow *Dropout* (0.3) \rightarrow *FC layer* (50 neurons) \rightarrow *Output layer*

In contrast, the classification head for the text stream consists of these layers:

FC layer (512 neurons) \rightarrow *Dropout* (0.5) \rightarrow *Output layer*

In both streams, the output layer refers to a FC layer with 10 neurons, one for each category, and a softmax activation function.

Additionally, a meta-classifier is trained, to combine the outputs of the image models and the text model. However, no tuning is performed and only the class probabilities are taken as input for the meta-classifier. The meta-classifier consists of 2 hidden layers, each with 256 neurons, and a dropout with a 0.30 dropout rate is applied after each hidden layer. In addition, two combinations for the whole document image classification system are tested. The first one with all image models and the text model and the second one with just the image models.

According to Wolpert [83], when performing stacked generalization, the meta-classifier should be trained on another dataset, e.g., the validation set. However, in the case of Tobacco3482, the validation data consists of 200 images. Setting aside 10% from that validation set to evaluate the meta-classifier during training means, that training is done on 180 images, while it is evaluated on just 20 images. This number is significantly smaller, compared to the 36,000 images, on which the meta-classifier for the RVL-CDIP dataset is trained on. Thus, the training set of Tobacco3482 is chosen to train the meta-classifier. However, to test whether overfitting occurs and what the impact of using the class probabilities of the training set as an input for the meta-classifier is, the validation data is also used to train the meta-classifier.

All models are trained with the Adam optimizer, and ReLU is used as the activation function. Based on initial experimentation, the learning rate for the image models is set to 0.001, for the text model to 0.0005, and for the meta-classifier to 0.001. All models are trained with early stopping on the validation loss and a patience of 3. The model of the last epoch is taken each time to predict on the training, validation and test set. Only the base classifiers have performed predictions on the training set in order to generate the data set for the meta-classifier to train on.

4.5 Summary

The methodological approach to answer the main research question, as described in Section 1.1, consists of building a two stream neural network with a meta-classifier, which is another neural network. All models in this work are developed and evaluated on the RVL-CDIP dataset, as well as finetuned and evaluated on the Tobacco3482 dataset.

The first stream is based on image features, whereby multiple CNNs are used. These CNNs are based on the EfficientNet [49] architecture, a SOTA CNN architecture. One CNN takes the whole document image as input, while the remaining CNNs focus on the header, footer, left body and right body region. First, a hyperparameter tuning

is conducted, to find the optimal settings for the CNNs. To reduce the computational complexity, the hyperparameter tuning is split into two parts. To further reduce the computational complexity, the training is done via a three level transfer learning. This way, the CNN models focusing on the header, footer, left body and right body region can be trained in parallel. When training the holistic model, first its classification head is trained, then the whole model.

The second stream is based on textual features, whereby DistilBERT, a transformer, is used. DistilBERT is based on BERT, but 40% smaller in size, while retaining 97% of its language understanding capabilities. Similarly to the image stream approach, a two stage hyperparameter tuning is first conducted to find the optimal hyperparameters. These are then used to train the DistilBERT model. Moreover, the training is split as well, such that the classification head is trained first before the whole model is trained.

The output of both streams is combined, which is used as an input for a meta-classifier, which outputs the final prediction. The input for the meta-classifier consists of either the softmax outputs of both streams, or the features from the last convolutional layer (CNN) and hidden layer (transformer). Additionally, to test the impact of the different CNNs as well as the transformer, different combinations of submodels are tested. Moreover, further experiments are done, such as measuring the impact on the test set accuracy of different interpolation methods.

These trained models are then finetuned and evaluated on the Tobacco3482 dataset. Apart from initial experimentation, no hyperparameter tuning is conducted. It is evaluated, whether an image only system achieves a higher test set accuracy compared to a multimodal system, i.e., a system using visual and textual features. Additionally, it is evaluated, whether using the softmax output of the base models on the training set or the validation set to train the meta-classifier, leads to a higher test set accuracy.



Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Results

This chapter consists of the final results on both datasets, as well as the meta-classifier results on the RVL-CDIP dataset. Furthermore, results are compared to past and present SOTA document image classification methods. The runtimes of modules are also presented, along with their number of parameters. The results are presented and compared, using the overall accuracy as metric. The system and libraries used to get the results are described in the Appendix, Section D.

5.1 Hyperparameter tuning

This section contains all hyperparameter tuning results, which are performed with a subset of the RVL-CDIP dataset. The results are shown first for the image stream, then for the text stream and finally for the meta-classifier. For the image stream and text stream, the hyperparameter tuning consist of two parts each, while for the meta-classifier, different combinations of submodels are considered in the hyperparameter tuning approach, to evaluate their impact on the overall accuracy.

5.1.1 Image stream

Table 5.1 presents the hyperparameter tuning results on the image stream, i.e., on all 5 image models, using an image size of 224×224 . Only the hyperparameter settings which lead to the best validation accuracy for each image model are shown. Note that the Adam optimizer is better than the SGD optimizer in each of the best hyperparameter settings.

From Table 5.1 it can be observed, that the best set of hyperparameters, on average, is the following: Dropout = 0.3, Learning rate = 0.0001, Batch normalization = True and FC Layer = True (with 50 neurons), with Adam being the best optimizer for every image model. This set of hyperparameters is used in the next step, to find the best

Optimal hyperparameters						
Image model	Base model	Dropout	Learning rate	BN	FC layer ¹	Validation accuracy
Holistic	EfficientNet-B0	0.3	0.0001	True	True	67.70
Header	EfficientNet-B0	0.3	0.0001	True	True	61.15
Footer	EfficientNet-B0	0.3	0.001	False	False	58.90
Left Body	EfficientNet-B7	0.3	0.0001	True	True	62.64
Right Body	EfficientNet-B0	0.1	0.0001	True	True	60.77

Table 5.1: Hyperparameter tuning results on a RVL-CDIP subset (10,016 training set, 5,024 validation set), using a 224×224 image size. Adam is the best optimizer for all image models. The validation accuracy is rounded to two decimal places.

¹ Each FC layer contains 50 neurons

base classifier. The EfficientNet models B0, B1, B2 and B3 are only considered due to memory restrictions, since larger EfficientNet models can not be loaded in memory with a batch size of at least 32. The batch size of 32 is fixed for all models in this work, since it generally provides a better generalization ability than larger batch sizes [85], and is an appropriate default value to start with [86]. Intuitively, it makes sense to have the highest validation accuracy on the holistic model, while the footer model, i.e., the model which focuses on the footer region of document images, has the lowest. This also holds true for the test set, once all weights of the model are unfrozen, which is explained in Section 5.3.

As a second hyperparameter tuning step, noisy student weights along with different EfficientNet base classifiers are considered. Even though the noisy student training increases the prediction accuracy on the ImageNet test set for EfficientNets by more than 3 percentage points, no such improvement is observed, when applying the noisy student weights for EfficientNets on the RVL-CDIP dataset, as Table 5.2 shows. In order to reduce the computational complexity, only the holistic model is considered, but the image size is increased to 384×384 .

Table 5.2 shows that the noisy student weights actually reduce the validation accuracy. The base classifier EfficientNet-B1, with weights from ImageNet, achieves the best results. Thus, this classifier, along with the best hyperparameters shown in Table 5.1, is used to train on the full RVL-CDIP training dataset. First, only the classification head is trained, and then the training is performed on the whole model.

The results for further tuning, which includes the BN, interpolation method and the evaluation metric (validation loss vs validation accuracy), can be seen in the Appendix, Section A. In particular, the result in the evaluation metric leads to the decision to choose the submodels with the highest validation accuracy.

Different base classifiers and weight initialization		
Classifier	Weight initialization	Validation accuracy
EfficientNet-B0	ImageNet	78.80
EfficientNet-B1	ImageNet	79.18
EfficientNet-B2	ImageNet	78.66
EfficientNet-B3	ImageNet	78.65
EfficientNet-B0	Noisy student	78.56
EfficientNet-B1	Noisy student	78.60
EfficientNet-B2	Noisy student	78.09
EfficientNet-B3	Noisy student	78.33

Table 5.2: Comparison of different base classifiers (EfficientNets) and weight initializations, evaluated on a RVL-CDIP subset (10,016 training set, 5,024 validation set), using a 384×384 image size. The validation accuracy is rounded to two decimal places.

5.1.2 Text stream

Similarly to the image stream, the text stream follows a two part hyperparameter tuning. Due to restricted GPU memory size, however, only the maximum sequence lengths 128 and 256 can run; 512 is not possible. Setting the hyperparameters, as described in Section 4.2.5 fixed, the validation accuracy for the maximum sequence length of 128 is 73.57%, compared to 74.15% for 256 (rounding to two decimal places). Thus, 256 is chosen as the sequence length. Next, a hyperparameter tuning is conducted to find the optimal dropout rate, number of neurons in the FC layer and learning rate. Table 5.3 depicts the optimal set of hyperparameters, which leads to the highest validation accuracy:

Optimal hyperparameters			
Dropout	Neurons in FC layer	Learning rate	Validation accuracy
0.3	512	0.00001	75.56

Table 5.3: Hyperparameter tuning results on a RVL-CDIP subset (10,016 training set, 5,024 validation set). The validation accuracy is rounded to two decimal places.

These hyperparameters are then used to train the model on the full RVL-CDIP training dataset. Only one FC layer is used on top of the base DistilBERT model (excluding the final output layer with a softmax activation). As in the image stream, only the classification head of DistilBERT is trained first, followed by a training on the whole model. While training the classification head, the validation loss turned out to decrease too slowly with a learning rate of $1e-5$. After 50 epochs it is still decreasing with no sign of slowing down. Thus, the learning rate is increased by a factor of 10, to $1e-4$. Even then, the validation loss is decreasing very slowly. However, after 120 epochs, the training is stopped after no significant change in the loss is observed.

5.1.3 Meta-classifier results on RVL-CDIP

The results in this section are from those different meta-classifier architectures, as well as different inputs for the meta-classifiers, mentioned in Section 4.3. Those different meta-classifiers are evaluated on the RVL-CDIP dataset. An early stopping with a patience of 3 is used. From each fold, the last reported accuracy is taken, and then averaged over all folds. Only the highest validation accuracy on average is reported. The results can be seen in Tables 5.4 and 5.5, respectively.

Meta-classifier results					
Models	Dropout	FC layers	Neurons in FC layer	Learning rate	Validation accuracy
Full model	0.5	4	512	0.0001	94.63
Holistic, text	0.3	2	1024	0.0001	93.87
Holistic, header	0.5	3	1024	0.0001	92.89
Holistic, header, text	0.3	4	512	0.0001	94.16
Holistic, header, footer	0.1	3	512	0.00005	93.37
Holistic, header, footer, text	0.3	4	256	0.0001	94.49
Holistic, header, left body	0.3	1	128	0.0005	93.23
Holistic, header, left body, text	0.5	3	256	0.0005	94.34
Holistic, header, left body, right body	0.1	3	512	0.0001	93.37
Holistic, header, left body, right body, text	0.3	3	1024	0.00005	94.44
Holistic, header, footer, left body, right body	0.3	3	128	0.0005	93.72

Table 5.4: Results from the meta-classifier on the validation set, using the class probabilities as input. The validation accuracy is rounded to two decimal places and 'text' refers to the DistilBERT model.

The approach for Table 5.4 is also done in [9], while the approach for Table 5.5 is done in [11]. Clearly, the results from Table 5.4 are better. In fact, for each model combination, the results are better, except for the combination of holistic + header + left body + text. Moreover, in Table 5.5, the dropout rate is generally higher and the learning rate is generally lower, compared to all other models in Table 5.4 on average. Lastly, the number of FC layers is generally lower for Table 5.5. This is due to the much larger input size for the meta-classifier. For Table 5.5, the input vector has a dimension of 7,168, while for Table 5.4, the dimension is 96 (assuming the full model for each case). Thus, this result shows that one should use class probabilities when working with a stacked model. This might also explain, why Harley et al. [11] report worse results on the RVL-CDIP test set for a stacked model, compared to a single holistic model.

Meta-classifier results					
Models	Dropout	FC layers	Neurons in FC layer	Learning rate	Validation accuracy
Full dataset	0.5	1	512	0.00005	94.54
Holistic, text	0.5	1	256	0.0001	92.37
Holistic, header	0.5	1	512	0.00005	92.56
Holistic, header, text	0.3	1	256	0.00005	94.14
Holistic, header, footer	0.5	1	512	0.00005	93.20
Holistic, header, footer, text	0.5	1	512	0.00005	94.39
Holistic, header, left body	0.5	1	256	0.0001	93.11
Holistic, header, left body, text	0.3	2	256	0.00005	94.35
Holistic, header, left body, right body	0.5	0	512	0.00005	93.25
Holistic, header, left body, right body, text	0.3	1	256	0.00005	94.39
Holistic, header, footer, left body, right body	0.3	1	256	0.00005	93.64

Table 5.5: Results from the meta-classifier on the validation set, using features from the last convolutional and hidden layer as input. The validation accuracy is rounded to two decimal places and 'text' refers to the DistilBERT model.

5.2 Results on Tobacco3482

Table 5.6 shows the results on the Tobacco3482 dataset, along with the number of parameters.

Results on Tobacco3482		
Author	Accuracy	# Parameters
Harley et al. (2015) [11]	79.90	220.97
Kang et al. (2014) [59]	65.35	4.21
Kumar et al. (2014) [37]	43.27	-
Ferrando et al. (2020) [10]	94.04	73.86
Afzal et al. (2015) [15]	91.13	25.60
Proposed approach (full model, train)	96.25	99.96
Proposed approach (full model, val)	96.17	99.96
Proposed approach (image models, train)	95.65	33.20
Proposed approach (image models, val)	95.75	33.20

Table 5.6: Tobacco3482 test set results. Accuracy in %. The number of parameters (in million) is either explicitly stated in the work or an estimation. Otherwise, it is omitted.

