Kelly Lais Wiggers

Deep Learning Approaches for Image Retrieval and Pattern Spotting in Ancient Documents

Curitiba
2019

Kelly Lais Wiggers

Deep Learning Approaches for Image Retrieval and Pattern Spotting in Ancient Documents

Thesis presented to the Post-graduate Program in Informatics (PPGIa) of the Pontifical Catholic University of Parana as a partial requirement to obtain the title of Phd degree in Informatics.

Advisor: Prof. Dr. Alceu de Souza Britto Jr.
Coadvisor: Prof. Dr. Laurent Heutte

Curitiba
2019

Dados da Catalogação na Publicação
Pontifícia Universidade Católica do Paraná
Sistema Integrado de Bibliotecas – SIBI/PUCPR
Biblioteca Central
Luci Eduarda Wielganczuk – CRB 9/1118

W655d
2019

Wiggers, Kelly Lais
Deep learning approaches for image retrieval and pattern spotting in ancient documents / Kelly Lais Wiggers ; advisor: Alceu de Souza Britto Jr. ; coadivisor: Laurent Heutte. – 2019.
xx, 153 f. : il. ; 30 cm

Tese (doutorado) – Pontifícia Universidade Católica do Paraná, Curitiba, 2019
Bibliografia: f. 117-134

1. Redes neurais (Computação). 2. Processamento de imagens. I. Britto Junior, Alceu de Souza. II. Heutte, Laurent. III. Pontifícia Universidade Católica do Programa de Pós-Graduação em Informática. IV. Título.

CDD 22. ed. – 006.32

DECLARAÇÃO

Declaro para os devidos fins que a aluna KELLY LAIS WIGGERS, defendeu sua tese de doutorado intitulada **"Deep Learning Approaches for Image Retrieval and Pattern Spotting in Ancient Documents"**, na área de concentração Ciência da Computação, no dia 19 de setembro de 2019, no qual foi aprovada.

Declaro ainda que foram feitas todas as alterações solicitadas pela Banca Examinadora, cumprindo todas as normas de formatação definidas pelo Programa.

Por ser verdade, firmo a presente declaração.

Curitiba, 20 de novembro de 2019.

Programa de Pós-Graduação em Informática
PPGIa
PUCPR

[illegible]

Prof. Dr. Emerson Cabrera Paraiso
Coordenador do Programa de Pós-Graduação em Informática
Pontifícia Universidade Católica do Paraná

Para Luiz, Serlei e Fábio.

Acknowledgements

I thank God for my life, providing me the strength, wisdom, and tranquility to move on with my goals and not be discouraged by the difficulties.

I thank my mother Serlei Correia Wiggers and my father Luiz Auri Wiggers, who always encouraged and helped me in all possible ways, understanding my moments of absence to complete this stage of my life. To Fabio Leandro Janiszevski, for the patience, support, and encouragement so that I never gave up of my dreams, staying by my side at all the good and bad times.

To my advisors Dr. Alceu de Souza Britto Junior, Dr. Laurent Heutte and Professor Dr. Alessandro Lameiras Koerich, for the opportunity, encouragement and contribution in this work developed.

I am grateful to all the professors and colleagues who contributed to the progress of my studies, especially my friends from PPGIa Gustavo Bonacina and Márcia Dadalto Pascutti for their support at all times I needed it. I also thank my friend Jehan Esheh from ÉTS for all the help and attention in Montreal.

To the Pontifical Catholic University of Paraná for the opportunity and structure for the studies. This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

To Global Affairs Canada, which enabled the development of sandwich-period research at the *École de technologie supérieure* (ÉTS) in Montreal through the *Emerging Leaders in the Americas Program.*

To all who directly or indirectly contributed to the elaboration of this doctoral thesis.

Contents

List of Figures

List of Tables

List of abbreviations

AP	Average Precision
BING	Binarized Normed Gradients
BoVW	Bag of Visual Words
CBIR	Content Based Image Retrieval
CDIP	Complex Document Image Processing
CK1	Campana- Keogh-1
CNN	Convolutional Neural Network
DB	DeepBox
DL	Deep Learning
DNN	Deep Neural Network
DTW	Dynamic Time Warping
EB	Edges Boxes
ES	Exhaustive Search
FV	Fisher Vector
IoU	Intersection over Union
IR	Image Retrieval
IVF	Inverted File Structure
HoG	Histograms of Oriented Gradients
HSV	Hue Saturation Value
LSH	Locality Sensitive Hashing

LWP	Longest Weighted Profile
mAP	mean AP
MCG	Multiscale Combinatorial Grouping
MLP	Multilayer Perceptron
PS	Pattern Spotting
RGB	Red Green Blue
SNN	Siamese Neural Network
SCNN	Siamese Convolution Neural Network
SIFT	Scale-Invariant Feature Transform
SGD	Stochastic Gradient Descent
SS	Selective Search
SURF	Speeded Up Robust Features
SVM	Support Vector Machine
SW	Sliding Windows
VLAD	Vector of Locally Aggregated Descriptor
TL	Transfer Learning
TM	Template Matching
WS	Word Spotting

Resumo

Este trabalho descreve abordagens para recuperação de objetos baseados em *Contend-Based Image Retrieval* (CBIR). O desafio é procurar por objetos em imagens de documentos que podem variar em termos de cor, forma, tamanho e qualidade, bem como a localização. Tal variabilidade torna o problema não trivial. Assim, foram propostas duas abordagens para CBIR em imagens de documentos mediante aprendizagem profunda. O principal desafio é realizar a tarefa de recuperação sem conhecimento prévio sobre as imagens a serem recuperadas. A primeira abordagem utiliza um modelo de Rede Neural Convolucional (CNN) pré-treinado, que é ajustado para obter uma representação compacta, porém discriminante, de imagens de consultas e candidatos. A segunda abordagem utiliza uma CNN Siamesa treinada com um subconjunto de pares de imagens da base ImageNet, para fornecer os mapas de características baseados em similaridade. Em ambos os métodos, o esquema de representação considera mapas de características de tamanhos diferentes que são avaliados em termos de desempenho. Um protocolo experimental robusto usando dois conjuntos de dados públicos (Tobacoo-800 e DocExplore) mostrou que os métodos propostos se comparam favoravelmente ao estado da arte. Assim, as abordagens propostas são esquemas interessantes para encorajar outros pesquisadores a continuarem trabalhando com diferentes problemas de recuperação de objetos.

Palavras-chave: Redes Neurais Convolucionais, Extração de Características, Redes Siamesas, Geração de Candidatos

Abstract

This work describes approaches for content-based graphical object retrieval and spotting in document images (CBIR). The challenge is to search for graphical objects in document images that can vary in terms of color, shape, size, and quality as well as the localization. Such variability in the graphical objects makes the problem not trivial. Thus, we propose two approaches for CBIR and pattern spotting in document images using deep learning. The main challenge is to perform a retrieval task without previous knowledge about the images to be retrieved. The first approach uses a pre-trained Convolution Neural Network model to cope with the lack of training data, which is fine-tuned to achieve a compact and discriminant representation of queries and candidate images. The second approach uses a Siamese Convolution Neural Network trained on a previously prepared subset of image pairs from the ImageNet dataset to provide the similarity-based feature maps. In both methods, the learned representation scheme considers feature maps of different sizes which are evaluated in terms of performance. A robust experimental protocol using two public datasets (Tobacoo800 and DocExplore) has shown that the proposed methods compare favorably against the state-of-the-art. Thus, the proposed approaches are interesting schemes to further encourage other researchers to continue tackle with different retrieval and spotting problems.

Keywords: Convolutional Neural Network, Feature extraction, Siamese Networks, Candidate Generation

Chapter 1

Introduction

The recent advances in technology have made it possible to store large amounts of information in digital libraries composed of images and videos related to our cultural heritage. Such storage was done through the digitalization of documents, pictures, books, and entire historical collections.

More than a valuable resource for researchers in the areas of Pattern Recognition and Image Retrieval, some additional steps to better explore these libraries may reveal new perceptions, knowledge, and context (ZHU; KEOGH, 2010). Thus, the development of solutions dedicated to retrieving objects from digital libraries based on their content is crucial (YARLAGADDA et al., 2011), and, the objects of an image, such as stamps, graphs, decorative objects, logos, dropped initial letters, and others may contain relevant information related to its source.

Image Retrieval (IR) and Pattern Spotting (PS) are very active research topics in the field of Pattern Recognition. The main motivation is the increasing demand for solutions capable of performing the retrieval of an image, or specific objects in it, from wide digital libraries stored in the last decades of modern society. Some efforts in this direction are the recent studies described in IR and PS (THUY et al., 2017; WU et al., 2017b; XU et al., 2017; EN et al., 2016c), where IR consists of finding relevant image candidates in digital collections based on a given query represented by the whole image or just by a pattern available on it, while additional information is necessary for the PS, which is related to the exact location of the query in the retrieved images.

Different techniques can be used to retrieve information from a collection of documents, but they are usually organized in a similar two-step process (MARINAI; MIOTTI; SODA, 2011), as follows: a) in an offline step, object proposals (image candidates) are extracted from the document images and indexed using a suitable representation schema (a feature vector); b) in an online phase, given an input query image, a measure of similarity is used to compare it with image candidates extracted from the stored documents, returning a ranked list.

Therefore, some difficulties can also contribute to the unsatisfactory performance of IR and PS methods, such as:

- the document image collections can present documents degradation due the poor quality of the scanning process;
- the document image collections may have different nature, and they can present small structures;
- the objects may be overlaped in the document, such as signatures and stamps;
- there is a lack of previous information about the objects in the document image collections;
- some queries can vary in terms of color, shape and texture or noisy environment;
- the queries and candidates may present different sizes.

A recent and exciting challenge on IR and PS has been the need for performing the retrieval task without previous knowledge about the images or patterns to be retrieved. The idea is to produce generic solutions able to work on different digital image libraries. This is an interesting challenge that increases considerably the level of difficulty of the retrieval task.

The lack of contextual information combined with the wide variability in terms of scale, color, and texture of the patterns present in document collections, which may include seals, logos, faces, and initial letters, make the definition of a robust representation scheme (feature extraction) a real challenging problem. In the literature, one may find some promising results of traditional representation methods based on local descriptors such as bag-of-visual-words (BoVW) (SIVIC; ZISSERMAN, 2003). However, some significant contributions have been recently made through the use of deep learning-based methods, mainly by doing feature extraction using Convolutional Neural Networks (CNN)(BABENKO; LEMPITSKY, 2015) (YUE-HEI; YANG; DAVIS, 2015) (GORDO et al., 2016). However, the main challenge is to develop a generic solution that is independent of the query images to work on different document collections by using deep architectures to provide a robust representation.

With this in mind, in this work we evaluated the following hypotheses: a) A pre-trained Convolutional Neural Network (CNN) model constructed to deal with the lack of training data can provide a discriminant representation of queries and image candidates for IR and PS tasks applied in document image collections; b) Deep metric based on Siamese Convolutional Neural Network (SCNN) trained on a general image dataset can be used to construct dataset-independent solutions for IR and PS tasks in the context of document image collections.

1.1 Objectives

The main objective of this work is to develop image retrieval and pattern spotting solutions for document image collections while reducing their dependency on previous knowledge of the image dataset in hands. To this end, we plan to represent the problem (query and image candidates) using deep learning. To reach such objective, the following efforts are necessary:

1. Elaborate a literature survey about IR and PS, deep features, and searching for object candidates;

2. Define a method for object candidate generation from the document image collections;
3. Define a deep representation that should be compact and discriminant for queries and image candidates;
4. Implement the solutions using parallel computing for fast image retrieval and spotting;
5. Evaluate the proposed solutions of IR and PS with a benchmarking in terms of precision/recall, comparing the proposed approaches with the state-of-the-art .

1.2 Proposal

To reach our main objective, in this work we describe two approaches based on deep learning. In the first proposed approach, we applied a pre-trained CNN as feature extractor considering feature maps of different sizes and explored the idea of obtaining a compact representation to provide a fast image retrieval. Thus, by transfer learning, we succeeded in obtaining a robust representation, while reducing the importance of the training phase which was represented in this case by a fine-tuning considering data augmentation on a small set of image queries.

Going deeper into the idea of avoiding training based on queries, in the second approach, two deep models based (CNN) were organized in a Siamese CNN and they were trained for feature extraction and similarity estimation. In both approaches, a strategy for reducing the dimensionality of the feature maps generated by the CNNs was proposed with the aim of reducing the computational cost of the retrieval and storage processes.

1.3 Contributions

The contributions of this thesis can be summarized, as follows:

- A CNN based method in which a pre-trained model is fine-tuned on a small set of query images reducing the dependence on previous knowledge of the used dataset;
- An additional implementation in the Selective Search algorithm to reduce noise and improve the quality of candidate images;
- A schema for feature map reduction to produce a compact and discriminant set of features to find the best trade-off between feature dimension and feature representation;
- A solution based on a implementation of a deep metric and trained in a Siamese CNN that allows us to avoid the training using queries and candidate images;
  - The weights of this trained network can be used in different solutions and datasets as a generic feature extractor model;
- The representation based on CNN models that reached competitive results when compared to state-of-the-art using a robust experimental protocol.

1.4 Publications

The contributions of this PhD work have been published in the following papers:

1. K. L. Wiggers, A. S. Britto, L. Heutte, A. L. Koerich and L. E. S. Oliveira, "Document Image Retrieval Using Deep Features," 2018 International Joint Conference on Neural Networks (IJCNN), Rio de Janeiro, 2018, pp. 3185-3192.
2. K. L. Wiggers, A. S. Britto, L. Heutte, A. L. Koerich and L. E. S. Oliveira, "Image Retrieval and Pattern Spotting using Siamese Neural Network",

2019 International Joint Conference on Neural Networks (IJCNN), Budapest, 2019.

3. K. L. Wiggers, A. S. Britto, L. Heutte, A. L. Koerich and L. E. S. Oliveira, "Deep Learning Approaches for Image Retrieval and Pattern Spotting in Ancient Documents". 2019. The paper is under review at Pattern Recognition Letters.

1.5 Organization

This document is organized into four chapters: **Chapter 2** presents the literature review of IR and PS. This chapter is organized into four sections, namely: Query Spotting; Pre-processing; Feature extraction based on Learning; Similarity measures and Distances; Parallel Computing; Final considerations.

In **Chapter 3** we present the proposed IR and PS methods for candidate generation, feature extraction approaches with CNN, the evaluation protocol, and the similarity measure. This chapter finishes with the description of the datasets used in our work.

The experiments are presented in **Chapter 4** with the results of our IR and PS tasks. We also compared the performance of the proposed IR and IS methods with those available in the literature.

The conclusions are presented in **Chapter 5**, where one may find the final considerations about the results and the directions of future work.

Chapter 2

Literature Review

The storage of documents, books, and other data collections results in a relevant source of information that can be explored by specialists from different areas. However, in traditional datasets, the document information is well organized in predefined structures, but in a dataset made up of document images, it is not so simple to define adequate queries due to the difficulty of understanding the semantics of the data. This challenging problem has received special attention from the information recovery community in the last few years, and various approaches for indexation and recuperation have been proposed.

Several techniques can be used to retrieve information from a collection of images. These techniques frequently follow a conventional process (MARINAI; MIOTTI; SODA, 2011), in this way: indexation of the candidate extracted from the images, and representation in an adequate feature space. This process can be accomplished in an offline phase; then during an online phase, the user conducts queries for the retrieval system and a measure of similarity is used to compare it with the candidates of the stored images, returning a ranked list. This process must be repeated until some stop condition is reached.

Therefore, this chapter presents the main definitions and processes selected to create a retrieval and spotting system. Section 2.1 presents the literature review on systems of query spotting and the main steps to implement a CBIR process.

Thus, Section 2.2 presents the pre-processing step with some algorithms from the literature to find object proposals (image candidates) given an input image and its applications. The feature extractor as a second step, responsible for the representation of images, is presented in Section 2.3, with emphasis placed on the use of deep learning, while similarity measures to complete the third step are presented in Section 2.4. Section 2.5 presents some concepts about parallel computing for IR to complement our research.

2.1 Query Spotting: image retrieval and pattern localization

The approach of interest for this study is known as Content Based Image Retrieval (*CBIR*). The main objective of the CBIR methods is to recuperate the information without previous knowledge about the domain. That is, the system must evaluate any variations in color, texture, shape, and localization of the query of the image collection. This query can be presented as a whole image or small areas of images. Datta et al. (2008) studied 300 articles related to CBIR systems published from 1995 to 2005. The authors describe the challenge of developing systems that are useful in the real world. In their opinion, aspects related to character extraction, feedback, and the evaluation process should receive special attention.

In other work (DOERMANN, 1998) one may find, a detailed report on the initial research on recuperating images from documents. Each study has a different focus, such as image retrieval in manuscripts, logos, medieval documents, signature. The authors conclude that the indexing and retrieval steps will make use of the powerful features offered in the context of the text, graphics, and images. In this way, the systems need to address complex trade-offs between image quality, retrieval recall, and precision and algorithm speed.

Figure 2.1 shows a general overview of a CBIR process. It begins with the input of a query image and ends with a list of retrieved images. For Wan et al. (2014), the performance of a CBIR system is crucially dependent on a robust process for feature extraction and an appropriate similarity measure. Such information based on content is extracted to retrieve similar images from the dataset. The similarity is computed based on the difference between two feature vectors (the query and the candidate). The IR system returns a ranked list, corresponding to the level of similarity between the query and all the candidates (IQBAL; ODETAYO; JAMES, 2012).

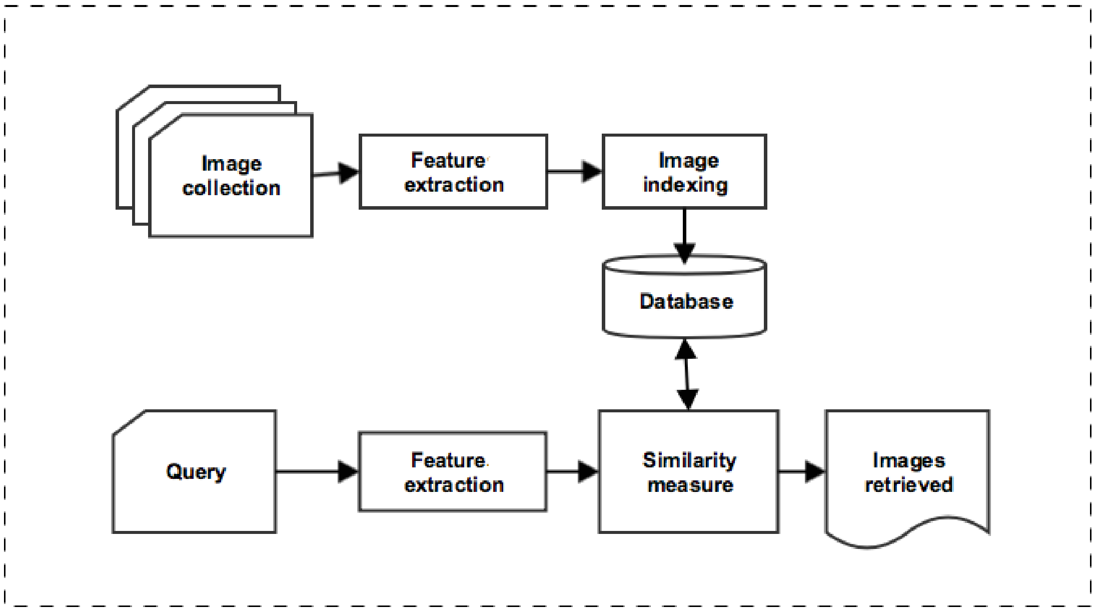
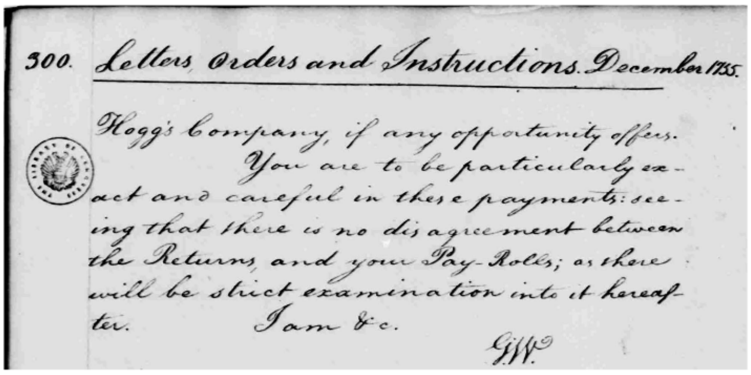


Figure 2.1: A typical CBIR system

Source: Adapted from Iqbal, Odetayo and James (2012)

A subclass of CBIR that aim to detect occurrences of a query and its precise location in a document image is query spotting. Due to noise and the complexity of the systems, image spotting tasks can be challenging, leading to false positives, which consequently can reduce the precision of the algorithm (CHATBRI; KWAN; KAMEYAMA, 2014). The spotting task can be based on Word Spotting, for retrieve of words or lines of text, or on Pattern Spotting for the recovery of other graphic objects from documents, such as decorative objects, coats of arm, or decorative letters. This section presents the approaches that can be used for these tasks, as well as the principal directions for this work.

