University of Brasília

Institute of Exact Sciences
Department of Computer Science

Layout-Aware Zero-Shot Learning for Visual Document Matching

Lucas de Almeida Bandeira Macedo

Dissertation submitted as a partial requirement for
completion of the Master's Degree in Informatics

Advisor
Prof. Dr. Pedro Garcia Freitas

Co-advisor
Prof. Dr. Bruno Luiggi Macchiavello Espinoza

Brasília
2026

Ficha Catalográfica de Teses e Dissertações

Está página existe apenas para indicar onde a ficha catalográfica gerada para dissertações de mestrado e teses de doutorado defendidas na UnB. A Biblioteca Central é responsável pela ficha, mais informações nos sítios:

http://www.bce.unb.br
http://www.bce.unb.br/elaboracao-de-fichas-catalograficas-de-teses-e-dissertacoes

Esta página não deve ser inclusa na versão final do texto.

University of Brasília

Institute of Exact Sciences
Department of Computer Science

Layout-Aware Zero-Shot Learning for Visual Document Matching

Lucas de Almeida Bandeira Macedo

Dissertation submitted as a partial requirement for
completion of the Master's Degree in Informatics

Prof. Dr. Pedro Garcia Freitas (Advisor)
CiC/UnB

Prof. Dr. Patricia Medyna Lauritzen de Lucena Drumond
CEAD/UFPI

Prof. Dr. Luis Paulo Faina Garcia
CiC/UnB

Prof. Dr. Cláudia Nalon
Coordinator of the Graduate Program in Informatics

Brasília, 02 Fevereiro 2026

Dedication

To everyone who has crossed my path and made me who I am.

Acknowledgements

I thank my family, for taking care of me and lightening my spirit.
I thank my friends, for being there for me and calming my emotions.
I thank my study group, for thinking with me and alleviating my mind.

Disclaimer

During the preparation of this work, the author used ChatGPT™ and Claude Sonnet™ to review English usage and LaTeX syntax correction. After using these tools, the author reviewed and edited the content as needed and takes full responsibility for the content of the publication.

This work was carried out with support from the Coordination for the Improvement of Higher Education Personnel - Brazil (CAPES), through access to the Portal of Journals.

Aprendizado com Zero Exemplos Utilizando Layout para Correspondência Visual de Documentos

Resumo

Conformidade documental é um processo crucial em ambientes corporativos, certificando que toda forma de documento segue os devidos padrões regulatórios. Este processo assegura que todo documento nesse ambiente segue uma série de verificações que podem ser humanas ou automáticas. Uma dessas verificações é a classificação documental. Na maioria dos casos, a classificação documental segue uma fórmula simples e tradicional: o modelo é treinado com exemplos de todas as classes e, posteriormente, categoriza novos documentos nas classes aprendidas. O desafio surge quando as estruturas desses documentos são atualizadas. Nesse cenário, o classificador tradicional precisa ser retreinado, pois o novo padrão não está mapeado no modelo original. Uma solução promissora para esse problema é o paradigma de Aprendizado com Zero Instâncias (do inglês, *Zero-Shot Learning* – ZSL). Nele, o modelo é capaz de classificar tipos de documentos nunca vistos anteriormente, tornando-o mais resiliente a mudanças temporais. Dessa forma, este trabalho aborda a classificação de imagens de documentos utilizando ZSL e introduz a abordagem de Correspondência Visual de Documentos, que reformula a tarefa de classificação como um problema de pareamento. Para viabilizar esse modelo, este trabalho também introduz a base de dados *Layout-Aware Complex Document Information Processing* (LA-CDIP), especializada em classificação documental ZSL. Essa base oferece uma ampla diversidade de estruturas, permitindo que os modelos atinjam maior generalização. O dataset conta com 17.295 documentos distribuídos em 665 classes, onde cada classe apresenta uma estrutura distinta. Sua construção contou com o auxílio de modelos in-

termediários treinados no próprio conjunto, acelerando o processo de rotulação humana. Para demonstrar a eficácia do que é proposto, este trabalho provê uma avaliação de desempenho extensiva, considerando diversas arquiteturas de redes neurais e hiperparâmetros. Em cenários ZSL, o método proposto atingiu uma Taxa de Erro Igual (EER) inferior a 5% em testes de verificação com validação cruzada, provando-se uma solução eficiente para a classificação documental dinâmica.

Palavras-chave: Classificação de imagens de documentos, Aprendizado com Zero Instâncias, Correspondência Visual, Conformidade Documental

Abstract

Document compliance is a crucial process in company environments, certifying that every form of document follows regulatory standards. This process ensures that every document within this scenario follows a series of human and automatic verifications, and one of these steps is document classification. In most cases, document classification follows a simple and traditional classification: a model is trained with examples of every class and later categorizes documents into these classes. The challenge arises when these documents update their structure. In this case, the traditional classification model must be retrained, as the new document type is not mapped in the current model. Therefore, a possible solution to this problem is to train the model under the Zero-Shot Learning (ZSL) paradigm. This means that the model will be able to classify completely unseen documents, making it more resilient to changes. Therefore, this work tackles the Document Image Classification (DIC) problem in a ZSL paradigm, introducing the Visual Document Matching (VDM) approach, which frames the classification task as a matching problem. To achieve such a model, this work also presents the Layout-Aware Complex Document Information Processing (LA-CDIP) dataset, a dataset specialized in ZSL-DIC, providing ample document layout diversity to allow great generalization in models trained from scratch. This dataset has 17,295 documents, with 665 different classes, where every class is a different layout arrangement. This dataset is constructed with a learning-based framework, where a Deep Learning (DL) model is trained with an intermediary dataset, and in return, uses its acquired knowledge to accelerate the human-labeling process. To further stimulate research with this framework, this work also provides an extensive benchmark, taking into consideration multiple DL backbones and hyperparameters. In zero-shot scenarios, the proposed method with the new dataset achieves an Equal Error Rate (EER) below 5% in 1-vs-1 verification with cross-validation, proving to be an efficient solution to the ZSL-DIC problem.

Keywords: Document image classification, Zero-shot learning, Visual document matching, Document compliance

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Contextualization

Documents are at the core of corporate society, and they often hold immeasurable value, representing contracts, employment relationships, loans, and identities. Recently, as technology becomes more accessible, documents have become increasingly more digital, some never being printed on paper. These documents often have all important information already separated in the file's metadata and require no extra intelligence to classify or extract the information. However, many organizations still receive documents in physical form, for example, when dealing directly with individual customers, which makes full digitization impractical in many cases.

In such a scenario, organizations require a document understanding pipeline to validate document types and extract data for business decision-making (see Figure 1.1). Historically, this process relied on manual human labor. Following the adoption of Optical Character Recognition (OCR) engines [1, 2], tasks shifted from manual data entry to back-office analysis, where human operators verified compliance and classified documents. However, given the scale of modern data, manual processing has become computationally and economically inefficient. Today, with the current technologies, especially with the advancement of Neural Networks (NNs) and the cheapening of computing systems, automated pipelines have replaced many manual workflows in many companies. Given this context, this work addresses the challenge of automatic document classification.

1.2 Problem Description

Document Classification is defined by the categorization of documents into predefined classes [3]. Document Image Classification (DIC) is a specialized subset of this task where textual data is not natively available, as inputs often consist of photos or scans of

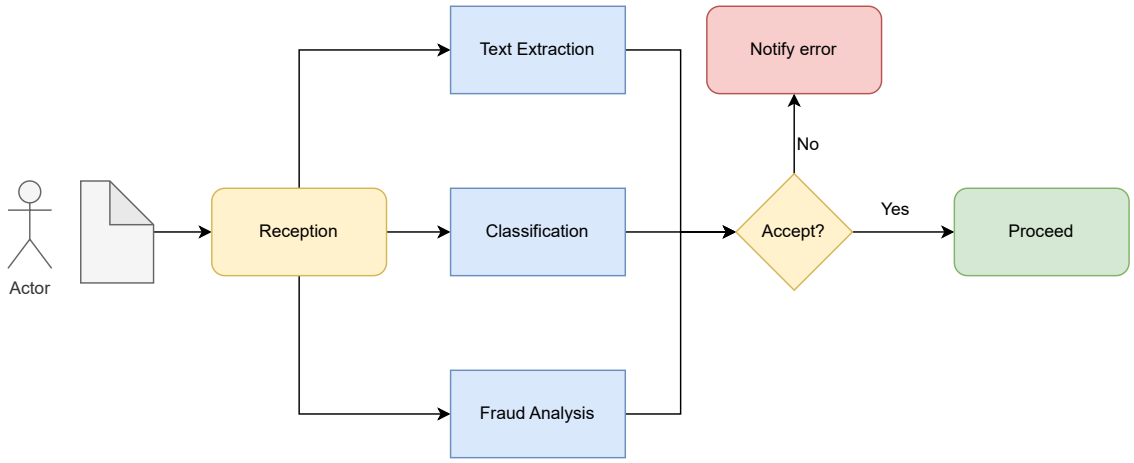


Figure 1.1: Simplified view of a document compliance pipeline. In this diagram, text extraction, document classification and fraud analysis are being used together to decide upon the acceptance of the received document.

physical documents. Consequently, textual analysis in DIC requires an initial extraction step using OCR algorithms. While Artificial Intelligence (AI) and Deep Learning (DL) can automate classification using image-only architectures, which reduces computational overhead by bypassing OCR, visual features alone may be insufficient for certain tasks. In such cases, models may utilize textual features or employ multimodal fusion techniques. This problem is already known and well studied. For instance, Bakkali et al. [4] achieved 97.70% accuracy on the Ryerson Vision Lab Complex Document Information Processing (RVL-CDIP) [5], a popular DIC dataset.

The problem arises when previously accepted documents change in form or when entirely new document classes, not previously considered, must be handled and categorized. In such cases, models are typically retrained, as traditional classification networks may struggle to generalize to new document layouts or entirely new classes. Traditional DL classification methods require a predefined set of labels and cannot output a label outside the original training set. Therefore, introducing new labels to the model means retraining the model, changing the output layer into a wider one with more options. Very often, this also means weeks or months of data engineering, labeling, and model training. The alternative is to use Zero-Shot Learning (ZSL) techniques, which allow models to generalize to classes that were not seen during training [6].

This work focuses on the ZSL on DIC, where the model must correctly classify a document image class that was not in the training set and therefore previously unseen by the model. ZSL techniques enable models to be considerably more flexible, as they are not limited by the classes used in training. Most ZSL systems work by semantically mapping embeddings into a feature space, where embeddings from the same class are all

clustered together, and different classes are placed far apart. This is called metric learning, where the model is optimized to map the training samples in regards to a given metric (e.g., Euclidean distance). Developing a robust ZSL system requires identifying a set of features that serve as discriminative representations, ensuring that document proximity in the feature space correlates strictly with class membership. Using a DL approach, this could also demand a dataset with a wide variety of classes, enough so that the model learns what makes two documents similar or different. These conditions allow ZSL models to group together documents, even if they were entirely novel document types.

However, not all classification datasets are suitable for effective ZSL applications. The scarcity of specialized datasets remains a primary obstacle in document-image ZSL, leading to a lack of standardized methodologies. As a consequence, another main challenge is the lack of a state-of-the-art methodology. Due to the challenge in achieving ZSL classification with the currently available datasets, researchers often employ diverse approaches that yield incomparable results [7], contrasting with the established frameworks and baselines found in traditional classification and information extraction. Recent approaches have used pretrained Large Language Models (LLMs) [8] to achieve ZSL by exploiting the models' previous knowledge. However, since it often requires a fine-tuning step, the potential cost-efficiency of this strategy may be reduced.

1.3 Research Objectives

The primary goal of this research is to develop a complete ZSL framework for DIC, that is, a classification model that does not require retraining every time a new class of document is introduced. To achieve this, this work pursues the following specific goals:

1. Design and curate a specialized dataset for zero-shot document image classification, ensuring proper class separation between training and testing sets;
2. Develop a framework using metric learning techniques to enable document classification based on visual similarity;
3. Evaluate and compare different NN backbones within the proposed framework to establish performance benchmarks;

1.3.1 Research Hypothesis

This work is grounded on the following hypothesis:

> *Document images that share similar visual layouts contain discriminative features that can be learned through metric learning, enabling accurate classification of pre-*

viously unseen document classes by measuring visual similarity, without requiring class-specific training.

This hypothesis is supported by the premise that document layouts within the same class exhibit consistent visual patterns (placement of text blocks, logos, signatures, etc.) that distinguish them from other classes, and that these patterns can be captured through vision models trained with a metric learning framework.

1.4 Research Contributions

The proposed framework measures the similarity between documents to determine if they belong to the same category, regardless of whether that class was present during training. Specifically, our Visual Document Matching (VDM) method formulates zero-shot classification as a binary matching problem, wherein the system determines if pairs of documents have visual correspondence (i.e., if they match visually or not).

This work makes several key contributions:

1. Introduces a novel document image dataset specifically designed for the task of document ZSL classification, enabling the development and evaluation of models in this domain.

2. Proposes the VDM approach, which is based on image similarity and leverages ZSL techniques to enable generalization across unseen document layouts without requiring additional intervention on the model itself.

3. Delivers a systematic evaluation of well-established backbones in the context of zero-shot document layout matching, offering valuable benchmarks for future research.

This research resulted in two publications. The first, titled *Towards Zero-Shot Document Image Classification* [9], was published at SIBGRAPI in Salvador, Brazil, and introduces the VDM solution and the method to use it with the novel LA-CDIP dataset. The second publication, titled *Visual Document Matching for Zero-Shot Document Classification* [10], was published at ICDAR in Wuhan, China, and extends the first work by comparing the results with the performance of Large Language Models. At the same time, the solution is currently implemented in one of the biggest banks in Brazil, having categorized in real time millions of documents into their respective classes, as soon as they are sent by the users.

1.5 About this work

This work is structured as follows: Chapter 2 establishes the theoretical foundation, while Chapter 3 reviews the state of the art in ZSL and visually-rich document understanding, highlighting existing methods and their limitations. Chapter 4 introduces the proposed VDM framework, detailing its architecture and components. Chapter 5 presents the data-labeling process and the experimental setup. Chapter 6 describes the experimental setup, datasets, evaluation metrics, and baseline comparisons, followed by a comprehensive analysis of the results. Finally, Chapter 7 concludes the work by summarizing the key contributions, discussing potential applications, and outlining directions for future research.

Chapter 2

Theoretical Foundation

This chapter is intended to introduce the core concepts, challenges, and progress related to the problem discussed in this work. Section 2.1 presents a brief summary of the traditional classification, a widely-used Machine Learning (ML) task of assigning an input (like an image, text, or data point) to one of a predefined set of categories or classes. Section 2.2 revisits the ZSL concept, an ML paradigm designed to overcome the limitations of traditional classification to allow a model to classify instances belonging to categories that have never been encountered during training. In Section 2.3, metric learning, a core enabling technique within the framework of ZSL, is presented. Next, the clustering techniques used to create the proposed Layout-Aware Complex Document Information Processing (LA-CDIP) dataset are discussed in Section 2.4. Finally, Section 2.5 reviews the main computer vision models that were used as basis to produce the embedding representations in our proposed ZSL-based VDM approach.

2.1 Traditional Classification

Traditional classification is the foundational ML paradigm where a model learns to map inputs, such as images or text, to one of a fixed, predetermined set of categories. A traditional classification model learns a direct mapping (a decision boundary) from the input features (X) into the predefined class labels (Y), i.e., Model$f : X \rightarrow Y_{seen}$. This model can only classify data into the classes it was explicitly trained on (seen classes). To create this mapping, a fundamental requirement is that the model must be trained on a large amount of labeled data to teach the model a direct mapping or a decision boundary between the input's features and the designated class labels. Consequently, the model is therefore limited by the class labels viewed in the training step. In other words, if a new class is introduced after the training stage, the model cannot identify it and must be completely or partially retrained using new labeled examples of that unseen class.