The proposed approach includes 4 different results. The full model refers to the architecture which is depicted in Figure 4.5. The image models refer to the 5 region based CNNs. The meta-classifier of both types of models are trained on either the training

set or the validation set. Table 5.6 shows that SOTA performance is achieved on all 4 different approaches. It can be seen, that the full model, i.e., image and textual features, achieves the highest test set accuracy. Concerning the model size, all models have less than 100 million parameters. The difference in the number of parameters between the full model and image models is due to DistilBERT.

Contrary to what is mentioned by Ting and Witten [84], using the class probabilities from the training set to train the meta-classifier has little impact in this case. Table 5.6 shows the results, using the class probabilities from the training set and from the validation set, for 2 different approaches. In the full model, using the class probabilities from the training set results in a slightly higher accuracy, while for the other approach it is the other way around. This suggests, when working with small datasets, such as Tobacco3482, using class probabilities from the training set as an input for the meta-classifier is justifiable.

More detailed results, which include a confusion matrix and the accuracies per class can be found in the Appendix, Section B2. These results refer to the overall best approach, i.e., the full model using the training set for the meta-classifier.

5.3 Results on RVL-CDIP

The results on the RVL-CDIP test set include the results from the base models, the results from the meta-classifier on the test set (i.e. model combinations) and the comparison with other SOTA approaches. Table 5.7 depicts the test results of only the base models.

Results of base models on RVL-CDIP		
Model	Das et al. (2018)	Proposed approach
Holistic	91.10	91.81
Header	86.00	88.10
Footer	81.20	82.88
Left body	85.20	86.71
Right body	82.20	83.95
DistilBERT	-	85.12

Table 5.7: Comparison of the test accuracy of base models with Das et al. (2018) in %, rounded to two decimal places. Das et al. (2018) use only image models.

Table 5.7 clearly shows the advantages of using an EfficientNet as a base model, compared to a VGG-16 model. Each base model in this proposed approach has a higher test accuracy. Note that Das et al. [9] have not used a NLP model in their work.

The final results in Table 5.8 might be counterintuitive. The full model has a lower test accuracy (93.50%) than all image models combined (93.70%). This "issue" is considered in great detail in Chapter 6. However, what makes intuitively sense is, that each added

Meta-classifier results on RVL-CDIP					
Models	Dropout	FC layers	Neurons in FC layer	Learning rate	Test accuracy
Full model	0.5	4	512	0.0001	93.50
Holistic, text	0.3	2	1,024	0.0001	89.36
Holistic, header	0.5	3	1,024	0.0001	90.48
Holistic, header, text	0.3	4	512	0.0001	91.69
Holistic, header, footer	0.1	3	512	0.00005	92.81
Holistic, header, footer, text	0.3	4	256	0.0001	92.71
Holistic, header, left body	0.3	1	128	0.0005	92.91
Holistic, header, left body, text	0.5	3	256	0.0005	93.29
Holistic, header, left body, right body	0.1	3	512	0.0001	93.50
Holistic, header, left body, right body, text	0.3	3	1,024	0.00005	93.15
Holistic, header, footer, left body, right body	0.3	3	128	0.0005	93.70

Table 5.8: Results from the meta-classifier on the test set, using class probabilities as input. The test accuracy is rounded to two decimal places and 'text' refers to the DistilBERT model.

image model increases the accuracy. Generally speaking, in almost all cases, when adding the DistilBERT model, the test accuracy is lower.

Table 5.9 shows the comparison of the proposed approach to other works, in terms of accuracy and parameters.

Results on RVL-CDIP		
Author	Accuracy	# Parameters
Afzal et al. (2017) [15]	90.97	138.36
Das et al. (2018) [9]	92.21	691.87
Audebert et al. (2020) [3]	90.60	3.64
Ferrando et al. (2020) [10]	92.31	85.47
Harley et al. (2015) [11]	89.80	58.35
Jain and Wigington (2019) [7]	93.60	138.36
Sarkhel and Nandi (2019) [46]	92.77	-
Tensmeyer and Martinez (2017) [13]	91.03	-
Xu et al. (2020) [8]	94.42	160.00
Xu et al. (2021) [16]	95.64	426.00
Proposed approach	93.70	40.72

Table 5.9: RVL-CDIP test set results. Accuracy in %. The number of parameters (in million) is either explicitly stated in the work or an estimation. Otherwise, it is omitted.

The proposed approach has the third highest test set accuracy on RVL-CDIP, and the second lowest number of parameters. More detailed results, which include a confusion matrix and accuracies per class, can be found in the Appendix, Section C. These detailed results are available for the final model, as well as for each base model. Figure 5.1 shows wrongly classified example predictions for each class.

5.4 Parameters and runtimes

Table 5.10 gives an overview of the different runtimes from each module. As seen in Figure 4.5, a total of 9 models are used to train the document image classification system.

Runtimes and # parameters for the proposed architecture				
Model	# Total parameters	# Trainable parameters	Epochs	Seconds per epoch
Holistic (classification head)	6,645,225	67,426	33	7,500
Holistic (whole model)	6,645,225	6,580,610	14	13,300
Header	6,645,225	6,580,610	13	13,300
Footer	6,645,225	6,580,610	15	13,300
Left body	6,645,225	6,580,610	14	13,300
Right body	6,645,225	6,580,610	12	13,300
DistilBERT (classification head)	66,764,816	401,936	120	3,200
DistilBERT (whole model)	66,764,816	66,764,816	8	8,180
Meta-classifier (class probabilities)	845,840	845,840	6	2

Table 5.10: Runtimes and number of parameters for the proposed document image classification architecture.

Note that in inference mode, only 7 of those models are used. The models, in which only the classification head is trained (Holistic and DistilBERT), are not used in inference mode. Ignoring these 2 models, the total number of parameters of this model amounts to 107,785,110.

However, not only the training of the models, but also the text preprocessing took a significant amount of time. The text preprocessing is done locally, on an Intel i7-10750H CPU machine, with 24GB of RAM. The following analysis is only based on the Tobacco3482 dataset. The total preprocessing run time is 5,307 seconds, with an average runtime of roughly 1.5 seconds per document image and a huge variability. The maximum time for a file is more than 9 seconds, while the smallest time is around 0.2 seconds. The 9 second difference in preprocessing is due to the noise in the document image. The more noise, which in this case is mostly salt-and-pepper noise, the longer the preprocessing time. Extrapolating these numbers to the RVL-CDIP dataset, the preprocessing took around 169.4 hours, or approximately one week.

0 / 1	0 / 2	0 / 3	0 / 11	0 / 13	0 / 15
1 / 0	1 / 3	1 / 5	1 / 7	1 / 11	1 / 13
2 / 0	2 / 9	2 / 10	2 / 12	2 / 13	2 / 15
3 / 0	3 / 1	3 / 4	3 / 7	3 / 13	3 / 13
4 / 3	4 / 6	4 / 8	4 / 9	4 / 12	4 / 13
5 / 1	5 / 3	5 / 6	5 / 9	5 / 10	5 / 12
6 / 1	6 / 4	6 / 5	6 / 8	6 / 9	6 / 12
7 / 0	7 / 1	7 / 3	7 / 5	7 / 6	7 / 13

Figure 5.1: Example predictions. Blue represents the actual class, red represents the predicted class. 0 = Letter, 1 = Form, 2 = Email, 3 = Handwritten, 4 = Advertisement, 5 = Scientific report, 6 = Scientific publication, 7 = Specification, 8 = File folder, 9 = News article, 10 = Budget, 11 = Invoice, 12 = Presentation, 13 = Questionnaire, 14 = Resume, 15 = Memo.

5. RESULTS



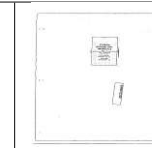






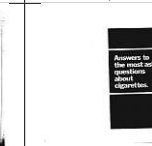





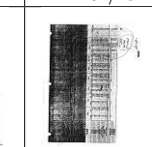
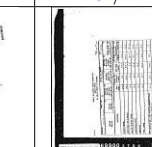
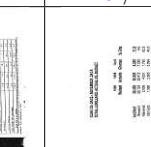


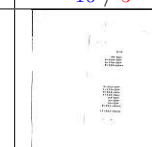
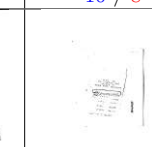

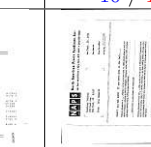


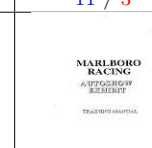
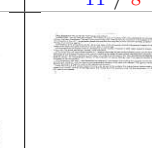
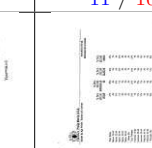

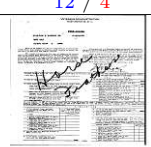

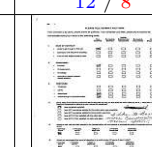

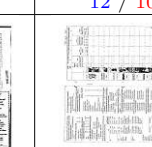
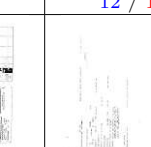

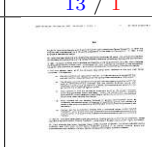

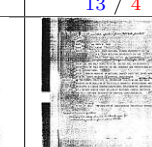
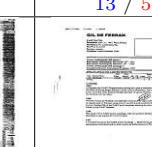
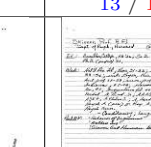

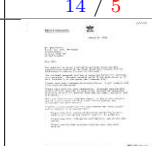

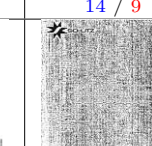
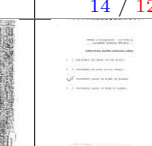

 8 / 1	 8 / 4	 8 / 5	 8 / 6	 8 / 12	 8 / 13
 9 / 4	 9 / 5	 9 / 6	 9 / 8	 9 / 12	 9 / 12
 10 / 1	 10 / 3	 10 / 5	 10 / 8	 10 / 11	 10 / 12
 11 / 0	 11 / 1	 11 / 3	 11 / 8	 11 / 10	 11 / 13
 12 / 4	 12 / 5	 12 / 8	 12 / 9	 12 / 10	 12 / 14
 13 / 0	 13 / 1	 13 / 3	 13 / 4	 13 / 5	 13 / 12
 14 / 1	 14 / 5	 14 / 6	 14 / 9	 14 / 12	 14 / 13
 15 / 0	 15 / 1	 15 / 5	 15 / 8	 15 / 10	 15 / 12

Figure 5.1 (cont.): Example predictions. Blue represents the actual class, red represents the predicted class. 0 = Letter, 1 = Form, 2 = Email, 3 = Handwritten, 4 = Advertisement, 5 = Scientific report, 6 = Scientific publication, 7 = Specification, 8 = File folder, 9 = News article, 10 = Budget, 11 = Invoice, 12 = Presentation, 13 = Questionnaire, 14 = Resume, 15 = Memo.

5.5 Summary

This chapter shows the hyperparameter tuning results and the results of each submodel for the image stream and text stream. It further shows the results of different submodel combinations, e.g. holistic model + header model. Additionally, the results of the finetuned models on the Tobacco3482 dataset are displayed.

The hyperparameter tuning results for both streams are obtained by performing a two stage tuning, in order to reduce the computational complexity. In both streams, the tuning is performed only on the added classification head on top. Moreover, a separate smaller dataset is created for the tuning process, which follows the class distribution of the original dataset. It consists of 10,016 training files and 5,024 validation files. The best validation accuracy during the tuning process is 79.18% for the holistic model, and 75.56% for the DistilBERT model.

Once the best hyperparameter settings are found, a training on the whole dataset is conducted. The ranking of the best to the worst submodel is the same as in [9], i.e. the holistic model is the best, followed by the header, left body, right body and footer model. The DistilBERT model achieves a test set accuracy of 85.12%.

In total, 11 different combinations of submodels (including the full model) are evaluated, by feeding the outputs of the submodels into a meta-classifier. Using only image submodels leads to a higher test set accuracy than the full model (93.70% vs 93.50%), i.e., the text submodel (DistilBERT) reduces the final test set accuracy. More about this result in Chapter 6.

These submodels are then taken to perform a finetuning on the Tobacco3482 dataset. No hyperparameter tuning is performed. The hyperparameters, such as learning rate, are chosen after an initial experimentation. It is evaluated, whether the full model achieves a higher test set accuracy than only the image submodels. Moreover, it is evaluated, what impact the choice of the dataset on the test set accuracy has. This dataset is used to train the meta-classifier, which consists of either the softmax output of the submodels on the training set or on the validation set. The full model with the training set achieves the highest test set accuracy, 96.25%.

Additionally, results about the number of parameters and runtimes are reported. The total number of parameters of the model in inference mode is 107,785,110. One epoch on one of the image models takes about 220 minutes, while on the text model it takes about 136 minutes on the RVL-CDIP dataset, using the system as described in Section D of the Appendix. The time to preprocess the text of the RVL-CDIP dataset is about one week. This, however, is done locally on a i7-10750H CPU machine with 24GB RAM.



Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

CHAPTER 6

Discussion

This proposed model achieves a test set accuracy of 93.70% on the RVL-CDIP test set, and a test set accuracy of 96.25% on the Tobacco3482 test set (both rounded to two decimal places). Even though there are minor differences in the datasets between the cited document image classification papers in this work due to the overlap of document images between the RVL-CDIP and Tobacco3482 datasets, these differences are insignificant. In total, the overlap would be 258 images between the training set of RVL-CDIP and the Tobacco3482 dataset. In this work, the overlap is not removed since the majority of the cited document image classification papers in this work use the full RVL-CDIP dataset [18], and this work is more focused around the RVL-CDIP dataset. The newest SOTA models published on document image classification use only the RVL-CDIP dataset, to test their algorithms, such as [8] and [16].