The initial concept of spotting methods emerged in 1996 through Manmatha, Han and Riseman (1996), named Word Spotting (*WS*). WS methods join forces to detect words in documents without knowledge of the context. Handwritten documents constitute a separate research area with many challenges due to differences between the quality of documents (KUMAR; GOVINDARAJU, 2016). An example of a handwritten document is shown in Figure 2.2. For this proposal, WS approaches describe each word as a sequence of features with varying sizes and they are successful in a limited domain, in which the documents are of good quality. However, documents can be degraded, by inks, stains, faded or other factors, which can make the recovery process more difficult.


300. Letters, Orders and Instructions. December 1755.

Hogg's Company, if any opportunity offers.
You are to be particularly exact and careful in these payments: seeing that there is no disagreement between the Returns, and your Pay-Rolls; as there will be strict examination into it hereafter. I am &c.

G.W.

Figure 2.2: Part of scanned document of George Washington collection

In documents written by the same person, patterns tend to be similar. However, if a dataset is from different authors, traditional word recovery processes can fail. Rath and Manmatha (2007) show the idea of WS, according to Figure 2.3 using a method of finding the shortest distances between the search word and document images. This method is used to group all word occurrences. The groups that contain terms of interest are selected and labeled. In this way, all instances are indexed to the respective words and documents.

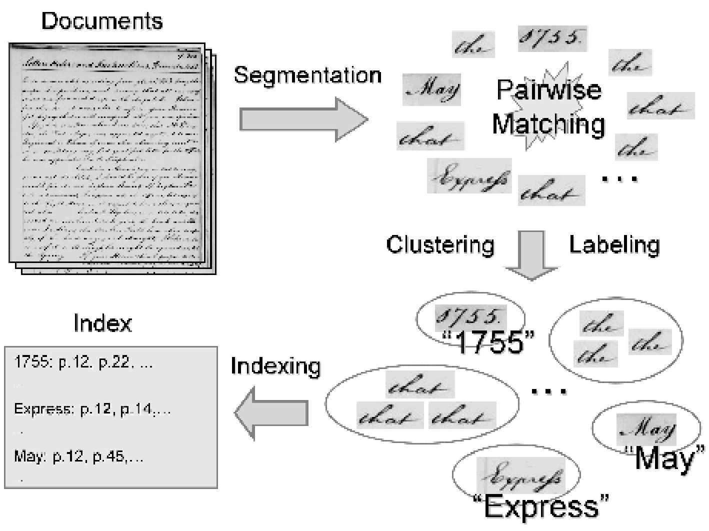


Figure 2.3: WS process example

Source: Rath and Manmatha (2007)

Handwritten historical documents are frequently of low quality and the pre-processing with the segmentation of a page into words is a challenge that is being explored by some authors (TOSELLIA et al., 2016; RUSIÑOL et al., 2015; TABIBIAN; AKBARI; NASERSHARIF, 2016; THOMAS et al., 2015; KANADJE et al., 2016), including documents with low printing quality (KUO; AGAZZI, 1994). The initial stage of creating a spotting method is related to pre-processing, where the WS systems can be based on *query-by-example* (DOVGALECS et al., 2013; RUSIÑOL et al., 2015), in which a query can be an image to be searched for or *query-by-string* (ALMAZÁN et al., 2014), in which a string contains a word to be searched for. In this stage, the system must define what to index from each image in the collection. The study presented by Dovgalecs et al. (2013) used Sliding Windows (SW) to find candidates in a collection of handwritten documents. The same idea was also applied by En et al. (2016c), creating sub-windows of at least 20x20 pixels. However, SW should be used judiciously, due to a large number of sub-windows that can be generated in this process, leading to exhaustive computational processes for feature extraction and matching.

A possibility to reduce the number of sub-windows to be indexed in this process would be to use a method of finding zones of interest in the document. In this way, the process seeks to separate graphic text from the rest of the document, as (ALMAZÁN et al., 2014) that defined a fixed hierarchy of blocks, and used histograms to combine the areas of interest, while (ROY; RAYAR; RAMEL, 2015) used a tool to define the areas of each image in a handwritten book.

Besides, WS systems can be categorized into two approaches by the segmentation procedures utilized: segmentation-based and segmentation-free. In the segmentation-free approach, the search can be done without a segmentation process or with only line-segmentation, as used by Roy, Rayar and Ramel (2015). However, in the segmentation-based approach, each word must be segmented before carrying out the matching. In other words, each document must be divided into small parts, which will be represented by feature sequences (MONDAL et al., 2016). However, if the document is badly degraded, it may not be possible to obtain good segmentation.

The second stage of development of a spotting system is the representation of the images, which can be done with the whole image or the sub-windows. Approaches based on Scale-Invariant Feature Transform (*SIFT*) can be found (ALMAZÁN et al., 2014; RUSIÑOL et al., 2015), but many authors based their work on the Bag of Visual Words (*BoVW*) (DOVGALECS et al., 2013; RUSIÑOL et al., 2015; EN et al., 2016c), in which a document is represented by word occurrences, independent of position in the document. Recent proposals for the use of Vector of Locally Aggregated Descriptor (*VLAD*) and Fisher Vector (*FV*) have obtained better results than the BOVW approach (EN et al., 2016c; ALMAZÁN et al., 2014). Other feature extraction methods can be used in spotting systems: Connected component and glyph codebook (ROY; RAYAR; RAMEL, 2015), Grayscale and geometric vectors (TOSELLIA et al., 2016), Binary vectors (RIBA et al., 2015; THOMAS et al., 2015), color histogram (ZHU; KEOGH, 2010).

Systems based on deep learning also lead to an interesting direction for spotting systems, especially if a Convolutional Neural Network (CNN) is used for feature extraction (JADERBERG; VEDALDI; ZISSERMAN, 2014). Thomas et al. (2015) observed that the models based on deep architectures have a high level of data representation and had the best performance in their experiments.

Finally, to carry out the matching between objects in the third step of a spotting system, a similarity measure must be defined between the query and all the indexed regions. This measure can be based on the approach of the nearest neighbor, for example, Euclidean distance and cosine (EN et al., 2016c; RIBA et al., 2015) or search pixel by pixel as used by Delalandre, Ogier and Lladós (2008). Some other approaches can be used to obtain better performance, such as Dynamic Time Warping (*DTW*). One example is the study applied to WS by Kanadje et al. (2016) who presented a new procedure using *Mel Frequency Cepstral Coefficients* for feature representation and relevant words identified considering DTW. Thomas et al. (2015) presented a system of recuperation using Hidden Markov Modeling (*HMM*). One disadvantage of models based on DTW is that they have slow execution time. On the other hand, the algorithms based on nearest neighbor approaches are compatible with the sliding windows strategy.

Based on WS systems, the procedures can be improved to retrieve not only words or lines of text but also other graphic objects present in image collections. In this way, Dovgalecs et al. (2013) and Hu et al. (2012) introduced the concept of Pattern Spotting (*PS*) to find tools with the objective of mining handwritten documents, or recover medieval capital letters, decorative objects, coats of arms, and other objects from images (TRANOUEZ et al., 2012).

PS consists of finding occurrences of a graphic object given an image collection, considering its location. This object can have different features from others. In this way, during the process, an image that has variability in its color and texture is compared with the written word or symbol. However, some traditional methods used in WS cannot be applied in the context of PS (EN et al., 2016c), for example, *query by string*, since the strings cannot express patterns, this would limit the systems capacity to detect patterns.

The first stage of a PS system is also the pre-processing. SW is also widely applied in PS, as presented by Rakthanmanon, Zhu and Keogh (2011), Hu et al. (2012) and En et al. (2016c). Zhu and Keogh (2010) indexed these images using patches, with each patch being a rectangular window on the pages. The graphic objects recuperated in PS systems can bring greater complexity due to the variations found in the queries. In this way, the set of features must also be robust, to form the best representation for the images. Many authors use different color spaces to limit objects, such as HSV (YARLAGADDA et al., 2011) and RGB (ZHU; KEOGH, 2010). Afterward, the objects can be represented by Histograms of Oriented Gradients (*HoG*). En et al. (2016c) compared three representation models, BOVW, VLAD and FV, in which the best results were obtained for the representation based on VLAD and FV. In this study, the authors recovered the query image and its respective location in the corresponding image. Dovgalecs et al. (2013) chose to use SIFT descriptors and represent them by BOVW and the pattern searched was considered correct only with overlap greater than 50%. SIFT descriptors were also used by Le et al. (2013) and Le et al. (2014).

As similarity measures, in PS the methods based on nearest neighbor are also chosen for the system, such as k-NN (ZHU; KEOGH, 2010), GHT distance (RAKTHANMANON; ZHU; KEOGH, 2011), dot product (EN et al., 2016c). However, expansion can be observed for other measures like Campana- Keogh-1 (*CK1*) used by Hu et al. (2012), Longest Weighted Profile (*LWP*) used by Dovgalecs et al. (2013) or SVM-base classifier, proposed by Yarlagadda et al. (2011).

2.1.1 Related Works

Recently, many articles have been dedicated to IR and PS problems, such as Chen et al. (2016), Thuy et al. (2017), Hangarge et al. (2016), He et al. (2016). Torres and Falcão (2006) presented a detailed survey with the use of the CBIR approach in several applications for medical systems and digital libraries.

In Shirdhonkar and Kokare (2011) the authors performed handwritten document image retrieval and used Contourlet Transform descriptors, followed by Euclidean and Camberra measures for similarity analysis, while (LIN et al., 2015a) explored deep learning to retrieve manuscripts and shop images, applying deep binary codes and an exhaustive search.

Logo retrieval was recently explored by Jain and Doermann (2012) that used SURF descriptors, proposing a new indexing model and candidate filter. They observed that the recognition of text along with logos could produce better results. However, this strategy applied only with logo images does not produce such satisfactory results.

Alaei, Delalandre and Girard (2013) presented an interesting system for logo retrieval based on administrative documents, using a decision tree and a small number of features for patches. They evaluated the proposed approach using public and two large industrial datasets. Based on the experiments, most of the errors occurred because some documents were very poor quality, noisy and skewed.

Table 2.1 shows a summary of the IR methods explained previously. The type of data and the method used to represent images and generate candidates are considered. Relevant papers were selected from research done in the last years.

Table 2.1: Summary of the recent works for Image Retrieval task

Author	**Dataset**	**Representation**	**Image segmentation**
(CHEN et al., 2016)	MPEG-7, (N-S) and AT&T	Shape context	Whole image
(THUY et al., 2017)	Corel dataset	Color and texture	Whole image
(HANGARGE et al., 2016)	Document images	Gabor wavelets	Word segmentation and bounding boxes are fixed
(HE et al., 2016)	Historical document	Singular structural feature	Patches with a fixed image size
(SHIRDHONKAR; KOKARE, 2011)	Historical handwritten	Contourlet Transform	Entire textual handwriting
(LIN et al., 2015a)	Cifar and Mnist	Deep binary hash code	Whole image
(JAIN; DOERMANN, 2012)	Administrative documents	SURF features	Interest point detector
(ALAEI; DELALANDRE; GIRARD, 2013)	Industrial datasets	Frequency and Gaussian probability, height, width, average density	Piece-wise painting algorithm

Table 2.2 shows a summary of WS and PS methods found in the literature. The type of data and the method used to represent images and generate candidates are considered. For both WS and PS, the candidate generation used by the authors was studied because it is important to evaluate similar approaches or alternative algorithms. Relevant papers were selected from research done in the last years.

Zhu and Keogh (2010) performed objection detection in books, and the RGB color space was used to represent units of low level to retrieval objects in books, while Hu et al. (2012) presented a search by occurrences of small regions of decorative letters of books. Yarlagadda et al. (2011) studied the object detection problem in 27 medieval manuscripts from Heidelberg University Library, presenting suitable results for multiple scales and orientations. However, there are many processes to produce a robust representation, which can be avoided by deep learning approaches.

Rakthanmanon, Zhu and Keogh (2011) introduced a system to detect occurrences of the patterns in historical texts, however, the authors did not use the exact location of symbols. Dovgalecs et al. (2013) proposed a method for the retrieval process using their dataset of historical documents, presenting some qualitative results about precise location. In our experiments we want to find the best object and its respective location, returning a comparison to the state-of-the-art, which in some cases was not described by the authors.

Riba et al. (2015) coded the images using graphs and binary vectors to detect plant pattern. In the experiments, they had influence of symmetric symbols that modify the entire topology of the plant. The use of binary vector can result in poor representation. However, En et al. (2016c) proposed information retrieval in historical documents, with a focus on graphical patterns. The methods were evaluated in DocExplore database, created by the authors in their research project. The images were represented in gray-scale and, in some cases, there may be confusion in some patterns.

Often we observe that the authors used shape features or binary representation, which is difficult to represent the different levels of abstractions of the

Table 2.2: Summary of the recent works for Word Spotting and Pattern Spotting

Author	**Dataset**	**Representation**	**Image segmentation**
		Word Spotting	
(DOVGALECS et al., 2013)	Historical handwritten	BOVW	Sliding Windows
(ALMAZÁN et al., 2014)	Historical handwritten	Fisher Vectors and CSR	Spatial Pyramids
(ROY; RAYAR; RAMEL, 2015)	Historical book	Connected component and glyph codebook	AGORA framework
(RUSIÑOL et al., 2015)	Historical handwritten	BOVW	Local patches
(THOMAS et al., 2015)	Handwritten letters	Binary pixels	Frames
(RIBA et al., 2015)	Historical handwritten	Binary vectors	Graphs
(TOSELLIA et al., 2016)	Handwritten document	Grayscale and geometric vectors	Text line
(EN et al., 2016c)	Manuscripts	BOVW, VLAD and Fisher Vectors	Sliding Windows
(KANADJE et al., 2016)	Lecture videos	MFCC	Frames
(MONDAL et al., 2016)	Historical handwritten	RLSA	Text line
(TABIBIAN; AKBARI; NASERSHARIF, 2016)	Lecture videos	Presence and Duration Confidence Function	Frames
		Pattern Spotting	
Zhu and Keogh (2010)	Historical Manuscripts	Color Histogram	Patches
Yarlagadda et al. (2011)	Medieval Images	HoG	Shape based HSV representation
Rakthanmanon, Zhu and Keogh (2011)	Historical Document	Pixels	Sliding Windows
Hu et al. (2012)	Books	Binary Vectors	Sliding Windows
Dovgalecs et al. (2013)	Author book	BOVW	Sliding Windows
Riba et al. (2015)	Document images	Binary Vectors	Graphs
En et al. (2016c)	Historical document	BOVW, VLAD and FV	Sliding Windows
Le et al. (2013)	Administrative document	SIFT feature space	Keypoints
Le et al. (2014)	Administrative document	SIFT and BRIEF	Keypoints

images. Thus, the use of deep learning in IR and PS approaches can allow a robust representation of images, however, it is important to avoid training steps because of the lack of information, that is, to perform IR and PS without any previous knowledge about the patterns. Therefore, the next sections present the principal considerations of the methods that can be used in this study, based on the steps of CBIR. First, pre-processing is a crucial task to find the candidates present in each document. Then definitions of feature extraction will be present, ending the chapter with the main similarity measures for IR.

2.2 Pre-processing: search for candidate objects

Pre-processing is a crucial task in IR and PS systems since it aims to search for candidate objects in each document of the data set. The main objective is to create a set of regions defined by bounding boxes corresponding to the objects that can be present given an image (CHENG et al., 2014). In the past few years, this process has become an important area of image processing. In CBIR solutions, we need to find all the possible object proposals (candidates) present in each document, but avoiding previous information about the type of object to be searched for. As observed in the literature, SW is a common process for generating candidates with fixed sizes, however, several approaches have been proposed to try to balance computational cost and high quality of the objects (ARBELÁEZ et al., 2014; GIRSHICK et al., 2014). So, this section presents some pre-processing methods for candidate generation given an image, used by authors in recent publications.

The most classic approach (SW) segments the image in homogenous regions, however, recent methods show a more advanced concept, called object proposals. Though a variety of approaches exist, such as Arbeláez et al. (2014), Cheng et al. (2014), Zitnick and Dollár (2014), Uijlings et al. (2013), the SW approach is still used in some studies, as mentioned by Hu et al. (2012), En et al. (2016c), Dovgalecs et al. (2013).

Forsyth and Ponce (2002) explain that the SW approach shows higher feasibility when working with high-quality images without deformation. The objects can be generated easily, using a set of fixed-size windows (m x n). However, care is necessary for this process, since objects can vary in size and thus an adequate scale must be defined for the images. Another important topic to keep in mind is the overlap of the windows, that is, considering a limit of the overlap between the windows generated in the image. So, it would interesting to find alternative methods that search for candidate regions to be used in the image recovery task, that could divide the image into small areas with various search parameters, not just fixed-size windows (ARBELÁEZ et al., 2014).

Accordingly, He and Lau (2015) presented a new approach to generate oriented object proposals based on different positions, scales and proportions. They formulated a probabilistic model, in which objects are searched for and located with the method of maximum likelihood. Then, the shape of the object is described by a covariance matrix. They used the Freestyle Motocross data set containing positive and negative samples. The results surpassed those found in the literature, however, the data set is composed of only one type of object to be searched for.

Faktor and Irani (2013) presented a method of co-segmenting common objects by composition. They suggested an approach for inducing affinities between parts of the image. In this way, the results were evaluated in three different data sets. The results surpassed the state of the art, but in some cases, the method failed when the background was complex (containing multiple images).

As a different alternative, Endres and Hoiem (2010) proposed an approach for automatically generating a small number of regions in an input image. They used binary segmentation (CFR Segmentation) based on a few regions as a seed and computed the probability of other regions belonging to the same object. However, they encountered difficulty finding regions in small objects.

Bhattacharjee et al. (2016) proposed a method based on the invariance of occlusion, geometric shape, and deformations to find objects in a large data set. The objective was to identify a set of regions of interest in which there is a high probability of the presence of any object. They developed their work based on the implementation of the Edge Boxes algorithm proposed by Zitnick and Dollár (2014) integrated with BoVW to represent the images. Although the authors developed an exhaustive search, the results were discriminative for outliers and it is prone to disorder, deformation, and variations in the conditions of the image.

A new pre-processing scheme was proposed by Liu, Lu and Jia (2015). First, they used R-CNN to generate the box proposals and then statistical information was codified to aggregate the boxes in a prediction of the final object. The data sets used were from the Pascal challenge and the experiments showed an improvement of 3.7% of mAP compared with the literature. However, the search process needs a seed defined based on the ground truth.

Zhang et al. (2016) presented an approach to building the top of a hierarchical segmentation. Candidate objects of multiples scales were selected with this approach, being obtained by the maximization of an objective submodular function. The results obtained in two data sets surpassed other methods, but some classes were not recognized.

Thus, many popular methods in the literature are based on object segmentation, as follows:

- Objectness (ALEXE; DESELAERS; FERRARI, 2010): it is considered one of the earliest methods of segmentation proposals. It uses a probability function based on spatial location, colors, and edges to define if the region of interest is an object, and then the proposals are generated.