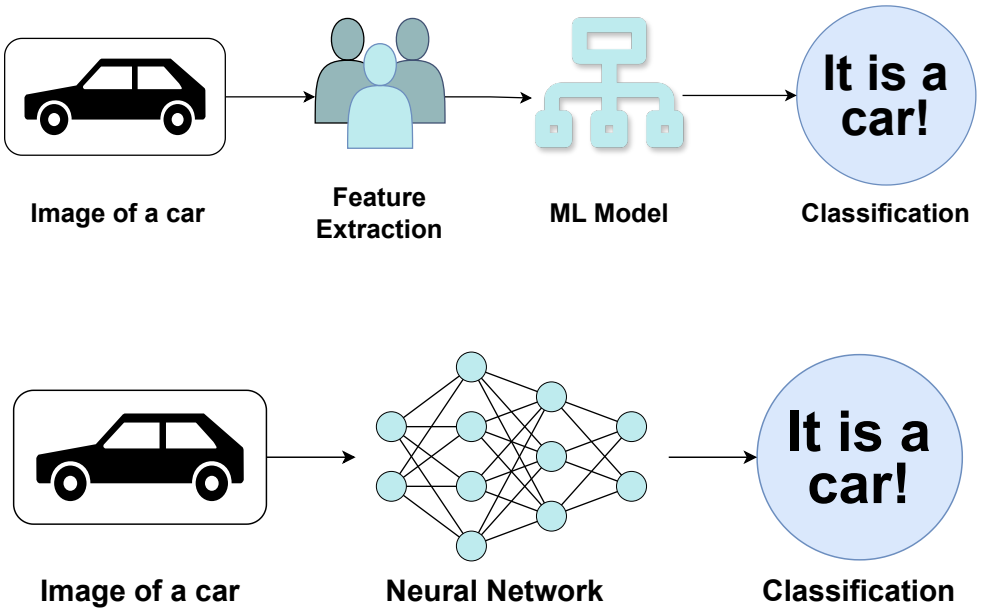


Figure 2.1: Comparison between traditional machine learning and deep learning approaches for classification. Top: Traditional ML requires manual feature extraction before classification. Bottom: Deep learning performs automatic end-to-end feature extraction and classification.

For example, if a business introduces a new product category, the existing traditional classification model is completely blind to it and requires a full or partial retraining phase with new labeled examples to update its knowledge base. In consequence, in this paradigm, the train-test data split must both represent the whole scenario, in other words, all the classes must appear in both the train and test split.

There are many known methods to achieve traditional classification in the ML paradigm, and each comes with their own strengths and weaknesses. If the data is tabular, for example, traditional machine learning techniques such as random forest [11] or xgboost [12] are usually a good fit, as they make good use of pre-extracted features to learn the decision boundary. If the data is unstructured, such as raw images and natural language text, then these techniques usually fall short, as the features are not pre-extracted. In these cases, DL comes as usual State-of-the-Art (SOTA) solutions [13]. They are capable of extracting meaningful features from raw, unstructured data, and subsequently using them to make the final classification. Figure 2.1 shows this difference, where the feature extraction is framed as a pre-processing step performed by a human (or could, alternatively, be performed algorithmically) in the traditional ML model, while the whole process is automatically done, end-to-end, by a DL model.

A very popular framework for traditional classification using DL is using a cross-entropy training framework. The cross-entropy loss computes the difference between two

probability distributions, and in this case they are the predicted classification distribution and the ground-truth distribution (which is 100% for the correct labels, and 0% otherwise). For a single instance, the cross-entropy equation is as follows:

$$CE_{loss} = -\sum_{c=1}^{M} y_c \log(p_c), \tag{2.1}$$

where p_c denotes the predicted probability of the input element belonging to class c, y_c is a label that is 1 if the element belongs to c and 0 otherwise, and M is the number of classes. When a model is trained using a class-disjoint split, meaning that no class appears in both training and test sets, a cross-entropy classifier is unable to learn anything about the unseen classes. Since these classes are not represented in the training data, it is not possible to compute a meaningful cross-entropy loss for them. Therefore, to tackle ZSL, another approach is needed.

2.2 Zero Shot Learning

Formally, let $\mathcal{X} = \{x_1, x_2, \ldots, x_n\}$ denote the set of document images, where each $x_i \in \mathbb{R}^{H \times W \times C}$ represents an image with height H, width W, and C color channels. Let $\mathcal{Y} = \mathcal{Y}^{\text{seen}} \cup \mathcal{Y}^{\text{unseen}}$ be the complete set of document classes, where $\mathcal{Y}^{\text{seen}} = \{y_1, y_2, \ldots, y_k\}$ represents the set of classes available during training and $\mathcal{Y}^{\text{unseen}} = \{y_{k+1}, y_{k+2}, \ldots, y_{k+m}\}$ represents the unseen classes encountered during testing, with $\mathcal{Y}^{\text{seen}} \cap \mathcal{Y}^{\text{unseen}} = \emptyset$. In the traditional supervised learning setting, a model $f : \mathcal{X} \to \mathcal{Y}^{\text{seen}}$ is trained on a labeled dataset $\mathcal{D}^{\text{train}} = \{(x_i, y_i)\}_{i=1}^{N}$, where $y_i \in \mathcal{Y}^{\text{seen}}$. However, this approach fails when encountering samples from $\mathcal{Y}^{\text{unseen}}$ during inference.

The ZSL classification problem aims to learn a function $g : \mathcal{X} \to \mathcal{Y}$ that can correctly classify documents from both seen and unseen classes. In the context of metric learning, this is achieved by learning an embedding function $\phi : \mathcal{X} \to \mathbb{R}^d$ that maps document images into a d-dimensional feature space $\mathcal{Z}$, such that:

$$\begin{cases} d(\phi(x_i), \phi(x_j)) < \tau & \text{if } y_i = y_j, \\ d(\phi(x_i), \phi(x_j)) \geq \tau & \text{otherwise} \end{cases} \tag{2.2}$$

where $d : \mathcal{Z} \times \mathcal{Z} \to \mathbb{R}^+$ is a distance metric (e.g., Euclidean distance) and τ is a decision threshold. The embedding function ϕ is optimized during training using only samples from $\mathcal{Y}^{\text{seen}}$, but must generalize to samples from $\mathcal{Y}^{\text{unseen}}$. The success of ZSL depends on the

model's ability to learn discriminative visual features from $\mathcal{Y}^{\text{seen}}$ that transfer effectively to $\mathcal{Y}^{\text{unseen}}$, capturing the structural properties that characterize document classes.

Unlike traditional classification, which requires the same classes on both the train and the test split, ZSL requires the train and test data split to be class-disjoint [6]. In ZSL, the model is trained also on data from seen classes Y_{seen}, however, it is trained in such way that it recognizes unseen classes. In other words, the idea is, instead of learning Model $f : X \rightarrow Y_{seen}$, it learns Model $g : X \rightarrow Y_{\text{unseen}}$. An extension of ZSL is the Generalized Zero-Shot Learning (GZSL) [6]. While both ZSL and GZSL aim to classify instances into categories without direct training, they differ in the scenario assumed during testing. In ZSL, the testing scenario is restricted to only unseen classes (Y_{unseen}). The model is trained on a set of seen classes (Y_{seen}) and it only sees test data belonging to the classes it has never seen before. On the other hand, GZSL considers the entire combined class space ($Y_{\text{seen}} \cup Y_{\text{unseen}}$) in the test scenario. The model must not only classify unseen instances but also be able to accurately identify and classify the instances belonging to the original seen classes it was trained on. This is intended to simulate a realistic implementation, where a trained model will receive unseen classes as input for inference, but also classes seen during training, as seen in Figure 2.2. The major challenge in ZSL is the unbalanced performance as models often bias classification towards the seen classes because the unseen class domain is often very different from the seen class domain, making it difficult to maintain high accuracy on the familiar seen classes while simultaneously generalizing to the unseen ones.

There are many paths to achieve ZSL in the context of DL. Recently, many tasks are being solved following a ZSL paradigm with LLMs. Using their pre-trained knowledge and general reasoning capabilities, they can perform unseen tasks with only natural language instructions [14]. Another possible ZSL solution is an attribute based approach, where the model learns a relation between attributes and objects [15]. A classic example of this approach is animal image classification. Instead of learning exclusively based on the images, the model learns the relations between the images and the attributes of each animal, for example: zebras have stripes and foxes are orange. This way, the model can classify an image of an unseen animal without retraining, only receiving a manually defined set of attributes that describe that species: tiger is orange and has stripes. Each of these approaches has a meaningful drawback: the LLM approach uses large, generalist models, and lack in cost-efficiency, while the attribute-based approach requires manual definition of the description.

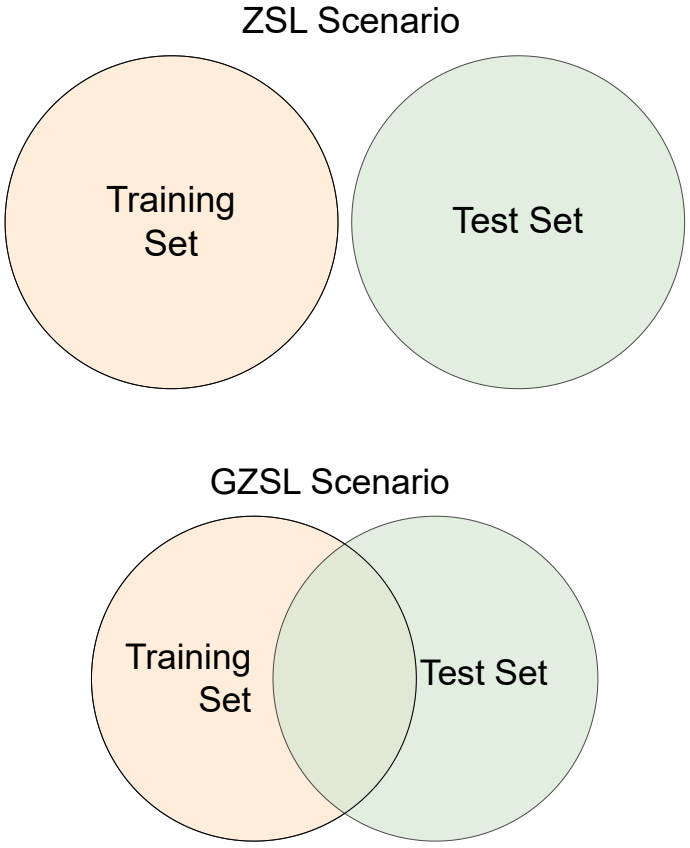


Figure 2.2: Zero-Shot Learning (top) and Generalized Zero-Shot Learning (bottom) scenarios. ZSL has no class intersection on train and test, while GZSL has a partial intersection.

2.3 Metric Learning

One possible solution to produce a ZSL classification model is to employ a metric-learning framework. As in ZSL the challenge is to classify data into unseen categories, metric learning provides the mechanism to solve it by learning a similarity-based embedding space that can be effectively transferred from seen to unseen classes. Specifically, the metric learning model $f(\cdot)$ is trained such that, over an input element, it yields a feature vector v, instead of a traditional classification [16]. The model $f(\cdot)$ is optimized under a given metric $d(\cdot, \cdot)$, such that two feature vectors that share the same class v_1, v_1' are closer together than two feature vectors from different classes v_1, v_2. In other words, the model is optimized so that $d(v_1, v_1') < d(v_1, v_2)$ and $d(v_1, v_1') < d(v_1', v_2)$. Therefore, to summarize, metric learning is the method used to learn the distance function that defines similarity in the cross-modal space, as shown on Figure 2.3, which is critical for knowledge transfer in ZSL.

The metric learning model $h(\cdot)$ is trained such that, over an input element, it yields a

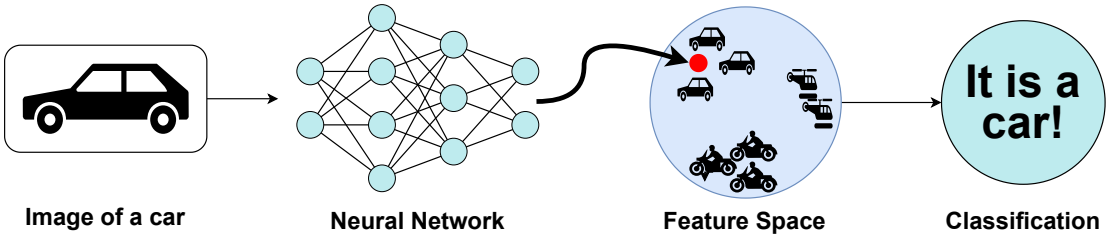


Figure 2.3: Representation of the metric learning paradigm in DL models. The model is capable of classifying a car by finding the nearest neighbor in the feature space.

feature vector v, instead of a traditional classification [16]. The model $h(\cdot)$ is optimized under a given metric $d(\cdot, \cdot)$, such that two feature vectors that share the same class v_1, v_1' are closer together than two feature vectors from different classes v_1, v_2. In other words, the model is optimized so that $d(v_1, v_1') < d(v_1, v_2)$ and $d(v_1, v_1') < d(v_1', v_2)$. Therefore, to summarize, metric learning is the method used to learn the distance function that defines similarity in the embedding space, which is critical for knowledge transfer in ZSL.

In the case of images, the key of ZSL is to associate the visual features of an image into its semantic representation (like a vector of attributes) of its class. This is where metric learning comes in. The idea is typically to map both the visual data and the semantic data into a common shared embedding space. Metric learning is then used to train the projection function for this space. Its objective is to ensure that the distance between a visual feature and its corresponding correct semantic feature is minimized while the distance to incorrect semantic features is maximized. At test time, when an image from an unseen class is used as input, the model maps its visual features into the shared embedding space and then classifies the image by calculating the distance to all the semantic class representations. The unseen class whose semantic representation is closest in the learned metric space is chosen as the predicted label.

A DL model following a metric learning framework can be constructed as a Siamese network [17]. They are named Siamese because they can be seen as bipartite architectures with two parallel paths that converge in the end, such as depicted in Figure 2.4, even though both sides share the same weights. To construct a visual Siamese network, an image model $\phi(\cdot)$ is required to transform an input image I into a feature representation in an arbitrary n-dimensional space, $\phi(I)$. The $\phi(\cdot)$ image model is referred to as the backbone of the siamese network, which is an arbitrary neural architecture (e.g., ResNet [18]).

Once feature vectors are extracted from images, they enable two main inference strategies: verification and identification. Verification addresses the binary problem of determining whether two images belong to the same class. This is accomplished by computing

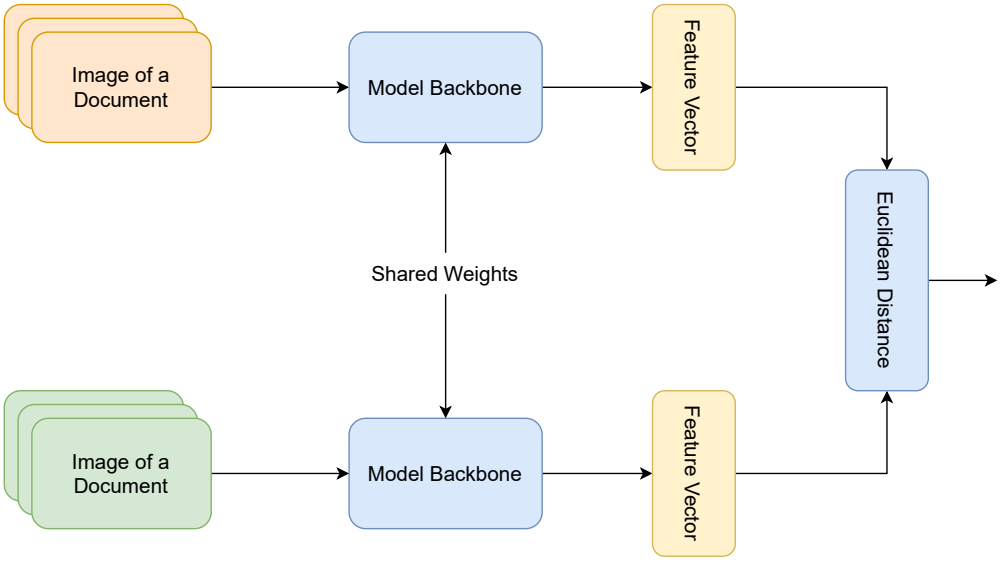


Figure 2.4: Simplified architecture of a siamese network.

the distance between their feature vectors and comparing them against each other with a predefined threshold, where pairs with distance below the threshold are classified as positive matches (same class), while those above it are classified as negative matches (different classes). Identification, on the other hand, solves the multi-class problem of determining which class a query image belongs to. This is typically achieved through a nearest neighbor search in the feature space, where the query is assigned to the label of its closest neighbor. In this way, verification serves as a binary classifier, while identification functions as a multi-class classifier.