Working with a large dataset, such as RVL-CDIP, requires either huge computational resources or a lot of compute time. Since in this work, the computations are done on a NVIDIA T4 GPU, the proposed model has to be more efficient than the SOTA model. For comparison, the second best model in terms of overall accuracy, [8], uses 8 NVIDIA Tesla V100 32GB GPUs, where their BASE model takes 80 hours to finish one epoch on 11 million documents, while their LARGE model takes nearly 170 hours to finish one epoch on the same dataset, or approximately one week.¹

The huge dataset is the primary reason why the hyperparameter tuning for the image and text stream is done on a smaller subset and only on the classification head. Moreover, the tuning is done in two parts for both streams to further reduce the complexity. Interestingly, when the tuning is done on the image stream on all 5 image based models,

¹As of April 2022, training the LARGE model for one epoch would cost around 4,900€ on an aws p3dn.24xlarge instance (8 NVIDIA Tesla V100 32GB), in the Oregon region. The Oregon region is, together with Nord-Virgina, the cheapest for this instance type as of April 2022. In Ireland, for example, one epoch would cost around 5,300€.

almost the same set of hyperparameters on all these models turned out to achieve the highest validation accuracy. Moreover, by far the biggest influence on the validation accuracy is the image size. This can be seen when comparing Table 5.1 and Table 5.2. The holistic model in Table 5.1, with image size 224×224 , achieves a validation accuracy of 67.70%, compared to the same model with an image size of 384×384 , achieving 78.80%. Table 5.2 further shows that bigger models do not necessarily perform better on this dataset, and that EfficientNet-B1 achieves the highest validation accuracy, which is the base model of the final proposed approach in Figure 4.5. On the other hand, in the text stream, the hyperparameter tuning does not increase the validation accuracy as much as the image stream. However, similarly to the image size in the image stream, the maximum sequence length in the text stream has the biggest influence. The hyperparameter tuning on the meta-classifier, however, is computationally orders of magnitude faster, which allows for an extensive tuning as well as for testing several different model combinations.

Due to the modular nature of the proposed system, it can be parallelized. The system contains 9 modules, 5 of them are image models and based on a CNN, 1 is a transformer and another one is a MLP which serves as the meta-classifier. The last two modules are used just for training. In the first step, only the classification head is trained for both modules, while in the second step, the whole model is trained. 4 of the 5 image based models can be trained in parallel, i.e., the header, footer, left body and right body part. Note that these 4 models are initialized with the weights from the holistic model, as can be seen in Figure 4.5.

The holistic model and the DistilBERT model are initialized with certain weights. These weights are the weights from the same model, where only the classification head is trained. As mentioned, these two models with only the classification head trained are only used during training. It turns out, that the separate training of the classification head only, plus the subsequent training of the whole model, results in a higher validation accuracy, than the training of the whole model alone. The training of the whole holistic model only has a validation accuracy of 91.97% (or 92.04%, respectively, as can be seen in Figure 1 in Section A of the Appendix), while the other way, of training the classification head first and subsequent the whole model, has a validation accuracy of 92.13%. This approach turns out to be even better for the text model. When training the DistilBERT model without training the classification head, the validation accuracy is 84.64%, while the other way has a validation accuracy of 85.09%. Section A in the Appendix contains the results and analysis of other hyperparameters, such as BN, the interpolation method for resizing and the validation metric (loss vs accuracy). In summary, the interpolation method has practically no impact, the BN has some impact, and the validation metric has the highest impact of these 3 hyperparameters, when evaluating on the test set.

From Table 5.7, it can be seen, that the proposed approach is far better for each image model, increasing the accuracy up to 2.1 percentage points on the header model. Das et al. [9] utilize a VGG-16 model, showing that a EfficientNet-B1 model, which is much smaller in terms of parameters, is superior. Table 5.9 shows the total number of

parameters. The final model for the proposed approach is over 10 times smaller in terms of parameters than the approach in [9], while the test accuracy is around 1.5 percentage points higher. However, regarding the total number of parameters and the training time, it would be more appropriate to report the number of trainable parameters as well, if the number of trainable parameters is lower than the number of total parameters, since this reduces the training time. Unfortunately, this is not reported in the cited papers about document image classification. Table 5.10 shows that the runtime for the holistic model and the DistilBERT model is very different, based on how many parameters are trainable. Training only the classification head for both these models takes roughly half the time. Table 6.1 shows the accuracies per class from the final model and all base models.

Accuracies for each class from the final model and all base models							
Class	Final model	Holistic	Header	Footer	Left body	Right body	DistilBERT
Letter	91.23	90.22	86.20	67.57	84.09	76.99	87.30
Form	86.79	84.00	80.09	74.18	80.17	74.94	79.13
Email	99.05	98.85	97.62	96.78	86.37	90.42	95.43
Handwritten	97.04	95.10	93.68	94.00	92.22	91.51	72.16
Advertisement	95.19	93.08	87.83	86.52	88.79	88.35	66.00
Scientific report	89.11	81.06	81.22	73.90	76.18	69.58	78.74
Scientific publication	93.62	92.80	88.91	89.73	90.51	90.90	87.32
Specification	94.86	93.85	92.56	90.49	90.45	89.40	92.96
File folder	97.35	96.16	94.30	89.04	93.75	87.53	89.39
News article	93.99	92.04	86.68	85.63	84.98	83.27	85.22
Budget	92.93	91.82	83.99	80.88	83.59	81.48	82.28
Invoice	94.59	93.34	85.75	85.51	88.05	89.99	90.59
Presentation	88.19	85.74	78.51	75.05	81.20	74.73	81.52
Questionnaire	92.53	89.98	84.15	77.74	85.95	83.41	87.56
Resume	98.15	96.69	95.70	92.12	95.11	94.48	97.67
Memo	94.30	93.94	91.93	66.05	85.47	75.56	88.76

Table 6.1: Test set accuracies on the RVL-CDIP dataset for each class from the final model and all base models, rounded to two decimal places.

Except for three classes, the test accuracy is always above 90% for the final model. In all classes, the final model has a higher test accuracy than the holistic model, with email having the highest accuracy of all classes. Interestingly, in two cases, the holistic model is worse than a region based model (header) and the DistilBERT. For the scientific report class, the header model achieves a higher test accuracy, compared to the holistic model, and for the resume class, the DistilBERT model achieves a higher accuracy. When comparing only the region based models (header, footer, left body and right body), it becomes clear that for certain classes, certain region based models perform better, showing that it is beneficial to focus on certain regions of a document. Among those 4 models, the header model achieves the highest test accuracy 9 times, followed by the

left body model (4 times), right body model (2 times) and footer model (1 time). The number of times of highest test accuracies achieved by those 4 models makes sense, when comparing it with the overall accuracy of the region based models (see Table 5.7). More detailed results, which contain the confusion matrices for each base classifier and the final model, can be seen in Section C of the Appendix, in Tables 3 to 9.

Looking once again at Table 6.1, the DistilBERT model has a higher test accuracy in 6 classes than any of the 4 region based CNN models. This suggests that combining all models leads to a higher test accuracy. This, however, is not the case. As can be seen from Table 5.8, the highest test set accuracy is achieved when combining all image models. Adding the text model decreases the test set accuracy by 0.20 percentage points. Intuitively, it does not make sense that adding textual information would lead to a worse test set accuracy. In fact, all recent models and the current SOTA on RVL-CDIP ([8], [10], [7], [3]) and [16] utilize textual information.

Analyzing the overlap of correct predictions of all models shows that the overlap between the image models is relatively higher than the overlap between the image models and the text model. As an example, Table 6.2 shows the overlap between the holistic model and all other models on the RVL-CDIP validation data.

Absolute and relative overlap of correct predictions		
Model	Absolute overlap	Relative overlap
Header	34,227	97.04
Footer	32,245	96.99
Left body	33,830	97.15
Right body	32,861	97.20
DistilBERT	32,609	95.81

Table 6.2: Absolute and relative overlap of correct predictions with the holistic model on the RVL-CDIP validation data.

Table 6.2 shows that the correct predictions of the DistilBERT model have the lowest relative overlap with the correct predictions of the holistic model. The footer model, for example, has a lower absolute overlap because its validation set accuracy is almost 2 percentage points lower (as can be seen in Section A of the Appendix), compared to DistilBERT's, but it has a higher relative overlap, which considers the number of correct predictions for the footer model in this case. A more detailed analysis as to why the test set accuracy decreases when adding textual information can be seen when analyzing the softmax output of each base model. Figure 6.1 displays this analysis.

The histograms in Figure 6.1 are based on the softmax output of each model on the RVL-CDIP validation data. Since the validation data has 16 classes and 40,000 images, this results in 640,000 values for each of the six histograms. The histograms are plotted on a base 10 logarithmic scale, ranging from $1e-35$ to 1, where each histogram contains 100 bins. The last bin ranges from around 0.4431 to 1, which approximately shows the

number of correct predictions in each histogram. The clear difference in the distributions can be seen. The image models show approximately a normal distribution, while the DistilBERT model in the bottom right has a skewed distribution. This might explain why the model performs worse, once the textual information is added. The same applies to other model combinations, e.g., Holistic + text vs Holistic. Generally, in most cases, where the DistilBERT model is added, the test accuracy decreases. However, looking at Table 5.6, the model on the Tobacco3482 dataset does perform better when adding textual information. Both full models achieve a higher test set accuracy than the image models. For a better understanding of why textual information now helps to achieve a higher test set accuracy, the distributions of the softmax output on both the training and validation set is plotted and can be seen in the Appendix, Section B2, Figures 2 and 3. In both cases, the distribution of the DistilBERT model on the bottom right of the graph is similar to all other image models. Thus, this strongly suggests that the final accuracy, i.e., the output of the meta-classifier, depends on the softmax output distribution of the base classifiers.

Another observation that can be seen in Table 5.8, is the fact that adding more and more region based models to the holistic model increases the accuracy, albeit the increase is less and less, i.e., it brings diminishing returns. The only exception to that is, that adding the header model to the holistic model decreases the test accuracy (90.48% vs 91.81%).

Figure 5.1 shows a collection of document images, which are not correctly classified by the final model. Each row represents one class. The blue number stands for the correct class, while the red number stands for the predicted class. While creating Figure 5.1, a few issues regarding the data quality of the RVL-CDIP dataset have been found. Dauphinee et al. [18] find that an estimated 2,259 document images in the RVL-CDIP dataset are duplicates. Moreover, of these 2,259 document images, the image of 426 document images is not available and displayed as shown in Figure 6.3 (a). Other dataset quality issues include different languages for the document images. In particular German, Dutch, Hebrew, Swedish, Spanish and French have been seen while creating Figure 5.1. Furthermore some document images are rotated by 90°. For example, the image in row 2 and column 3 (with the caption "1 / 5") is rotated by 90°. More rotated examples can be found in Figure 5.1. Since CNNs are not rotation invariant, these rotated images could be an issue. However, further study would be needed to determine the exact amount of different languages and rotated images. When it comes to different languages, it makes sense that CNNs are largely stable in this regard in contrast to a transformer, since filters in CNNs focus more on the layout and form of the document. To give an example of what the filters in a CNN, specifically in the RVL-CDIP dataset, capture, see Figure 6.2. It shows gradient ascent visualizations for filters, where a randomly initialized input image is modified such that the mean activation of a given filter is maximized [87]. The filters in Figure 6.2 are from increasing depths in the InceptionV3 [88] network.

Figure 5.1 (cont.) further shows, what kind of image quality is present in the dataset. For example, the document image in the last row and fourth column in Figure 5.1

(cont.) has a lot of noise all over it. The final dataset issue observed concerns the labels themselves. Of the 2,521 wrongly classified document images by the final model, at least 169 document images are wrongly labeled. One such example can be seen in Figure 6.3 (b), where the prediction of the final model makes more sense than the actual label.

Finally, considering the training architecture, and the fact that training the classification head first and then the whole model, which is done for the holistic model and the DistilBERT model, achieves a higher accuracy, one might think about doing this approach for all image based models. However, this would result in longer training times.

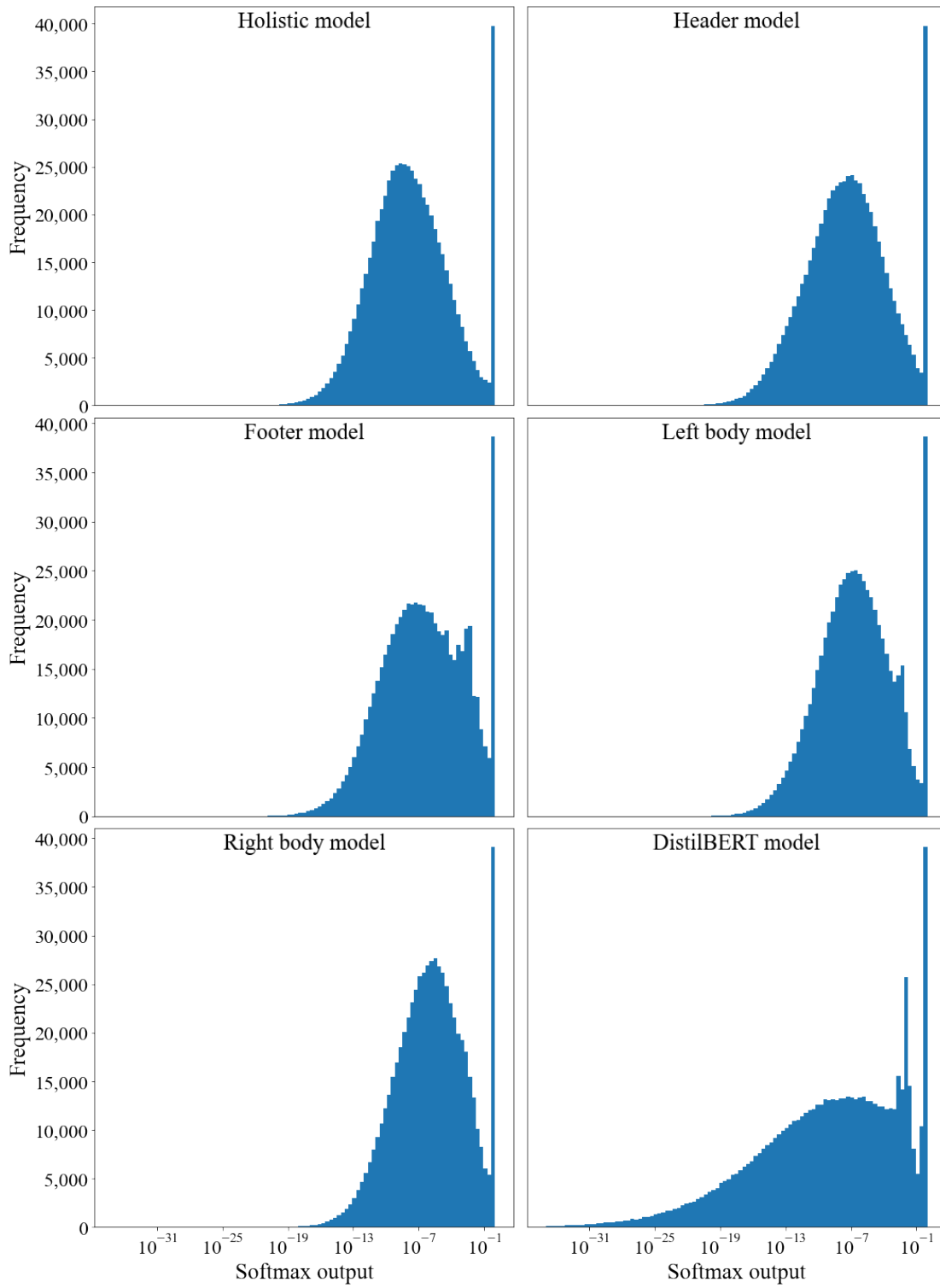


Figure 6.1: The softmax output distribution from the base classifiers on the RVL-CDIP validation data.