- CPMC (CARREIRA; SMINCHISESCU, 2012): it performs automatic segmentation to generate and classify object hypotheses in images, without prior knowledge of the individual properties of each class.

- Binarized Normed Gradients (*BING*) (CHENG et al., 2014): it presents a thresholding version of standard gradient resource to estimate objects in an image.

- Edges Boxes (*EB*) (ZITNICK; DOLLÁR, 2014): each object is found by means of its edges. The closest contour number by a bounding box indicates the probability of containing an object.

- RPrim (MANEN; GUILLAUMIN; GOOL, 2013): it uses a merge function to calculate weights between low-level pixels to generate the proposals.

- DeepBox (*DB*) (KUO; HARIHARAN; MALIK, 2015): the proposals are generated with *bottom-up* method, and ranked by a deep learning algorithm.

- Multiscale Combinatorial Grouping (*MCG*) (ARBELÁEZ et al., 2014): it is an approach to image segmentation and object generation with efficient use of multiscale information. This information is then grouped to generate the objects.

- Selective Search (*SS*) (UIJLINGS et al., 2013): it is a combination between exhaustive search (capture locals of objects) and segmentation to generate objects. This algorithm considers that the objects can occur at different scales.

Zitnick and Dollár (2014) made comparisons among several algorithms available in the literature to generate objects from the images. They presented EB and SS as promising algorithms for this purpose since they have a high number of object proposals and recall. Bing is an algorithm that can be faster than EB and SS, but it has low precision. Table 2.3 shows a brief review of the methods studied for object proposals. Some studies did not use a specific algorithm to generate the objects.

Therefore, the result of this step is a set of candidates indexed from each image of the collection. In the next step, a robust feature extractor must be defined to represent these images. The definitions and approaches to implement a feature extractor method for spotting systems are presented in the next section.

Table 2.3: Literature review for candidate generation methods

Author	Algorithm	Dataset	Methods	Advantages	Disadvantages
He and Lau (2015)	-	Freestyle Motocross	Probabilistic model	Different positions and scales	Depends training samples
Faktor and Irani (2013)	-	MSRC iCoseg Pascal	Co-segmentation	Search objects in common	Confusion with background in some cases
Endres and Hoiem (2010)	CRF	BSDS VOC2008	Binary segmentation	Find probability to each region	Difficult in small objects
Bhattacharjee et al. (2016)	EB	Belgalogo FlickerLogos-27 Tobacco-800	EdgeBox BoVW	Discriminative to outliers	Exhaustive search
Liu, Lu and Jia (2015)	R-CNN	VOC2007 2010	Statistical Information	Agregation of box proposals	Depends of labels
Zhang et al. (2016)	-	BSDS 300 VOC2012	Hierarquical segmentation	Select object from multiples scales	Slower than other methods
Carreira and Sminchisescu (2012)	CPMC	MSRC VOC2009	Pool of segments	Multiple spatial scales	Problems with occlusion
Arbeláez et al. (2014)	MCG	BSDS500 VOC2012	Hierarquical segmentation	Multiscale regions	Slower than other methods
Cheng et al. (2014)	BING	VOC2007	Norm of the gradients	Fast and high quality objects windows	Limited number of regions proposals
Zitnick and Dollár (2014)	EB	VOC2010	Measure the number of edges that exist in the box minus	Fast and larger number of proposals	Many arests are removed, losting useful information
Uijlings et al. (2013)	SS	VOC2007 VOC2010	Hierarquical grouping	Selective search and larger number of proposals	Slower than other methods
Kuo, Hariharan and Malik (2015)	DB	VOC2007 COCO	Semantic approach	Faster and high quality objects windows	Object trained to be generated

2.3 Image representation: Feature Extraction based on Learning

The feature extraction process plays an important role in traditional methods of machine learning. An adequate method for extracting features from images is essential for representing the queries and candidate images generated in the pre-processing stage. The feature extraction process aims to describe the object based on the chosen representation, and it begins with the result of a segmentation stage to provide the representation of each object in the data set. This representation results in quantitative information that is used to differentiate one object class from another (GONZALEZ; WOODS, 2006). Therefore, feature extraction methods determine an appropriate subspace of dimensionality m (set of m features) from a space of dimensionality d (the image), considering $m \leq d$ (JAIN; DUIN; MAO, 2000). This will result in a feature vector for each instance of the data set.

The feature vectors extracted are introduced to an inductor algorithm, such as SVM, decision tree, Bayesian networks, etc., that can carry out the learning task. Typically, the representation task is costly and challenging, because it is necessary to find a set of features that adequately represent all the data. Several researcher use a specialized feature extractor, such as SIFT descriptors (LOWE, 2004), SURF (BAY et al., 2008) or HOG (DALAL; TRIGGS, 2005).

Studies of image representation have received special attention from the computational vision community for many years. When the main focus is on shape features, an external representation is chosen. However, when the focus is related to local properties of the image, then an internal representation is chosen, such as color and texture (GONZALEZ; WOODS, 2006). In document images, the features can be complex, with variability in shape, color, and texture. During most of the last decade, the use of BoVW was considered a reference in the state of the art, especially when used with SIFT descriptors. However, this representation works with only local descriptors, motivating researchers to also use alternative methods.

Recent studies show that Deep Learning (*DL*) has become a new alternative for retrieval systems, The use of the learning process in feature extraction when possible is an attractive approach with promising results. This approach is called Feature Learning or Representation Learning (LECUN et al., 1998). In this way, feature learning can be accomplished through DL techniques, which represent a new area of machine learning. These techniques have increased the efficacy of different solution in pattern recognition and one objective is to improve the representation in multiple levels of abstraction, such as images, sounds, and texts.

The generation of a feature map from a deep learning architecture is considered one of the best solutions for many computational vision problems, including searching for images in repositories. It is common to acquire an image of an object, place or person using a camera and then create a query image for the recovery system (BOSCH, 2015). In the last few years, deep architectures have been used to recover images using feature learning methods (KRIZHEVSKY; HINTON, 2011) (XIA et al., 2014). Recently, Gordo et al. (2016) evaluated approaches using public data sets, like ImageNet, Cifar-10 e Mnist. The principal focus is to learn high-level features by the composition of low-level features (MITCHELL, 1997)

Another current focus of the DL models is the ability to work with data in large scale, due to the high number of parameters and the type of deep architecture. The performance of traditional machine learning models (for example SVM) has caused problems of *underfitting*, which present large errors during the training stage when the training becomes too large. The solution to this problem is to use DL, as can be observed in Figure 2.4 (WANG, 2016).

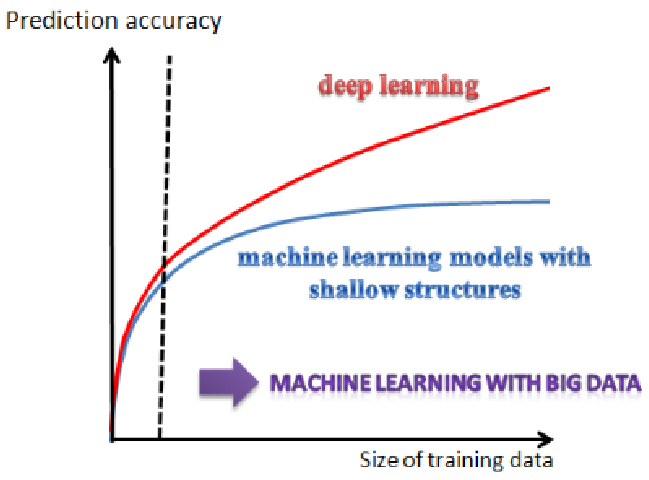


Figure 2.4: Comparison of machine learning models by training data size

Source: Wang (2016)

However, DL architectures are not more advantageous than other learning methods and can even result in worse results when the amount of training is small. The solution to this problem is the use of transfer learning. This method will be explained in Section 2.3.2. However, DL makes a big difference in large scale data, since in experiments, it has been observed that the performance was improved when the dimensionality was increased. This is known as "dimensionality blessing" because a large training capacity is necessary. Bengio and LeCun (2007) defined DL architectures as the composition of multiple layers with adaptive non-linear components. In this sense, the architectures can:

- learn the previous knowledge with input data;
- have multiple layers where two adjacent layers may connect themselves with non-linear components;
- have several parameters of training;
- be well dimensioned;
- support different paradigm of learning, such as multi-task learning or semi-supervised.

In the last few years, several researchers have worked with DL. Babenko and Lempitsky (2015), Kalantidis, Mellina and Osindero (2016), Razavian et al. (2014b) and Gong et al. (2014) recently worked with deep features applied to image representation. Lin et al. (2015a) and Wu et al. (2017b) used deep binary representation for fast IR, due to the large data set used in their experiments.

Many architectures have been proposed as presented by (BENGIO, 2009), (VINCENT et al., 2008), (HINTON; OSINDERO; TEH, 2006) and (LECUN et al., 1989), especially those that use Convolutional Neural Network (*CNN*) as a feature extractor (LIN et al., 2015a) (GORDO et al., 2016) (RADENOVIC; TOLIAS; CHUM, 2016). In the following sections, attention will be given to CNN architecture, since it was used in the present study as a feature extraction method thanks to its high performance as reported in recent studies.

2.3.1 Convolutional Neural Networks

The study of CNNs began with neurobiological studies and with an original idea of HUBEL and WIESEL (1962) related to local sensitivity and selective neurons of a cat's visual cortex. The hypothesis of a good internal representation must be hierarchical, in which pixels generate edges, edges become patterns of shape, and the combined standards will form objects, which consequently form scenes (LECUN; KAVAKCUOGLU; FARABET, 2010). Accordingly, CNN is an architecture that is widely used in DL, making it a good candidate algorithm for resolving problems involving pattern recognition. This section presents the CNN architecture and the operation of this network.

The concept of CNN was proposed in 1989 by Lecun et al. (1989), as a multiple-layer perceptron. This network was designed precisely to recognize two-dimensional shapes with a high level of invariance for translation, scale and other forms of distortion.

This network combines three architectural ideas: the local receptive fields share weights and spatial or temporal subsampling. This permits the use of a filter on the visual data, maintaining the neighbor relationship between the pixels of the image during the network processing.

Two decades after this concept of architecture was introduced, Krizhevsky, Sutskever and Hinton (2012) presented this architecture model for the recognition process applied to a data set from Imagenet Large Scale Visual Recognition Challenge (ILSVRC) in 2012 that is a dataset with millions of labeled examples. The algorithm was able to learn features of different natures, that is, it could work with spatial information, scales, and colors. Since then, the interest in this architecture model has grown, as presented by (SERMANET et al., 2014), (RAZAVIAN et al., 2014a), (BOSCH, 2015), (TOLIAS; SICRE; JÉGOU, 2015), (GORDO et al., 2016), (RADENOVIC; TOLIAS; CHUM, 2016) (KALANTIDIS; MELLINA; OSINDERO, 2016).

Zeiler and Fergus (2014) defined CNN as a feature hierarchy producer, and the learning is evaluated through the analysis of the filters obtained at the end of the training stage. A feature map is generated in each layer of the CNN model (LECUN; KAVAKCUOGLU; FARABET, 2010) and the input represents a matrix of pixel intensity. The feature map of any internal layer is a multichannel induced image, in which each pixel corresponds to a specific feature. All the neurons are connected to a small number of neurons adjacent to the last layer (DRUZHKOV; KUSTIKOVA, 2016).

In this architecture, the first layers can learn filters for edges, borders, and colors. At deeper levels, greater detail and complex features of the images are detected. In the last layer, a final feature map is generated (LECUN et al., 1998). In this way, feature extraction occurs when the learning of the layer is finished, generating a feature map. Each feature map will serve as standard input for the next layer. In IR or PS, it is necessary to define which layers will be used to provide the final feature map, as done by Yue-Hei, Yang and Davis (2015) that investigated the effect of different layers and the use of dimensioned input images, evaluating the performance of convoluted features in the IR task.

Besides, CNN architectures can create filters with different dimension. The filters are defined during the training process of the network, and they can have different lengths, widths, and depths, crossing multiple channels. The classic filters in image processing rarely have these features, since most are 2D. The possibility of this network generating feature maps is a great benefit of this model, and at the same time, it follows a complex process of extraction. In this way, the training data information is maximized. So, CNN has some important characteristics (LECUN et al., 1998)(ZEILER; FERGUS, 2014)(LECUN; KAVAKCUOGLU; FARABET, 2010):

- the images are often large and have enough variability in the information of the pixels;
- the input data goes through convolution layers;
- a convolution layer is composed of several neurons, each one responsible by filter application in some specific piece of the image;
- the filter size defines the neighbor rate that each neuron of the layer will process;
- an activation function is commonly used after the convolution layer and it is responsible for applying a transformation in the data received;
- a reduction layer is applied after convolution and activation layers to reduce the dimensionality of data;
- the parameter stride also must be defined, representing how the pixels will be shifted between each window of the filter;
- a neuron is connected to the pixels of the superior layer, and each connection receives a weight. The weights are shared between layers;
- the weights are assigned to the connections in each neuron and they are interpreted as a matrix representing the convolution filter;
- when a classification is performed at least one fully connected layer is increased after convolution layers and grouping. This layer will draw a decision path from the filter responses;

- a classification function is defined after fully connected layer. This layer influences the filter learning and final results;

Unlike the formulation of the traditional perceptron, in a CNN only the input set is connected to each neuron analyzing the local field, and the neurons of the same layer are grouped into maps. In a CNN it is not necessary to define an initial feature model, but only the architecture of the filters: quantity, size and stride (ZEILER; FERGUS, 2014).

Once the architecture is defined, a feature map is obtained with the convolution in an input image by a linear filter, a bias, and an activation function. That is, with the k layer, the filters are determined for a set of weights W^k, input feature map x, a bias term b_k and a convolution operator $*$. Equation 2.1 defined by Lecun, Kavakcuoglu and Farabet (2010) shows the output of the feature map h^k obtained by an activation function f.

$$h_{ij}^k = f((W^k * x)_{ij} + b_k) \tag{2.1}$$

The convolution process is made up of three phases: filter definition, activation function, and reduction. A typical CNN structure is composed of one or more stages with these three phases. Figure 2.5 shows the basic CNN structure composed of two convolutional stages. This type of network is widely used in problems involving classification, detection and image recognition.

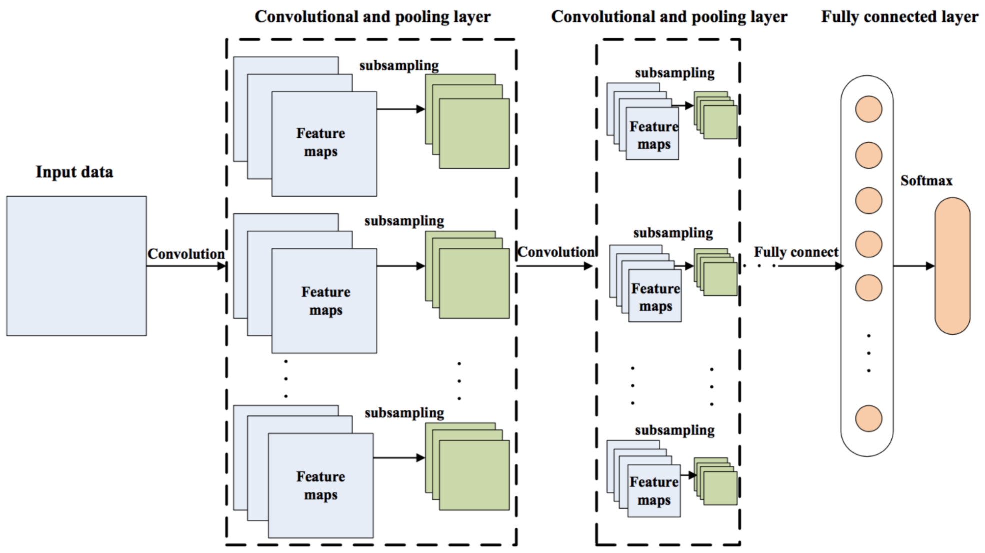


Figure 2.5: Typical structure of CNN

Source: Adapted from Jing et al. (2017)

Thus, at each step of the filter definition, the convolution of the filters W_i^k corresponding to the $i-th$ filter of the layer k of size l_1 and l_2. The filters are responsible for detecting a particular feature in all locations of the input image. Figure 2.6 shows an example of this convolution process with an image $4x3$ and convolution filter $3x3$.

$$\begin{array}{ccccc} \text{Image} & & \text{Convolution filter} & & \text{Result} \\ \begin{array}{|c|c|c|c|}\hline 1 & 0 & 1 & 0 \\ \hline 0 & 1 & 0 & 1 \\ \hline 0 & 0 & 1 & 1 \\ \hline \end{array} & * & \begin{array}{|c|c|c|}\hline 1 & 0 & 1 \\ \hline 0 & 1 & 0 \\ \hline 1 & 0 & 1 \\ \hline \end{array} & = & \begin{array}{|c|c|}\hline 4 & 1 \\ \hline \end{array} \end{array}$$

$$\text{c1} = 1*1+0*0+1*1+0*0+1*1+0*0+1*0+0*0+1*1 = 4$$

$$\text{c2} = 0*1+1*0+0*1+1*0+0*1+1*0+0*1+1*0+1*1 = 1$$

Figure 2.6: Example of convolution and filter

Source: The author

The next step defines the use of an activation function (frequently non-linearity is applied). This step is responsible for applying a function to each element of the feature map. Equations 2.3 to 2.5 correspond to the principal functions used and their respective graphs, which depend on z obtained through Equation 2.2.

$$z = (W^T * x) + b \tag{2.2}$$

$$sigm(z) = \frac{1}{1 + e^{-z}} \tag{2.3}$$

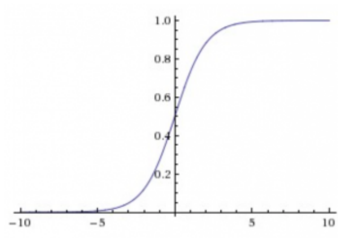


Figure 2.7: Sigmoid function

$$tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}} \tag{2.4}$$

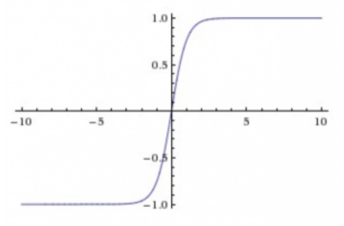


Figure 2.8: Hyperbolic tangent function

$$relu(z) = max(0, z) \tag{2.5}$$

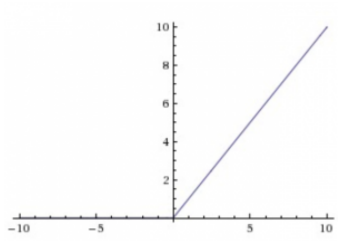


Figure 2.9: ReLu Function

The non-linear (sigmoid) function (Eq. 2.3) has values between [0,1] and Menon et al. (1996) carried out a detailed study of this function. The hyperbolic tangent function (Eq. 2.4) has interval between [-1,1]. Comparing these two functions, the tangent has shown better performance with deep architectures. However, both face problems when units of the uppermost level become saturated, and leakage gradients near 0 are propagated to lower levels. This can decelerate the convergence of the training or generate a poor local minimum. On the other hand, the ReLU function (Eq. 2.5) proposed by Glorot, Bordes and Bengio (2011) offers an alternative for non-linear functions, solving this problem, since there is no escape from gradients along with the actively hidden units. In this way, when a unit is activated above 0, it has a partial derivative of 1 (MAAS; HANNUN; NG, 2013)(HAYKIN, 2001). Thus, the ReLU activation function has been more widely used in CNN models.