There are many loss functions that support metric learning [19, 20], and they all follow the same strategy: cluster together elements that share a label and split apart elements that do not. When training a siamese network, the training step should consider at least two distinct elements, and the loss function scores the step over the distance between the elements. For example, the contrastive loss [19] considers only two elements per training step, and they can either share the same class or belong to different classes. In this work, the contrastive loss function is used in the experiments.

2.3.1 Contrastive Loss

The contrastive loss function $\mathcal{L}{=}\mathcal{L}(m, y, x_1, x_2)$ is an objective function used for metric learning. The conceptual idea behind it is to minimize a given metric distance $d(\cdot, \cdot)$ between representations of data instances assumed similar (referred to as positive pairs) while maximizing the distance between representations of instances considered dissimilar (negative pairs), within a learned embedding space.

More formally, for a pair of input embeddings, x_1 and x_2, the contrastive loss function is constructed to enforce a specific geometry on the embedding space. It is defined as follows:

$$\mathcal{L} = y \cdot d(x_1, x_2)^2 + (1 - y) \cdot \max(0, m - d(x_1, x_2))^2, \tag{2.3}$$

where d is the metric function, (x_1, x_2) is an input pair, and $m > 0$ is a hyperparameter defining a distance margin representing the boundary to discriminate what is different and what is similar, as illustrated in Figure 2.5. Lastly, y is a label with value 1, if both elements share the same class, or 0 otherwise. In other words, this function can be alternatively expressed as

$$\mathcal{L} = \begin{cases} d(x_1, x_2)^2, & \text{if } y = 1 \\ \max(0, m - d(x_1, x_2))^2, & \text{otherwise.} \end{cases} \tag{2.4}$$

If the pair (x_1, x_2) consists of similar (often denoted by a label $y_{1,2} = 0$), the loss penalizes the model when the distance $d(x_1, x_2)$ between their embeddings is large, encouraging the model to pull them closer. On the other hand, if the pair (x_1, x_2) is a negative pair (i.e., they are dissimilar, $y_{ij} = 1$), the loss is designed to penalize the model only if the distance $d(x_1, x_2)$ is less than a predefined margin, m, pushing dissimilar samples farther apart until they cross this threshold.

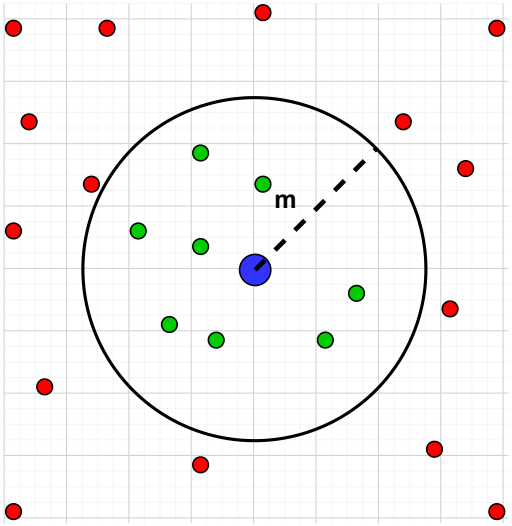


Figure 2.5: Figure demonstrating the effect of the margin hyperparameter m on the contrastive loss function. The model optimizes the feature representations so every element within radius m around certain reference shares the same class, while the elements outside this area have different classes.

In this work, we use two different metric functions d: the Euclidean distance and the Cosine distance. The Euclidean distance is defined as the distance between two points $P = (x_{11}, x_{12}, \cdots, x_{1n})$ and $Q = (x_{21}, x_{22}, \cdots, x_{2n})$ in an n-dimensional space, defined mathematically as

$$d(P,Q) = \sqrt{(x_{11} - x_{21})^2 + \cdots + (x_{1n} - x_{2n})^2} = \sqrt{\sum_{i=1}^{n} (x_1 i - x_2 i)^2}. \tag{2.5}$$

The Cosine distance measures the dissimilarity between two vectors by evaluating the cosine of the angle between them. For two points $P = (x_{11}, x_{12}, \cdots, x_{1n})$ and $Q = (x_{21}, x_{22}, \cdots, x_{2n})$ in an n-dimensional space, the cosine distance is defined as:

$$d(P,Q) = 1 - \frac{\sum_{i=1}^{n} x_{1i} \cdot x_{2i}}{\sqrt{\sum_{i=1}^{n} x_{1i}^2} \cdot \sqrt{\sum_{i=1}^{n} x_{2i}^2}}. \tag{2.6}$$

2.4 Clustering

Clustering techniques are used extensively in this research, as part of the labeling process. Clustering is an unsupervised learning technique that groups data points into distinct clusters based on their similarity, without requiring labeled data [21]. The fundamental objective is to maximize intra-cluster similarity while minimizing inter-cluster similarity, aggregating the data into meaningful groups. In the context of metric learning, clustering becomes particularly powerful as feature vectors learned through metric learning frameworks are specifically optimized to be close together when they belong to the same class and far apart otherwise. This natural alignment between metric learning objectives and clustering goals makes clustering an effective approach for tasks such as class discovery, data exploration, and ZSL scenarios where traditional supervised methods cannot be applied.

K-means [22, 23] is a very popular clustering algorithm. This algorithm begins with k initial seeds, which we call centroids. Then, it follows a two-step iterative process: first, each element is associated with the closest centroid. Next, the centroids are recalculated as the average of the associated elements. This dissertation follows a clustering framework to discover new classes and increase the dataset (see Section 5.1), and k-means can be used in this scenario when the number of possible classes in the clustering process is previously known. However, since part of the goal is to discover new, unseen classes, it is not possible to set a k value.

Therefore, a more suitable clustering algorithm is a hierarchical clustering algorithm [24]. Hierarchical clustering builds a hierarchy of clusters through a tree-like structure called a dendrogram, which represents the nested grouping of data points and the similarities at

which various clusters are merged or split. There are two main approaches to hierarchical clustering: divisive (top-down) and agglomerative (bottom-up). The agglomerative approach is more commonly used and begins by treating each data point as its own individual cluster. At each iteration, the two closest clusters are merged together based on a chosen linkage criterion.

The chosen linkage criterion significantly impacts the resulting cluster structure. One of the possible strategies is Ward's method [24], which is particularly effective for creating compact, spherical clusters. Ward's linkage minimizes the intra-cluster variance when merging clusters, choosing at each step the pair of clusters on which fusion results in the minimum increase in total intra-cluster variance. Formally, the distance between two clusters C_i and C_j using Ward's method is defined as:

$$d(C_i, C_j) = \frac{|C_i| \cdot |C_j|}{|C_i| + |C_j|} \|\mu_i - \mu_j\|^2, \tag{2.7}$$

where $|C_i|$ and $|C_j|$ are the number of elements in clusters C_i and C_j, respectively, and μ_i and μ_j are their respective centroids. This algorithm supports some stopping conditions, for example, stopping the iterations when a specified number of classes is achieved. A suitable criterion for this work is to stop merging clusters when their distance is greater than m, where m is the margin hyperparameter used in the contrastive loss of the metric learning model, as this model will be optimized to treat m as a frontier to what is similar and what is different.

2.5 Vision Models

This section presents the neural architectures used as backbones in this work. All models discussed here are evaluated with the metric learning framework, and their comparative performance analyzed in Chapter 6.

2.5.1 Convolutional Neural Networks

A vision-based DL model is a NN that accepts images as inputs. Vision models can handle a wide variety of tasks, including image classification [13], image segmentation [25], and object detection [26]. Traditionally, these models are based on Convolutional Neural Networks (CNNs) [27], which are neural structures specialized in extracting rich local representations of given data. In vision models, this often reflects on a two-dimensional kernel applied over the input image:

$$(I * K)_{(i,j)} = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} I_{(i+m,j+n)} \cdot K_{(m,n)}, \tag{2.8}$$

where I is the input image, K is the convolutional kernel of size $M \times N$, and (i, j) represents the position in the output feature map.

AlexNet [28], the first NNs to win the ImageNet competition [13], used CNNs as feature extractors to correctly classify images in a traditional classification methodology. AlexNet achieved a 15.3% top-5 error rate in the ImageNet competition, while the second place in that year achieved a 26.2% top-5 error rate, showing the potential of CNN-based models against other classifiers. AlexNet's success rapidly transformed the ImageNet competition, and CNNs became increasingly common. Following this trend, VGG [29] is another CNN-based work that focused its efforts on the ImageNet competition. While AlexNet used only 8 neural layers in total, VGG experimented with increasing the depth of the architecture and its effects on performance. The second biggest model, VGG-16, uses 16 neural layers, and surpassed AlexNet performance with a 8.43% error rate, showing that increasing the depth of a CNN yields better results.

But still, training deep NNs is not as simple as just stacking more layers, as the problem of exploding and vanishing gradient becomes a real impediment [30, 31]. And yet, ResNet [18] is built with even more layers than the previous works. The great breakthrough of this work are the skip-connections, represented in Figure 2.6. Mathematically, instead of learning a desired underlying mapping $H(x)$, the network learns a residual function $F(x) = H(x) - x$, and the final output becomes $H(x) = F(x) + x$. This makes the model easier to optimize, and allows for very deep CNN models to keep learning, even with very deep architectures [18]. The shallowest proposed ResNet with 18 neural layers, ResNet-18, is almost as deep as the deepest VGG, VGG-19. The deepest variant, ResNet-152, achieved a 3.57% top-5 error rate on the ImageNet competition.

As CNNs became increasingly more complex, a new wave of works arose aiming to simplify the current landscape. MobileNet [32, 33, 34] is a series of works that progressively achieves better image classification performance while aiming at an economical architecture capable of being executed on mobile devices. MobileNetV3 [34] is the third iteration of this work, and the largest proposed model achieved a 7.8% top-5 error rate in the ImageNet benchmark, with 5.4 million parameters. In the same year, EfficientNet [35] was also published, following the same cost-economic motivation but following a different path. Instead of focusing on devices with low processing power, EfficientNet changes the scaling logic for CNNs. Instead of focusing exclusively on depth, this work proposes to also increase the width (number of filters on each convolution) and the resolution of the input image. This allows the smallest proposed model, EfficientNet-B0, to achieve a 6.7% top-5

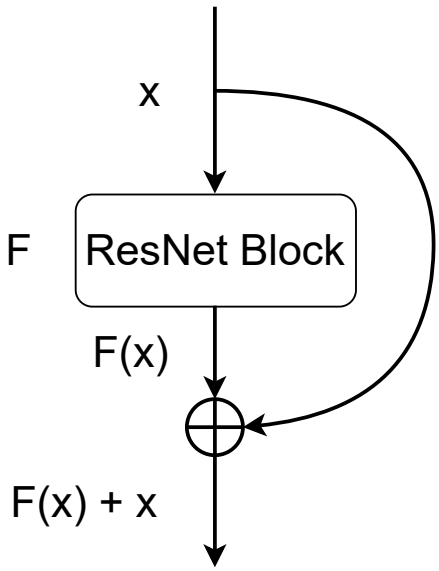


Figure 2.6: Simplified diagram of a ResNet skip-connection.

error rate with 5.3 million parameters, and the biggest proposed model, EfficientNet-B7, to achieve a 3% top-5 error rate, with 66 million parameters.

2.5.2 Transformers

Transformers [36] is a NN architecture that completely revolutionized the field. The core innovation of transformers is the self-attention mechanism, which allows the model to weigh the importance of different parts of the input when processing each element, enabling the capture of long-range dependencies without the sequential processing constraints of recurrent networks, effectively replacing them. This architecture stands at the top of a wide variety of benchmarks, as the trained models can perform completely new tasks, effectively being able to zero-shot with only natural-language instructions [37].

This revolution, for a while, stayed concentrated in Natural Language Processing (NLP) research, while vision research remained with ResNet-like architectures as state of the art [38]. Then, there was ViT [38]. This work takes the Transformers from NLP research and makes an effort to use it in an image context, making the least amount of adaptation possible. To achieve this, Vision Transformer (ViT) applies this architecture by dividing an input image into fixed-size patches, linearly embedding each patch, and processing the sequence of embedded patches through transformer encoder layers. This

approach treats image patches analogously to tokens in NLP, allowing the model to learn global relationships between different regions of an image. Despite requiring large amounts of training data to achieve competitive performance, ViT-based models have achieved state-of-the-art results on various vision benchmarks, usually by pre-training the model on a large dataset and then fine-tuning to solve a specific problem.

Table 2.1 provides an aggregation of the discussed vision models and their performance on the ImageNet classification task [13]. The performance metrics are taken from their own work. AlexNet [28] and VGG [29] do not report top-1 error rate, while ViT does not report top-5 error rate. Note that the models presented in this table are all used in this research, along with a few other variants.

Table 2.1: Comparison of popular neural network architectures in computer vision.

Model	**Year**	**Parameters (M)**	**ImageNet Performance**
AlexNet	2012	60	Top-5 err. 15.3%
VGG-16	2014	138	Top-5 err. 7.3%
ResNet-152	2015	25.6	Top-1 err. 19.38%, Top-5 err. 4.59%
MobileNetV3-Large	2019	5.4	Top-1 err. 24.8%, Top-5 err. 7.8%
EfficientNet-B0	2019	5.3	Top-1 err. 22.9%, Top-5 err. 6.7%
EfficientNet-B7	2019	66	Top-1 err. 15.7%, Top-5 err. 3.0%
ViT-B/16	2020	86	Top-1 err. 11.63%

Chapter 3

Related Work

This chapter analyzes the literature about automatic document processing, focusing on the available datasets and document classification methods. The relevance and direction of this work are justified by the limitations of each dataset and classification framework. This chapter also helps to provide the boundaries for the problem.

3.1 Document Understanding

Document Understanding is a broad field that acts as an umbrella to a wide variety of document-related tasks, and its goal is to transfer physical information, presented in paper, to a digital version that can be stored in databases and used in computer processes [39]. The increasing digitization of documents has led to significant advancements in document analysis techniques, particularly through DL and transformer-based architectures [40]. Among the key challenges in this domain is handling the inherent complexity of scanned documents, which often exhibit noise, structural variability, and diverse formatting styles.

Several fundamental tasks define the scope of document understanding. Document Layout Analysis involves detecting and categorizing different structural components of a document, such as text blocks, tables, images, and forms, to facilitate higher-level information extraction [41]. Information extraction focuses on retrieving relevant information, such as named entities and key-value pairs, from structured and unstructured document sources [40]. This work focuses on DIC, which focuses on correctly classifying a document image into a class. It is important to point out that document image tasks do not provide the document class as input for the experiments, and if the text is required for the proposed solution, it needs to be extracted from the image, for example, using OCR solutions.

3.2 Document Understanding Databases

This chapter reviews existing datasets commonly used for document classification and discusses their limitations in supporting zero-shot scenarios. By analyzing these datasets, the need for a new benchmark that explicitly enforces zero-shot constraints is highlighted. Since the document understanding field is so broad, there are many document-image datasets available. This chapter goes through many of them and points out the limitations they show when seen through the ZSL-DIC optic.

PubLayNet [41] and DocLayNet [42] are large-scale datasets specifically designed for document layout analysis tasks. PubLayNet contains over 360,000 document images, with labels for five layout categories: text, title, list, table, and figure. DocLayNet extends this scope by providing 80,863 manually annotated pages from more diverse document sources, categorized into 11 layout classes, instead of 5. While these datasets have significantly advanced document layout analysis research and serve as foundational resources for training models to detect and segment structural components, they are not designed for document classification tasks. Their labels include pixel-by-pixel class annotations, making them unsuitable for directly training or evaluating DIC approaches.