6. DISCUSSION

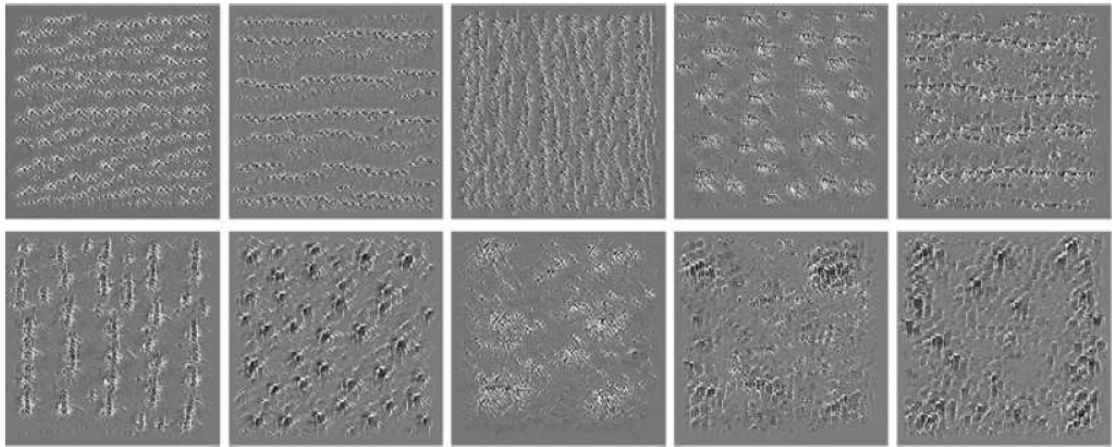
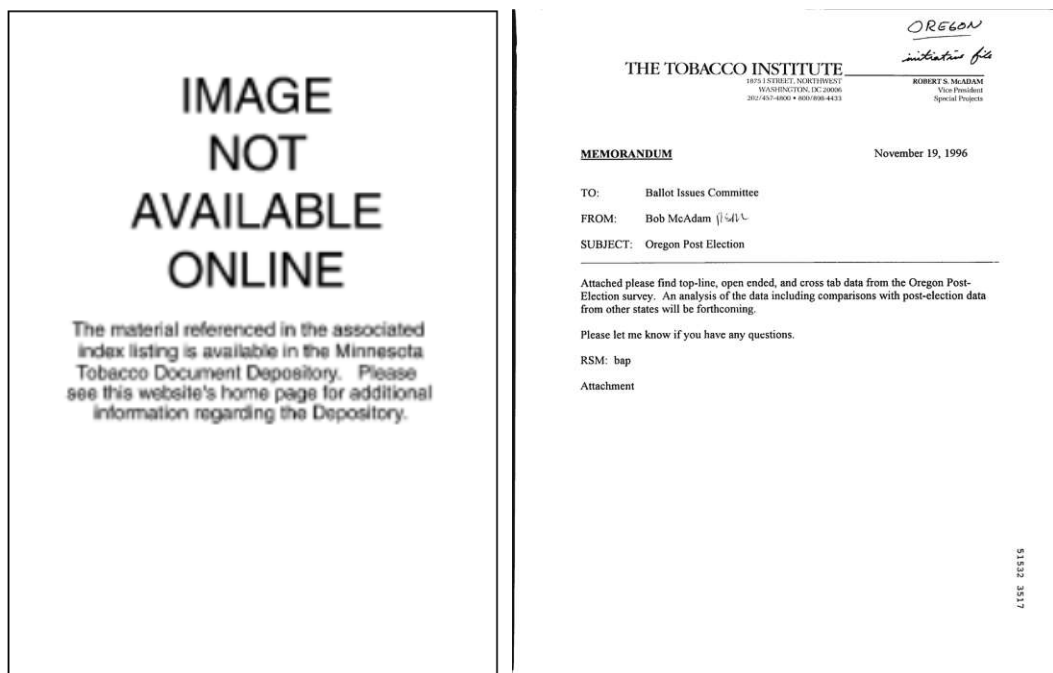


Figure 6.2: Gradient ascent visualization of filters learned from the RVL-CDIP dataset, where patterns for text lines, columns and paragraphs appear [87].



(a) Image occurring 426 times

(b) Label: Questionnaire, prediction: letter

Figure 6.3: Issues regarding the dataset quality.

Conclusion

The results show that the proposed model can compete with current SOTA methods. In fact, with a test set accuracy of 93.70% on the RVL-CDIP test set, the proposed method achieves SOTA, when excluding those models, which use an extra dataset with document images for training.

This chapter summarizes the work and discusses some limitations. Furthermore, the contribution and findings are outlined, together with some future work.

7.1 Summary and limitations

The architecture of the proposed model has both advantages and disadvantages in terms of time complexity. Because it is a stacked model, it requires multiple stages of hyperparameter tuning to find the best hyperparameters. However, because of the way transfer learning is used, 4 image based models can run in parallel, which greatly reduces the time complexity. Nevertheless, the running time is small, compared to the model by Xu et al. [8], which requires approximately one week on the LARGE model to finish just one epoch, and using 8 NVIDIA Tesla V100 32GB GPUs. The system used in this work is mentioned in Section D of the Appendix.

One reason for that long running time is the IIT-CDIP Test Collection, which contains more than 11 million document images. The current SOTA, [16] and the previous SOTA, [8], are both based on a transformer architecture without any CNN and have used this extra training dataset. The dataset size is also the deciding limitation factor. Probably the most surprising observation in this work is that the test accuracy for the RVL-CDIP test set gets worse when adding textual information. The reason for that is, that the distribution of the softmax output probabilities of the image models follow a normal distribution, while the softmax output probabilities for the DistilBERT model follow a skewed distribution (see Figure 6.1). However, the current SOTA model [16], as well as

the previous SOTA results, such as [8] or [7], utilize a combination of image and textual features. Nevertheless, in general, for document image classification, image features are preferred [14]. This work confirms that, when looking at Table 5.7. The holistic model achieves a test accuracy of 91.81%, compared to 85.12% for the DistilBERT model. This, however, is largely due to the image quality and different languages, as discussed in Chapter 6.

Aside from the dataset size used to train the model and the model size (e.g., a maximum sequence length of 512 for DistilBERT could not be used due to limited GPU memory), another limitation is the OCR system itself. All cited document image classification papers in this work, which are based on the RVL-CDIP dataset and use textual information, use the tesseract OCR system, except the current SOTA model [16], which uses the Microsoft OCR system. Further research would be needed in order to decide which OCR system is the most suitable, considering also other OCR systems, such as those from Google or Amazon. Finally, the last limitation is the type of input data that can be processed. Aside from a few document images, the language in this dataset is English. Thus, if one wants to use this proposed approach or the SOTA model, it would probably not work well out of the box for other languages. A larger finetuning step, i.e., more data to finetune with and/or unfreezing more layers, is thus necessary.

7.2 Contributions and future work

The main aim of this thesis is to achieve a high accuracy with a combination of visual and textual features, which can compete with current SOTA systems, without pretraining on millions of document images. The high accuracy is achieved, albeit without the use of textual information. To summarize, the contributions of this work are as follows:

- Developing a model which can compete with current SOTA models on the RVL-CDIP dataset, without the need of millions of document images. Moreover, the developed model has over 10 times less parameters than the current SOTA model.
- Improving the SOTA on the Tobacco3428 test dataset by 2.21 percentage points.
- Exploring and finding the most discriminative parts within a document based on the different model combinations.
- Analyzing the unintuitive result (less test accuracy with textual information), which could lead to new future work about increasing the test accuracy in a stacked model.
- Showing dataset quality issues in RVL-CDIP.
- Showing that meta-classifiers can also learn on class probabilities from the training set, without achieving a lower test accuracy.

Next to these contributions, the findings of this work are the following:

- Adding textual information to a stacked model of CNNs reduces the test accuracy.
- The interpolation method for resizing the document image has little to no impact for this dataset (see Figure 1 in Section A of the Appendix).
- The BN has some impact its weights also be updated when unfreezing the model (see Figure 1 in Section A of the Appendix).
- The model with the highest validation accuracy leads to a higher test set than the model with the lowest validation accuracy. The difference can be as high as almost 1% (see Table 1 in Section A of the Appendix).
- For the EfficientNet-B1 model, the image size has the biggest influence on the test set. A size of 384×384 is used, instead of 224×224 or 227×227 , which is used in most papers on document image classification, such as [9], [7], [15], [14], [6], [21], [46], [65].
- Noisy student weights perform worse than Imagenet weights for RVL-CDIP.
- Adding more and more region based models improves the test accuracy (with diminishing returns).
- A meta-classifier trained on class probabilities instead of features from the last layers achieves a higher validation accuracy (see Tables 5.4 and 5.5).

Due the importance of this topic, document image classification has been explored extensively [15]. One reason for the importance is the fact that document image classification systems are an initial step in document processing tasks [9]. For example, first, all incoming documents are classified, and then only those documents, which have been classified as invoice move to a next system, which extracts the key information automatically. Due to this importance, the work on document image classification systems is likely to continue. This work shows how an efficient model, in terms of parameters, with a high test accuracy is build, and has laid the foundation for further research.

The last two SOTA methods, [8] and [16], suggest that transformer based architectures might be superior to CNN architectures. However, the last two SOTA methods have also used much more training data. Thus, future work could look whether training the SOTA method on the same amount of training data would beat the test accuracy of this proposed approach of 93.70%.

Moreover, the fact that the softmax output distribution of the DistilBERT model is different than those softmax outputs from image models (see Figure 6.1), leads to the following question: What impact would transforming the skewed distribution into a normal distribution have on the test set accuracy? Given that on the Tobacco3482 dataset, all softmax distributions look similar and that the full model performs better than the image models, i.e., textual information increases the test accuracy for this dataset (see Table 5.6), transforming the distribution might help to increase the test accuracy.

7. CONCLUSION

Another area, which could be explored more extensively, is the stacked generalization architecture for the image stream. More concretely, Table 5.8 shows that each additional image model, starting from the holistic + header combination, increases the test accuracy. The following question naturally arises: How many image regions can be added, until the test accuracy starts to decrease? For example, the document image could be sliced into 4 equal parts: top left, top right, bottom left and bottom right. What would the impact on the test set accuracy be, if these 4 image region models are added to the already 5 proposed image regions? Adding these 4 models would increase the total number of parameters by about 26.6 million parameters, to a total of around 67.32 million parameters, which is still far below the current SOTA model.

The ideas in the previous two paragraphs could also be combined. Another area for future work is to look into different OCR systems, or perhaps try the fifth version of the open source tesseract system. Finally, another possible way to improve the system is to try other transformer models, such as BERT, which generally has a better performance than DistilBERT, but contains many more parameters (110 million for the base model and 340 million for the large model).