The next step for defining a CNN model presents the reduction function called subsampling. This function calculates the mean or maximum value of a predetermined neighborhood in each feature map. Then, another map is generated, with a lower resolution than the original image. These main functions are defined as:

- Average (AVE): it uses the arithmetic average between the elements a_i of each region R_j (ZEILER; FERGUS, 2013), and it is defined by Equation 2.6

$$s_j = \frac{1}{|R_j|} \sum_{i \in R_j} a_i \tag{2.6}$$

- Maximun (MAX): it selects a neighborhood 2 x 2 without overlap (ZEILER; FERGUS, 2014) according to Equation 2.7:

$$s_j = max_{i \in R_j} a_i \tag{2.7}$$

- Stochastic (STO): it firstly computes the probability p for each region R_j, normalizing the activations inside of each region. The calculation is defined by Equation 2.8:

$$p_j = \frac{a_i}{\sum_{k \in R_j} a_i} \tag{2.8}$$

  A local $L = \{l_1, l_2, ..., l_n\}$ based on p that is selected inside the region. The activation is grouped in L resulting in $s_j = a_l$ where $l \sim P(p_1, p_2, ..., p_{|R_j|})$ (ZEILER; FERGUS, 2013).

Figure 2.10 shows an example of the reduction process in a feature map using the MAX function. This function is often used and decreases training time without loss of quality (ZEILER; FERGUS, 2014).

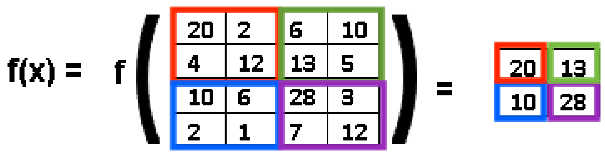


Figure 2.10: Example of reduction function

Source: The author

Space arrangement. CNN requires some hyperparameters in addition to the previously defined functions. These are:

- *Stride*: in convolution layers and reduction is necessary to define the value of filter shifting which determines where must be the next subsample region. If the stride is 1 the filter is shifted 1 pixel by time. If the stride is 2 then it will be shifted 2 pixels by time. Stride equal or greater than 3 is not used.

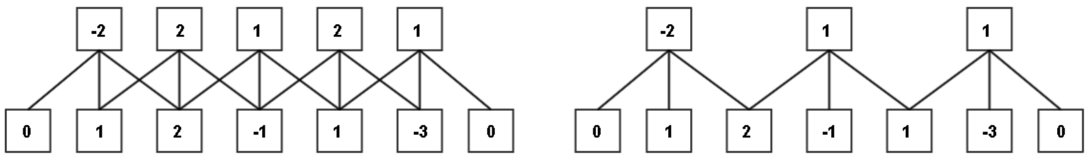


Figure 2.11: Example of stride applied in a region (stride 1 and 2)

Source: The author

- *Padding*: sometimes, it can be convenient increase the input image with pixels around it to consider the borders of the image. The most common approach is *zero-padding*, that adds zeros around of image. Figure 2.12 shows an example of *zero-padding*, while Figure 2.11 shows an example of *stride* 1 and 2, respectively.

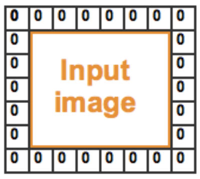


Figure 2.12: Example of padding applied in an image

Source: The author

- *Weight initialization:* Gaussian distribution can be used, in which case the weights are initialized according to the mean and standard deviation of the input. Alternatively, the Xavier algorithm can be used, and this depends on the number of the connection on the neuron and its output.

- *Optimization function:* is used to minimize the error in a network with w weights in k samples. A sample z_k is randomly chosen, and an optimization function is applied. Some optimization function have already been used in CNN models, like Adadelta (ZEILER, 2012) and RMSPROP (TIELEMAN; HINTON, 2012). Although, Gradient Descent is one of the most popular algorithms for carrying out optimization. The gradients are used to update parameters once the analytic gradient is computed with backpropagation. The

Stochastic Gradient Descent (*SGD*) function can often be applied (RECHT et al., 2011). This function is defined by Equation 2.9.

$$w(t+1) = w(t) - \lambda \bigtriangledown Q(z_t, w_t) \tag{2.9}$$

in which $w(t+1)$ is a new weight, λ is an adequately chosen learning radius, and Q is the empirical risk with a sample z_t. The risk is expected to decrease significantly. Some hyperparameters can be used to accelerate the SGD function in a relevant direction, improving the performance as follows:

- The parameter *momentum* γ does an addition of a fraction γ of the update vector of the past time step to the current update vector. The value $\gamma = 0.9$ or similar is often used.
- The *dropout* avoids the sensibility in the parameters turning off a percent of connections of neurons randomly and uniformly.

According to the example in Figure 2.11, the spatial size of the output of the convolution and reduction layer is calculated defining the volume W, the respective filter F, the stride S, and the padding value P. They are:

- Convolution Layer: the output size will be $(W-F+2P)/S+1$. For example, in an input 7×7, a filter $F = 3 \times 3$ results in a volume $W = 5$. Considering a stride 1 and zero-padding $P = 1$, the size of map is 5×5 because $(5 - 3 + 2(1))/1 + 1 = 5$. So, the volume may be calculate by $W_2 \times H_2 \times D_2$ (length, width, depth), being $W_2 = (W_1 - F)/S + 1$, $H_2 = (H_1 - F)/S + 1$ and $D_2 = K$. The parameter K corresponds to the number of filters and (W_1, H_1) the length and width respectively.

- Reduction Layer: The output volume will be $W_2 \times H_2 \times D_2$, being height, width, and depth respectively. These values are obtained with $W_2 = (W_1 - F)/S + 1$, $H_2 = (H_1 - F)/S + 1$ e $D_2 = D1$.

Once the CNN layers are implemented, the layer responsible for the output interpretation is defined. This function is called Softmax, in which the sum of the outputs must be 1. This output is interpreted as a distribution of the discrete probability of the input belonging to each of the classes.

The Softmax function has as input a vector $\hat{y}_i$ and it calculate the probability distribution for each class. The similarity x of the input is calculated by each vector, and the vector weights W_i. The activation values are transformed in positive values using an exponential function, and they are normalized (BISHOP, 2006). Then, cross-entropy is used as a measure of error where each error derivative (Err) related to the weights W_i takes on $\frac{\partial Err}{\partial W_i} = (\hat{y}_i - y_i)x$.

Various CNN architectures have been developed based on these specifications, such as LeNet (LECUN et al., 1998), with two convolution layers followed by grouping, finishing with one convolution layer; AlexNet (KRIZHEVSKY; SUTSKEVER; HINTON, 2012), with five convolution layers following by grouping, ending with two fully connected layers; the VGG network (SIMONYAN; ZISSERMAN., 2015) with two convolution layers followed by grouping between them; and GoogLenet (SZEGEDY et al., 2015) with 9 inception models, each with three convolution layers followed by grouping. All these architectures showed promising results when applied to the area of pattern recognition, as well as when they were adapted to be used in retrieval systems.

2.3.2 Transfer Learning

CNN architecture is frequently used as a feature extraction method based on trained models (*Transfer Learning*). The study of Transfer Learning (*TL*) has gained attention since 1995 and is motivated by the fact that people can intelligently apply the knowledge previously learned to solve new problems quickly or with better solutions. Currently, TL methods appear in several areas, most notably in data mining, machine learning, and their applications (PAN; YANG, 2010).

The TL is widely used when the amount of training data is small. It focuses on developing methods of transferring knowledge learned in one or more source tasks and using it to improve the learning of a related target task, as shown in Figure 2.13 (TORREY; SHAVLIK, 2009).

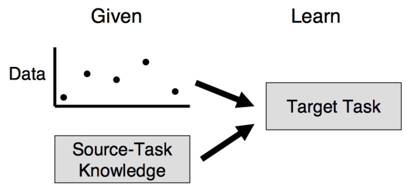


Figure 2.13: Transfer learning procedure. It is a machine learning method with an additional source of information apart from the standard training data: knowledge from one or more related tasks

Source: Torrey and Shavlik (2009)

The definition of TL is: *Given a source domain* D_S *and the learning task* T_S*, a target domain* D_T *and the learning task* T_T*, transfer learning aims to improve the learning of the target predictive function* $f_T(.)$ *in* D_T *using the knowledge in* D_S *and* T_S*, in which* $D_S \neq D_T$ *or* $T_S \neq T_T$ (PAN; YANG, 2010).

Many researchers have attempted to overcome the lack of training samples in some categories using classifiers trained in other data sets. Examples of TL include Zhuang et al. (2015), who used an algorithm based on auto-encoders to find a good representation of instances of different domains. Other methods using different architectures use the same objective, such as Yosinski et al. (2014) and Oquab et al. (2014) that use a CNN as a generic feature extractor that was pre-trained on a data set, the source task with the data set ImageNet, and then reused for another target task.

Akçay et al. (2016) also made use of TL with a CNN that was pre-trained using the ImageNet data set in the context of X-ray baggage security screening. They obtained good results using an AlexNet architecture sharing the weights of different layers and analyzing the results. In studies involving image retrieval, such as Ou et al. (2014), a deep learning algorithm with the TL method can be used to learn a robust image representation. In this way, the focus on TL for the present study is to evaluate the transfer of knowledge that can be used as a feature extractor to find good representation in different domains based on a pre-trained CNN.

Besides, Babenko et al. (2014) suggest one possible interesting direction for investigation, but they did not implement in the paper. The idea is the feature vectors can be obtained directly by training the whole deep architecture using the pairs of matched images, through of Siamese Neural Network (*SNN*), which can be implemented based on CNN architectures. This network will be shown in the following section.

2.3.3 Siamese neural networks

The name "siamese" historically comes from the necessity to collect the state of a single network for two different activation samples during training. It can be seen as using two identical, parallel neural networks sharing the same set of weights (BERLEMONT et al., 2018). The first idea of the siamese network was published by Bromley et al. (1993) for signature verification problem. The Siamese Neural Network (SNN) is usually implemented in face recognition (SCHROFF DMITRY KALENICHENKO, 2015) (WU et al., 2017a), signatures or symbols problems (KOCH; ZEMEL; SALAKHUTDINOV, 2015) to compute similarity between images. Chopra, Hadsell and LeCun (2005) implement a method for training a siamese network based on Contrastive Loss function and the reported results were promising. During the test, the output from one of the subnets of the SNN is a feature vector of the input image of the subject. A model is constructed of each subject by calculating the mean feature vector and the variance-covariance matrix using the feature vectors generated from the first five images of each subject. The likelihood that a test image is genuine is found by evaluating the normal density of the test image on the model of the concerned subject. Figure 2.14 presents the architecture implemented, where X_1 and X_2 are pair of images shown in the network. The Y is a binary label, where $Y = 0$ if the images are similar to each other or $Y = 1$ otherwise. The architecture returns two points (G_X and G_Y) as feature vectors in the low-dimensional space. W is a shared parameter vector W which is subject to learning. Then, the system calculates the compatibility between the images.

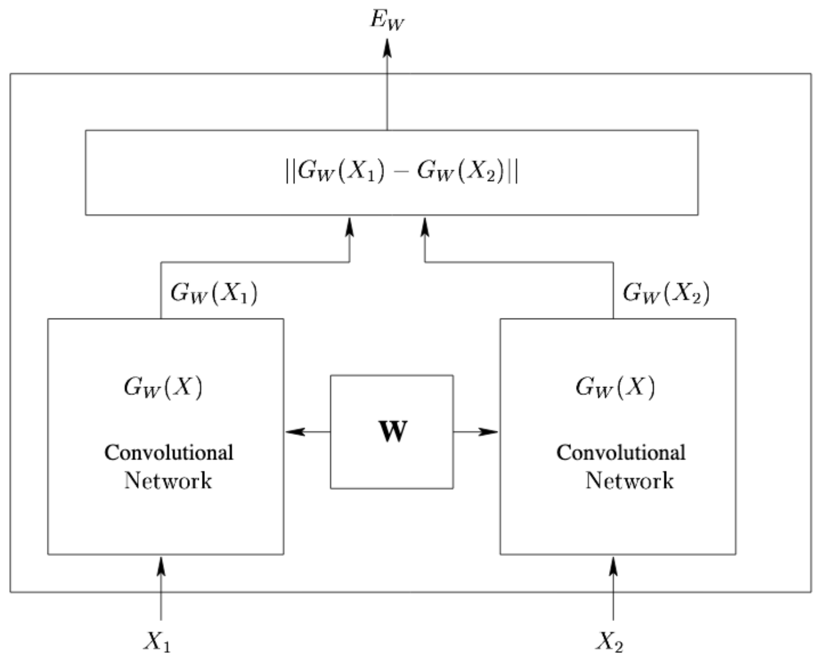

E_W
$||G_W(X_1) - G_W(X_2)||$
$G_W(X_1)$
$G_W(X_2)$
$G_W(X)$ Convolutional Network
W
$G_W(X)$ Convolutional Network
X_1
X_2

Figure 2.14: The siamese network implemented by Chopra, Hadsell and LeCun (2005)

Recently, some authors are using SNN for different purposes, as shown in Table 2.4.

Table 2.4: Summary of recent works for Siamese Networks in IR

Author	Dataset	Model
Qi et al. (2016)	Flickr15K	A novel architecture of CNN
Ong, Husain and Bober (2017)	Oxford and Paris	VGG + Fisher Vector Aggregation
Chung and Weng (2017)	Medical images	A novel architecture of CNN
Lin et al. (2015b)	Aerial images	A novel architecture of CNN

Qi et al. (2016) proposed a sketch-based image retrieval to pull output feature vectors closer for input pairs that are labeled as similar and push them away if irrelevant. They used query images to train the SNN. However, IR and PS tasks do not rely on any prior information regarding the query, nor predefined class of graphical objects (EN et al., 2016c).

Recent work by Chung and Weng (2017) focused on a Siamese Convolutional Neural Network (*SCNN*) pre-trained in Imagenet dataset for IR. The proposed method computed contrastive loss function and showed good performance in Diabetic Retinopathy Fundus image dataset. The results are comparable to the state-of-the-art, but this architecture needs high computational power although the author uses only binary image pair information. Besides, the contrastive loss function limits the use of a deep metric in the experiments. In contrast, in document images, the features can be more complex, with variation in shape, color, and texture, and in this case, both internal and external representation should be used. Figure 2.15 shows the SCNN used by authors, in which two identical CNN 2.15(a) learn to differentiate an image pair by evaluating the similarity and relationship between the given images.

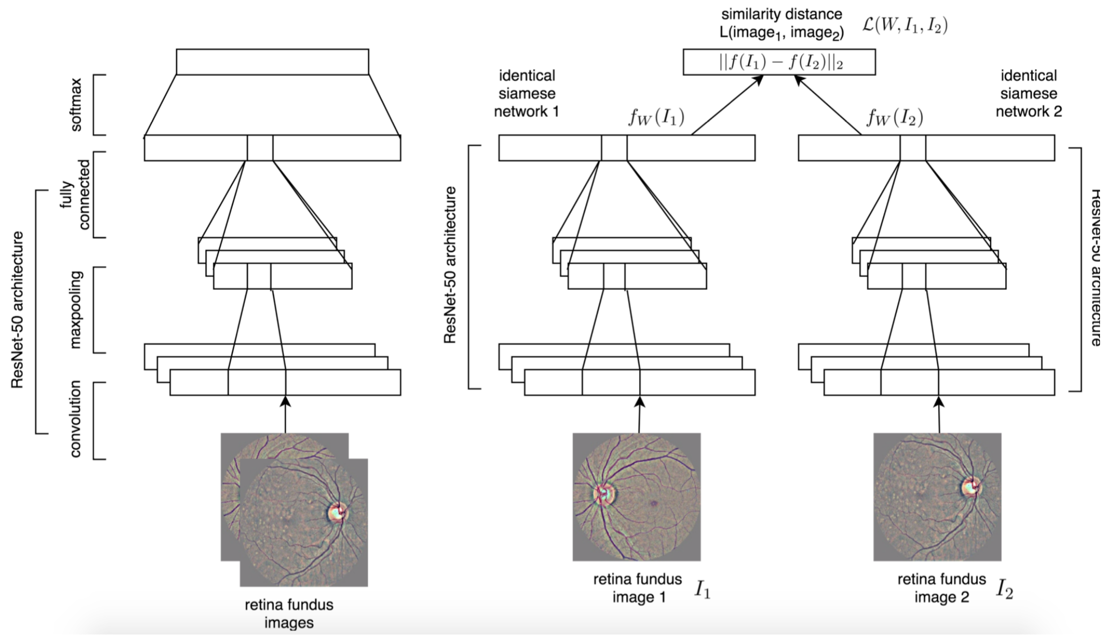


Figure 2.15: Architecture of used model in the study of Chung and Weng (2017). (a) Single CNN, and (b) proposed SCNN.

Ong, Husain and Bober (2017) proposed a novel SCNN that simultaneously learn both the CNN filter weights and Fisher Vector model parameters, although they perform supervised learning. Euclidean distance is metric learning. The method improves on the retrieval performance of the following state-of-the-art approaches: SCNN with max-pooling and Triplet loss with max-pooling. Lin et al (LIN et al., 2015b) also proposed a SCNN to learn a feature representation and finding matches between street view and aerial view imagery. However, they used the dataset to train and test the representation, but again, this is different from IR and PS objective because of the information regarding the queries.

We observe that the authors often do not explain the computational cost of using SCNN. However, it can require high computational power because of the number of combinations to calculate the similarity between the pair of images. Thus, a possible solution is to add a new layer to produce a compact representation, or similarity calculations can be added to solve IR problems. These methods are described in the next few sections.

2.3.4 Learning additional layers

Region proposal methods often produce a large number of candidates in each document, and storing all these regions, feature extraction, and calculation of the distance between vectors can become very costly. It is expensive because the IR process involves searching for the nearest neighbor, in which the main idea is to find the regions that are closest to the query image. This problem can be solved by techniques that generate the data in a compact dimensional space.

In this way, a layer can be developed that will receive the feature map from the higher layers and carry out a new function. This function will be responsible for generating a feature space with smaller size without losing all the information from the higher levels. But, to perform this task, some crucial choices are necessary, such as an adequate activation function and the number of outputs.

Some authors use strategies to generate binary codes through the networks as feature extractors. First, a $n-bit$ code is generated to form a hash code. Then, the hash code is transformed with a function f_k that normally considers Equation

2.10:

$$h_k(x) = \begin{Bmatrix} 1 & f_k(x) \geqslant b_k \\ 0 & f_k(x) < b_k \end{Bmatrix} \tag{2.10}$$

in which x is the representation of a data sample, f_k is a hashing function, and b_k is the corresponding threshold. Based on the method for receiving f_k, can use data-dependent (or learning-based) in supervised or not-supervised methods.

Several hashing algorithms such as Locality Sensitive Hashing (*LSH*) and Spectral Hashing have been proposed as a solution for identifying relevant data. LSH is the more popular approach and was originally introduced by Indyk and Motwani (1998) for problems with the nearest neighbor search. This technique aims to maximize the probability that the data is mapped to a similar binary code, and there is a projection of data points for a random hyperplane. However, LSH requires a great number of bit or multiple hash tables to reach a satisfactory search performance (LU; ZHENG; LI, 2017).

Unsupervised methods use only unlabeled data as a training set to overcome this limitation as Kernelized LSH (KULIS; GRAUMAN, 2009), Semantic Hashing (KRIZHEVSKY; HINTON, 2011), and Spectral Hashing (WEISS; TORRALBA; FERGUS, 2009). Some authors such as Kulis and Grauman (2009), Gionis, Indyk and Motwani (1999) and Xu et al. (2017) present methods of describing an improved version of this algorithm and obtaining better execution time and the analysis are generalized.

Recent studies have proposed methods based on CNN for hashing. Guo, Zhang and Li (2016) introduced a hash layer in a hashing learning task in a CNN. The objective was to learn k hash functions that are used to codify images in k-bit hash codes. The codes are used for recovery, and the shared images that share the same label with the query image will be ranked at the top of the list of results. They proposed to learn semantic-preserving hash function, defining an original softmax loss layer, a hash layer and hinge softmax layer.

Xia et al. (2014) proposed a two-stage supervised method for IR. The first state is pre-processing, in which a matrix-decomposition algorithm learns the representation of the datum. They used CNN to learn k hash codes and image representation. However, the first stage requires the input of a pairwise similarity matrix of approximate target hash codes and this is unfavorable when the size of the data is large because it takes up considerable storage space.