More recently, the DocVQA dataset [43] was introduced, leveraging a subset of documents from the same collection. It comprises over 12,000 document images paired with 50,000 question-answer pairs, designed to evaluate models' abilities in visual question answering on document images. DocVQA has since become a standard benchmark for assessing the performance of LLMs in document understanding tasks, particularly in multimodal reasoning scenarios. The FUNSD dataset [44], introduced in 2019, has been widely used for OCR-based semantic relation extraction from scanned forms. Similarly, SROIE [45], CORD [46] and XFUND [47] target key-value pair extraction in receipts and invoices, focusing on structured entity retrieval, such as store names, monetary values, and transaction dates. All three of these datasets focus on, in one way or another, extracting specific information from documents. Again, these labels cannot be directly used to train a classification model, as they do not provide a class-based separation, nor a class-disjoint split we expect in a ZSL dataset. Figure 3.1 shows examples of document layout analysis and information extraction datasets.

At the end of the day, the most suitable datasets for the ZSL-DIC task are the traditional classification datasets. The biggest source of documents for these datasets is the IIT-CDIP dataset [48], introduced in 2006, which is a large-scale repository used for document classification and information retrieval. It originated from the Tobacco Documents Library, from the University of San Francisco Industry Documents Library collection, and contains millions of scanned documents. The most popular classification dataset derived from the IIT-CDIP is the RVL-CDIP dataset [5], introduced in 2015. This dataset is a

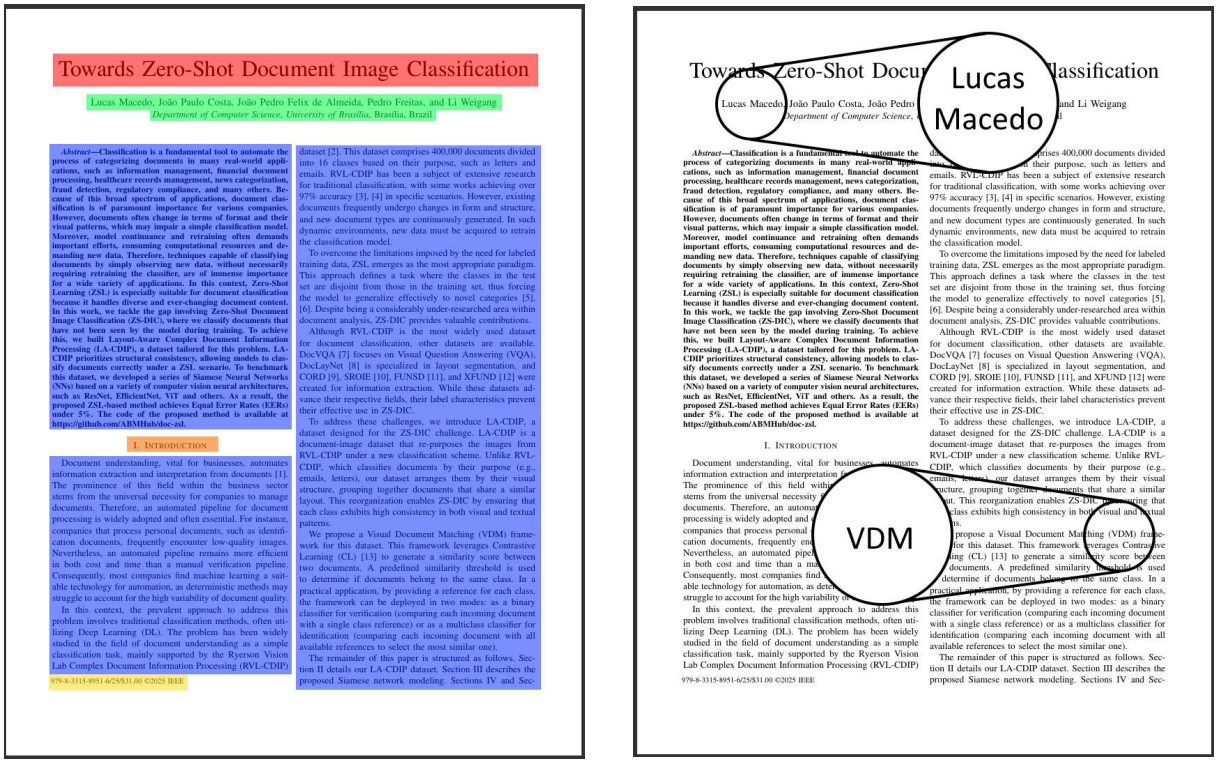


Figure 3.1: Examples of document understanding datasets. The left image illustrates a document layout analysis dataset, where the goal is to visually segment the layout information. The right illustrates an information extraction dataset, where the goal is to find specific information about the document, i.e., the author name.

labeled subset of the IIT-CDIP dataset and organizes 400,000 document images into 16 predefined categories such as letters, forms, and emails. This classification is driven by the document's purpose and uses, which hinders the performance of a ZSL model trained from scratch, with no external real-world knowledge. This observation is confirmed by Larson et al. [49], who highlighted this difficulty by stating that each RVL-CDIP class comprises various subclasses, as illustrated in Figure 3.2. They further argue that this may constrain ZSL solutions, as no single reference could specify an entire class without overlaps. Other traditional classification datasets suffer from the same problem, as they share very similar classes and labeling logic [50].

Therefore, even with these available datasets, there is still a gap that can be explored in the ZSL-DIC domain. The existing datasets either lack the appropriate structure for classification tasks or present challenges in zero-shot scenarios due to overlapping class definitions and the absence of explicit class-disjoint splits. This limitation motivates the need for a new benchmark that explicitly addresses these constraints and enables more robust evaluation of zero-shot document classification approaches.

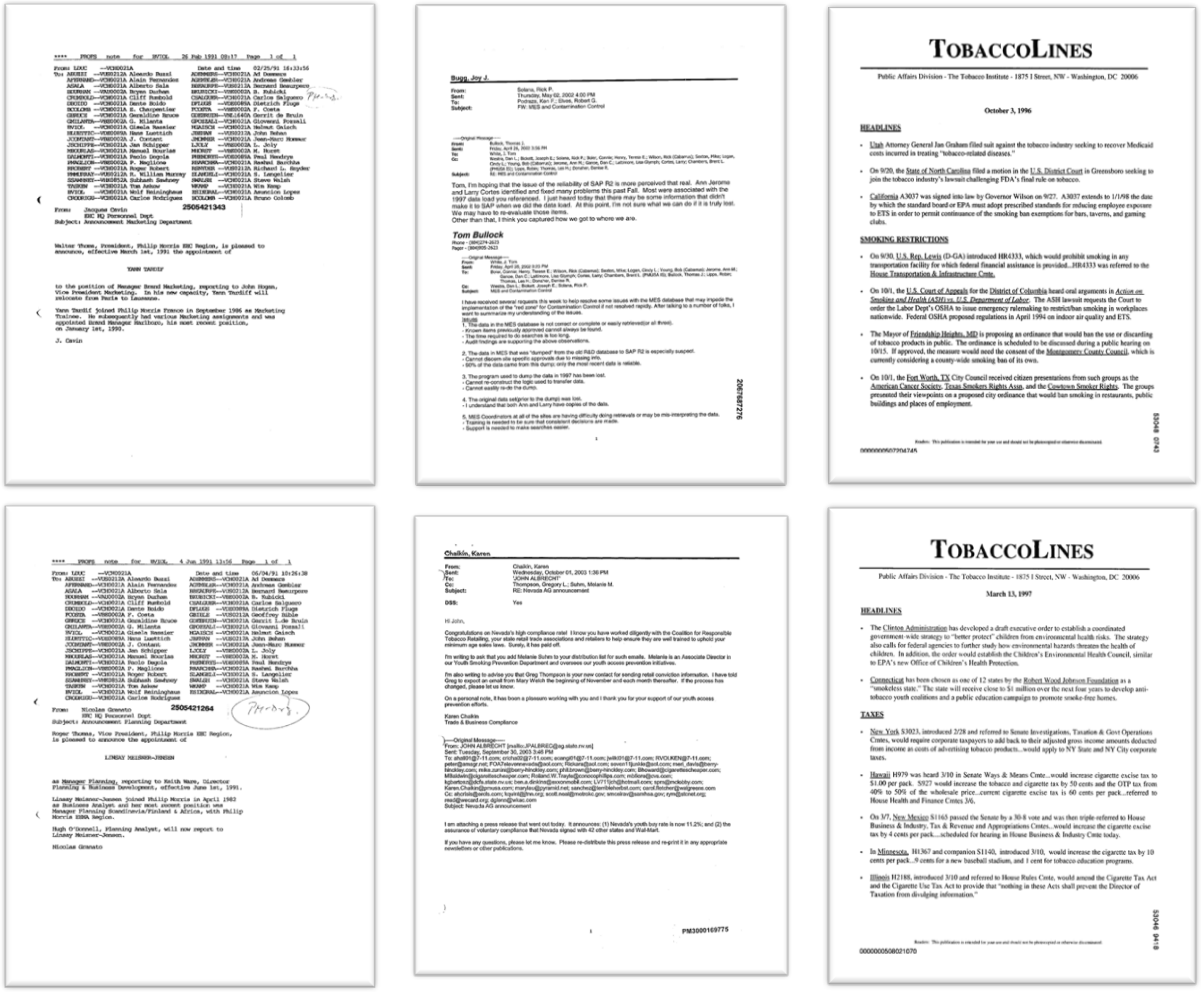


Figure 3.2: Six examples of the RVL-CDIP dataset. Note that, despite the visual differences, these six documents all share the same class under the RVL-CDIP organization.

3.3 Zero-Shot Document Classification

The ZSL-DIC field is historically underdeveloped. One of the possible main reasons is the lack of a proper dataset. Even then, the research community took some different approaches to ZSL-DIC. Most of the research is done with RVL-CDIP dataset, even with the presented limitations.

Khalifa et al. [51], using the RVL-CDIP dataset, proposed a matching solution for ZSL-DIC. Using contrastive learning, they encode both the document itself and the label string in a metric learning approach. They achieve a 39.63% F1-score in an unsupervised setting, that is, with no task-specific training, using exclusively pre-trained models. Then, they train the models in a contrastive-learning framework and achieve 37.23% F1-score. Note that the performance actually decreased. They mention that training the models in a subset of the classes in RVL-CDIP, then testing with the remaining classes, hinders the

generalization on the unseen classes. They close the paper suggesting that the RVL-CDIP dataset is not suited for the ZSL-DIC problem, and that a new dataset must be created to continue the research, arguing that the RVL-CDIP classes are comprised of various subclasses.

Scius et al. [8] investigate the application of LLMs, such as GPT-4 [52] and RoBERTa [53], for zero-shot prompting and few-shot fine-tuning in document image classification. Focusing on their zero-shot experiments, their framework is based on describing to the LLM, through a prompt, the classification task and how it should classify each document. Their study demonstrates that LLMs can achieve competitive performance with minimal labeled data, challenging the traditional reliance on large annotated datasets. Their best performance with open source LLMs achieved an average of 54.4% accuracy, using a model of 123 billion parameters, and achieved 69.9% accuracy with GPT-4-Vision, which has an undisclosed parameter count.

Based on the work of Khalifa et al. [51], Sinha et al. [7] introduce CICA, a framework that enhances CLIP's [54] performance in zero-shot classification by joining it with a content-encoder, which aligns its own feature representations with the feature space used by CLIP in a contrastive-learning approach. CICA manages to bypass the RVL-CDIP limitations by exploring CLIP's pre-trained knowledge, achieving a 67.29% average accuracy. Compared to the previous works, this study emphasizes data-efficient approaches to document classification. However, their approach is not suitable for a from-scratch training, which would allow an even greater efficiency gain.

Chapter 4

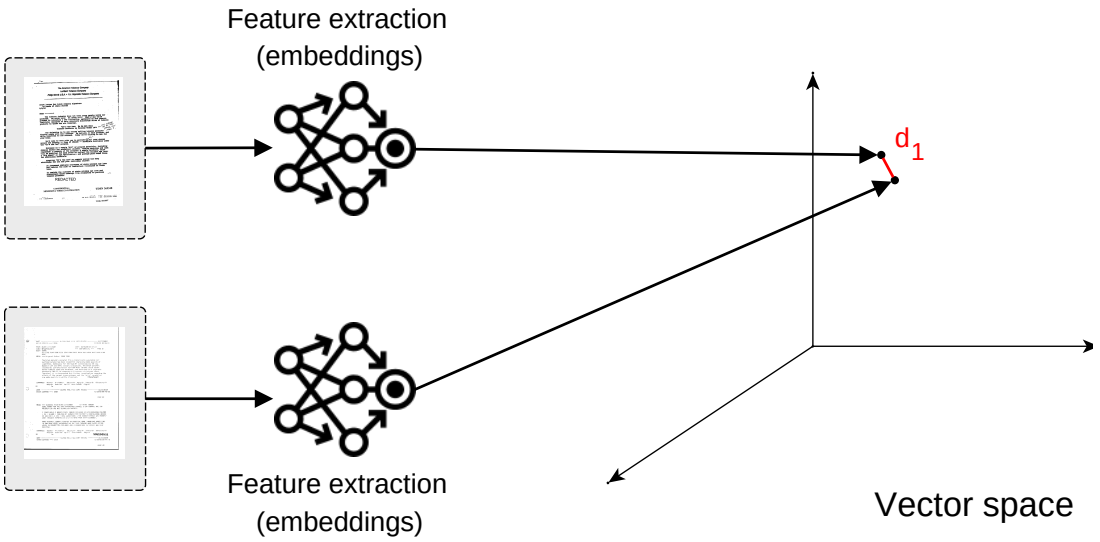
Proposed Solution

4.1 Visual Document Matching

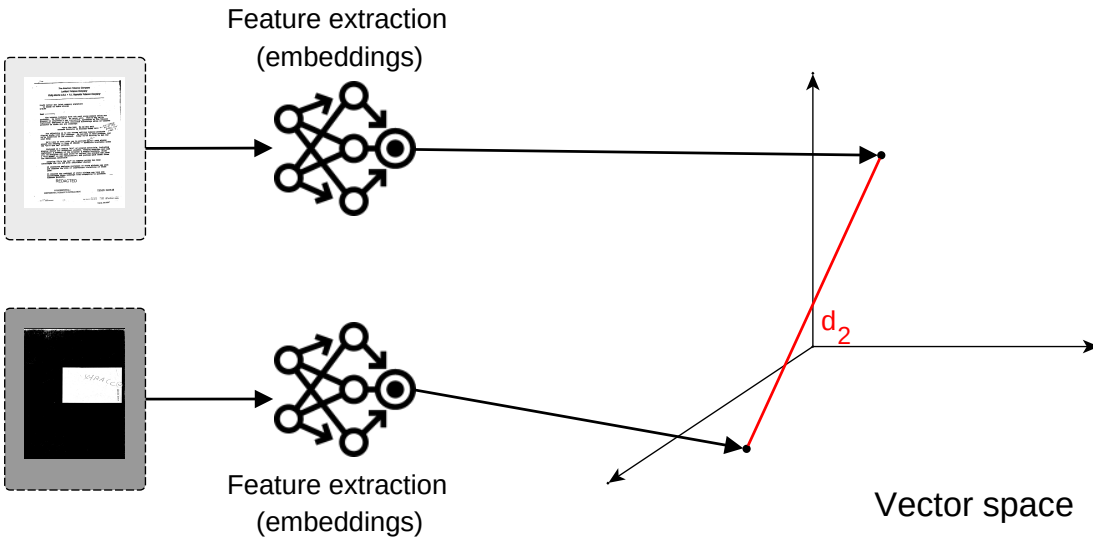
Traditional classification methods, such as entropy learning, lock the generality of the model to the set of classes it has been trained on. Therefore, the use of Zero Shot Learning techniques is essential to be able to classify document layouts that have not been seen on the training set. Among the possible ZSL solutions, we use metric learning with Siamese networks to achieve this goal. The proposed metric learning framework, VDM, takes two different document images, and compares them to decide whether they belong to the same class or not, using vision models to match documents that share the same class.