List of Figures

1.1	The images are from different categories, but show relatively low interclass variability [11].	2
1.2	The images are from the same category, but show relatively high intraclass variability [11].	3
2.1	Number of files per category, where no text is left, after applying OCR and text preprocessing.	8
2.2	An example document image for each class from the RVL-CDIP [11] dataset. Notice the visual similarities, for example in (c) Email and (p) Memo, showing a low interclass variability.	9
2.3	Class distribution of the Tobacco3482 [21] dataset.	10
3.1	Overview of different features for a document image classification system, adapted from [6].	19
4.1	The architecture of LeNet-5 [61].	22
4.2	The four components described and their general procedure, for a 2D input. [63].	23
4.3	The training strategy for the image stream	28
4.4	The vanilla transformer architecture [51].	29
4.5	Proposed architecture for document image classification based on SOTA architectures, with an image - and text stream and utilizing different levels of transfer learning.	35
5.1	Example predictions. Blue represents the actual class, red represents the predicted class. 0 = Letter, 1 = Form, 2 = Email, 3 = Handwritten, 4 = Advertisement, 5 = Scientific report, 6 = Scientific publication, 7 = Specification, 8 = File folder, 9 = News article, 10 = Budget, 11 = Invoice, 12 = Presentation, 13 = Questionnaire, 14 = Resume, 15 = Memo.	49
6.1	The softmax output distribution from the base classifiers on the RVL-CDIP validation data.	59
6.2	Gradient ascent visualization of filters learned from the RVL-CDIP dataset, where patterns for text lines, columns and paragraphs appear [87].	60
6.3	Issues regarding the dataset quality.	60
		65

1	Effect of batch normalization and image interpolation methods on the validation accuracy, using the holistic model, based on EfficientNet-B1.	72
2	The softmax output distribution from the base classifiers on the Tobacco3482 training data.	74
3	The softmax output distribution from the base classifiers on the Tobacco3482 validation data.	75

List of Tables

3.1	Overview of different methods and their used features.	15
3.2	Overview of different methods and their used features on the RVL-CDIP dataset.	17
5.1	Hyperparameter tuning results on a RVL-CDIP subset (10,016 training set, 5,024 validation set), using a 224×224 image size. Adam is the best optimizer for all image models. The validation accuracy is rounded to two decimal places.	42
5.2	Comparison of different base classifiers (EfficientNets) and weight initializations, evaluated on a RVL-CDIP subset (10,016 training set, 5,024 validation set), using a 384×384 image size. The validation accuracy is rounded to two decimal places.	43
5.3	Hyperparameter tuning results on a RVL-CDIP subset (10,016 training set, 5,024 validation set). The validation accuracy is rounded to two decimal places.	43
5.4	Results from the meta-classifier on the validation set, using the class probabilities as input. The validation accuracy is rounded to two decimal places and 'text' refers to the DistilBERT model.	44
5.5	Results from the meta-classifier on the validation set, using features from the last convolutional and hidden layer as input. The validation accuracy is rounded to two decimal places and 'text' refers to the DistilBERT model.	45
5.6	Tobacco3482 test set results. Accuracy in %. The number of parameters (in million) is either explicitly stated in the work or an estimation. Otherwise, it is omitted.	45
5.7	Comparison of the test accuracy of base models with Das et al. (2018) in %, rounded to two decimal places. Das et al. (2018) use only image models.	46
5.8	Results from the meta-classifier on the test set, using class probabilities as input. The test accuracy is rounded to two decimal places and 'text' refers to the DistilBERT model.	47
5.9	RVL-CDIP test set results. Accuracy in %. The number of parameters (in million) is either explicitly stated in the work or an estimation. Otherwise, it is omitted.	47
5.10	Runtimes and number of parameters for the proposed document image classification architecture.	48
		67

6.1	Test set accuracies on the RVL-CDIP dataset for each class from the final model and all base models, rounded to two decimal places.	55
6.2	Absolute and relative overlap of correct predictions with the holistic model on the RVL-CDIP validataion data.	56
1	Comparison between models with the lowest validation loss and models with the highest validation accuracy.	72
2	Tobacco3482 test set results from the best model (full model, meta-classifier trained on softmax output of training set). Accuracy in % and rounded to two decimal places. Overall accuracy: 96.25%.	73
3	RVL-CDIP test set results from the best model (meta-classifier with only the image models as base classifiers). Accuracy in % and rounded to two decimal places. Overall accuracy: 93.70%.	76
4	RVL-CDIP test set results from the holistic model. Accuracy in % and rounded to two decimal places. Overall accuracy: 91.81%.	77
5	RVL-CDIP test set results from the header model. Accuracy in % and rounded to two decimal places. Overall accuracy: 88.10%.	77
6	RVL-CDIP test set results from the footer model. Accuracy in % and rounded to two decimal places. Overall accuracy: 82.88%.	78
7	RVL-CDIP test set results from the left body model. Accuracy in % and rounded to two decimal places. Overall accuracy: 86.71%.	78
8	RVL-CDIP test set results from the right body model. Accuracy in % and rounded to two decimal places. Overall accuracy: 83.95%.	79
9	RVL-CDIP test set results from the DistilBERT model. Accuracy in % and rounded to two decimal places. Overall accuracy: 85.12%.	79
10	Libraries used in this work and their version.	80

Acronyms

CNN Convolutional Neural Network

SOTA state-of-the-art

RVL-CDIP Ryerson Vision Lab Complex Document Information Processing

IIT-CDIP Illinois Institute of Technology Complex Document Information Processing

LTDL Legacy Tobacco Document Library

OCR Optical Character Recognition

DP Dynamic Programing

OC1 Oblique Classifier 1

HMM Hidden Markov Model

ARGs Attributed Relational Graphs

FORGs First Order Random Graphs

PCA Principal Component Analysis

AdaBoost Adaptive Boosting

BOW Bag-of-words

SVM Support Vector Machine

TAS Triple Adjacent Segment

RF Random Forest

SURF Speeded up Robust Features

SIFT Scale-Invariant Feature Transform

MLP Multi layer perceptron

VGG Visual Geometry Group

ResNet Residual Network

BERT Bidirectional Encoder Representations from Transformers

LSTM Long Short Term Memory

Adam Adaptive moment estimation

SGD Stochastic gradient descent

FC Fully connected

GPT Generative Pre-trained Transformer

GPU Graphics processing unit

CPU Central processing unit

RAM Random access memory

NLP Natural Language Processing

BN Batch Normalization

ReLU Rectified Linear Unit

RNN Recurrent neural networks

Appendix

A. Further tuning

In this section, some of the effects of other hyperparameters are shown. An extensive tuning on these hyperparameters is not done due to high computational complexity and, as it turns out, very little gain in terms of accuracy on one particular hyperparameter.

The three hyperparameters, which are explored additionally are BN, the interpolation method for image resizing and the evaluation metric for validating the models, i.e., whether to use validation accuracy or validation loss. The evaluation of the first two hyperparameters can be seen in Figure 1. Note that when testing without BN, only one interpolation method is tested, since there is practically no difference between the lanczos and bilinear interpolation method. The highest validation accuracy for the bilinear interpolation method is 92.04%, while for lanczos it is 91.97%. When using no batch normalization, the highest validation accuracy is 91.74%. Figure 1 shows, that between the two interpolation methods is practically no difference, while not having the batch normalization updated is worse. These methods are evaluated on the holistic model, which is based on the EfficientNet-B1 model. All weights are unfrozen, except in the no BN case, where, the BN weights of the base model (EfficientNet-B1) are kept frozen. Moreover, these models are finetuned with unfrozen weights from the beginning, i.e. they are not finetuned on the classification head first and then on the whole model. On each epoch, the validation accuracy is lower for the no BN case, compared to the other two methods.

Table 1 shows the difference in the test accuracy and validation accuracy, when using models with the lowest validation loss and models with the highest validation accuracy. A difference in the test set for each model between these two different validation metrics can be seen. For each image model, the test accuracy is higher when using the model with the highest validation accuracy. The difference can also be significant, as is in the case of the header model; the test accuracy is almost 1 percentage point higher when using the model with the highest validation accuracy. Interestingly, for the DistilBERT model, the test accuracy is higher in both cases, compared to the validation accuracy.

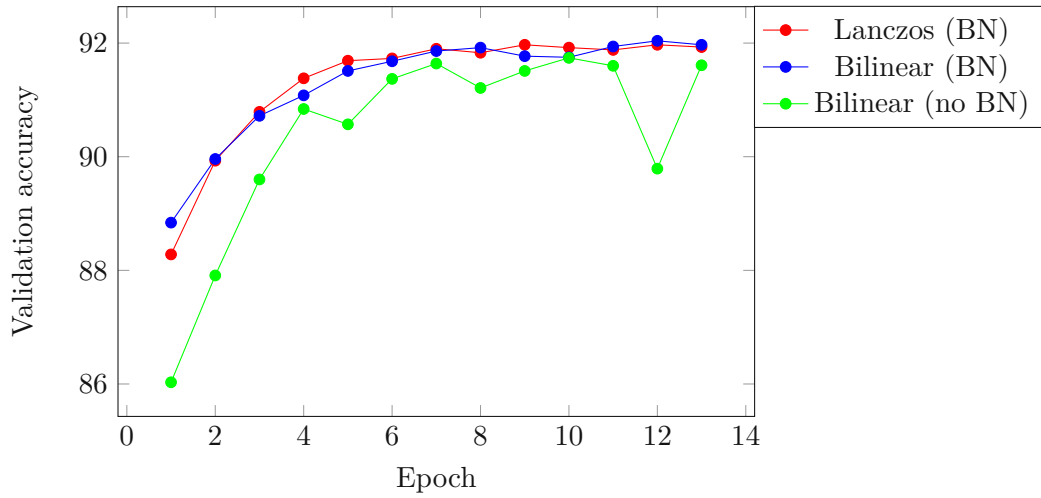


Figure 1: Effect of batch normalization and image interpolation methods on the validation accuracy, using the holistic model, based on EfficientNet-B1.

Model	Lowest validation loss models		Highest validation accuracy models	
	Test accuracy	Validation accuracy	Test accuracy	Validation accuracy
Holistic	91.50	91.66	91.81	92.13
Header	87.13	87.58	88.10	88.18
Footer	82.44	82.82	82.88	83.11
Left Body	86.18	86.43	86.71	87.05
Right Body	83.71	83.87	83.95	84.52
DistilBERT	84.59	84.53	85.12	85.09

Table 1: Comparison between models with the lowest validation loss and models with the highest validation accuracy.

B. Tobacco3482 results

The results for the Tobacco3482 dataset include the confusion matrix from the model with the highest test accuracy, and the softmax output of base classifiers, both on the training and validation set. Table 2 shows the confusion matrix, along with the accuracy from each class, while Figures 2 and 3 depict the softmax output.

B1. Confusion matrix

		Actual class									
		Advertisement	Email	Form	Letter	Memo	News	Note	Report	Resume	Scientific
Predicted class	Advertisement	124	0	0	0	0	0	0	0	0	0
	Email	1	496	0	0	0	0	0	0	0	0
	Form	0	0	322	0	0	0	2	0	0	2
	Letter	0	2	2	448	1	0	1	6	0	1
	Memo	0	1	2	3	516	0	4	2	0	2
	News	2	0	0	0	0	85	0	0	0	1
	Note	1	0	1	0	2	1	93	0	0	2
	Report	2	0	4	16	0	0	0	139	0	7
	Resume	0	0	0	0	0	0	0	1	20	0
	Scientific	0	0	0	0	1	2	1	17	0	146
	Accuracy (%)		95.38	99.40	97.28	95.93	99.23	96.59	92.08	84.24	100.00

Table 2: Tobacco3482 test set results from the best model (full model, meta-classifier trained on softmax output of training set). Accuracy in % and rounded to two decimal places. Overall accuracy: 96.25%.

B2. Softmax output

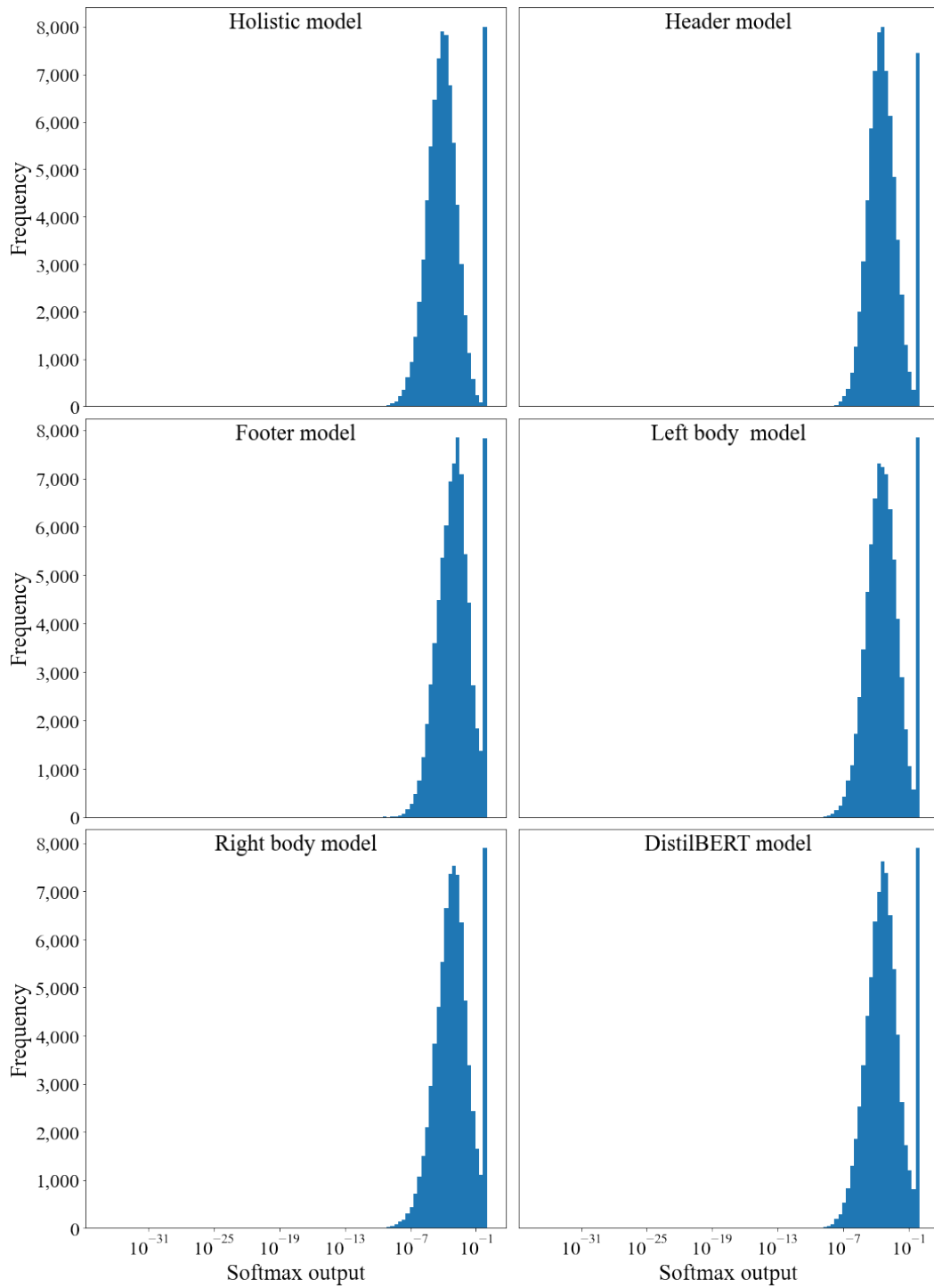


Figure 2: The softmax output distribution from the base classifiers on the Tobacco3482 training data.