Lin et al. (2015a) proposed the use of TL based on data from ImageNet for fine-tuning the network in a new layer for learning the representation and generating hash codes. They adopted a coarse-to-fine search strategy in which the first n outputs of a new layer are extracted and k binary codes are obtained by binarization of the activations. Then, given a query image and its k binary codes, a pool of m candidates is generated by the Hamming distance. Finally, in the IR task, given a query image and a candidate pool, the features of the images were extracted from the last fully connected layer before the hash layer ($fc7$ for AlexNet) to then generate the image ranking by similarity using Euclidean distance.

In this way, the use of hashing for deep architectures proceeds to simultaneously learn a specific domain and a set of a hash function are satisfactorily applied in large data sets as shown in the literature. However, such information is lost using hashing methods, so other approaches should be considered, such as the use of additional layers in the architectures. Functions and numbers of outputs of the layers can be chosen without converting to binary codes. This methodology will be shown in Chapter 3, as well as in the experiments, creating an additional layer in the network model to reduce the number of features with a single CNN or SCNN.

2.4 Similarity Measures and Distances

In spotting or retrieval systems, the analysis of the performance of the method can be carried out by using a numerical measure capable of defining the level of similarity between two images: the query image and the candidate present in the document. The representation of images through feature vectors is used in the similarity calculation. Besides, there are methods of appropriate distance for comparing image representations, such as Euclidean distance and cosine. This section presents several similarities and dissimilarity measures frequently used in computational vision. More details about the measures used in this study will be provided in Chapter 3.

According to the features used in image representation, different similarity measures can be used. Euclidean distances and cosine are frequently used to work with clustering analysis (EN et al., 2016c), but if the information is binary, the Hamming distance is the appropriate measure (RIBA et al., 2015). In the case of template matching, in which the images are compared pixel by pixel (DELALANDRE; OGIER; LLADÓS, 2008) a similarity measure can also be considered (SAIKRISHNA et al., 2012).

The use of the term *distance* is commonly used in dissimilarity calculations. In the case of the present study, distance is calculated between feature vectors extracted by a CNN to carry out a retrieval task. The choice depends on the image set and its representation. In this sense, first all the dissimilarity functions have some criteria (TAN; STEINBACH; KUMAR, 2005):

- $d(x, y) \geq 0$ for all x and y;
- $d(x, y) = 0$ only if $x = y$;
- $d(x, y) = d(y, x)$, when the distance between two elements is the same;
- $d(x, y) + d(y, z) \geq d(x, z)$, known as a triangular difference to x, y e z points.

In document image retrieval, the image set is organized as feature vectors, with $c_j = \{a_{1,j}, a_{2,j}..., a_{n,j}\}$ and $q_i = \{a_{1,i}, a_{2,i}, ...a_{n,i}\}$, in which c_j is the candidate from document j and q is the query vector with index i. The n corresponds to the number of features from each image. The query must be compared with all the vectors of the candidates in the data set to obtain a ranking of the results. These results must be ordered according to the chosen measure. The main distances found the literature will be described below.

Minkowski distance

Minkowski distance is the generalization of the distance between two points x, y, in an n-dimensional characteristic space (TAN; STEINBACH; KUMAR, 2005). The distance is defined by Equation 2.11.

$$d(c_j, q_i) = \left(\sum_{k=1}^{n} |c_{kj} - q_{ki}|^r \right)^{1/r} \tag{2.11}$$

in which k is the index of c_j e q_i. The parameter r can represent different values and variations. The most widely used are:

- $r = 1$. Manhattan distance is used between two object with binary characteristic vectors, defined by equation 2.13.

$$d(c_j, q_i) = \sum_{k=1}^{n} (c_{kj} - q_{ki}) \tag{2.12}$$

- $r = 2$. Euclidean distance is a function that is traditionally used, and corresponds to the Equation 2.13:

$$d(c_j, q_i) = \sqrt{\sum_{k=1}^{n} (c_{kj} - q_{ki})^2} \tag{2.13}$$

- $r = \infty$. Chebychev distance is the maximum difference between any atributes of the objects, defined by Equation 2.14.

$$d(c_j, q_i) = \lim_{r \to \infty} \left(\sum_{k=1}^{n} |c_{kj} - q_{ki}|^r \right)^{1/r} \tag{2.14}$$

Mahalanobis distance.

This distance is different from the Euclidean distance because the correlation between the data set is considered. Firstly, a matrix of covariance C is computed and the distance between two vectors of the same distribution is defined by Equation 2.15 (MAESSCHALCK; JOUAN-RIMBAUD; MASSART, 2000):

$$d(\vec{c_j}, \vec{q_i}) = (\vec{c_j} - \vec{q_i})^T C^{-1} (\vec{c_j} - \vec{q_i}) \tag{2.15}$$

If the covariance matrix is an identity matrix, this measure is reduced to Euclidean distance, and, if the matrix is diagonal, it is reduced to normalized Euclidean distance, defined by Equation 2.16.

$$d(\vec{c_j}, \vec{q_i}) = \sqrt{\sum_{n=1}^{p} \frac{(c_{nj} - q_{ni})^2}{\sigma_n^2}} \tag{2.16}$$

Cosine similarity

In this distance, the attributes are used as a vector to find the normalized dot product of a pair of bit vectors. Two vectors with the same direction have similarity equal to 1 and two opposite vectors have similarity equal to -1. The similarity is calculated by the equation 2.17 (TAN; STEINBACH; KUMAR, 2005)(KUO; CHOU; CHANG, 2016).

$$Similarity(c_j, q_i) = \frac{\vec{c_j} \bullet \vec{q_i}}{|\vec{c_j}| \cdot |\vec{q_i}|} = \frac{\sum_{k=1}^{n} c_j \cdot q_i}{\sqrt{\sum_{k=1}^{n} c_j^2} \cdot \sqrt{\sum_{k=1}^{n} q_i^2}} \tag{2.17}$$

in which $\vec{c_j} \bullet \vec{q_i}$ is the dot product between the vectors c_j and q_i. in this way, the distance between two correlated vectors is presented in Equation 2.18.

$$d_{cosine}(c_j, q_i) = 1 - cos^{-1}(Similarity(c_j, q_i)) \tag{2.18}$$

Sum of absolute difference

It is a simple measure of similarity. The distortion is calculated between the current object and the reference (query), using the absolute difference between them. This difference calculated considering n-dimensions and is defined by Equation 2.19 (OLIVARES et al., 2004).

$$SAD(c,q) = \sum_{i=1}^{M} \sum_{j=1}^{N} |c_{i,j} - q_{i,j}| \tag{2.19}$$

in which $q_{i,j}$ is each element in the query image and $c_{i,j}$ is each element of the candidate object, with $M \times N$ dimensions. However, the computation of SAD is divided into three steps: 1. Calculate the difference between the query and the candidate; 2. Determine the absolute value of these distances; 3. add all the absolute values.

Malik and Baharudin (2013) carried out an analysis between different measures to compare the performance in image recovery approaches, including SAD, city block distance, Euclidean, maximum value metric and Minkowski difference. They concluded that the Euclidean distance and SAD have good precision. However, the Euclidean and cosine distance also returns good results, as presented by (EN et al., 2016c; RUSIÑOL et al., 2015; RIBA et al., 2015). In this way, to end this section, the table 2.5 shows a summary of the main similarity measures used in the literature, where it can be observed that the Euclidean distance and cosine have been used recently in retrieval tasks and, the authors reported good results using these methods.

2.5 Parallel Computing

The increasing amount of image data and massively parallel hardware have opened new opportunities for researchers investigating the use of both CPUs and GPUs. Hence, IR and PS applications are data-intensive and computation-intensive, which creates significant challenges to process image data in real-time (FANG et al., 2011). Although parallel computing implementations are not main objectives for IR and PS tasks, the study and development of approaches to reduce the computational cost can improve the performance, mainly with limited resource

Table 2.5: Summary of the matching techniques

Work	**Representation**	**Method**
(ALMAZÁN et al., 2014)	FV	Euclidean distance
(EN et al., 2016c)	BoVW, VLAD and FV	Cosine similarity
(RUSIÑOL et al., 2015)	BoVW	Cosine similarity
(RIBA et al., 2015)	Graphs	Cosine similarity
(TOSELLIA et al., 2016)	Grayscale or geometric vectors	DTW
(KANADJE et al., 2016)	MFCC	Segmented DTW
(MONDAL et al., 2016)	RLSA	DTW
(TABIBIAN; AKBARI; NASERSHARIF, 2016)	PCF and DCF	M-Viterbi
(HU et al., 2012)	Binary vectors	CK1 distance
(THUY et al., 2017)	Color and texture features	Euclidean distance
(HANGARGE et al., 2016)	Gabor features	Cosine similarity
(HE et al., 2016)	Adapted BoVW	Manhattan and Cumulative Score
(SAIKRISHNA et al., 2012)	Primitives of color moments	SAD
(KAVITHA et al., 2011)	Texture features	L2 distance and Euclidean distance

computational. In this sense, this section presents some parallel computing concepts.

The compute resources are typically a single computer with multiple processors/cores or an arbitrary number of such computers connected by a network, thus the use of parallel computing concept is promising. In this way, parallel computing is the use of multiple computes resources simultaneously to solve a specific problem (BARNEY, 2018):

- A problem is broken into discrete parts that can be solved concurrently
- Each part is further broken down to a series of instructions
- Instructions from each part execute simultaneously on different processors
- An overall control/coordination mechanism is employed

Figure 2.16 shows the computational problem should be able to be broken apart into discrete pieces of work that can be solved simultaneously; Execute multiple program instructions at any moment in time; Be solved in less time with multiple compute resources than with a single compute resource.

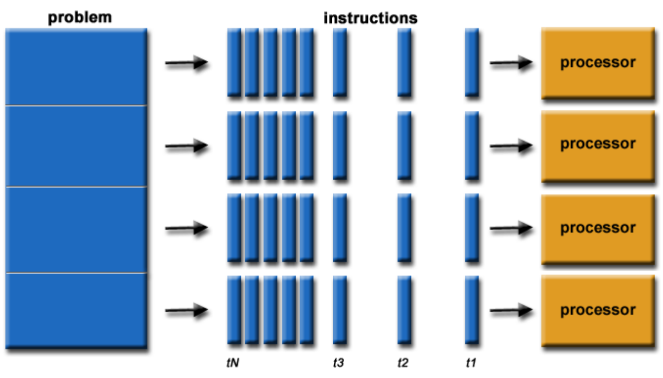


Figure 2.16: Example of parallel problem

Source:(BARNEY, 2018)

2.5.1 Process and subprocess

The concept of process is divided into two: a set of resources to execute a program and, a line or context execution called thread. A process is an abstract entity that uses the code and data to produce the output of that task activated by the parallel program, within a finite amount of time. That is, a process is just an instance of an executing program, including the current values of the program counter, registers, and variable. During this time, in addition to performing computations, a process may synchronize or communicate with other processes, if needed (GRAMA et al., 2003)(TANENBAUM, 2014). Four principal events cause processes to be created:

- System initialization.
- Execution of a process creation system call by a running process.
- A user request to create a new process.
- Initiation of a batch job.

For each one of these events, the process can be divided into three basic elements (see Figure 2.17): hardware context, software context and address space, which together keep all the information necessary to execute the program (MACHADO; MAIA, 1999):

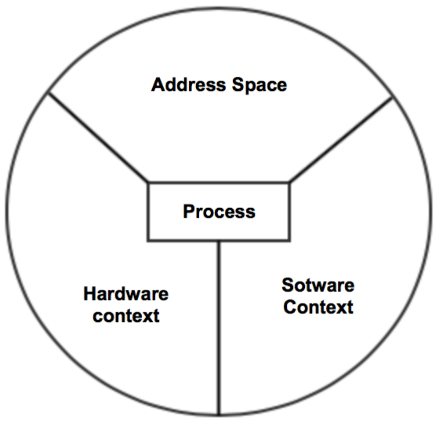


Figure 2.17: Example of process

Adapted from (MACHADO; MAIA, 1999)

- Hardware context: is the content of registers with the program counter (PC), the stack pointer (SP) and state bits. When a process is running, the hardware context is stored in the processor registers. At the time the process loses the use of the CPU, the system saves the information.
- Software context: specifies process characteristics that will influence the execution of a program, such as the maximum number of open files simultaneously or the size of the buffer for I/O operations. These characteristics are determined at the time of the process creation, but some can be changed during its existence. The software context defines three groups of information in a process: identification, quotas, and privileges.
- Address space: is the memory area of the process where the program will run, in addition to the space used for the data. Each process has an address space, which must be protected from access to other processes. This area of

memory as well as files, registers, and I / O devices can be shared between processes, which can compromise the execution of applications. To avoid this kind of problem, the processes must be synchronized from the mechanisms offered by the operating system. Figure 2.18 provides an example where two concurrent processes share a buffer to exchange information through write and read operations. Thus, the data will be written only if the buffer is not full, the same to read. In both cases, the processes must wait for the buffer to be ready for operations. There are some algorithms to perform this synchronization, such as semaphore, monitors and messaging

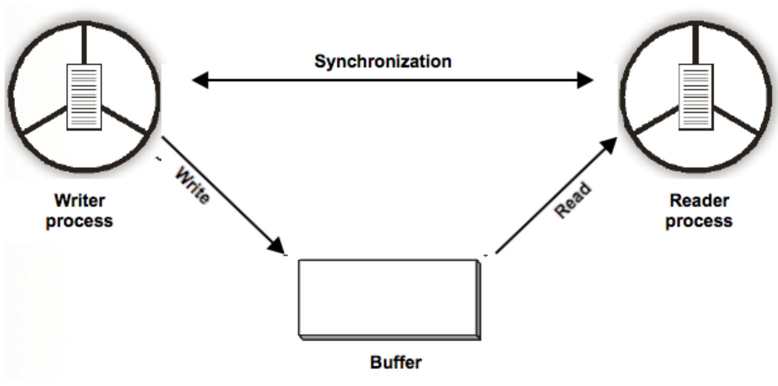


Figure 2.18: Example of synchronization between two process

Adapted from (MACHADO; MAIA, 1999)

For different applications, the initiation of a batch job creates processes. In some cases, a process can create another process, thus the main process and child process continue to be associated in certain ways. The child process can create more processes, forming a process hierarchy, as shows Figure 2.19.

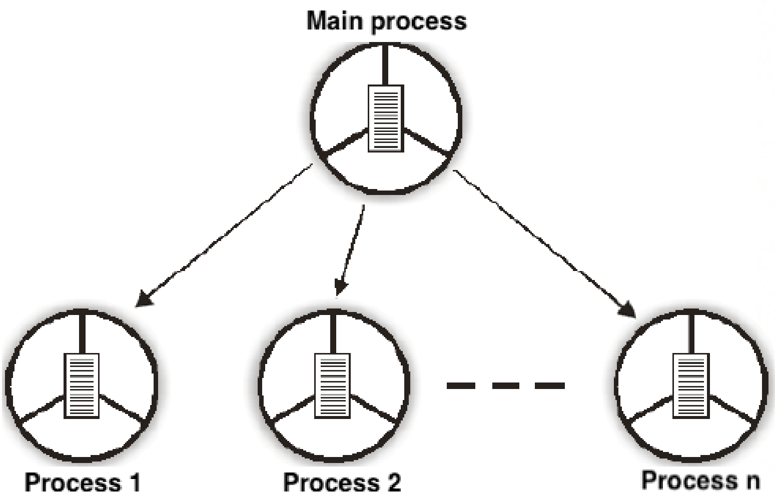


Figure 2.19: Example of sub-process

Adapted from (MACHADO; MAIA, 1999)

However, in a multi-processing, the structure of the memory is quite complex. Efficient memory management is very critical for good performance of the entire system. Shared memory allows several processes to access the same portion of main memory and very common in many applications. Figure 2.20 shows how shared memory works. In this example, processing elements share memory (either directly or indirectly); the communication among processing elements can be achieved by carefully reading and writing in main memory and data and load distribution can be hidden from the programmer (BEKAS, 2009).

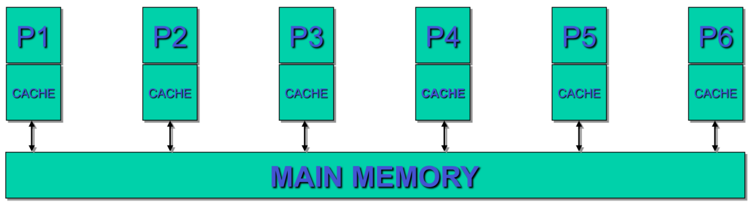


Figure 2.20: Shared memory

Source:(BEKAS, 2009)

2.5.2 Parallel implementation for Image Retrieval

Some parallel computing methods (KAO; STEINERT; DREWS, 2001) (LU et al., 2007) (TERBOVEN et al., 2006) have been introduced into the CBIR. However, most of these systems were designed only for some specifical task, as feature extraction step, reading files or similarity calculation.

According to Terboven et al. (2006) given the IR system, three different layers can be identified that offer potential for parallelization:

- Queries are mutually independent. The queries can be processed in parallel, that is, several queries are run in batch mode.
- The scores calculated for each query and candidates from a document can be calculated in parallel because the database images are independent of each other.
- The distances between the feature vector from query and candidates can be calculated in parallel.

For this purpose, Lu et al. (2007) present a parallel process for reading the parameters for the image feature extraction from the configuration file, identifying all image files in the specified directory, reading the image, and then generating a feature vector. However, there was no obvious modification to retrieval precision because they didn't do any improvement to the retrieval methods, only focused on the efficiency of the retrieval system.

Tungkasthan and Premchaiswadi (2013) propose a parallel processing technique via Hadoop MapReduce framework, contributing to the index creation process and similarity measurement process through MapReduce. Figure 2.21 shows the method proposed by the author to increase the performance of calculation, where the functions feature extraction and similarity measurement is applied. However, experiments with any dataset were not presented to prove the efficiency of the method.

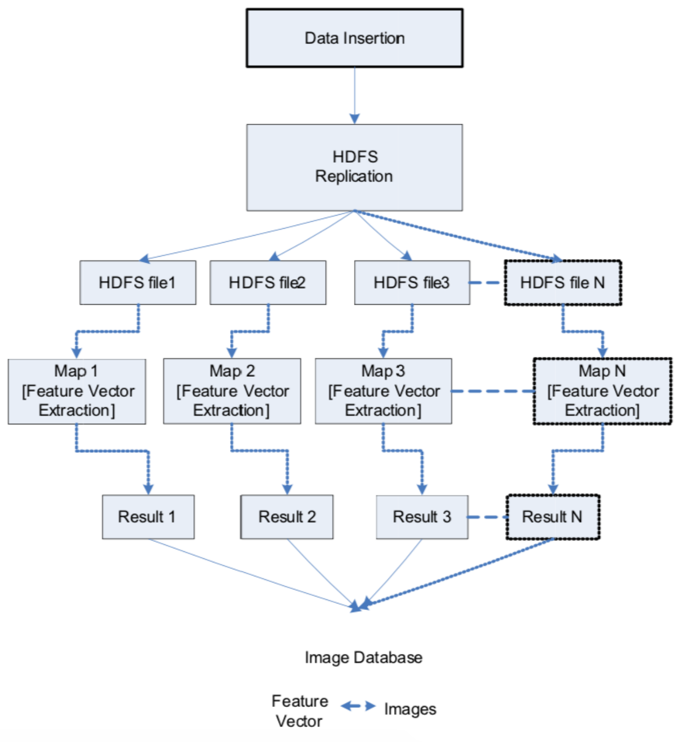


Figure 2.21: The proposed a distributed processing framework using MapReduce for data insertion module in CBIR system. Source:(TUNGKASTHAN; PREMCHAISWADI, 2013)

Lu et al. (2007) describes the importance of more research on finding the factors that would affect the searching efficiency of IR retrieval systems. In this sense, the influence of hardware devices on the efficiency of an IR system, such as the I/O equipment, the processor ability, and the message exchange system could be fully investigated.