This approach is inspired by biometric signature verification approaches [55], where a system has a repository of ground-truth signature references (signatures that they know are from a certain person), and they compare them with a query signature, to verify if they belong to the same person or not. Therefore, VDM works by learning to draw representations of documents in a feature space with n dimensions, such that documents that share the same class are represented as clustered together, and documents that have different classes are far apart, as illustrated in Figure 4.1.

A wide variety of backbones are used to experiment on the proposed dataset, but they all follow the same metric learning architecture with Siamese Networks. They are AlexNet [28], VGG [29], ResNet [18], MobileNetV3 [34], EfficientNet [35] and ViT [38]. To adapt the architectures to a Siamese network, only the last linear layer of each model — the classification layer — is modified into a new linear layer with an arbitrary size n, suitable for the problem.



(a) Similar documents with small distance d_1 between them.



(b) Dissimilar documents with large distance d_2 between them.

Figure 4.1: Illustration of two documents mapped to the vector space that are similar (a) and dissimilar (b). This document mapping to a vector space is performed by the proposed learned method for similarity analysis. Visually similar documents have a smaller distance in the projected space than dissimilar ones.

Chapter 5

Methodology

5.1 LA-CDIP Dataset Construction

The construction of this dataset is motivated by the absence of a specialized ZSL-DIC dataset and the fact that RVL-CDIP is unsuitable for the proposed VDM method [51] (see Section 3.3). To support VDM in achieving ZSL-DIC, this dataset must provide two main characteristics: great class diversity, so that models trained on it can generalize across unseen classes, and consistent intra-class and inter-class consistency, ensuring that the classes are atomic (with no possible subdivision).

Therefore, the proposed dataset, LA-CDIP, comes as a reorganization of the RVL-CDIP dataset, focusing on increased granularization, in the sense that no two document layouts are seen in the same class. Visual cues such as a company logo, the layout of a table, the position of certain text, etc., all contribute to differentiating between distinct document patterns, and different patterns imply different classes. While the RVL-CDIP dataset was used as source content, it was re-labeled to align with the problem of differentiating documents, with a particular focus on their visual structure (layout). Figure 5.1 illustrates the impacts of the new labels. All eight documents in the figure used to belong to the same class, but now belong to two different classes under the new organization.

The dataset labeling follows two stages: preliminary clustering, followed by a manual reorganization. The clustering is initially constructed using a private metric learning model trained on a private document dataset. Both the model and the dataset follow the same methodology as this dissertation. Subsequently, the model is used to generate feature vectors across the entire RVL-CDIP dataset. These feature vectors are then used to create clusters of documents, which are effectively predicted classes. The Hierarchical Agglomerative Clustering algorithm with Ward's [24] strategy is used (see Section 2.4). This clustering served the purpose of accelerating the manual labeling process.

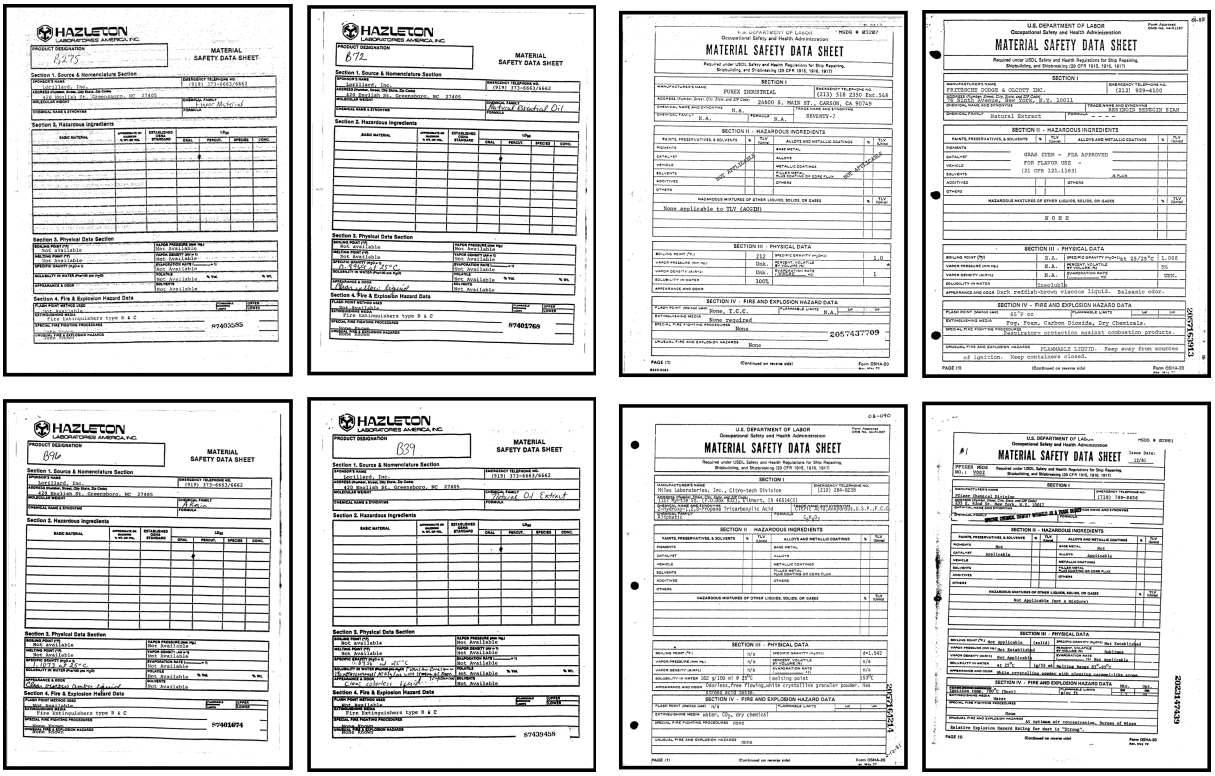


Figure 5.1: Examples of two distinct classes under the proposed LA-CDIP dataset. Those classes originally belonged to the same class under RVL-CDIP organization.

The manual labeling process has two main tasks: first, verify the integrity of the clusters. As the clusters were made with a DL model, they are prone to errors. Therefore, it is possible that one cluster holds many different document patterns, which must be corrected by a human annotator to ensure intra-class consistency. Second, verify if the cluster is unique. The model may have created two distinct clusters with the same layout, therefore, a step to merge clusters is necessary to ensure inter-class consistency.

This training pipeline operates in a loop (illustrated in Figure 5.2): given a stop condition (which can be a time frame or a goal in terms of the number of labeled documents), the annotated documents are compiled into a dataset containing more elements than the previous iteration. This dataset is then used to train a new model using the same VDM framework. This new, enhanced model is then used to create new clusters, which should exhibit greater intra-cluster and inter-cluster consistency, therefore reducing the human annotation time required per document. To further optimize the labeling process, each cluster is ranked based on the average pairwise distance between every element in the cluster: clusters with the lowest average distances are provided first to the human annotators. This is meant to optimize the amount of work to achieve intra-class consistency.

Note that the proposed data labeling loop is similar to active learning [56], with a few key differences. While the LA-CDIP dataset is created to support VDM, it is still a

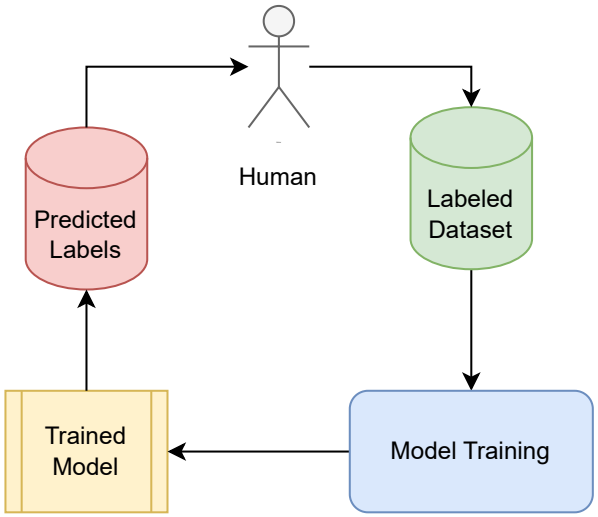


Figure 5.2: The proposed data labeling loop. At each loop, the labeled dataset is increased, which is used to train a better model, which results in more consistent predicted labels, reducing the time the human annotator spends per document.

contribution by itself. Therefore, the purpose of this labeling loop is to create a large and complete dataset, which differs from the active learning philosophy of achieving greater accuracy with fewer labeled instances [56]. The sampling strategy frequently used in active learning involves focusing human efforts on samples for which the model provided low classification confidence, so that they provide the most value for the next trained model. This is opposite to the approach proposed in this work, which prioritizes clusters with the highest confidence to optimize the time spent per annotation.

The efficiency of this labeling loop can be seen by the increasing quality of the produced labels. Consequently, specific metrics are used to verify the validity of this framework. It is assumed that good automatic clustering yields an organization as close as possible to the final, human-annotated dataset. Therefore, to measure progress for each loop, we calculate the distance or similarity between two sets of clusters: the one automatically produced and the human-annotated one. We use two metrics: Adjusted Rand Index (ARI) and Normalized Mutual Information (NMI).

ARI is a similarity metric that compares two sets of clusterings to evaluate whether the relationship between elements remains the same [57]. In other words, this metric evaluates whether, for each pair of elements within a cluster, they still share the same label after the changes, and whether, for each pair of elements across different clusters, they still belong to different clusters. NMI also traces the similarity between two clusters, but it quantifies the amount of information one cluster provides about the other [58].

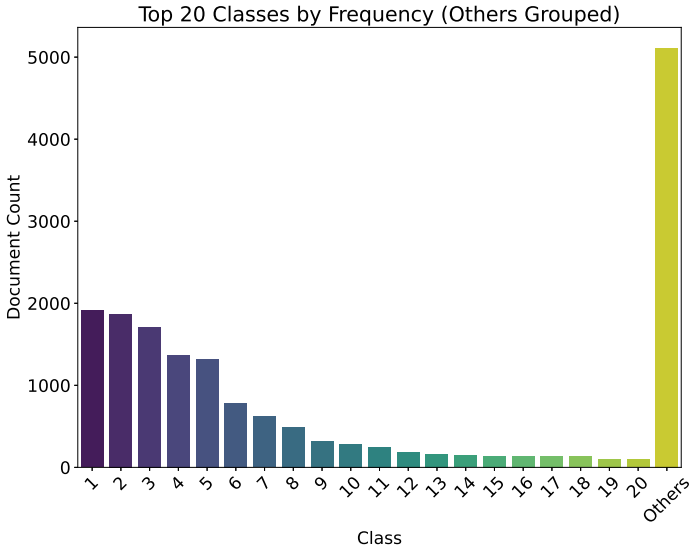


Figure 5.3: Top 20 classes by frequency; remaining 645 aggregated as “others”.

These two metrics result in a value of 1 if the clusters are exactly the same and 0 if they share no relation. Therefore, if the proposed labeling method reduces the amount of work necessary per loop, these metrics should increase in each loop.

5.2 LA-CDIP Dataset

The proposed dataset is composed of 17,295 documents, divided into 665 different classes. Each class has at least two documents, and the largest class has 1913 documents, presenting an extra challenge in dealing with an unbalanced dataset. Figure 5.3 shows the class distribution.

To maintain train-test data consistency, the data is split following the ZSL and GZSL protocols proposed by Xian et al. [6] (see Section 2.2). In both scenarios, the data is split into train and test sets, and the training data is further divided using 5-fold cross-validation [59]. The data for the test scenario and for each split are chosen randomly and remain the same for every experiment. The test scenario is treated as a sixth split for the cross-validation, so it follows the same rules as the other five splits. For the ZSL scenario, one-sixth of the classes are randomly chosen for each split, with no overlap. Naturally, this scenario creates splits with variable sizes, as the classes themselves have variable sizes. For the GZSL scenario, half of the classes in the entire dataset are chosen as classes that do not overlap between splits, while the other half are distributed across the splits. When constructing a split, the non-overlapping classes are chosen first, and

then the remaining slots are filled with the overlapping classes, achieving splits of constant size for this scenario.

This dataset can be normally used for traditional visual classification problems; however, since the problem is tackled with metric learning, additional considerations must be taken into account. When using a Siamese network, running inference on a single data point returns no information, as the architecture requires a comparison between different data points. Randomly selecting pairs for every test can cause results to fluctuate; therefore, to maintain test consistency, a test protocol is used that ensures the same pairs are tested every time. For every document in the set, two other documents are chosen: one that shares the same class and one that does not. The second document is chosen randomly by its class; therefore, every class has the same probability of being picked. A protocol is established for the test set and for every cross-validation split.

5.3 Evaluation Criteria

Since a metric learning model yields a distance between two data points (in this case, two documents), a threshold needs to be defined to declare whether the two data points belong to the same class or not, to calculate how accurate the model is. For this, Equal Error Rate (EER) is used as the metric for the experiments [60]. EER is the point at which the False Acceptance Rate (FAR) and False Rejection Rate (FRR) are equal. The EER is calculated by setting the FAR equal to the FRR and finding the corresponding threshold [61]. EER occurs when $FAR(\tau_{EER}) = FRR(\tau_{EER})$. These rates are defined as:

$$FAR(\tau) = \frac{\text{False Acceptances}}{\text{Negative Samples}} \quad \text{and} \quad FRR(\tau) = \frac{\text{False Rejections}}{\text{Positive Samples}} \tag{5.1}$$

representing the respective probabilities of false positive and false negative classifications at threshold τ.

Ten models are trained for each chosen backbone (see Section 4.1): 5-fold cross-validation for each of the two scenarios, ZSL and GZSL. As shown in Section 5.1, each split follows a fixed validation protocol to ensure consistency. The trained models from each split are also evaluated on the independent test set to compare them with LLMs. The mean EER is reported for every cross-validation scenario, as well as the mean EER on the independent test set.

Chapter 6

Results

6.1 Dataset Construction

The dataset construction followed a sequence of incremental steps, following the methodology presented in Section 5.1. The proposed labeling method assumes that a model trained with the current iteration of the dataset will be able to create better (i.e., more precise) labels if compared to training with the previous iteration (should have less data), which will then speed up the human annotation process. Table 6.1 shows this behavior. This table details every iteration of the training loop: the achieved number of documents, followed by number of classes, and the similarity between what was automatically clustered versus the resulting human annotated labels. Each iteration begins with an automatic clustering produced by a model trained on the dataset achieved by the previous iteration. The first iteration takes the dataset from a previous work [10] and creates a new, independent dataset.

Table 6.1: Iterative dataset construction metrics. Each row shows information of an iteration of the dataset, such as the total number of documents and classes in the dataset at that iteration. ARI and NMI are similarity metrics between the automatic labeling process and the final, human annotated labels.

Iteration	Number of Documents	Number of Classes	ARI	NMI
1	10,659	512	0.131	0.665
2	12,246	560	0.141	0.650
3	17,295	665	0.440	0.784

The table shows an increase in cluster similarity with the growth of the dataset. In particular, the increase from the second to the third iteration is greater than the increase from the first to the second. It is possible that the data achieved by the first iteration had the same overall quality as the data achieved by the previous work, meaning that models

trained on these datasets achieved similar behavior. The labeling loop then follows with a leap of cluster quality in the third iteration, revealing evidence that the labeling method can optimize the manual process. Furthermore, the second and the third iterations both lasted the same duration, and yet, the second iteration achieved 1,587 new documents, and the third, 5,049. This is empirical evidence that the cluster similarity is inversely related to the time spent per document on the manual annotation process. The remaining experiments in this work are done with the last iteration of the dataset.

6.2 VDM Experimental Setup

The method was implemented in Python, primarily using the PyTorch library [62]. All experiments were executed using four NVIDIA GPUs: three RTX 4090 (24GB VRAM each) and one GTX 1060 (6GB VRAM). The experiments were parallelized across these GPUs, with each device training different model configurations independently. Despite the different hardware, all models used consistent batch sizes and hyperparameters to ensure a fair comparison. The hyperparameters used in these experiments are shown in Table 6.2. Note that, while most of the architectures use the same hyperparameters, MobileNetV3 and ViT use different configurations, following suggestions from their original works.