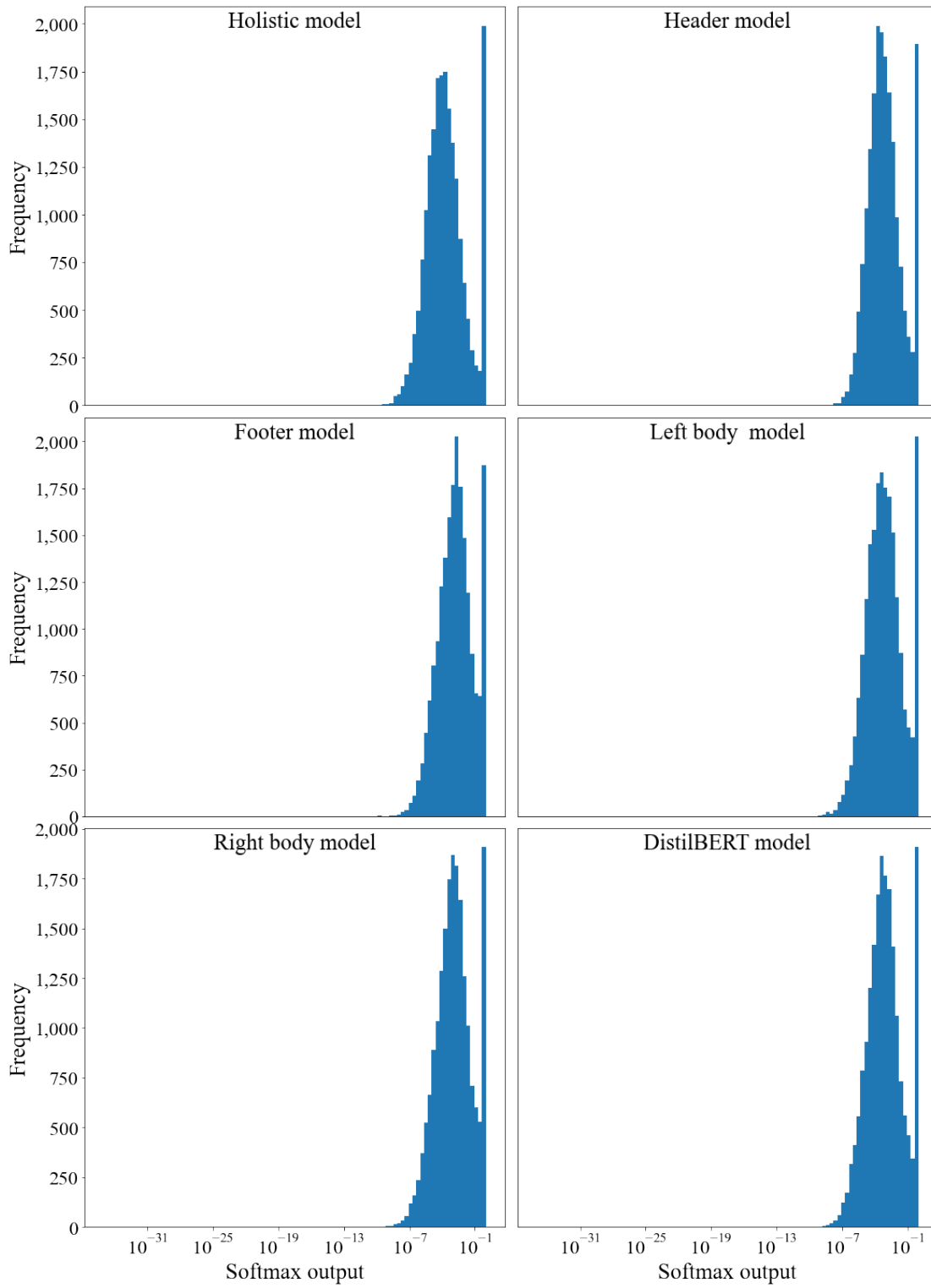


Figure 3: The softmax output distribution from the base classifiers on the Tobacco3482 validation data.

C. RVL-CDIP results

The results for RVL-CDIP include the confusion matrices of all 6 base models, as well as the confusion matrix of the overall best model. Tables 3 to 9 display the confusion matrices along with the accuracy for each class. To fit the confusion matrices on a page, the classes are encoded as follows: 0 = Letter, 1 = Form, 2 = Email, 3 = Handwritten, 4 = Advertisement, 5 = Scientific report, 6 = Scientific publication, 7 = Specification, 8 = File folder, 9 = News article, 10 = Budget, 11 = Invoice, 12 = Presentation, 13 = Questionnaire, 14 = Resume, 15 = Memo.

Predicted class	Actual class															
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	2,248	47	4	7	2	3	0	13	1	3	1	20	5	14	2	36
1	26	2,175	4	9	7	47	3	33	3	2	18	59	5	33	4	25
2	19	1	2,492	1	0	2	0	4	0	3	1	0	2	1	0	1
3	41	28	0	2,457	18	11	1	13	2	0	4	16	0	31	0	3
4	9	17	0	16	2,394	9	9	4	6	31	11	5	24	29	2	5
5	6	40	0	1	2	2,226	61	24	11	12	34	2	100	18	9	10
6	5	2	0	0	5	29	2,407	6	2	25	1	0	4	2	3	0
7	7	24	1	6	0	11	2	2,345	0	0	2	0	0	3	1	7
8	10	10	0	0	10	22	22	2	2,460	11	17	5	38	9	1	14
9	6	5	4	1	38	9	43	1	2	2,315	12	0	43	3	6	2
10	4	22	1	0	2	23	0	4	4	2	2,328	20	38	8	2	16
11	10	65	0	0	0	8	1	5	0	1	25	2,343	0	2	0	5
12	5	10	2	1	17	67	17	3	27	51	37	2	2,195	22	11	10
13	18	41	3	33	17	9	1	12	6	1	6	3	17	2,253	4	2
14	2	1	0	0	1	8	3	0	2	2	0	0	10	1	2,490	6
15	48	18	5	0	2	14	1	3	1	4	8	2	8	6	2	2,350
Accuracy (%)	91.23	86.79	99.05	97.04	95.19	89.11	93.62	94.86	97.35	93.99	92.93	94.59	88.19	92.53	98.15	94.30

Table 3: RVL-CDIP test set results from the best model (meta-classifier with only the image models as base classifiers). Accuracy in % and rounded to two decimal places. Overall accuracy: 93.70%.

Predicted class	Actual class															
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	2,223	45	10	21	8	10	0	14	1	5	2	17	13	22	4	53
1	43	2,105	5	23	8	76	5	32	4	2	22	63	8	57	6	18
2	19	3	2,487	3	1	0	0	5	1	1	0	1	2	5	0	2
3	24	16	0	2,408	13	0	1	9	2	1	3	12	0	27	0	0
4	9	15	0	18	2,341	5	14	7	7	46	10	6	20	21	3	5
5	12	38	1	0	2	2,025	66	31	10	17	36	1	116	19	20	7
6	6	1	0	0	8	44	2,386	3	9	22	2	1	9	6	8	1
7	11	42	1	12	3	24	8	2,320	0	0	6	2	6	13	2	8
8	4	8	0	0	11	18	13	0	2,430	3	10	5	33	9	1	10
9	8	5	1	1	37	26	35	2	2	2,267	8	3	36	2	7	3
10	8	29	1	0	8	46	3	9	12	6	2,300	37	51	16	4	12
11	24	107	0	8	6	13	0	9	2	2	22	2,312	2	8	1	4
12	19	11	3	1	40	125	28	6	36	69	60	5	2,134	30	18	19
13	24	51	4	37	25	38	6	20	10	9	14	3	32	2,191	8	5
14	3	3	0	0	2	29	5	3	1	4	1	2	21	4	2,453	4
15	27	27	3	0	2	19	1	2	0	9	9	7	6	5	2	2,341
Accuracy (%)	90.22	84.00	98.85	95.10	93.08	81.06	92.80	93.85	96.16	92.04	91.82	93.34	85.74	89.98	96.69	93.94

Table 4: RVL-CDIP test set results from the holistic model. Accuracy in % and rounded to two decimal places. Overall accuracy: 91.81%.

Predicted class	Actual class															
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	2,124	62	12	21	19	11	5	18	6	17	15	77	20	29	9	75
1	33	2,007	6	17	16	53	3	34	3	13	39	83	22	63	7	24
2	19	6	2,456	3	1	1	0	5	4	5	0	3	0	1	1	4
3	40	26	3	2,372	21	4	0	11	4	0	2	15	1	41	1	3
4	19	26	3	25	2,209	15	32	10	20	78	16	10	37	39	6	4
5	22	63	3	7	12	2,029	88	22	21	37	78	16	152	35	21	15
6	9	6	0	0	16	49	2,286	6	4	61	6	2	17	9	8	1
7	5	44	1	9	5	23	3	2,288	2	1	17	17	16	17	2	11
8	16	38	16	10	63	43	31	5	2,383	22	52	35	89	28	6	22
9	11	14	4	2	57	17	73	9	3	2,135	19	7	48	9	10	7
10	11	32	2	4	9	36	6	12	9	6	2,104	43	53	27	4	10
11	47	71	2	6	9	26	2	7	6	12	46	2,124	14	10	2	6
12	32	27	2	8	49	135	29	13	42	65	77	20	1,954	57	23	12
13	28	48	1	45	26	21	6	19	9	3	18	10	35	2,049	2	4
14	5	8	1	1	1	13	4	5	3	5	1	2	14	6	2,428	3
15	43	28	4	2	2	22	3	8	8	3	15	13	17	15	7	2,291
Accuracy (%)	86.20	80.09	97.62	93.68	87.83	81.22	88.91	92.56	94.30	86.68	83.99	85.75	78.51	84.15	95.70	91.93

Table 5: RVL-CDIP test set results from the header model. Accuracy in % and rounded to two decimal places. Overall accuracy: 88.10%.

Predicted class	Actual class															
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	1,665	89	13	39	20	33	5	37	11	10	15	37	35	64	14	141
1	103	1,859	10	14	13	92	15	48	17	25	43	83	42	86	10	74
2	24	15	2,435	1	51	10	2	3	21	14	9	14	22	18	8	14
3	56	40	2	2,380	41	4	2	31	7	0	2	17	0	56	2	1
4	18	17	4	24	2,176	12	20	6	16	46	9	10	26	46	3	14
5	50	73	8	1	14	1,846	67	30	43	34	72	30	140	44	33	127
6	18	10	0	1	12	31	2,307	3	11	26	3	5	9	4	22	8
7	51	41	3	22	6	13	2	2,237	1	0	3	6	7	26	4	24
8	21	18	4	1	25	27	19	4	2,250	16	26	24	48	11	9	31
9	24	29	3	1	46	45	56	3	15	2,109	39	13	71	15	20	48
10	22	27	1	0	8	36	3	5	10	11	2,026	32	35	20	4	50
11	59	124	4	8	23	31	17	16	34	30	52	2,118	25	35	9	75
12	62	54	4	0	40	146	21	15	32	78	79	23	1,868	53	36	164
13	48	42	3	38	19	15	7	15	7	8	20	21	29	1,893	4	38
14	48	10	8	1	3	44	20	4	11	17	5	9	30	4	2,337	37
15	195	58	14	1	18	113	8	15	41	39	102	35	102	60	22	1,646
Accuracy (%)	67.57	74.18	96.78	94.00	86.52	73.90	89.73	90.49	89.04	85.63	80.88	85.51	75.05	77.74	92.12	66.05

Table 6: RVL-CDIP test set results from the footer model. Accuracy in % and rounded to two decimal places. Overall accuracy: 82.88%.

Predicted class	Actual class															
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	2,072	57	16	15	5	18	1	12	2	6	8	37	16	27	6	109
1	55	2,009	11	21	15	69	8	43	7	6	39	97	12	66	1	40
2	40	15	2,173	5	7	2	0	9	8	9	6	3	8	13	4	10
3	35	28	3	2,335	24	19	2	15	11	3	8	11	4	30	0	10
4	11	11	3	27	2,233	17	18	9	13	77	10	7	27	29	2	9
5	18	60	8	7	8	1,903	65	44	20	27	76	14	127	24	23	59
6	7	6	3	0	19	66	2,327	3	6	91	5	0	19	7	11	1
7	7	42	6	11	2	21	5	2,236	1	1	17	8	8	10	1	7
8	16	20	240	29	47	35	34	7	2,369	29	48	22	56	19	9	26
9	13	6	6	3	59	15	74	4	7	2,093	14	2	49	4	13	5
10	5	44	3	14	11	70	2	26	14	3	2,094	65	46	30	13	18
11	19	98	2	9	4	15	4	11	4	4	63	2,181	3	11	1	8
12	34	21	22	13	57	127	18	7	43	85	71	5	2,021	40	25	39
13	30	52	6	39	18	36	7	20	20	8	13	5	40	2,093	5	15
14	8	7	2	2	4	17	5	5	0	6	11	9	14	8	2,413	6
15	94	30	12	2	2	68	1	21	2	15	22	11	39	24	10	2,130
Accuracy (%)	84.09	80.17	86.37	92.22	88.79	76.18	90.51	90.45	93.75	84.98	83.59	88.05	81.20	85.95	95.11	85.47

Table 7: RVL-CDIP test set results from the left body model. Accuracy in % and rounded to two decimal places. Overall accuracy: 86.71%.