All the authors presented above showed the importance of the use of parallel computing in the implementation of IR tasks. Perhaps without these methodologies, the adopted procedures become unfeasible. In this case, the present work presents the parallel computing approach used in Section 3.

2.6 Final Considerations

This chapter presented the literature review related to CBIR systems, describing each step in the process: candidate generation, feature extraction based on CNN models, computation of similarity measures and some definitions about parallel computing. Recent studies for WS and PS were also described, showing the methodology used. The next chapter (Chapter 3) presents the methodology used in the present study to develop a new and efficient IR and PS system.

Chapter 3

Proposed Method

Motivated by the significant growth of digital content storage observed in the last years, we propose here two approaches that contemplate both image retrieval and pattern spotting, which are the main tasks of an image search engine. The image retrieval consists of finding every document that contains a given query, while additional information is necessary for the pattern spotting, which is related to the exact location of the query in the retrieved images. Thus, Section 3.1 describes the pre-processing, while Sections 3.2 and 3.3 present the two proposed approaches with an overview of them. Section 3.4 presents the experimental protocol, while Section 3.5 describes the database used for the experimental protocol.

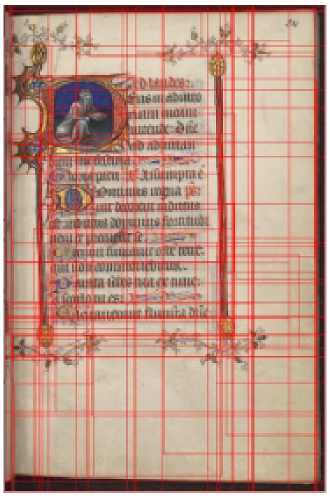
3.1 Pre-processing

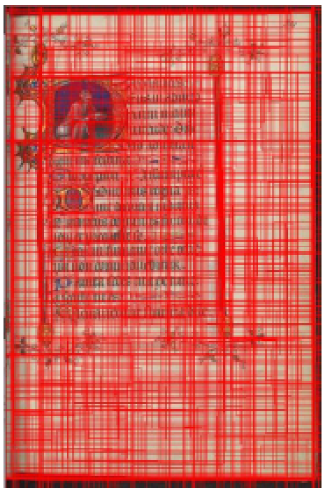
An algorithm to find candidate objects was defined to generate the list of candidates to be retrieved. The idea is to use an algorithm based on segmentation methods, which will be explained in this section.

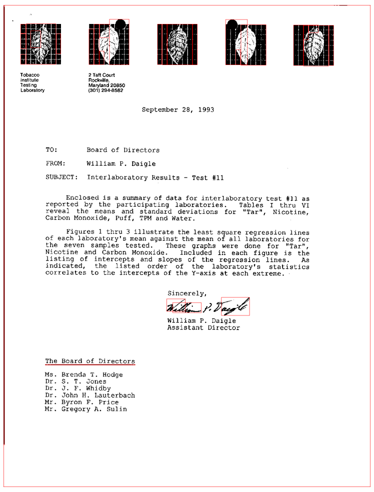
According to literature (ZITNICK; DOLLÁR, 2014) , the Selective Search (*SS*) is considered an algorithm with good performance to generate candidate objects. It is used to find and generate candidates in the document images. The objective of this algorithm is to generate a selective search strategy independent of class, based on data, that generates a small set of high-quality object locations (UIJLINGS et al., 2013). Besides, this algorithm considers that the objects can occur in different scales. There are other algorithms for this step, as shown in Section 2.2, but some of them can not be used in IR and PS tasks. For example, the algorithms proposed He and Lau (2015), Kuo, Hariharan and Malik (2015) and Liu, Lu and Jia (2015) depend on labels and, we can not use ground truth as previous information to search for candidates.

Two versions of SS were used in this study. The first approach, named SSv1, is available at `http://dlib.net/imaging.html#find_candidate_object_locations` that is based on the implementation of Uijlings et al. (2013). This version was used without modifications of the parameters and it does not remove background. Figures 3.1a and 3.1b show two examples of the SS algorithm applied on a medieval document and Figures 3.1c and 3.1d of an administrative document. For the example presented in Figure 3.1c, the maximum candidate size was limited (for example 160 x 160), while in the example in Figure 3.1d, all the possible candidates were selected. The experiments used an unlimited size.

In the second version presented in Figure 3.2, named SSv2, the tool available at `https://github.com/belltailjp/selective_search_py` was used. This is a SS tool implemented in Python and also based on Uijlings et al. (2013) and Sande et al. (2011). In this tool, it is possible to modify the search parameters for possible candidates, selecting the color space, size of the initial search seed (s), and the merges (similarity between regions), which consider color, fill, size and texture.


(a) Example 1 - limited size


(b) Example 2 - unlimited size


Tobacco
Institute
Testing
Laboratory

2 Taft Court
Rockville,
Maryland 20850
(301) 294-8582

September 28, 1993

TO: Board of Directors

FROM: William P. Daigle

SUBJECT: Interlaboratory Results - Test #11

Enclosed is a summary of data for interlaboratory test #11 as reported by the participating laboratories. Tables I thru VI reveal the means and standard deviations for "Tar", Nicotine, Carbon Monoxide, Puff, TPM and Water.

Figures 1 thru 3 illustrate the least square regression lines of each laboratory's mean against the mean of all laboratories for the seven samples tested. These graphs were done for "Tar", Nicotine and Carbon Monoxide. Included in each figure is the listing of intercepts and slopes of the regression lines. As indicated, the listed order of the laboratory's statistics correlates to the intercepts of the Y-axis at each extreme.

Sincerely,

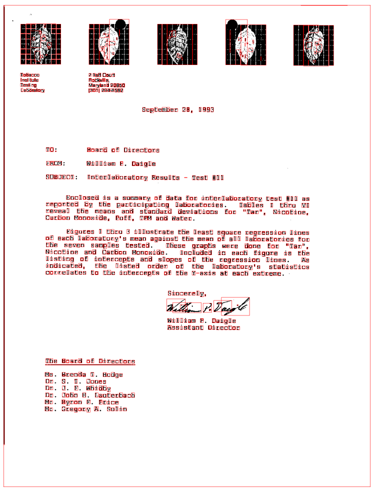
[illegible]

William P. Daigle
Assistant Director

The Board of Directors

Ms. Brenda T. Hodge
Dr. S. T. Jones
Dr. J. F. Whidby
Dr. John H. Lauterbach
Mr. Byron F. Price
Mr. Gregory A. Sulin

(c) Example 3 - limited size


Tobacco
Institute
Testing
Laboratory

2 Taft Court
Rockville,
Maryland 20850
(301) 294-8582

September 28, 1993

TO: Board of Directors

FROM: William P. Daigle

SUBJECT: Interlaboratory Results - Test #11

Enclosed is a summary of data for interlaboratory test #11 as reported by the participating laboratories. Tables I thru VI reveal the means and standard deviations for "Tar", Nicotine, Carbon Monoxide, Puff, TPM and Water.

Figures 1 thru 3 illustrate the least square regression lines of each laboratory's mean against the mean of all laboratories for the seven samples tested. These graphs were done for "Tar", Nicotine and Carbon Monoxide. Included in each figure is the listing of intercepts and slopes of the regression lines. As indicated, the listed order of the laboratory's statistics correlates to the intercepts of the Y-axis at each extreme.

Sincerely,

[illegible]

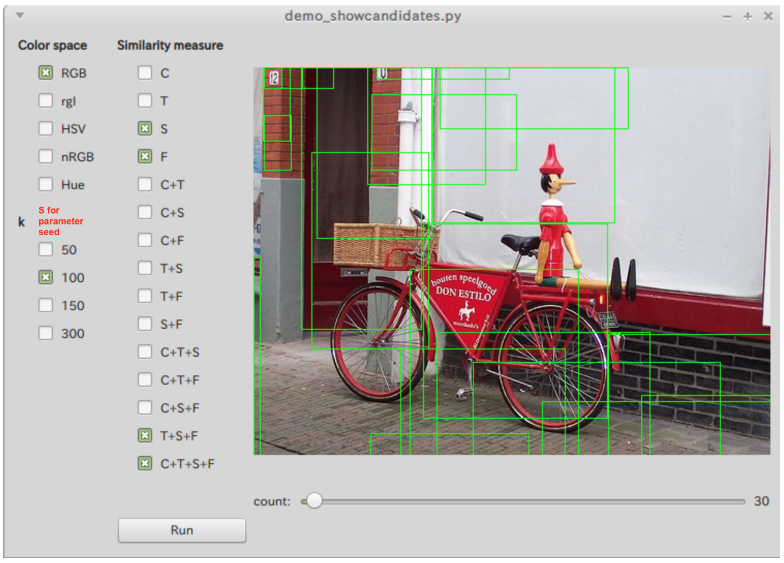
William P. Daigle
Assistant Director

The Board of Directors

Ms. Brenda T. Hodge
Dr. S. T. Jones
Dr. J. F. Whidby
Dr. John H. Lauterbach
Mr. Byron F. Price
Mr. Gregory A. Sulin

(d) Example 4 - unlimited size

Figure 3.1: SSv1 algorithm applied in an image of the medieval document (examples 1 and 2) and administrative document (examples 3 and 4)


demo_showcandidates.py

Color space
- ☒ RGB
- ☐ rgI
- ☐ HSV
- ☐ nRGB
- ☐ Hue

k — S for parameter seed
- ☐ 50
- ☒ 100
- ☐ 150
- ☐ 300

Similarity measure
- ☐ C
- ☐ T
- ☒ S
- ☒ F
- ☐ C+T
- ☐ C+S
- ☐ C+F
- ☐ T+S
- ☐ T+F
- ☐ S+F
- ☐ C+T+S
- ☐ C+T+F
- ☐ C+S+F
- ☒ T+S+F
- ☒ C+T+S+F

count: 30

Figure 3.2: Original tool of SSv2

Source: `https://github.com/belltailjp/selective_search_py`

After analysis of these initial parameters, the tool was improved to also perform thresholding on the image (global and adaptive). For the global threshold, the value of the standard histogram 127 may be used, since it is the equivalent of middle gray in density, while in the adaptive, there is the possibility of modifying the values of the block size (deciding the size of the neighborhood) and offset (constant subtracted from the mean of the neighborhood). Figure 3.3 shows the result of the use and adaptation of the tool for the implementation of SSv2.

- (A) it is possible to select to work with the original image (*ndimage*), global threshold or adaptative threshold.
- (B) it is possible to write the parameters of the threshold model, as well the block size and offset.
- (C) it is possible to put a precise location of an object of interest to observe the result of segmentation for one document example.
- (D) it is possible to select the search algorithm (not only selective search but also sliding windows and OpenCV algorithm). Other algorithms were included to compare the results and the quality of objects.
- (E) the parameters color space, the size k of the initial candidate seed and similarity measure can be selected.
- (F) presents the number of candidates generated to the page.
- (H) presents the result of the threshold.
- A button was added to save all the objects for the pages with the parameters selected.

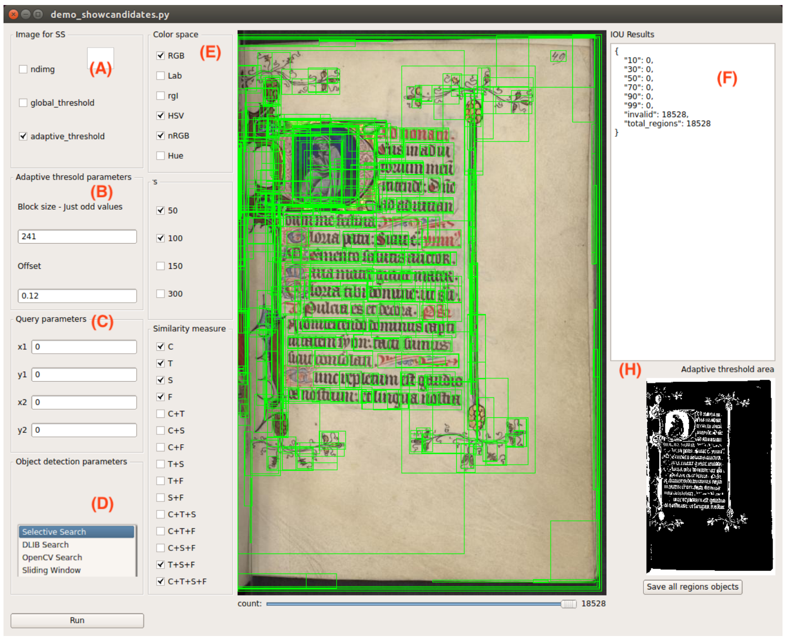


Image for SS
☐ ndimg
☐ global_threshold
☑ adaptive_threshold

Adaptive thresold parameters
Block size - Just odd values: 241
Offset: 0.12

Query parameters
x1: 0
y1: 0
x2: 0
y2: 0

Object detection parameters
Selective Search
DLIB Search
OpenCV Search
Sliding Window

Color space
☑ RGB
☐ Lab
☐ rgI
☑ HSV
☑ nRGB
☐ Hue

's
☑ 50
☑ 100
☐ 150
☐ 300

Similarity measure
☑ C
☑ T
☑ S
☑ F
☐ C+T
☐ C+S
☐ C+F
☐ T+S
☐ T+F
☐ S+F
☐ C+T+S
☐ C+T+F
☐ C+S+F
☑ T+S+F
☑ C+T+S+F

count: 18528

IOU Results
```
{
    "10": 0,
    "30": 0,
    "50": 0,
    "70": 0,
    "90": 0,
    "99": 0,
    "invalid": 18528,
    "total_regions": 18528
}
```

Adaptive threshold area

(a) Adapted tool - example 1

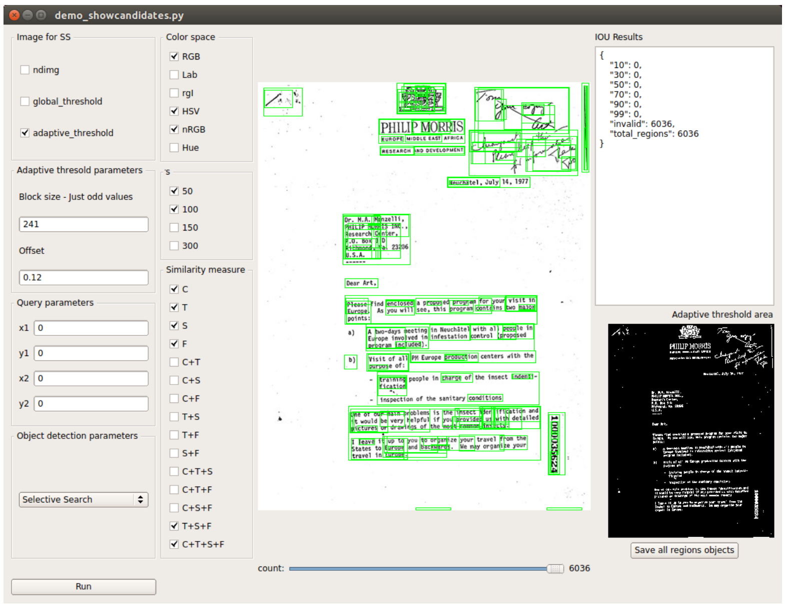


Image for SS
☐ ndimg
☐ global_threshold
☑ adaptive_threshold

Adaptive thresold parameters
Block size - Just odd values: 241
Offset: 0.12

Query parameters
x1: 0
y1: 0
x2: 0
y2: 0

Object detection parameters
Selective Search

Color space
☑ RGB
☐ Lab
☐ rgI
☑ HSV
☑ nRGB
☐ Hue

's
☑ 50
☑ 100
☐ 150
☐ 300

Similarity measure
☑ C
☑ T
☑ S
☑ F
☐ C+T
☐ C+S
☐ C+F
☐ T+S
☐ T+F
☐ S+F
☐ C+T+S
☐ C+T+F
☐ C+S+F
☑ T+S+F
☑ C+T+S+F

count: 6036

IOU Results
```
{
    "10": 0,
    "30": 0,
    "50": 0,
    "70": 0,
    "90": 0,
    "99": 0,
    "invalid": 6036,
    "total_regions": 6036
}
```

Adaptive threshold area

(b) Adapted tool - example 2

Figure 3.3: SSv2 algorithm applied in a document image of a medieval document (example 1) and administrative document (example 2)

For each document image, a different number of candidates was generated, depending on the quantity of information present in the image. However, it is necessary to do experiments to find the best configuration of parameters to return objects with high quality. The results of these experiments are shown in Chapter 4.

The next step is related to image representation. Thus, the proposed approaches are used for identification of queries in collections of document images, whether they be handwritten documents or symbols, drawing, letters, etc. In this sense, the next sections present a general overview of the methods used and the respective steps that make them up, as well as materials used in the application.

3.2 First approach - single CNN

The first approach is based on a single CNN architecture as a feature extractor. Figure 3.4 shows a general overview of the proposed method that works with IR and PS in collections of document images. This approach has two phases: online e offline. The offline phase begins with two steps: 1) data augmentation to train the feature extractor based on the representation of the queries, in which a set of transformations is applied to each query image; and 2) pre-processing to produce a list of candidates for each document image based on an object segmentation method. Then a fine-tuned (FT) CNN is used as a feature extractor. This is done considering two strategies: a) the training of a standard CNN model (SM) and b) training of a model with an additional layer for feature extraction (CM). The online phase is made up of the retrieval process, in which candidates are selected and compared with each query image in the collection, ending with the calculation of the similarity measure.

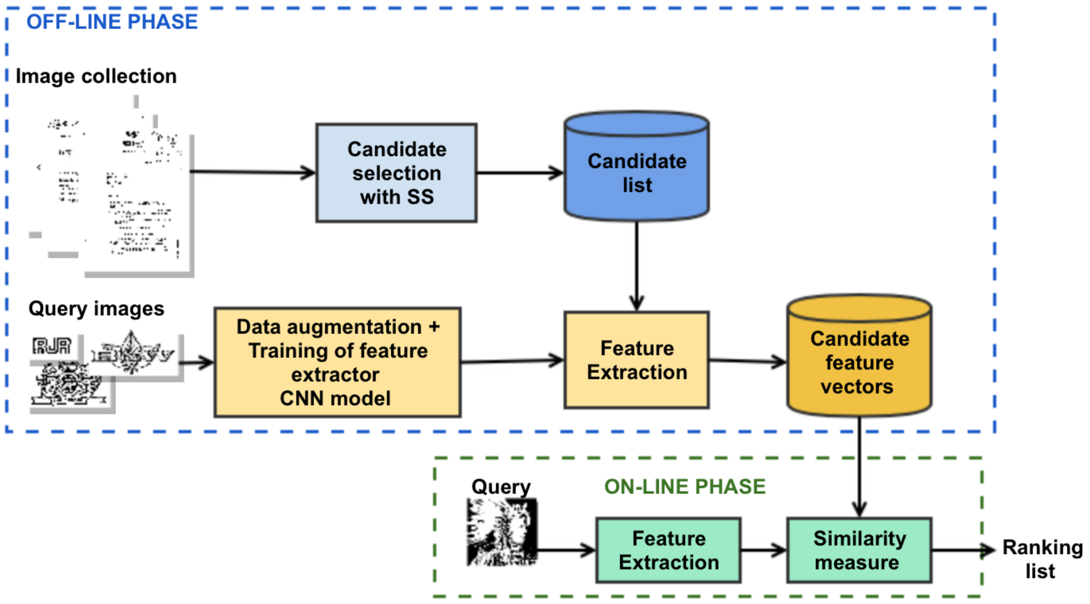


Figure 3.4: Overview of the proposed Single CNN

Source: The author

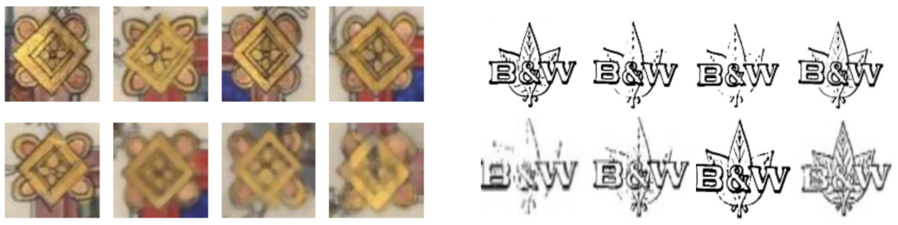
3.2.1 Pre-processing

The first step used data augmentation to train a feature extractor. The data augmentation process uses query images to generate the training set for the CNN. This set is composed of N images, the number of queries in the set $X = \{x_1, x_2, \cdots, x_N\}$. Then, the following transformations were applied to each query $x_i \in X$, based on Dosovitskiy et al. (2014) .