Table 6.2: Hyperparameters for different neural architectures in VDM experiments.

Architecture	**Optimizer**	**LR**	**LR Decay**	**WD**	**Epochs**
Default	SGD	0.01	Step (0.1/30 ep.)	1×10^{-4}	90
MobileNetV3	SGD	0.01	Step (0.973/2 ep.)	1×10^{-5}	300
ViT	AdamW	0.001	Cosine Annealing	1×10^{-5}	90

The remaining configurations: output size, distance metric, and model edition for each architecture, are determined by a hyperparameter search in Chapter 6. VDM is also used in the proposed labeling process as the DL model that provides feature vectors for the clustering process. The models trained for clustering have fixed configurations: ResNet-18 backbone, Euclidean distance, and 64-neuron output.

The VDM trained models are exclusively visual; therefore, their input data is the (R, G, B) document image matrix. First, the input image is resized into a shape compatible with each neural architecture. For most models, this shape is $(224, 224)$ in height and width. EfficientNet is the only architecture that does not follow this rule, as different model versions increase their input size while simultaneously increasing network depth and width. Then, every value in the image matrix is scaled from 0–255 to 0–1 and normalized using the mean and standard deviation of the current training split.

During training, each step processes a pair of documents that can be either similar (same class) or dissimilar (different classes). The pairing process follows a specific procedure: first, a random document is selected from the training set; then, the pair type (similar or dissimilar) is randomly determined. For similar pairs, another document from the same class is chosen. For dissimilar pairs, a different class is randomly selected first, followed by a random document from that class. This class-level randomization ensures that every class has equal probability of forming dissimilar pairs, preventing overfitting to larger classes. Each batch is composed of 16 pairs. Throughout an epoch, each document is chosen once to serve as the first element of the pair, and may be randomly chosen more times to close pairs in the second position. While training randomizes each pair in every epoch, during testing and validation, the pairs are fixed. The chosen pair is then processed by the Siamese network, generating two feature vectors, which are evaluated by the contrastive loss function (see Section 2.3.1).

6.3 Hyperparameter search

Before making a proper benchmark, a hyperparameter search is necessary. Most hyperparameters are already set, as they are taken from the original works (see Section 6.2), but hyperparameters related to the metric-learning nature have not been set. Table 6.3 shows the search for the remaining configurations: the architecture edition, the distance metric and the embedding size.

The experiments in Table 6.3 use the first fold of the cross-validation as the train and validation splits, and the reported values are from the independent test split. This table is used to find the best possible parameter configuration for each architecture, and the remaining experiments in this work are done with the best hyperparameters found. To better analyze these results, this table is split into three other different tables.

Table 6.4 shows the average EER for each combination of metric distance and embedding size. We can see that the best overall combination is an embedding of size 256, and using the cosine distance. We can see in the last row, where the results are averaged across every embedding size, that the cosine distance is indeed the best overall metric distance, with an absolute difference of 3.10% EER from the Euclidean distance. This is evidence that most architectures used in these experiments work better with the cosine distance for this problem. The last two columns on this table bring the average EER for each embedding size, averaging the distance metrics. In these columns, we can see that the smaller embedding size used, 16, is actually the best overall, with the lowest standard deviation. A smaller embedding size implies a smaller feature space, therefore, it is possible that models trained with an embedding size of 16 are less prone to overfit-

Table 6.3: Results of the hyperparameter search for each architecture in EER. Searching for: the architecture edition (e.g. ResNet-18 or ResNet-34), the distance metric (Euclidean or cosine), and the output embedding size.

		Euclidean						Cosine					
Model	Ed.	16	32	64	128	256	512	16	32	64	128	256	512
AlexNet		24.71	20.45	34.81	33.61	31.07	22.31	21.21	20.21	21.62	18.44	22.14	**15.26**
VGG	11	19.35	32.73	9.03	31.68	30.82	34.59	22.80	29.38	20.23	22.02	29.31	26.39
	13	22.38	8.83	24.05	14.82	33.05	11.69	18.22	23.61	13.85	19.59	24.22	30.01
	16	7.63	4.82	30.80	12.57	23.39	8.90	12.01	27.76	22.65	30.23	25.68	32.63
	19	13.26	1.81	29.79	18.08	27.67	11.86	23.48	23.17	19.69	21.18	1.66	**1.57**
ResNet	18	3.52	7.07	9.00	7.05	3.40	1.66	19.35	11.69	2.94	2.35	1.64	2.64
	34	9.10	10.30	7.36	8.07	17.71	3.30	3.23	4.82	1.54	5.48	**0.49**	0.61
	50	7.07	26.47	11.69	4.01	23.24	14.75	15.95	17.25	4.31	6.04	10.37	10.10
	101	21.80	19.06	20.72	21.23	31.99	19.06	13.04	6.73	3.84	2.42	7.75	24.93
	152	7.14	4.77	14.24	13.72	5.70	16.85	4.26	8.98	2.62	15.63	13.70	9.34
MobileNet	S.	7.44	13.60	48.29	16.24	29.87	39.97	20.38	6.36	5.53	14.04	5.14	5.60
	L.	22.82	43.79	38.01	46.16	44.32	31.60	17.34	16.44	17.10	10.57	**5.09**	12.84
EfficientNet	0	7.61	1.00	5.68	1.88	14.46	**0.73**	2.67	5.53	3.96	8.73	2.62	2.81
	1	4.99	7.44	5.31	3.33	6.73	5.09	2.54	4.84	3.86	5.75	1.37	2.40
	2	3.42	4.11	4.26	3.99	2.15	13.50	4.87	3.11	2.59	12.77	4.26	6.65
	3	19.50	5.28	4.26	8.71	11.15	8.00	10.10	4.35	3.99	12.87	10.59	6.09
ViT	B.	29.43	33.19	17.95	28.01	25.15	24.66	30.26	37.77	38.60	26.83	18.62	32.27
	L.	19.32	27.81	28.69	**16.81**	22.60	21.72	25.76	18.15	40.17	38.43	22.38	30.53

Table 6.4: Average performance in EER over each possible output dimension, considering the Euclidean distance, the Cosine distance, and considering both distances. The last row considers every embedding size, to measure the best average distance metric.

	Euclidean		Cosine		Both	
Embedding Size	Mean	Std	Mean	Std	Mean	Std
16	13.92	8.39	14.86	8.69	14.39	8.43
32	15.14	12.81	15.01	10.33	15.07	11.47
64	19.39	13.77	12.81	11.99	15.92	13.10
128	16.11	12.14	15.19	9.96	15.65	10.95
256	21.36	11.91	**11.50**	9.74	16.43	11.83
512	16.13	11.37	14.04	11.96	15.08	11.55
All	16.99	11.83	13.89	10.37	15.42	11.20

ting, achieving stable but moderate results. Higher embedding sizes, while more prone to overfitting, also hold the potential to handle more classes without entanglement. We can see this behavior in Table 6.3: while an embedding size of 16 is the best average value, it was not once the best hyperparameter for any architecture, and most best configurations use 256 or 512 embedding size.

Table 6.5: Average performance in EER over each possible architecture, considering the Euclidean distance, the Cosine distance, and considering both distances.

	Euclidean		Cosine		Both	
Architecture	Mean	Std	Mean	Std	Mean	Std
AlexNet	27.83	6.12	19.81	2.58	23.82	6.13
VGG	19.32	10.30	21.72	7.96	20.52	9.19
ResNet	12.30	8.01	8.01	6.26	10.08	7.42
MobileNetV3	31.84	13.93	11.37	5.69	21.61	14.75
EfficientNet	6.36	4.50	5.39	3.30	5.87	3.93
ViT	24.61	5.06	29.98	7.81	27.30	6.99

Table 6.6: Average performance in EER over each possible architecture, considering each output dimension.

	16		32		64		128		256		512	
Architecture	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
AlexNet	22.96	2.47	20.33	0.17	28.22	9.32	26.03	10.72	26.60	6.31	18.79	4.98
VGG	17.39	5.82	19.01	12.02	21.26	7.38	21.27	6.76	24.47	9.79	19.71	12.60
ResNet	10.45	6.73	11.71	7.10	7.83	6.17	8.60	6.26	11.60	10.19	10.33	8.37
MobileNetV3	16.99	6.75	20.05	16.39	27.23	19.44	21.75	16.44	21.10	19.39	22.50	15.99
EfficientNet	6.96	5.68	4.46	1.88	4.24	0.94	7.25	4.20	6.67	4.88	5.66	3.99
ViT	26.19	4.98	29.23	8.43	31.35	10.28	27.52	8.84	22.19	2.69	27.29	4.94

Table 6.5 shows the average EER for each combination of distance function and chosen architecture. We can see that the best combination of architecture and metric distance is EfficientNet with cosine distance. Most architectures perform better with cosine distance, with the exception of ViT and VGG. MobileNet performed particularly badly with the Euclidean distance, with almost three times the EER if compared against using the cosine distance. Table 6.3 shows that some experiments with this combination got near 50% EER, a score that would be as bad as a completely random decision. In these cases, the trained models mostly did not learn across all the 300 epochs they were trained on. Again, the last two columns show the performance of each architecture, independently of the chosen distance metric. The best architectures in these experiments are the EfficientNet variants, followed by ResNet and VGG.

Finally, 6.6 shows the performance of each architecture, for each possible embedding size. This table shows some of the behaviors pointed out in the two previous paragraphs: embedding size of value 16 having lower overall standard deviation and EfficientNet achieving the lowest average EER. The best combination of embedding size and architecture, in average, is EfficientNet with size 64, and is different from the best hyperparameter combination found on Table 6.3, which is using embedding size 512. This shows, again, that while bigger embeddings may increase the potential for better models,

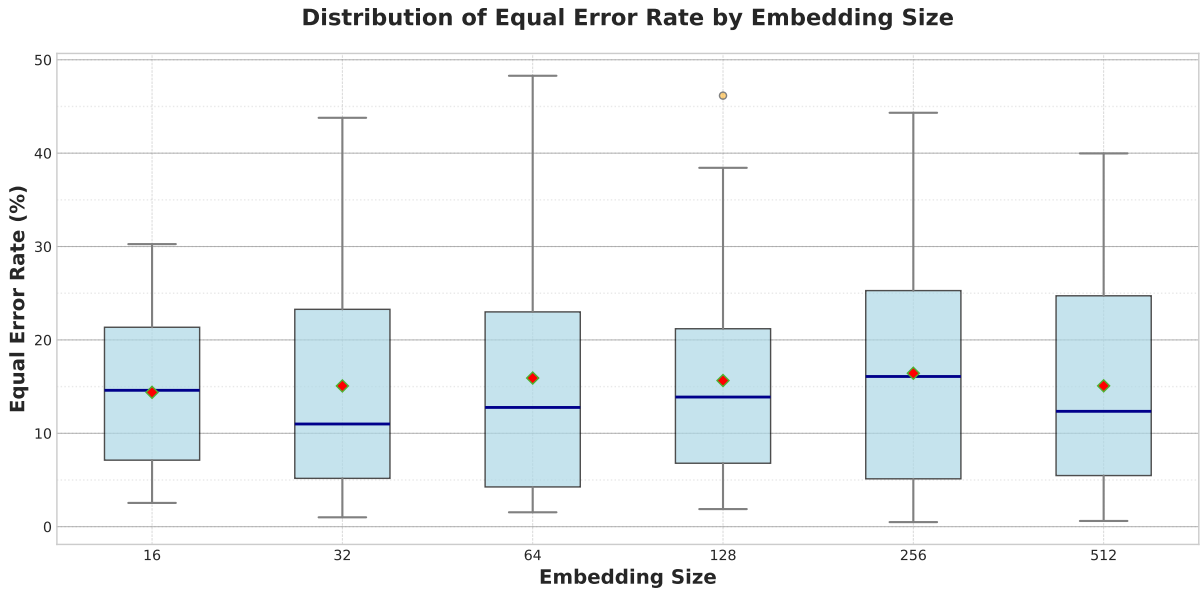


Figure 6.1: Boxplot of the EER over the different chosen embedding sizes. In each box, the blue line represents the median value, and the red shape represents the average value.

it can also increase the variability of the results. Figure 6.1 shows a boxplot of the EER in relation to the embedding size, across all experiments shown in Table 6.3. Note that the error oscillates across the size options and is neither strictly increasing nor decreasing, and the most stable parameter is still the 16 embedding size.

6.4 Benchmark

This section's discussion revolves around Table 6.7, where the results of every trained model are presented. The table shows the average EER value of the cross-validation in both scenarios (ZSL and GZSL). Every model is trained with the best set of hyperparameters, found on Table 6.3.

Immediately, we can see that the overall best models in this experiment are the ResNet-34 and EfficientNet-b0. They offer a better balance between performance and economy, being the smallest models that perform the best. Usually, the larger the model, the more data it needs to perform well, and with a small dataset, smaller models usually perform at least as good as the larger models [63]. With this observation, we can observe that ResNet and EfficientNet reside in the balance between model complexity and performance. Lowering the parameter count, we find MobileNetV3-Large, which does not perform as well as the other two models, despite being at least more economic. In the hyperparameter search, MobileNetV3-Small showed even higher error rate, showing that the MobileNetV3 architecture is already too economic for this problem. In the other direction, VGG and

Table 6.7: Comparative performance between different visual backbones. Following the columns: the architecture name, number of parameters for each trained model, cross-validation over the ZSL scenario and cross-validation over the GZSL scenario. Every value is a mean EER (%) value over the five Cross-Validation (CV) folds.

		ZSL		GZSL	
Architecture	Params	Mean	Std	Mean	Std
AlexNet	57M	10.05	8.80	13.09	1.43
VGG-19	139M	7.79	5.02	12.30	2.04
ResNet-34	21M	4.72	4.11	2.55	2.29
MobileNetV3-Large	4M	9.32	4.16	10.77	1.54
EfficientNet-b0	4M	6.92	4.07	6.51	2.28
ViT-Large	305M	15.93	5.93	13.50	1.32

ViT performed worse than EfficientNet and ResNet. This is evidence that the increased architecture complexity did not suit the size of the dataset. ViT, in particular, being a transformer architecture [36], performed the worst of all architectures in these experiments, as it is specially data demanding. Figure 6.2 shows the TSNE visualization of the ZSL scenario. Every plot shows the representations of every element in a single validation split, where a dot is a vector representation of a document, and each color is a different class. We can see that, on this particular fold, the representations of the ResNet, EfficientNet and VGG models are well defined and well separated, while the representations on ViT, the largest model, are very entangled.

The difference between ZSL and GZSL is the data split. While ZSL is class-disjoint, GZSL has both seen and unseen classes on the validation split (see Section 2.2). The GZSL split has some great advantages: first, it is likely to decrease the error on models prone to overfitting on the seen classes, as some of them will appear on the validation test as well. Second, the greater class diversity on the training set should increase the model potential to generalize to unseen classes. On the other hand, the GZSL also increases the amount of classes present on the validation set, raising the overall difficulty, especially for models that cannot make use of the advantages. For reference, each ZSL validation split has 111 classes, while the GZSL splits have an average of 253.8 classes. This difference separates the trained models into two groups: ResNet, EfficientNet and ViT perform better in the GZSL split, while AlexNet, VGG and MobileNet perform worse.