Predicted class	Actual class															
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	1,897	79	28	30	20	51	7	32	2	23	11	23	37	43	13	264
1	58	1,878	13	40	10	82	7	51	10	10	37	63	14	72	3	44
2	42	19	2,275	6	7	15	11	7	108	24	29	4	37	17	12	34
3	39	44	8	2,317	18	17	1	10	23	1	9	19	5	35	1	8
4	10	18	1	29	2,222	9	20	8	17	62	9	5	33	32	3	8
5	48	82	31	19	17	1,738	56	49	23	20	49	15	155	29	27	104
6	15	10	2	2	29	74	2,337	5	12	120	9	2	27	6	12	7
7	11	62	3	1	4	34	4	2,210	2	3	17	14	10	18	3	9
8	14	20	40	10	22	37	22	5	2,212	20	49	19	50	19	5	12
9	14	10	13	2	70	19	49	5	13	2,051	15	3	58	8	13	12
10	10	40	16	13	10	65	7	20	19	3	2,041	59	47	22	2	18
11	17	97	3	6	6	17	0	13	6	3	106	2,229	4	8	0	12
12	32	14	25	18	48	143	22	13	58	68	71	5	1,860	48	18	41
13	38	71	11	35	20	33	7	12	18	8	18	3	42	2,031	5	20
14	17	15	10	0	6	40	14	6	2	15	7	2	32	4	2,397	16
15	202	47	37	4	6	124	7	26	2	32	28	12	78	43	23	1,883
Accuracy (%)	76.99	74.94	90.42	91.51	88.35	69.58	90.90	89.40	87.53	83.27	81.48	89.99	74.73	83.41	94.48	75.56

Table 8: RVL-CDIP test set results from the right body model. Accuracy in % and rounded to two decimal places. Overall accuracy: 83.95%.

Predicted class	Actual class															
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	2,151	43	22	17	10	8	6	8	1	9	3	20	7	18	4	77
1	31	1,983	10	29	21	67	7	32	3	5	33	45	16	33	0	35
2	24	5	2,401	3	3	5	2	1	2	5	3	1	2	5	0	4
3	30	42	5	1,827	102	34	7	23	50	7	38	24	4	48	1	9
4	20	24	3	92	1,660	23	14	3	107	72	32	10	40	34	2	12
5	8	51	10	11	5	1,967	67	33	17	9	14	3	71	12	7	12
6	4	12	1	8	11	83	2,245	4	7	88	1	1	14	5	5	3
7	6	49	2	12	7	27	6	2,298	3	0	5	6	3	8	0	13
8	38	54	7	408	490	81	52	24	2,259	50	118	57	110	80	17	54
9	22	15	11	14	112	29	137	3	9	2,099	35	5	97	21	5	12
10	7	39	4	16	18	39	2	11	12	14	2,061	48	48	7	3	10
11	21	101	5	23	9	23	1	15	7	10	99	2,244	5	11	0	11
12	11	22	17	8	36	82	14	6	34	79	45	4	2,029	13	12	21
13	19	39	7	59	29	14	4	8	6	10	13	4	32	2,132	2	7
14	1	1	0	1	1	3	6	0	7	3	1	1	4	0	2,478	0
15	71	26	11	4	1	13	1	3	3	3	4	4	7	8	1	2,212
Accuracy (%)	87.30	79.13	95.43	72.16	66.00	78.74	87.32	92.96	89.39	85.22	82.28	90.59	81.52	87.56	97.67	88.76

Table 9: RVL-CDIP test set results from the DistilBERT model. Accuracy in % and rounded to two decimal places. Overall accuracy: 85.12%.

D. System and Libraries and used

The training is performed on a g4dn.2xlarge aws instance. This instance is equipped with a NVIDIA T4 GPU, 8 virtual CPUs and 32 GiB of RAM. The CPUs are aws-custom Intel Cascade Lake CPUs, which are optimized for machine learning inference and small scale training. However, text preprocessing with the tesseract OCR system is done locally on an Intel i7-10750H CPU machine, with 24GB of RAM.

The implementation is done in the Python programming language, version 3.8. Table 10 depicts the libraries,¹ which are used in this work, along with their version.

Library	Version
keras-tuner	1.0.4
numpy	1.20.3
matplotlib	3.4.1
opencv-python	4.5.3.56
pandas	1.3.3
Pillow	8.3.2
pytesseract	0.3.7
scikit-learn	1.0.1
tensorboard	2.4.0
tensorflow	2.4.1
tesseract	4.1.1
tqdm	4.62.3
transformers	4.11.2

Table 10: Libraries used in this work and their version.

¹excluding the python standard library, which contains modules such as *statistics* that are used in this work

Bibliography

- [1] F. W. Lancaster, *Toward Paperless Information Systems*. Academic Press, Inc., 1978, p. 179, ISBN: 0124360505. DOI: 10.5555/601095.
- [2] R. Smith, “The Environmental Sustainability of Paper”, *Graduate Studies Journal of Organizational Dynamics*, vol. 1, no. 1, 2011. [Online]. Available: <http://repository.upenn.edu/gsjod/vol1/>.
- [3] N. Audebert, C. Herold, K. Slimani, and C. Vidal, “Multimodal deep networks for text and image-based document classification”, *Communications in Computer and Information Science*, vol. 1167 CCIS, pp. 427–443, 2020, ISSN: 18650937. DOI: 10.1007/978-3-030-43823-4_{_}35.
- [4] O. B. P. L. Amelec Viloría Noel Varela, H. H. P. Nataly Orellano Llinás Yasmin Flores, C. V. M. Marín-González, and Freddy, “International Conference on Communication, Computing and Electronics Systems”, in *Lecture Notes in Electrical Engineering*, vol. 637, 2020, ch. Classifica, pp. 213–220, ISBN: 9789811526114. DOI: 10.1007/978-981-15-2612-1_{_}26.
- [5] L. O’Gorman and R. Kasturi, “Executive Briefing: Document Image Analysis.”, IEEE Computer Society Press, 1997, ISBN: 081867802X.
- [6] N. Chen and D. Blostein, “A survey of document image classification: Problem statement, classifier architecture and performance evaluation”, *International Journal on Document Analysis and Recognition*, vol. 10, no. 1, pp. 1–16, 2007, ISSN: 14332833. DOI: 10.1007/s10032-006-0020-2.
- [7] R. Jain and C. Wigington, “Multimodal document image classification”, *Proceedings of the International Conference on Document Analysis and Recognition, ICDAR*, vol. 3, pp. 71–77, 2019, ISSN: 15205363. DOI: 10.1109/ICDAR.2019.00021.
- [8] Y. Xu, M. Li, L. Cui, S. Huang, F. Wei, and M. Zhou, “LayoutLM: Pre-training of Text and Layout for Document Image Understanding”, *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1192–1200, 2020. DOI: 10.1145/3394486.3403172.

- [9] A. Das, S. Roy, U. Bhattacharya, and S. K. Parui, “Document Image Classification with Intra-Domain Transfer Learning and Stacked Generalization of Deep Convolutional Neural Networks”, *Proceedings - International Conference on Pattern Recognition*, vol. 2018-Augus, pp. 3180–3185, 2018, ISSN: 10514651. DOI: 10.1109/ICPR.2018.8545630.
- [10] J. Ferrando, J. L. Domínguez, J. Torres, R. García, D. García, D. Garrido, J. Cortada, and M. Valero, “Improving accuracy and speeding up document image classification through parallel systems”, *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 12138 LNCS, pp. 387–400, 2020, ISSN: 16113349. DOI: 10.1007/978-3-030-50417-5{_}29.
- [11] A. W. Harley, A. Ufkes, and K. G. Derpanis, “Evaluation of deep convolutional nets for document image classification and retrieval”, *Proceedings of the International Conference on Document Analysis and Recognition, ICDAR*, vol. 2015-Novem, pp. 991–995, 2015, ISSN: 15205363. DOI: 10.1109/ICDAR.2015.7333910.
- [12] F. Cesarini, M. Lastrì, S. Marinai, and G. Soda, “Encoding of modified X-Y trees for document classification”, *Proceedings of the International Conference on Document Analysis and Recognition, ICDAR*, vol. 2001-Janua, pp. 1131–1136, 2001, ISSN: 15205363. DOI: 10.1109/ICDAR.2001.953962.
- [13] C. Tensmeyer and T. Martinez, “Analysis of Convolutional Neural Networks for Document Image Classification”, *Proceedings of the International Conference on Document Analysis and Recognition, ICDAR*, vol. 1, pp. 388–393, 2017, ISSN: 15205363. DOI: 10.1109/ICDAR.2017.71.
- [14] A. Kolsch, M. Z. Afzal, M. Ebbecke, and M. Liwicki, “Real-Time Document Image Classification Using Deep CNN and Extreme Learning Machines”, *Proceedings of the International Conference on Document Analysis and Recognition, ICDAR*, vol. 1, pp. 1318–1323, 2018, ISSN: 15205363. DOI: 10.1109/ICDAR.2017.217.
- [15] M. Z. Afzal, A. Kolsch, S. Ahmed, and M. Liwicki, “Cutting the Error by Half: Investigation of Very Deep CNN and Advanced Training Strategies for Document Image Classification”, *Proceedings of the International Conference on Document Analysis and Recognition, ICDAR*, vol. 1, pp. 883–888, 2017, ISSN: 15205363. DOI: 10.1109/ICDAR.2017.149.
- [16] Y. Xu, Y. Xu, T. Lv, L. Cui, F. Wei, G. Wang, Y. Lu, D. Florencio, C. Zhang, W. Che, M. Zhang, and L. Zhou, “LayoutLMv2 : Multi-modal Pre-training for Visually-rich Document Understanding”, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics (ACL) 2021*, pp. 2579–2591, 2021.
- [17] D. Soekhoe, P. van der Putten, and A. Plaat, “On the Impact of Data Set Size in Transfer Learning Using Deep Neural Networks”, in *IDA*, 2016, ISBN: 978-3-319-46348-3. DOI: 10.1007/978-3-319-46349-0. [Online]. Available: <http://link.springer.com/10.1007/978-3-319-46349-0>.

- [18] T. Dauphinee, N. Patel, and M. Rashidi, “Modular multimodal architecture for document classification.”, *arXiv*, 2019.
- [19] D. Lewis, G. Agam, S. Argamon, O. Frieder, D. Grossman, and J. Heard, “Building a test collection for complex document information processing”, *Proceedings of the Twenty-Ninth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, vol. 2006, pp. 665–666, 2006. DOI: 10.1145/1148170.1148307.
- [20] H. Schmidt, K. Butter, and C. Rider, “Building Digital Tobacco Industry Document Libraries at the University of California, San Francisco Library/Center for Knowledge Management”, *D Lib Mag.*, vol. 8, no. 9, 2002. DOI: 10.1045/september2002-schmidt. [Online]. Available: <https://doi.org/10.1045/september2002-schmidt>.
- [21] J. Kumar and D. Doermann, “Unsupervised classification of structurally similar document images”, *Proceedings of the International Conference on Document Analysis and Recognition, ICDAR*, pp. 1225–1229, 2013, ISSN: 15205363. DOI: 10.1109/ICDAR.2013.248.
- [22] S. Abuelwafa, M. Pedersoli, and M. Cheriet, “Unsupervised Exemplar-Based Learning for Improved Document Image Classification”, *IEEE Access*, vol. 7, pp. 133 738–133 748, 2019, ISSN: 21693536. DOI: 10.1109/ACCESS.2019.2940884.
- [23] S. L. Taylor, M. Lipshutz, and R. W. Nilson, “Classification and functional decomposition of business documents”, *ICDAR*, vol. 2, pp. 563–566, 1995.
- [24] S. L. Taylor, M. Lipshutz, and C. Weir, “Document structure interpretation by integrating multiple knowledge sources”, *Symposium on Document Analysis and Information Retrieval*, pp. 58–76, 1992.
- [25] Y. Byun, Y. Lee, Y. W. Choi, G. Kim, and S. Yoon, “Form classification using DP matching”, *Proceedings of the ACM Symposium on Applied Computing*, vol. 1, pp. 1–4, 2000. DOI: 10.1145/335603.335611.
- [26] T. Kochi and T. Saitoh, “User-defined template for identifying document type and extracting information from documents”, *Proceedings of the International Conference on Document Analysis and Recognition, ICDAR*, pp. 127–130, 1999, ISSN: 15205363. DOI: 10.1109/ICDAR.1999.791741.
- [27] C. K. Shin and D. S. Doermann, “Classification of document page images based on visual similarity of layout structures”, *Document Recognition and Retrieval VII*, vol. 3967, no. January, pp. 182–190, 1999. DOI: 10.1117/12.373493.
- [28] S. K. Murthy, S. Kasif, and S. Salzberg, “A System for Induction of Oblique Decision Trees”, *Journal of Artificial Intelligence Research*, vol. 2, pp. 1–32, 1994, ISSN: 1076-9757. DOI: 10.1613/jair.63.
- [29] C. Shin, D. Doermann, and A. Rosenfeld, “Classification of document pages using structure-based features”, *International Journal on Document Analysis and Recognition*, vol. 3, no. 4, pp. 232–247, 2001, ISSN: 14332825. DOI: 10.1007/PL00013566.