- scale: resizing the image by a coefficient between -20 and 20 with stride 2;
- average filter: replacing the value of each pixel by an average of the intensity levels of its neighborhood in each scaled image.

These image transformations are carried out to increase the number of training samples since the deep architecture needs a large number of samples to find the proper representation of the pattern to be learned.

Dosovitskiy et al. (2014) applied in their research other transformations, such as vertical and horizontal translation, contrast and added color in the H channel of the HSV color space. They also trained a CNN model to use it as a feature extractor. Figure 3.5 shows two examples of data augmentation applied to a query of DocExplore and Tobacco800 dataset, respectively. Thus, the main idea of using data augmentation is to increase the number of samples in each class to perform the training on a CNN model, avoiding to use the original queries as a training set.



(a) Example 1 - Medieval document (b) Example 2 - Administrative document

Figure 3.5: Data augmentation applied to queries of medieval and administrative documents

Source: The author

3.2.2 Image Representation

Deep features will be used to represent the candidates generated by the algorithm in Section 3.1 and, the query images. With this representation model, an appropriate architecture model can work with the great variability in each object in terms of color, textures, and sizes. This training process uses transfer learning approach from a model pre-trained on ImageNet data set. The weights of first layers are frozen and the last fully convolutional layer is fine-tuned with transformed queries (data set from data augmentation step). This set is divided into 75% for the training set and 25% for validation, with the size of the images fixed at 256 x 256.

The parameters for configuration of the CNN model will follow the values most commonly used in the literature for the first experiments: learning rate: between 10^{-3} e 10^{-5} and linear or exponential function, optimization function: SGD and snapshot: every 10 iterations.

For this approach, the CNN model is based on the AlexNet architecture (KRIZHEVSKY; SUTSKEVER; HINTON, 2012), which is composed of eight layers of which five are convolutional followed by fully connected layers. This architecture was chosen because returns good performance in the Imagenet challenge (RUSSAKOVSKY et al., 2015). Note that there are many more network topologies: for example, Szegedy et al. (SZEGEDY et al., 2015) proposed GoogLenet, but their model is theoretically 50% more complex than AlexNet; Simonyan and Zisserman (SIMONYAN; ZISSERMAN., 2015) proposed architecture with very small convolution filters, which is however slow to train and needs more memory.

Therefore, the AlexNet has been considered as a good starting point for the proposed method. It is easy to implement and has shown to be effective in different deep learning scenarios, showing a good compromise between the number of layers and final accuracy. Besides, the non-linear part trains much faster than standard functions (Sigmoid and Tanh).

Different approaches to the AlexNet model will be evaluated as the feature extractor of the proposed method. The idea is to compare different architectures and the relevance of its features, as follows:

- AlexNet standard model (SM): the model is trained with original configurations;
- AlexNet + fine-tuned model (SM + FT): the model is trained using the transfer learning process, sharing the weights in high layers and training the last layer;
- AlexNet standard model + additional layer (SM + CM): the model is trained with original configuration and one layer is added to extract reduced features with n dimensions;

- AlexNet + fine-tuned model + additional layer (CM + FT): the model is trained using the transfer learning process, sharing the weights in high layers and training one additional layer to extract reduced features;

The additional layer in the CNN model

We intend to evaluate the use of an additional layer in the model to reduce the number of features of the original Alexnet model. The feature map of this layer is used to perform the IR and PS task. Figure 3.6 shows the model of the AlexNet architecture used as a reference and adapted for this study.

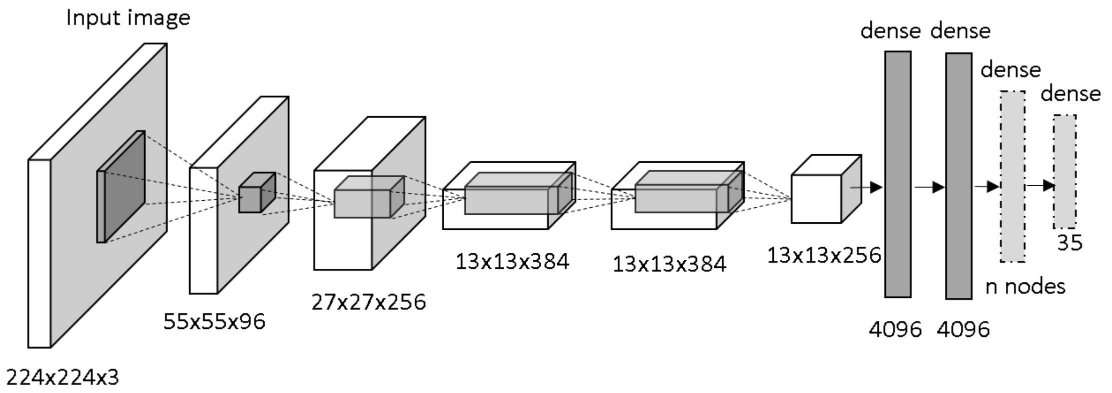


Figure 3.6: An illustration of the architecture proposed by Krizhevsky et al. (KRIZHEVSKY; SUTSKEVER; HINTON, 2012) and adapted for our first approach in the last layers. The n is the number of features

As shown in Figure 3.6, we added a layer in the fully connected part of pre-trained AlexNet architecture to reduce the dimensionality of the feature maps (LIN et al., 2015a). Afterward, we freeze the weights of the convolutional layers of the pre-trained network and fine-tune only the fully connected part using a dataset artificially generated from some query images with data augmentation. Thus, the final layer (with 1000 outputs) of the original AlexNet is replaced by a new final layer (with the number of classes considered in this problem, for example, 35 outputs of Tobacco800 dataset) Finally, the feature map generated by this model is used to represent both the query and candidate images.

The objective of this layer is to learn a nonlinear function to encode each point x in its compact k codes, such as a sigmoid function $\alpha(x) = \frac{1}{1+e^{-x}}$. Then, we define the loss layer to use during the training. Thus, a *Softmax loss layer* is trained, which is frequently used in multiclass problems, and the given predictions for the classes are transformed with a loss function $loss = -w_g log(\alpha_g)$, where g is the ground truth label and w_g is the weight for the g-th category. The number of parameters related to high-level features can change according to the network configuration, as follows:

- 128, 256 or 512 features of the AlexNet model with an additional layer ($fc8_{new}$ layer) (these values are based on Lin et al. (2015a) that worked with binary codes).

Therefore, we have the description of each image by a feature extractor based on CNN model to perform an exhaustive search. We can evaluate the differences in terms of performance and time consuming of the approach based on the standard Alexnet model, and that using the modified architecture with the additional layer.

3.2.3 Computational Resources

The following resources were used to develop the proposed method: The Caffe tool proposed by Jia et al. (2014); Digits NVIDIA; OpenCV Library; Postgres database; Python as a programming language; Numpy Library and Ubuntu OS 14.04 LTS, dlib Library as an algorithm for searching for candidate objects.

The computational resources used in this first approach are available at École de Technologie Supérieure (ETS), Montreal, Canada, and these consist of 2 Intel(R) Xeon(R) CPU E5-2620 v3 2.40GHz with 12 cores each; 64 GB of RAM; 2 GPU NDVIA Tesla K40 model with 11GB of memory and 2880 CUDA cores each; 1 HD 4TB WD RE Enterprise WD4000FYYZ.

3.2.4 Parallel computing

The use of parallel computing is essential in IR and PS tasks, thus the time to retrieve a single query, considering the whole set of candidates to search can be reduced. Let us recall the different layers that can be identified that offer potential for parallelization, according to Terboven et al. (2006):

- Queries are mutually independent. The queries can be processed in parallel, that is, several queries are run in batch mode.
- The scores calculated for each query and candidates from the document can be calculated in parallel because the dataset images are independent of each other.
- The distances between the feature vector from query and candidates can be calculated in parallel.

Figure 3.7 shows the parallel method implemented in this study for the online phase, considering that in the off-line phase all the features of candidates and queries were extracted.

This approach begins with the main process, where a batch of 4096 candidates is defined and get from the dataset. Each candidate is added in the queue 1 and 7 worker processes will be created to use this queue 1. Each worker process starts getting the features of all the query. Then, the process gets the candidate features of the queue 1 and the distance measure will be calculated for the queries and that candidate. Finally, the worker process writes at the disk the distances previously calculate.

Problems with synchronization were detected in our parallel processes. Some items of a worker process finished before a new queue 1 be created, overwriting the data already calculate. To avoid problems with queue synchronization, in other words, waiting a long time to finish previous processes, or ending up overwriting the data, semaphores were used. Semaphores have two operations, down and up (generalizations of sleep and wake up, respectively). The down operation in

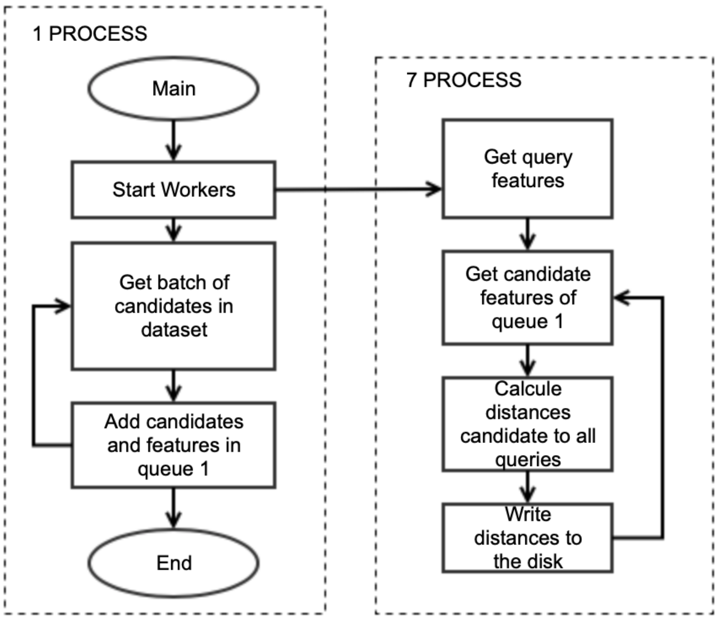

1 PROCESS
Main
Start Workers
Get batch of candidates in dataset
Add candidates and features in queue 1
End
7 PROCESS
Get query features
Get candidate features of queue 1
Calcule distances candidate to all queries
Write distances to the disk

Figure 3.7: First parallel approach to retrieve the candidates using CNN features

a semaphore verifies if the value is greater than 0. If so, it reduces the value (uses a stored activation) and continues. If the value is 0, the process is put to sleep without finishing the down at the moment. Checking the value, altering it and possibly sleeping, all of this is done as a unique atomic and indivisible action. It is guaranteed that, once a semaphore operation is initiated, no other process can access the semaphore until the operation is concluded or blocked. This approach is essential for solving synchronization problems and avoiding race conditions. The up operation increases the value of the addressed semaphore. If one or more processes were sleeping in that semaphore, incapable of concluding a previous descending operation, one of them is chosen by the system (for example, randomly) and it allowed to conclude the operation. In this way, after a signal in a semaphore with a sleeping process in it, the semaphore will continue to be 0, but there will be one fewer process sleeping in it. The operation of increasing the semaphore and waking up a process is also indivisible. No process blocks its execution, just as no process blocks the activation in the previous model (TANENBAUM, 2014).

3.3 Second approach - Siamese Convolutional Neural Network (SCNN)

This approach uses the SCNN implemented with Alexnet architecture to learn the image representation and the distance during the training stage. We decide to avoid training data augmented and to improve distance calculation. Figure 3.8 shows an overview of the proposed method that works with IR and PS in a collection of document images using a second approach to be investigated in the present study. This proposed approach has also two phases: online and offline. The offline phase begins with the Selective Search algorithm to find possible locations of objects and making them candidates for the image retrieval process. Also in the offline phase, the SCNN model is trained on the Imagenet dataset, that was prepared with a pair of images (similar and not similar pairs). The objective is to pull output feature vectors closer for input pairs that are labeled as similar and push them away if the input pair is not similar. Using the concept of transfer learning the SCNN is used as a feature extractor for our datasets. For the search process carried out during the on-line phase, given an image query, an exhaustive search is performed considering the whole list of generated candidates. Here, there are two possibilities to compare queries and candidates: (1) The feature map of each candidate is stored to be further used in the on-line phase or (2) distance calculation between queries and candidates can be done directly using the deep metric. Each step of the proposed method is detailed as follows.

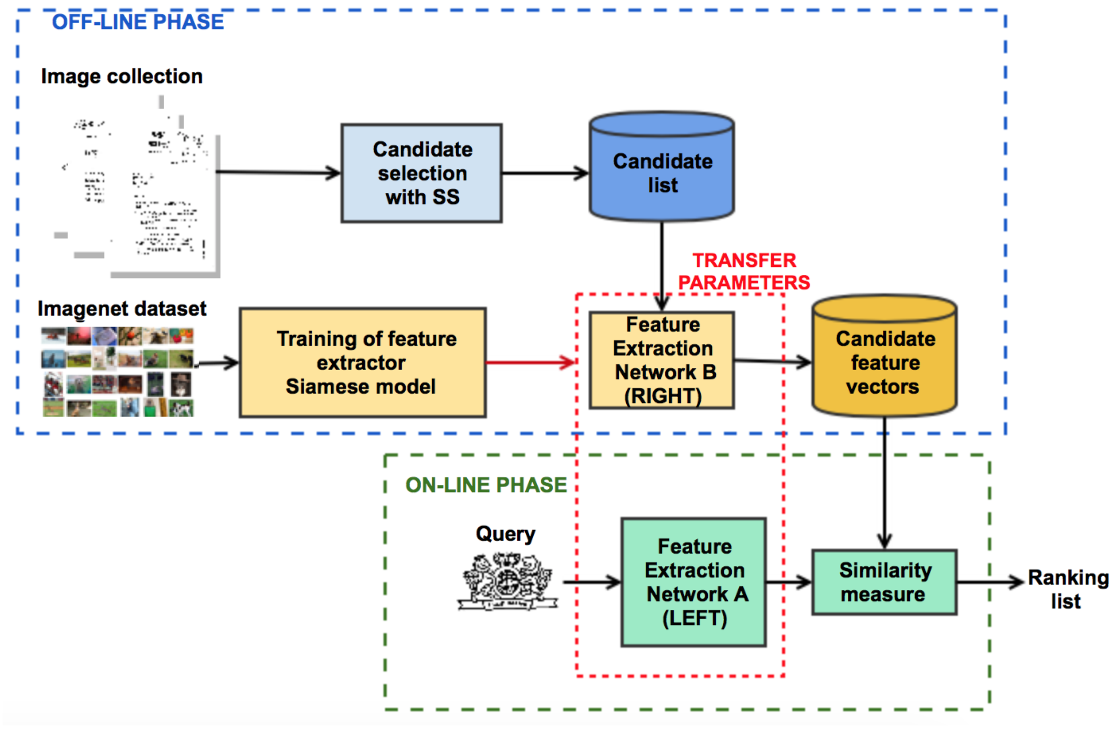

OFF-LINE PHASE
Image collection
Candidate selection with SS
Candidate list
TRANSFER PARAMETERS
Imagenet dataset
Training of feature extractor Siamese model
Feature Extraction Network B (RIGHT)
Candidate feature vectors
ON-LINE PHASE
Query
Feature Extraction Network A (LEFT)
Similarity measure
Ranking list

Figure 3.8: Overview of the proposed SCNN approach

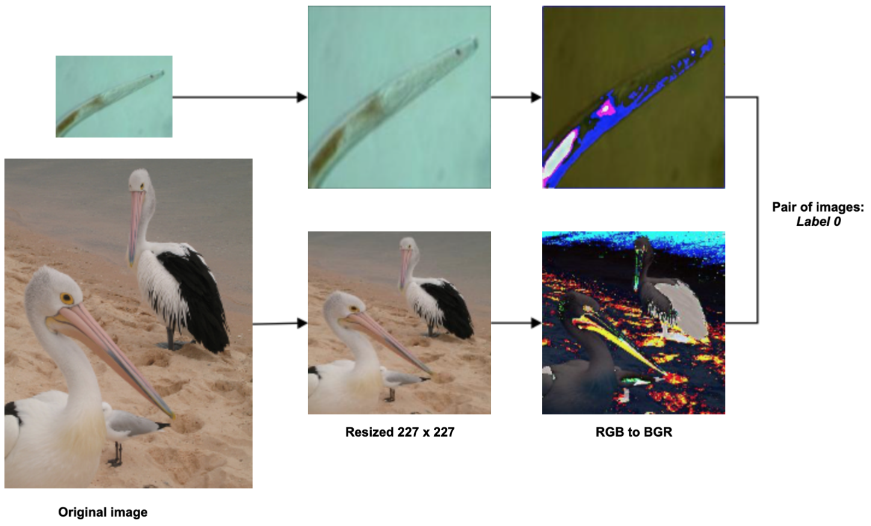
Source: The author

However, the main difference with the first approach (training a single CNN) is the training of the SCNN without data augmentation, offering only transfer of parameters to the online stage, as well as a network capable of learning the distance metric.

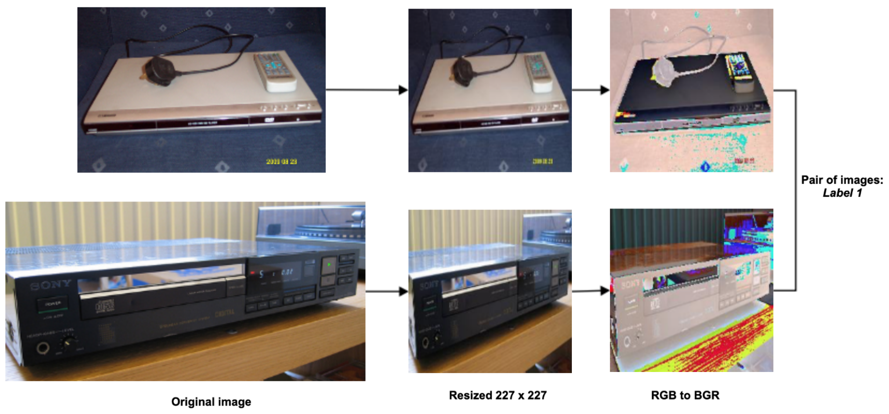
3.3.1 Pre-processing

The first stage of this approach is the preparation of the ImageNet data to carry out the training of the images. Firstly, the images are resized to 227 x 227 because the architecture Alexnet uses this size. To form this training set, all the images need to have 3 channels of RGB color. For input into the network, the images are transformed to BGR, since the tool being used requires this procedure to interpret the input. Finally, the pair of images are labeled as positive or negative, in the following way:

- Negatives (label 0): if the pairs of images do not belong to the same class and thus are not similar (see Figure 3.9a).
- Positives (label 1): if the pairs of images belong to the same class and therefore are similar (see Figure 3.9b);



(a) Pair of images Label 0



(b) Pair of images Label 1

Figure 3.9: Example of data preparation for the training set using Imagenet dataset

The images of Imagenet used to train the network were 100,000 similar pairs and 150,000 not similar pairs. We generated 1.5x more not similar images for training and test as recommended by Melekhov, Kannala and Rahtu (2016). We split the images into training (70%) and test (30%) sets. The configurations of the network are shown in the next section.

3.3.2 Image representation

The image representation by deep features extracted from the SCNN will be used. The SCNN of our approach was inspired by Melekhov, Kannala and Rahtu (2016) approach and Chung and Weng (2017), we used the network illustrated in Fig. 3.10 to learn deep representation from data for distinguishing similar and not similar image pairs. Based on our first approach presented in Section 3.3, the network is also based on the AlexNet architecture and we have added a layer to reduce the feature map.