The better performance on the GZSL split with the ResNet and EfficientNet models suggests that the generalization advantages exceed the difficulty increase. The worse performance on the AlexNet, VGG and MobileNet suggests that the difficulty increase weighed more than the advantages. ViT is a particular case. Figure 6.3 shows the ZSL loss value per epoch on both the train and test split on every architecture. This figure shows

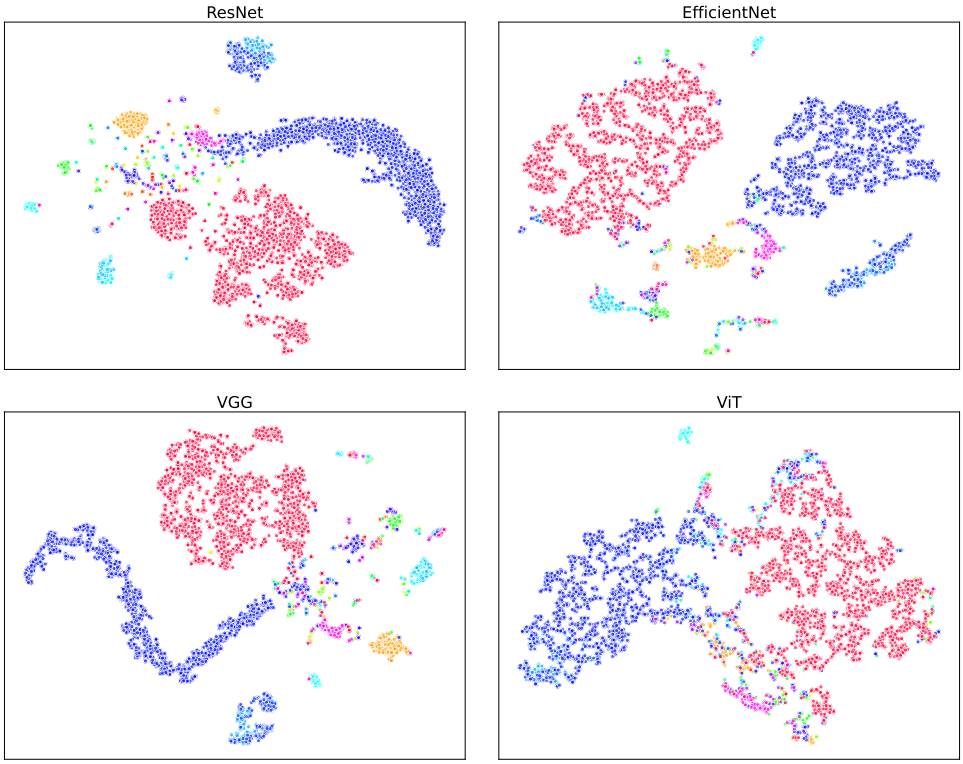
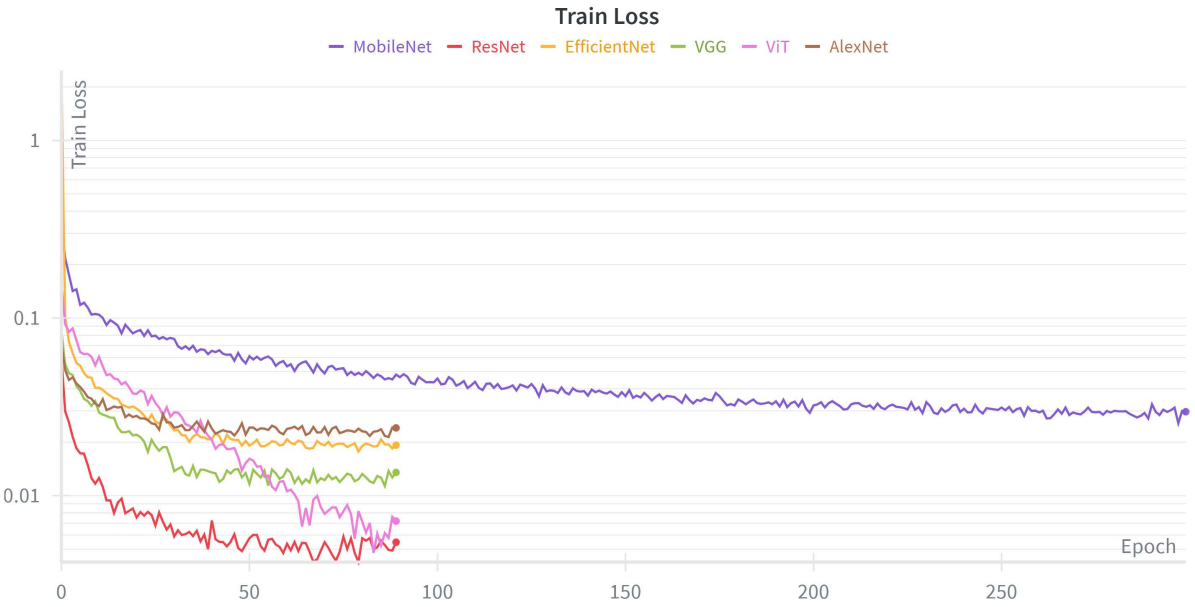


Figure 6.2: TSNE visualization of the Test ZSL scenario. These models were trained on the same cross-validation fold. Each dot represents a document positioned in this two-dimensional space through TSNE dimensionality reduction. Each color represents a different class.

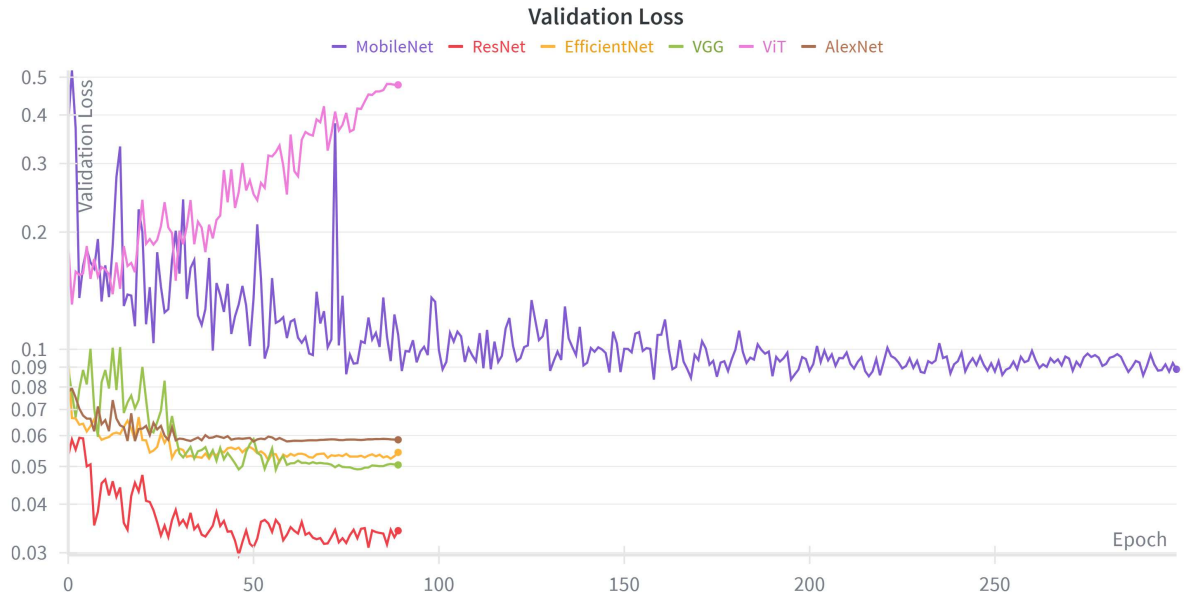
Table 6.8: Summary of error rates for each architectures in the GZSL scenario, for seen or unseen pairs.

Architecture	**Seen Error Rate**	**Unseen Error Rate**
AlexNet	13.06	13.84
EfficientNet	5.20	10.28
MobileNet	14.21	14.42
ResNet	3.07	8.64
VGG	15.14	13.99
ViT	13.11	16.00

that the ViT architecture is the only one to diverge on the validation loss score, while overfitting on the train loss score, explaining why the GZSL error decreased. Table 6.8 shows the different error rates for two different categories: the error rate on pairs that their classes were not seen during training, and the error rate on pairs where at least one of the elements was seen during training. With the exception of VGG, every architecture performed better on the seen scenario, which indicates that the increased difficulty on the GZSL scenario is, indeed, the culprit.



(a) Training loss.

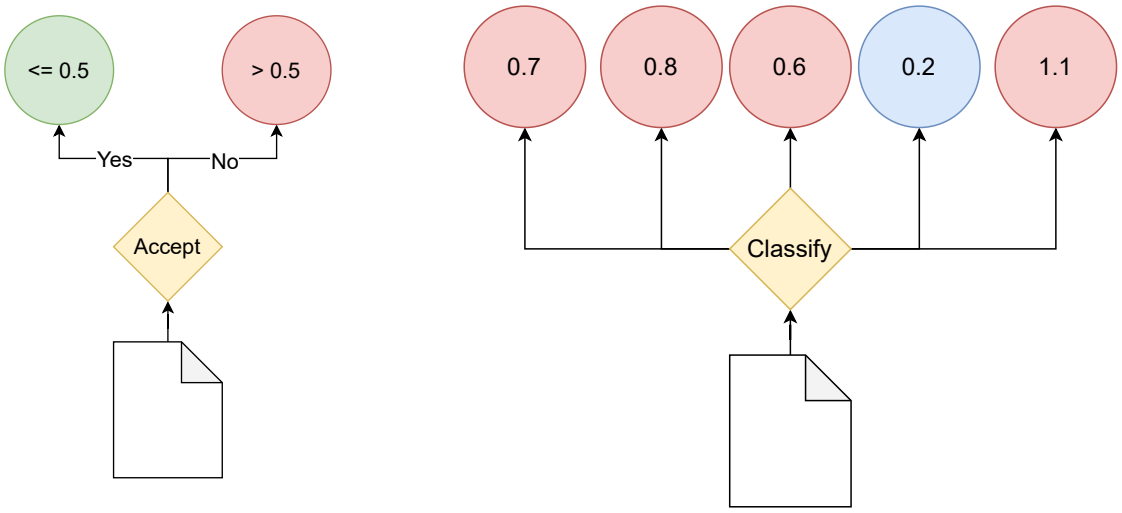


(b) Validation loss.

Figure 6.3: Loss curves over training epochs for all architectures in the ZSL scenario, showing both training (a) and validation (b) performance.

6.5 Industry Uses

With the trained models shown in this chapter, an industry system can leverage the metric learning nature of the solution to achieve zero-shot DIC in a couple of different ways. First, this model can be used in a verification system, within the same framework used for testing: given a document reference, and with a predefined acceptance threshold,



(a) Verification task. (b) Identification task.

Figure 6.4: Comparative illustration between the verification (a) and identification (b) tasks. The verification system approves documents when the distance between the references and the query is lower than or equal to 0.5, and rejects otherwise. The identification system chooses the nearest neighbor to classify the query document.

a query document must be accepted if the distance to the reference is smaller than the threshold, and rejected otherwise. As mentioned in 1, this task can be viewed as a binary classification problem, as the system will either accept or reject the document.

Second, the model can be used in an identification system, which becomes similar to a multiclass classification problem: given a document reference for each available class, and with a predefined acceptance threshold, a query document must be assigned to the closest class with respect to the embedding distances, or detected as outside of the label space if no class is within the predefined threshold. Figure 6.4 illustrates the difference between the verification and identification tasks.

Both tasks can be facilitated by a few design choices. For example, using multiple ground-truth references per class and computing the distance between the query document and the class centroid can improve classification robustness, as it prevents outlier samples from disproportionately representing an entire class. This decision demands more human annotation per class.

Another possible choice is to use a different threshold for each class. While the model is trained to have all classes share the same acceptance threshold, it is natural that some classes in a real-world scenario have less intra-class variance than others. This, of course, increases the maintenance required per class registered in the system, as calculating a unique threshold demands testing. On the other hand, it may be of interest to change

the threshold in order to increase the false rejection rate in scenarios where rejecting a correct document is cheaper than accepting an incorrect document, for example.

All these examples share the same need for a database to store the references. It is possible to store the embeddings directly, instead of the document itself, to reduce the number of inferences the model must process. In the verification scenario, the complexity of an inference should increase linearly with the number of references in the class, i.e., the time to calculate the centroid of the class, $O(r)$, where r is the number of references in a given class (the centroid coordinates can be stored to save time). Following the same strategy, in the identification scenario, along with the cost to calculate the centroid, there is also the cost of comparisons per class to find the nearest neighbor, which results in $O(rc)$, where c is the number of classes in the system.

Chapter 7

Conclusion

This work tackles an understudied area of the document analysis field: ZSL-DIC. One of the biggest obstacles of this area is the lack of a specialized dataset. Therefore, this work starts by introducing a novel document image dataset, LA-CDIP, categorized mainly by the layout information of the document image, which allows for knowledge generalization in a zero-shot setting. This dataset is constructed with a labeling framework assisted by DL models that were trained on the dataset itself. This labeling framework works in iterations, where the current loop should be easier to label manually than the previous iteration, allowing the manual labeling to become easier, as the iterations go by. The resulting dataset is specialized in ZSL-DIC, providing enough class diversity to allow DL models trained from scratch to achieve a high generalization potential.

To use this dataset in the ZSL-DIC task, this work also presents VDM, which is a framework that frames the classification problem as a matching problem, using embedding similarity to achieve the ZSL paradigm. Therefore, by having a set of ground-truth references, this method can classify any document class in a zero-shot scenario, if correctly trained. To further encourage research with this framework, the dataset is extensively benchmarked by training siamese networks with a wide variety of neural backbones, as well as an ample set of hyperparameters.

In conclusion, this work developed an understudied branch in the DIC research, providing a novel dataset and a method, resulting in a classification solution with real-world applications.

7.1 Future Works

There are some known limitations of the work and plans to tackle them. First, as mentioned, the architecture complexity is currently an obstacle, as the dataset is still relatively small. To solve this problem, it is already an ongoing work to keep iterating with the

techniques presented here, increasing both the number of classes and the number of documents. To further increase the quality of the dataset, DL-assisted dataset verification methods are being developed, where the classes are matched against each other to find possible duplicate patterns. This technique will be able to pinpoint possible human mistakes and increase the parallelization of human effort in creating this dataset. Another possible dataset enhancement is to increase the number of document sources by gathering documents from sources other than the RVL-CDIP dataset, increasing the generalization of the trained models.

With regard to the training framework, a possible step to enhance the results is to explore data augmentation techniques, which should alleviate the burden of the dataset labeling process. In the metric learning topic, more advanced techniques, such as tuple mining [64], are mapped as future experiments, together with exploring the combination of different loss functions and distance functions.