- [30] J. Hu, R. Kashi, and G. Wilfong, “Comparison and classification of documents based on layout similarity”, *Information Retrieval*, vol. 2, no. 2-3, pp. 227–243, 2000, ISSN: 13864564. DOI: 10.1023/a:1009910911387.
- [31] A. D. Bagdanov and M. Worring, “Fine-grained document genre classification using first order random graphs”, *Proceedings of the International Conference on Document Analysis and Recognition, ICDAR*, vol. 2001-Janua, pp. 79–83, 2001, ISSN: 15205363. DOI: 10.1109/ICDAR.2001.953759.
- [32] K. V. U. Reddy and V. Govindaraju, “Form classification”, *Document Recognition and Retrieval XV*, vol. 6815, no. January 2008, 68150Y-68150Y-6, 2008. DOI: 10.1117/12.766737.
- [33] J. Kumar, R. Prasad, H. Cao, W. Abd-Almageed, D. Doermann, and P. Natarajan, “Shape codebook based handwritten and machine printed text zone extraction”, *Document Recognition and Retrieval XVIII*, vol. 7874, no. May 2014, p. 787406, 2011, ISSN: 0277786X. DOI: 10.1117/12.876725.
- [34] V. Ferrari, L. Fevrier, F. Jurie, and C. Schmid, “Groups of Adjacent Contour Segments for Object Detection”, *Slides*, vol. 30, no. 1, pp. 36–51, 2003.
- [35] M. Agrawal and D. Doermann, “Voronoi++: A dynamic page segmentation approach based on voronoi and Docstrum features”, *Proceedings of the International Conference on Document Analysis and Recognition, ICDAR*, pp. 1011–1015, 2009, ISSN: 15205363. DOI: 10.1109/ICDAR.2009.270.
- [36] J. Kumar, P. Ye, and D. Doermann, “Learning document structure for retrieval and classification”, *Proceedings - International Conference on Pattern Recognition*, no. Icp, pp. 1558–1561, 2012, ISSN: 10514651.
- [37] J. Kumar, P. Ye, and D. Doermann, “Structural similarity for document image classification and retrieval”, *Pattern Recognition Letters*, vol. 43, no. 1, pp. 119–126, 2014, ISSN: 01678655. DOI: 10.1016/j.patrec.2013.10.030.
- [38] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks”, *Neural Information Processing Systems*, vol. 25, 2012. DOI: 10.1145/3383972.3383975.
- [39] H. Hotelling, “Analysis of a complex of statistical variables into principal components.”, *Journal of Educational Psychology*, vol. 24, pp. 498–520, 1933.
- [40] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition”, *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, pp. 1–14, 2015.
- [41] K. Weiss, T. M. Khoshgoftaar, and D. D. Wang, *A survey of transfer learning*, 1. Springer International Publishing, 2016, vol. 3, ISBN: 4053701600. DOI: 10.1186/s40537-016-0043-6.
- [42] A. Sharif Razavian, H. Azizpur, J. Sullivan, and S. Carlsson, “Cnn features off-the-shelf: an astounding baseline for recognition”, *Proc. CVPR Workshops*, pp. 806–813, 2014, ISSN: 0277786X. DOI: 10.1117/12.827526.

- [43] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, “ImageNet Large Scale Visual Recognition Challenge”, *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015, ISSN: 15731405. DOI: 10.1007/s11263-015-0816-y. [Online]. Available: <http://dx.doi.org/10.1007/s11263-015-0816-y>.
- [44] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, “Going deeper with convolutions”, *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 07-12-June, pp. 1–9, 2015, ISSN: 10636919. DOI: 10.1109/CVPR.2015.7298594.
- [45] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition”, *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2016-Decem, pp. 770–778, 2016, ISSN: 10636919. DOI: 10.1109/CVPR.2016.90.
- [46] R. Sarkhel and A. Nandi, “Deterministic routing between layout abstractions for multi-scale classification of visually rich documents”, *IJCAI International Joint Conference on Artificial Intelligence*, vol. 2019-Augus, pp. 3360–3366, 2019, ISSN: 10450823. DOI: 10.24963/ijcai.2019/466.
- [47] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. C. Chen, “MobileNetV2: Inverted Residuals and Linear Bottlenecks”, *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 4510–4520, 2018, ISSN: 10636919. DOI: 10.1109/CVPR.2018.00474.
- [48] A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov, “Bag of tricks for efficient text classification”, *15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017 - Proceedings of Conference*, vol. 2, pp. 427–431, 2017. DOI: 10.18653/v1/e17-2068.
- [49] M. Tan and Q. V. Le, “EfficientNet: Rethinking model scaling for convolutional neural networks”, *36th International Conference on Machine Learning, ICML 2019*, vol. 2019-June, pp. 10 691–10 700, 2019.
- [50] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of deep bidirectional transformers for language understanding”, *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, vol. 1, no. Mlm, pp. 4171–4186, 2019.
- [51] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, J. Llion, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need”, in *Advances in Neural Information Processing Systems*, 2017, pp. 6000–6010. DOI: 10.1109/2943.974352.

- [52] M. N. Asim, M. U. G. Khan, M. I. Malik, K. Razzaque, A. Dengel, and S. Ahmed, “Two stream deep network for document image classification”, *Proceedings of the International Conference on Document Analysis and Recognition, ICDAR*, pp. 1410–1416, 2019, ISSN: 15205363. DOI: 10.1109/ICDAR.2019.00227.
- [53] R. Smith, “An overview of the tesseract OCR engine”, *Proceedings of the International Conference on Document Analysis and Recognition, ICDAR*, vol. 2, pp. 629–633, 2007, ISSN: 15205363. DOI: 10.1109/ICDAR.2007.4376991.
- [54] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory”, *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997, ISSN: 08997667. DOI: 10.1162/neco.1997.9.8.1735.
- [55] B. Alan C., *The Essential Guide to Image Processing*. Academic Press, 2009, vol. 2nd ed, ISBN: 9780123744579. [Online]. Available: <http://search-ebscohost-com.uaccess.univie.ac.at/login.aspx?direct=true&db=nlebk&AN=249002&site=ehost-live>.
- [56] A. Hamdi, A. Jean-Caurant, N. Sidère, M. Coustaty, and A. Doucet, “Assessing and Minimizing the Impact of OCR Quality on Named Entity Recognition”, *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 12246 LNCS, pp. 87–101, 2020, ISSN: 16113349. DOI: 10.1007/978-3-030-54956-5{_}7.
- [57] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient estimation of word representations in vector space”, *1st International Conference on Learning Representations, ICLR 2013 - Workshop Track Proceedings*, pp. 1–12, 2013.
- [58] L. Liu, Z. Wang, T. Qiu, Q. Chen, Y. Lu, and C. Y. Suen, “Document image classification: Progress over two decades”, *Neurocomputing*, vol. 453, pp. 223–240, 2021, ISSN: 18728286. DOI: 10.1016/j.neucom.2021.04.114. [Online]. Available: <https://doi.org/10.1016/j.neucom.2021.04.114>.
- [59] L. Kang, J. Kumar, P. Ye, Y. Li, and D. Doermann, “Convolutional neural networks for document image classification”, *Proceedings - International Conference on Pattern Recognition*, pp. 3168–3172, 2014, ISSN: 10514651. DOI: 10.1109/ICPR.2014.546.
- [60] D. H. Hubel and T. Wiesel, “Receptive fields, binocular interaction and functional architecture in the cat’s visual cortex”, *Most*, pp. 106–154, 1962.
- [61] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition”, *Proceedings of the IEEE*, vol. 86, no. 11, p. 46, 1998. [Online]. Available: <http://ieeexplore.ieee.org/document/726791/#full-text-section>.
- [62] J. Schmidhuber, “Deep Learning in neural networks: An overview”, *Neural Networks*, vol. 61, pp. 85–117, 2015, ISSN: 18792782. DOI: 10.1016/j.neunet.2014.09.003.

- [63] Z. Li, F. Liu, W. Yang, S. Peng, and J. Zhou, “A Survey of Convolutional Neural Networks: Analysis, Applications, and Prospects”, *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–21, 2021, ISSN: 2162-237X. DOI: 10.1109/tnnls.2021.3084827.
- [64] H. Pham, Z. Dai, Q. Xie, M.-T. Luong, and Q. V. Le, “Meta Pseudo Labels”, pp. 1–12, 2020. [Online]. Available: <http://arxiv.org/abs/2003.10580>.
- [65] G. Csurka, D. Larlus, A. Gordo, and J. Almazan, “What is the right way to represent document images?”, pp. 1–28, 2016. [Online]. Available: <http://arxiv.org/abs/1603.01076>.
- [66] M. P. S. Parsania and D. P. V. Virparia, “A Comparative Analysis of Image Interpolation Algorithms”, *IJARCCCE*, vol. 5, pp. 29–34, 2016. DOI: 10.17148/ijarccce.2016.5107.
- [67] P. P. S. Parsania and D. P. V. Virparia, “A Review: Image Interpolation Techniques for Image Scaling”, *International Journal of Innovative Research in Computer and Communication Engineering*, vol. 02, no. 12, pp. 7409–7414, 2015, ISSN: 23209798. DOI: 10.15680/ijircce.2014.0212024.
- [68] L. Huang, J. Qin, Y. Zhou, F. Zhu, L. Liu, and L. Shao, “Normalization Techniques in Training DNNs: Methodology, Analysis and Application”, pp. 1–20, 2020. [Online]. Available: <http://arxiv.org/abs/2009.12836>.
- [69] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift”, *32nd International Conference on Machine Learning, ICML 2015*, vol. 1, pp. 448–456, 2015.
- [70] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: A Simple Way to Prevent Neural Networks from Overfitting”, *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, 2014, ISSN: 03702693. DOI: 10.1016/0370-2693(93)90272-J.
- [71] Q. Xie, M. T. Luong, E. Hovy, and Q. V. Le, “Self-training with noisy student improves imagenet classification”, *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 10 684–10 695, 2020, ISSN: 10636919. DOI: 10.1109/CVPR42600.2020.01070.
- [72] J. Feng and S. Lu, “Performance Analysis of Various Activation Functions in Artificial Neural Networks”, *Journal of Physics: Conference Series*, vol. 1237, no. 2, 2019, ISSN: 17426596. DOI: 10.1088/1742-6596/1237/2/022030.
- [73] T. Lin, Y. Wang, X. Liu, and X. Qiu, “A Survey of Transformers”, vol. 1, no. 1, 2021. [Online]. Available: <http://arxiv.org/abs/2106.04554>.
- [74] J. L. Ba, J. R. Kiros, and G. E. Hinton, “Layer Normalization”, *arXiv*, vol. abs/1607.0, 2016. [Online]. Available: <http://arxiv.org/abs/1607.06450>.

- [75] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, “Language models are few-shot learners”, *Advances in Neural Information Processing Systems*, vol. 2020-Decem, 2020, ISSN: 10495258.
- [76] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, “DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter”, pp. 2–6, 2019, ISSN: 2331-8422. [Online]. Available: <http://arxiv.org/abs/1910.01108>.
- [77] C. Bucila, R. Caruana, and A. Niculescu-Mizil, “Model Compression”, *Kdd*, vol. 54, no. 1, pp. 1–9, 2006, ISSN: 0218-0014. [Online]. Available: <http://users.iems.northwestern.edu/~nocedal/PDFfiles/dss.pdf%0Ahttp://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.67.7952&rep=rep1&type=pdf%255Cnhttp://books.nips.cc/papers/files/nips15/AA03.pdf%250Ahttps://ieeexplore.ieee.org/document/85783>.
- [78] Y. Zhu, R. Kiros, R. Zemel, R. Salakhutdinov, R. Urtasun, A. Torralba, and S. Fidler, “Aligning books and movies: Towards story-like visual explanations by watching movies and reading books”, *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2015 Inter, pp. 19–27, 2015, ISSN: 15505499. DOI: 10.1109/ICCV.2015.11.
- [79] K. Lakhotia, E. Kharitonov, W.-N. Hsu, Y. Adi, A. Polyak, B. Bolte, T.-A. Nguyen, J. Copet, A. Baevski, A. Mohamed, and E. Dupoux, “Generative Spoken Language Modeling from Raw Audio”, 2021. [Online]. Available: <http://arxiv.org/abs/2102.01192>.
- [80] A. K. Uysal and S. Gunal, “The impact of preprocessing on text classification”, *Information Processing and Management*, vol. 50, no. 1, pp. 104–112, 2014, ISSN: 03064573. DOI: 10.1016/j.ipm.2013.08.006. [Online]. Available: <http://dx.doi.org/10.1016/j.ipm.2013.08.006>.
- [81] S. Vijayarani, J. Ilamathi, and S. Nithya, “Preprocessing Techniques for Text Mining - An Overview”, *International Journal of Computer Science & Communication Networks*, vol. 5, no. 1, pp. 7–16, 2015.
- [82] M. Toman, R. Tesar, and K. Jezek, “Influence of word normalization on text classification”, *Proceedings of InSciT*, pp. 354–358, 2006. [Online]. Available: <http://www.kiv.zcu.cz/research/groups/text/publications/inscit20060710.pdf>.
- [83] D. H. Wolpert, “Stacked generalization”, *Neural Networks*, vol. 5, no. 2, pp. 241–259, 1992, ISSN: 08936080. DOI: 10.1016/S0893-6080(05)80023-1.

- [84] K. M. Ting and I. H. Witten, “Issues in stacked generalization”, *Journal of Artificial Intelligence Research*, vol. 10, pp. 271–289, 1999, ISSN: 10769757. DOI: 10.1613/jair.594.
- [85] N. S. Keskar, J. Nocedal, P. T. P. Tang, D. Mudigere, and M. Smelyanskiy, “On large-batch training for deep learning: Generalization gap and sharp minima”, *5th International Conference on Learning Representations, ICLR 2017 - Conference Track Proceedings*, pp. 1–16, 2017.
- [86] I. Kandel and M. Castelli, “The effect of batch size on the generalizability of the convolutional neural networks on a histopathology dataset”, *ICT Express*, vol. 6, no. 4, pp. 312–315, 2020, ISSN: 24059595. DOI: 10.1016/j.icte.2020.04.010. [Online]. Available: <https://doi.org/10.1016/j.icte.2020.04.010>.
- [87] A. Cosma, M. Ghidoveanu, M. Panaitescu-Liess, and M. Popescu, “Self-supervised representation learning on document images”, *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 12116 LNCS, pp. 103–117, 2020, ISSN: 16113349. DOI: 10.1007/978-3-030-57058-3_{_}8.
- [88] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “Rethinking the Inception Architecture for Computer Vision”, *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2016-Decem, pp. 2818–2826, 2016, ISSN: 10636919. DOI: 10.1109/CVPR.2016.308.