References

[1] Kay, Anthony: *Tesseract: an open-source optical character recognition engine*. Linux J., 2007(159):2, July 2007, ISSN 1075-3583. 1

[2] Mindee: *docTR: Document Text Recognition*, 2021. `https://github.com/mindee/doctr`. 1

[3] Liu, Li, Zhiyu Wang, Taorong Qiu, Qiu Chen, Yue Lu, and Ching Y. Suen: *Document image classification: Progress over two decades*. Neurocomputing, 453:223–240, September 2021, ISSN 0925-2312. `https://www.sciencedirect.com/science/article/pii/S0925231221006925`, visited on 2025-03-06. 1

[4] Bakkali, Souhail, Zuheng Ming, Mickaël Coustaty, and Marçal Rusiñol: *EAML: ensemble self-attention-based mutual learning network for document image classification*. Int. J. Doc. Anal. Recognit., 24(3):251–268, September 2021, ISSN 1433-2833. `https://doi.org/10.1007/s10032-021-00378-0`, visited on 2025-06-03. 2

[5] Harley, A. W., A. Ufkes, and K. G. Derpanis: *Evaluation of deep convolutional nets for document image classification*. In *2015 International Conference on Document Analysis and Recognition (ICDAR)*, pages 991–995. IEEE, 2015. 2, 20

[6] Xian, Yongqin, Christoph H. Lampert, Bernt Schiele, and Zeynep Akata: *Zero-Shot Learning—A Comprehensive Evaluation of the Good, the Bad and the Ugly*. IEEE Transactions on Pattern Analysis and Machine Intelligence, 41(9):2251–2265, September 2019, ISSN 1939-3539. `https://ieeexplore.ieee.org/document/8413121`, visited on 2025-03-06, Conference Name: IEEE Transactions on Pattern Analysis and Machine Intelligence. 2, 9, 29

[7] Sinha, Sankalp, Muhammad Saif Ullah Khan, Talha Uddin Sheikh, Didier Stricker, and Muhammad Zeshan Afzal: *CICA: Content-Injected Contrastive Alignment for Zero-Shot Document Image Classification*. In *International Conference on Document Analysis and Recognition*, pages 124–141. Springer, 2024. 3, 23

[8] Scius-Bertrand, Anna, Michael Jungo, Lars Vögtlin, Jean Marc Spat, and Andreas Fischer: *Zero-Shot Prompting and Few-Shot Fine-Tuning: Revisiting Document Image Classification Using Large Language Models*. In Antonacopoulos, Apostolos, Subhasis Chaudhuri, Rama Chellappa, Cheng Lin Liu, Saumik Bhattacharya, and Umapada Pal (editors): *Pattern Recognition*, pages 152–166, Cham, 2025. Springer Nature Switzerland, ISBN 978-3-031-78495-8. 3, 23

[9] Macedo, Lucas, João Paulo Costa, João Pedro Felix De Almeida, Pedro Freitas, and Li Weigang: *Towards Zero-Shot Document Image Classification*. In *2025 38th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*, pages 1–6, 2025. 4

[10] De Almeida Bandeira Macedo, Lucas, Joao Paulo Vieira Costa, Joao Pedro Felix De Almeida, Pedro Garcia Freitas, and Li Weigang: *Visual Document Matching for Zero-Shot Document Classification*. In Jin, Lianwen, Richard Zanibbi, and Veronique Eglin (editors): *Document Analysis and Recognition – ICDAR 2025 Workshops*, pages 192–208, Cham, 2026. Springer Nature Switzerland, ISBN 978-3-032-09368-4. 4, 31

[11] Breiman, Leo: *Random Forests*. Mach. Learn., 45(1):5–32, October 2001, ISSN 0885-6125. https://doi.org/10.1023/A:1010933404324. 7

[12] Chen, Tianqi and Carlos Guestrin: *XGBoost: A Scalable Tree Boosting System*. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, page 785–794, New York, NY, USA, 2016. Association for Computing Machinery, ISBN 9781450342322. https://doi.org/10.1145/2939672.2939785. 7

[13] Deng, Jia, Wei Dong, Richard Socher, Li Jia Li, Kai Li, and Li Fei-Fei: *ImageNet: A large-scale hierarchical image database*. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. 7, 15, 16, 18

[14] Brown, Tom B., Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei: *Language Models are Few-Shot Learners*, July 2020. http://arxiv.org/abs/2005.14165, visited on 2025-06-08, arXiv:2005.14165 [cs]. 9

[15] Lampert, Christoph H., Hannes Nickisch, and Stefan Harmeling: *Learning to detect unseen object classes by between-class attribute transfer*. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 951–958, 2009. 9

[16] Kulis, Brian: *Metric Learning: A Survey*. 5(4):287–364, ISSN 1935-8237, 1935-8245. https://www.nowpublishers.com/article/Details/MAL-019, visited on 2018-11-01. 10, 11

[17] Ilina, Olga, Vadim Ziyadinov, Nikolay Klenov, and Maxim Tereshonok: *A Survey on Symmetrical Neural Network Architectures and Applications*. Symmetry, 14(7):1391, July 2022, ISSN 2073-8994. https://www.mdpi.com/2073-8994/14/7/1391, visited on 2025-11-06, Publisher: Multidisciplinary Digital Publishing Institute. 11

[18] He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun: *Deep Residual Learning for Image Recognition*. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, June 2016. https://ieeexplore.ieee.org/document/7780459, visited on 2025-03-06, ISSN: 1063-6919. 11, 16, 24

[19] Chopra, S., R. Hadsell, and Y. LeCun: *Learning a similarity metric discriminatively, with application to face verification*. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 539–546 vol. 1, June 2005. https://ieeexplore.ieee.org/document/1467314, visited on 2025-03-06, ISSN: 1063-6919. 12

[20] Hoffer, Elad and Nir Ailon: *Deep metric learning using Triplet network*, December 2018. http://arxiv.org/abs/1412.6622, visited on 2025-06-09, arXiv:1412.6622 [cs]. 12

[21] Bishop, Christopher M.: *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, July 2006, ISBN 978-0-387-31073-2. 14

[22] MacQueen, J.: *Some methods for classification and analysis of multivariate observations*. 1967. https://www.semanticscholar.org/paper/Some-methods-for-classification-and-analysis-of-MacQueen/ac8ab51a86f1a9ae74dd0e4576d1a019f5e654ed, visited on 2025-11-18. 14

[23] Lloyd, S.: *Least squares quantization in PCM*. IEEE Transactions on Information Theory, 28(2):129–137, March 1982, ISSN 1557-9654. https://ieeexplore.ieee.org/document/1056489, visited on 2025-11-18. 14

[24] Ward Jr, Joe H: *Hierarchical grouping to optimize an objective function*. Journal of the American statistical association, 58(301):236–244, 1963. Publisher: Taylor & Francis. 14, 15, 26

[25] Ronneberger, Olaf, Philipp Fischer, and Thomas Brox: *U-Net: Convolutional Networks for Biomedical Image Segmentation*. In Navab, Nassir, Joachim Hornegger, William M. Wells, and Alejandro F. Frangi (editors): *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pages 234–241, Cham, 2015. Springer International Publishing, ISBN 978-3-319-24574-4. 15

[26] Redmon, Joseph, Santosh Divvala, Ross Girshick, and Ali Farhadi: *You Only Look Once: Unified, Real-Time Object Detection*. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, June 2016. https://ieeexplore.ieee.org/document/7780460, visited on 2025-10-18, ISSN: 1063-6919. 15

[27] LeCun, Y., B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel: *Backpropagation Applied to Handwritten Zip Code Recognition*. Neural Computation, 1(4):541–551, December 1989, ISSN 0899-7667. https://ieeexplore.ieee.org/document/6795724, visited on 2025-11-17. 15

[28] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton: *ImageNet classification with deep convolutional neural networks*. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 1*, volume 1 of *NIPS'12*, pages 1097–1105, Red Hook, NY, USA, December 2012. Curran Associates Inc. 16, 18, 24

[29] Simonyan, Karen and Andrew Zisserman: *Very Deep Convolutional Networks for Large-Scale Image Recognition*. In Bengio, Yoshua and Yann LeCun (editors): *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. http://arxiv.org/abs/1409.1556, visited on 2025-03-07. 16, 18, 24

[30] Bengio, Y., P. Simard, and P. Frasconi: *Learning long-term dependencies with gradient descent is difficult*. IEEE Transactions on Neural Networks, 5(2):157–166, March 1994, ISSN 1941-0093. https://ieeexplore.ieee.org/document/279181, visited on 2025-11-16. 16

[31] Pascanu, Razvan, Tomas Mikolov, and Yoshua Bengio: *On the difficulty of training recurrent neural networks*. In *Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28*, ICML'13, pages III–1310–III–1318, Atlanta, GA, USA, June 2013. JMLR.org. 16

[32] Howard, Andrew G., Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam: *MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications*, 2017. https://arxiv.org/abs/1704.04861. 16

[33] Sandler, Mark, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang Chieh Chen: *MobileNetV2: Inverted Residuals and Linear Bottlenecks*. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4510–4520, 2018. 16

[34] Howard, Andrew, Mark Sandler, Grace Chu, Liang Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, Quoc V. Le, and Hartwig Adam: *Searching for MobileNetV3*, November 2019. http://arxiv.org/abs/1905.02244, visited on 2025-03-06, arXiv:1905.02244 [cs]. 16, 24

[35] Tan, Mingxing and Quoc Le: *EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks*. In *Proceedings of the 36th International Conference on Machine Learning*, pages 6105–6114. PMLR, May 2019. https://proceedings.mlr.press/v97/tan19a.html, visited on 2025-03-06, ISSN: 2640-3498. 16, 24

[36] Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin: *Attention Is All You Need*, August 2023. http://arxiv.org/abs/1706.03762, visited on 2025-06-21, arXiv:1706.03762 [cs]. 17, 37

[37] Kojima, Takeshi, Shixiang (Shane) Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa: *Large Language Models are Zero-Shot Reasoners*. In Koyejo, S., S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (editors): *Advances in Neural Information Processing Systems*, volume 35, pages 22199–22213. Curran Associates, Inc., 2022. https://proceedings.neurips.cc/paper_files/paper/2022/file/8bb0d291acd4acf06ef112099c16f326-Paper-Conference.pdf. 17

[38] Dosovitskiy, Alexey, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby: *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. https://openreview.net/forum?id=YicbFdNTTy. 17, 24

[39] Nagy, G.: *Twenty years of document image analysis in PAMI*. IEEE Transactions on Pattern Analysis and Machine Intelligence, 22(1):38–62, January 2000, ISSN 1939-3539. https://ieeexplore.ieee.org/document/824820, visited on 2025-11-20. 19

[40] Abdallah, A., D. Eberharter, Z. Pfister, and A. Jatowt: *A survey of recent approaches to form understanding in scanned documents*. Artificial Intelligence Review, 57:342, 2024. 19

[41] Zhong, Xu, Jianbin Tang, and Antonio Jimeno-Yepes: *PubLayNet: Largest Dataset Ever for Document Layout Analysis*. In *2019 International Conference on Document Analysis and Recognition, ICDAR 2019, Sydney, Australia, September 20-25, 2019*, pages 1015–1022. IEEE, 2019. https://doi.org/10.1109/ICDAR.2019.00166. 19, 20

[42] Pfitzmann, Birgit, Christoph Auer, Michele Dolfi, Ahmed S. Nassar, and Peter Staar: *DocLayNet: A Large Human-Annotated Dataset for Document-Layout Segmentation*. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '22, pages 3743–3751, New York, NY, USA, August 2022. Association for Computing Machinery, ISBN 978-1-4503-9385-0. https://dl.acm.org/doi/10.1145/3534678.3539043, visited on 2025-06-20. 20

[43] Mathew, M., A. Kembhavi, J. Schreiber, D. Batra, D. Parikh, and M. Bansal: *DocVQA: A dataset for VQA on document images*. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 2200–2209, 2021. 20

[44] Jaume, G., H. K. Ekenel, and J. P. Thiran: *FUNSD: A dataset for form understanding in noisy scanned documents*. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 1–6. IEEE, 2019. 20

[45] Huang, Zheng, Kai Chen, Jianhua He, Xiang Bai, Dimosthenis Karatzas, Shijian Lu, and C. V. Jawahar: *ICDAR2019 Competition on Scanned Receipt OCR and Information Extraction*. In *2019 International Conference on Document Analysis and*

Recognition (ICDAR), pages 1516–1520, September 2019. https://ieeexplore.ieee.org/document/8977955, visited on 2025-03-07, ISSN: 2379-2140. 20

[46] Park, Seunghyun, Seung Shin, Byeongchang Kim, Junbum Cha, and Hwalsuk Lee: *CORD: A Consolidated Receipt Dataset for Post-OCR Parsing.* In *Document Intelligence Workshop at NeurIPS 2019*, 2019. https://arxiv.org/abs/1908.07414. 20

[47] Xu, Yiheng, Tengchao Lv, Lei Cui, Guoxin Wang, Yijuan Lu, Dinei Florêncio, Cha Zhang, and Furu Wei: *LayoutXLM: Multimodal Pre-training for Multilingual Visually-rich Document Understanding.* CoRR, abs/2104.08836, 2021. https://arxiv.org/abs/2104.08836, arXiv: 2104.08836. 20

[48] Lewis, D. D., S. Agarwal, N. Kappagantula, and P. Vora: *The IIT-CDIP test collection.* Technical report, Illinois Institute of Technology, 2006. 20

[49] Larson, Stefan, Gordon Lim, and Kevin Leach: *On Evaluation of Document Classification with RVL-CDIP.* In Vlachos, Andreas and Isabelle Augenstein (editors): *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2665–2678, Dubrovnik, Croatia, May 2023. Association for Computational Linguistics. https://aclanthology.org/2023.eacl-main.195/, visited on 2025-06-17. 21

[50] Kumar, Jayant, Peng Ye, and David Doermann: *Structural Similarity for Document Image Classification and Retrieval.* Pattern Recognition Letters, 43:119–126, 2014. 21

[51] Khalifa, Muhammad, Yogarshi Vyas, Shuai Wang, Graham Horwood, Sunil Mallya, and Miguel Ballesteros: *Contrastive Training Improves Zero-Shot Classification of Semi-structured Documents.* In Rogers, Anna, Jordan Boyd-Graber, and Naoaki Okazaki (editors): *Findings of the Association for Computational Linguistics: ACL 2023*, pages 7499–7508, Toronto, Canada, July 2023. Association for Computational Linguistics. https://aclanthology.org/2023.findings-acl.473/. 22, 23, 26

[52] OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan

Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Jan Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine B. Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, C. J. Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph: *GPT-4 Technical Report*, 2024. `https://arxiv.org/abs/2303.08774`, _eprint: 2303.08774. 23

[53] Liu, Yinhan, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov: *RoBERTa: A Robustly Optimized BERT Pretraining Approach*, July 2019. `http://arxiv.org/abs/1907.11692`, visited on 2025-11-22, arXiv:1907.11692 [cs]. 23

[54] Radford, Alec, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sand-

hini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever: *Learning Transferable Visual Models From Natural Language Supervision*. In Meila, Marina and Tong Zhang (editors): *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 8748–8763. PMLR, 2021. http://proceedings.mlr.press/v139/radford21a.html. 23

[55] Jiang, Jiajia, Songxuan Lai, Lianwen Jin, and Yecheng Zhu: *DsDTW: Local Representation Learning With Deep soft-DTW for Dynamic Signature Verification*. IEEE Transactions on Information Forensics and Security, 17:2198–2212, 2022, ISSN 1556-6021. https://ieeexplore.ieee.org/document/9787558, visited on 2025-06-14. 24

[56] Settles, Burr: *Active Learning Literature Survey*. Technical Report, University of Wisconsin-Madison Department of Computer Sciences, 2009. https://minds.wisconsin.edu/handle/1793/60660, visited on 2025-10-15, Accepted: 2012-03-15T17:23:56Z. 27, 28

[57] Rand, William M.: *Objective Criteria for the Evaluation of Clustering Methods*. Journal of the American Statistical Association, 66(336):846–850, 1971, ISSN 01621459, 1537274X. http://www.jstor.org/stable/2284239, visited on 2025-12-01. 28

[58] Strehl, Alexander and Joydeep Ghosh: *Cluster ensembles — a knowledge reuse framework for combining multiple partitions*. J. Mach. Learn. Res., 3(null):583–617, March 2003, ISSN 1532-4435. https://doi.org/10.1162/153244303321897735. 28

[59] Wong, Tzu Tsung and Po Yang Yeh: *Reliable accuracy estimates from k-fold cross validation*. IEEE Transactions on Knowledge and Data Engineering, 32(8):1586–1594, 2019. Publisher: IEEE. 29

[60] Agrawal, Pinki, Ravikant Kapoor, and Sanjay Agrawal: *A hybrid partial fingerprint matching algorithm for estimation of Equal error rate*. In *2014 IEEE International Conference on Advanced Communications, Control and Computing Technologies*, pages 1295–1299, 2014. 30

[61] Hofbauer, Heinz and Andreas Uhl: *Calculating a boundary for the significance from the equal-error rate*. In *2016 International Conference on Biometrics (ICB)*, pages 1–4, 2016. 30

[62] Paszke, Adam, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala: *PyTorch: An Imperative Style, High-Performance Deep Learning Library*, December 2019. http://arxiv.org/abs/1912.01703, visited on 2025-06-21, arXiv:1912.01703 [cs]. 32

[63] Hestness, Joel, Sharan Narang, Newsha Ardalani, Gregory Diamos, Heewoo Jun, Hassan Kianinejad, Md. Mostofa Ali Patwary, Yang Yang, and Yanqi Zhou: *Deep Learning Scaling is Predictable, Empirically*, 2017. https://arxiv.org/abs/1712.00409. 36

[64] Schroff, Florian, Dmitry Kalenichenko, and James Philbin: *FaceNet: A unified embedding for face recognition and clustering.* In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 815–823, June 2015. https://ieeexplore.ieee.org/document/7298682, visited on 2025-06-17, ISSN: 1063-6919. 43