The SCNN is composed of two identical Alexnet architectures trained on the Imagenet dataset (see Fig. 3.10). The last fully-connected layer with the number of categories was removed and the fully connected layer with n dimensions was trained as a feature representation. Thus, Euclidean distance learning is made with layers available by Caffe tool (JIA et al., 2014): *Eltwise* (an operation to sum between two blobs), *Power* (operation for power or square root) and *Reduction* (reduce input blob using sum). After that, we implement a fully connected layer with 1 output and loss layer. We set the learning rate in 10^{-3} with an exponential function. The activation function used is Stochastic Gradient Descent (SGD), as recommended by Recht et al (RECHT et al., 2011).

During the training, the input to the network is a pair of images X and Y, where X and Y are Imagenet positive and negative pairs from the training set, respectively. The inputs X and Y are fed into the two Alexnet networks A and B, both with an additional layer with n dimensions. The output distance of networks A and B are fed to a *sigmoid Cross-Entropy Loss* layer, that aims to minimize the difference of probability distribution between the predicted labels and ground

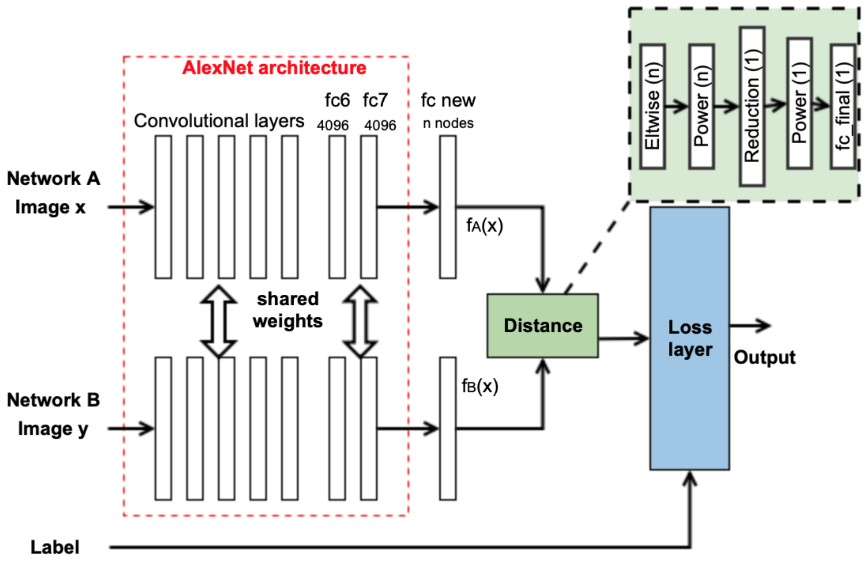


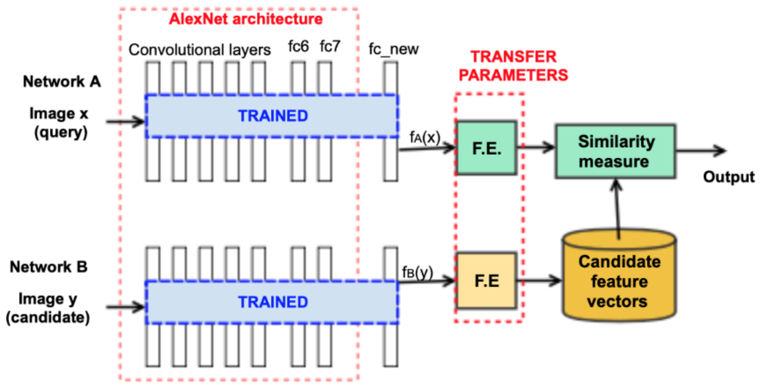
Figure 3.10: Overview of our Siamese network - Training
Source: The author

truth labels (LIU; QI, 2017). For each sample we have:

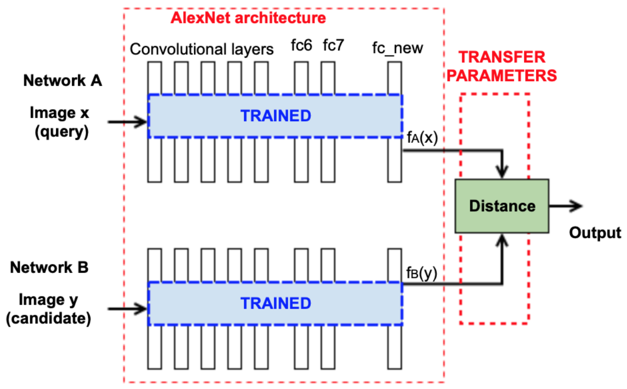
$$L_i = \sum_{k=1}^{C} t_k(y_i) \log P(y_i = k|b_i; w_k) \tag{3.1}$$

where $t_k(y_i)$ is the distribution of ground truth labels y_i; $P(y_i = k|b_i; w_k)$ the probability distribution of predicted labels. The sigmoid function is used for the two-class logistic regression, that is, when using a network, we try to get 0 and 1.

During testing, we use the concept of transfer learning, where the weights from previously trained network are used as a feature extractor to retrieve document images. The input of the network is a pair of images from a dataset. First, each image was converted from RGB to BGR and then merging an image x (network A) and image y (network B). There are two possibilities to calculate the similarity. (1) During the testing v1 (see Fig. 3.11a), the network is used as a feature extractor. The feature maps extracted by the networks are used to represent the query (network A) and candidate images (network B). Each image passes only one time to extract and index all the feature vectors of the dataset and then



(a) Testing v1



(b) Testing v2

Figure 3.11: Overview of our Siamese network - Testing

a similarity measure is calculated. (2) During testing v2 (see Fig. 3.11b), instead of storing the feature vectors, the distance calculation is done in real-time and the SCNN returns the similarity between a query and a candidate. This method uses the layers implemented to calculate Euclidean distance detailed previously. However, there is a combination of images presented several times as input to the network, for example, query q_1 will be compared to candidates $c_1, c_2, c_3, ...c_n$ and, it requires a lot of computational resources.

3.3.3 Computational Resources

The following resources were used to develop the proposed method: The Caffe tool proposed by Jia et al. (2014); Digits NVIDIA; OpenCV Library; Postgres database; Python as a programming language; Numpy Library and Ubuntu OS 14.04 LTS, dlib Library as an algorithm for searching for candidate objects.

The computational resources used in this second approach consist of: (1) 1 Notebook Acer Predator Helios 300, Core i7-7700HQ@ 2.80Ghz processor and 32GB of RAM DDR4; 1 GTX 1060 with 6GB; 1 SSD NVME 512GB and 1 HD SATA 2TB. (2) Desktop with motherboard MSI X370 KRAIT Gaming MS-7A33, AMD RYZEN 5 1600@ 3.2Ghz processor and 32GB of RAM DDR4; 1 GTX 1080 with 8GB; 1 SSD 256GB and 1 HD SATA 4TB.

3.3.4 Parallel computing

In this approach, parallel computing was also implemented. Figures 3.12 and 3.13 show the methods implemented in this approach for the online phase.

The approach of Figure 3.12 begins with the main process, where it gets the features of queries. Each query feature is added in the queue 1 and 10 worker processes will be created to use this queue 1. Each worker process starts getting the features of all the query. Then, the process gets the features of a candidate. We define some constraints based on context information before retrieval phase starts. The candidate c will only be considered for retrieval if the ratio $\frac{height_c}{width_c}$ has a maximum of 25% difference with the query q. For example, if a query has $\frac{height_q}{width_q} = 0.5$, the candidate similarity calculation will only be performed if the candidate has a $\frac{height_c}{width_c}$ between 0.375 and 0.625. Thus, the distance measure will be calculated for the query and the candidate selected. Finally, the worker process writes at the disk the distances previously calculate. In this approach synchronization with semaphores was applied as presented in Section 3.3.5.

The search task presented in Figure 3.13 begins loading the network model with the configurations. Then, a worker process (child process) was created to

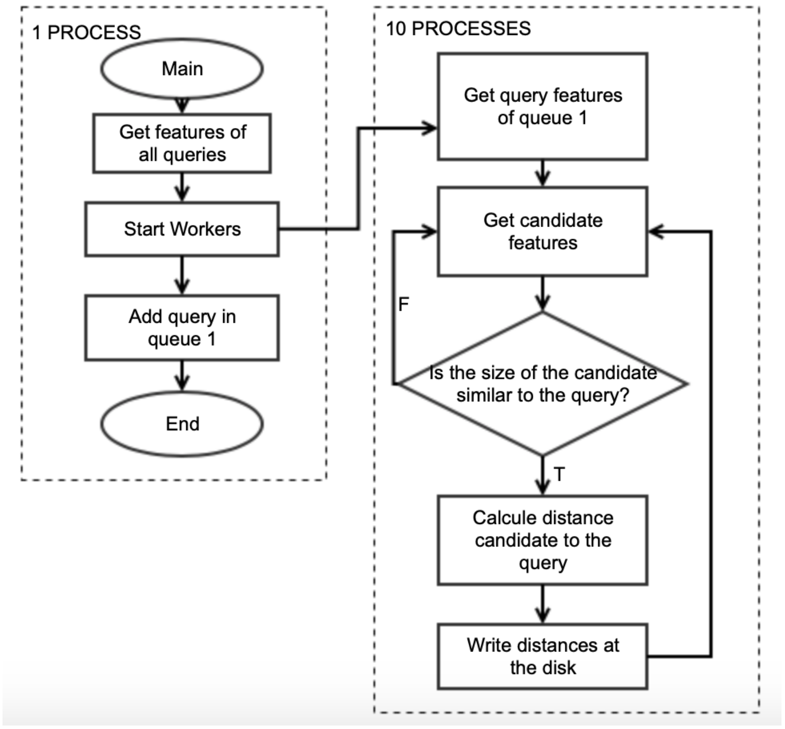

1 PROCESS
Main
Get features of all queries
Start Workers
Add query in queue 1
End
10 PROCESSES
Get query features of queue 1
Get candidate features
F
Is the size of the candidate similar to the query?
T
Calcule distance candidate to the query
Write distances at the disk

Figure 3.12: Parallel approach to retrieve candidates using features of SCNN (using testing V1)

select queries and candidates that will be used in the retrieval phase (distance calculation). This child process adds in a queue one query (image x) with all candidates from the dataset (image y), to merge the image x (network A) and image y (network B). Back to the main process, 6 workers processes will do the merging between query and candidates until the batch size number defined in the network model. After, these batches will be added in a queue of distance calculation using the layer implemented (retrieval). Back to the main process, this queues will be used to do the retrieval method using GPU and write the results at the disk. All this process depends on a good performance of CPU and GPU.

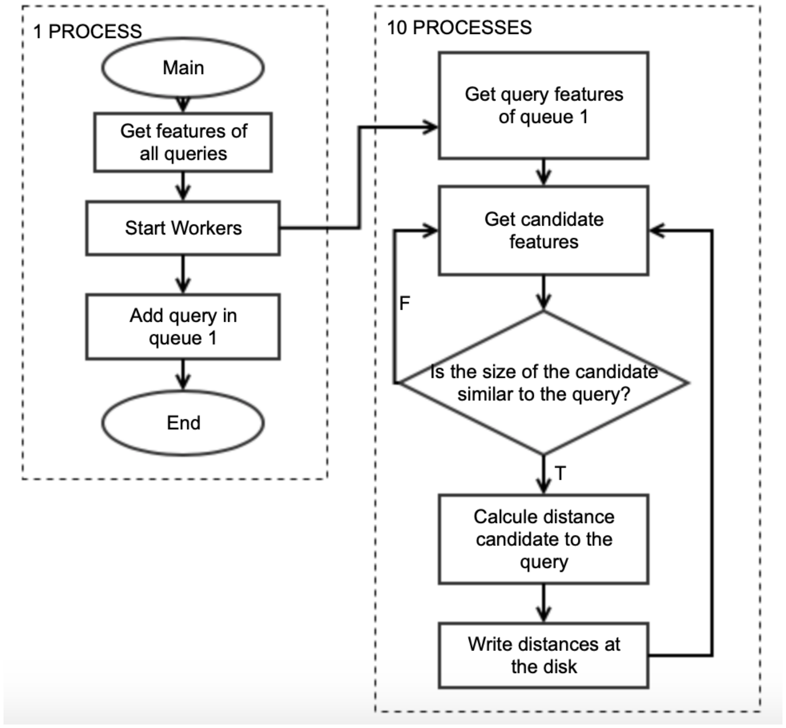

1 PROCESS
Main
Get features of all queries
Start Workers
Add query in queue 1
End
10 PROCESSES
Get query features of queue 1
Get candidate features
F
Is the size of the candidate similar to the query?
T
Calcule distance candidate to the query
Write distances at the disk

Figure 3.13: Parallel approach to retrieve candidates using features of SCNN (using testing V2)

3.4 Search Procedure - Performance Evaluation Protocol

For both proposed approaches, Single CNN and SCNN, it is necessary to compute the distances between the queries and image candidates. Thus, during an exhaustive search, all candidates generated by the document images are compared with all the queries, and a ranking is made with the most similar candidates. That is, a comparison is made in pairs between all the feature vectors extracted from the candidates and all the queries, where the smallest distance is ranked first. This comparison is carried out using some similarity measure. In this work, we can evaluate the results using euclidean and cosine distances, which were explained in Section 2.5.

After this, a threshold number must be defined to generate a ranked list.

The top k candidates were defined to evaluate the correct candidates, with the value of k based on the literature (RUSIÑOL et al., 2015)(JAIN; DOERMANN, 2012)(BHATTACHARJEE et al., 2016). This threshold number will be used in the IR and PS tasks to evaluate the performance of the approaches.

3.4.1 Image Retrieval Task

The objective of this task is to retrieve the maximum number of relevant images given an input query, independent of its position in the retrieved document. There are no previously defined restrictions or categories regarding the object to be detected, and there is also no training stage. The output is a list of document images retrieved, and the confidence level of each image to the query.

In CBIR approaches, good performance depends on the process of extracting features and the use of an adequate similarity measure (WAN et al., 2014). Thus, a vector is extracted from the images according to the feature extractor based on deep learning. Given a new query, this vector is extracted and compared with the feature vectors of the candidates through a distance measure.

Figure 3.14 shows how the results should appear in this image retrieval task, in which the shortest distance will be ranked first.

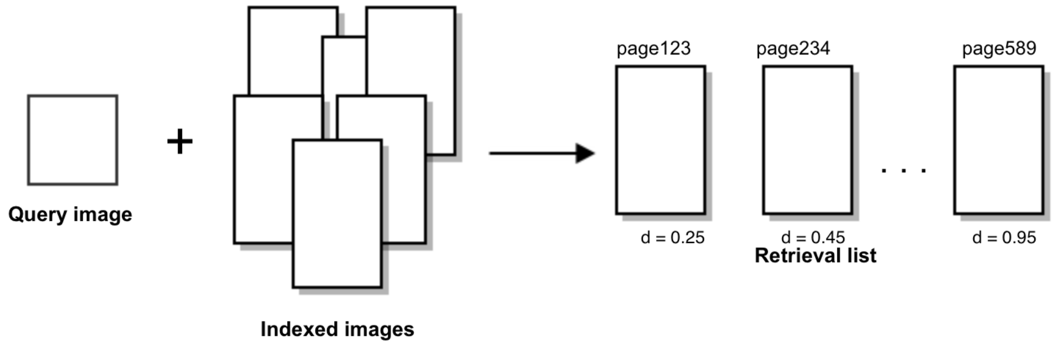


Figure 3.14: Results presented in images retrieval task. The output is a list of images ordered by confidence level

Source: The author

Evaluation metric. The performance of the present retrieval system is evaluated using Precision (Pr) and $Recall$ for each query (POWERS, 2011). Then, the mean AP (mAP) is calculated. The AP is the area above the precision and recall curve for each query, according to the equations 3.2 and 3.3, respectively.

$$precision = \frac{relevant\ images \cap retrieved\ images}{retrieved\ images} \tag{3.2}$$

$$recall = \frac{relevant\ images \cap retrieved\ images}{relevant\ images} \tag{3.3}$$

where the precision is the ratio of the number of positive images retrieved to the total number of images retrieved. The recall is the ratio of the number of positive images retrieved to the total number of positive images in the corpus. Thus, the precision of each candidate retrieved for each query is evaluated by a label identifying the page to which the candidate belongs. All the candidates are organized by an ascending rank for each query.

3.4.2 Pattern Localization Task

Unlike CBIR systems, the pattern location task considers the overlap between the query and the image retrieved from the document. Thus, the system must find a precise match between the candidate and the query. It will return a list of images (without repetition) and the precise location of each candidate. Figure 3.15 shows an example of retrieved images given a query, as well as the location of the object (red square).

The overlap is computed to recover the location. It considers the position of the query (x, y), and its area q_1. The positions of the candidate (x_1, y_1) are also considered, as well as its area o_1, as described in Equation 3.4(NOWOZIN, 2014).

$$U_{iou}(x, y) = \frac{q_1 \cap o_1}{q_1 \cup o_1} \tag{3.4}$$

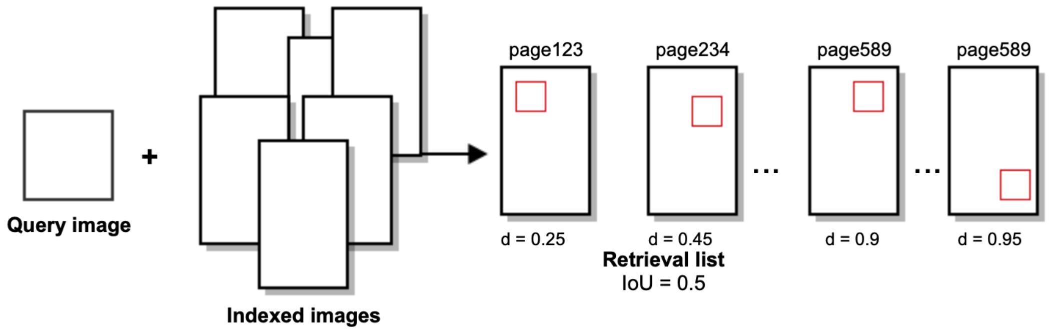


Figure 3.15: Example of the results presented in the pattern spotting task. The output is a list of images, ordered by confidence level and their precise location

Source: The author

Evaluation metric. The same evaluation approach used in the IR task is applied. However, an image will only be considered relevant if it overlaps enough to identify the query. The criterion used to establish this is Intersection over Union (IoU). Figure 3.16 shows an example of this overlap calculation.

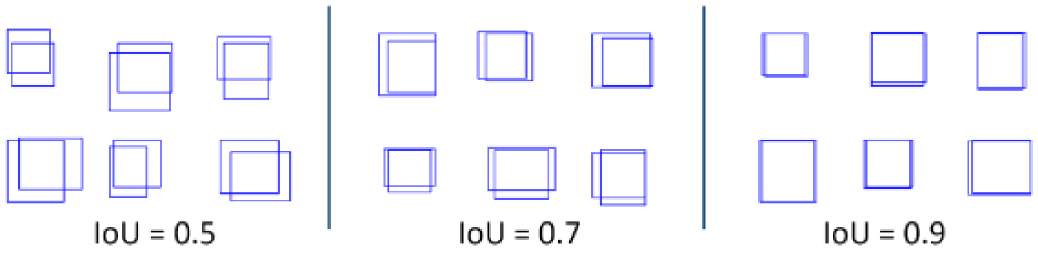


Figure 3.16: Example of IoU = 0.5, 0.7 and 0.9 in objects

Fonte: Zitnick and Dollár (2014)

$IoU = 0.7$ is considered to be a reasonably good result, since $IoU = 0.5$ is considered too small and $IoU = 0.9$, very restrictive (ZITNICK; DOLLÁR, 2014). However, the present study considers the analyses: $0.1 \leq IoU \leq 0.7$ to determine that a positive candidate is retrieved, and in the end, the precision and recall are calculated. Finally, the mAP is calculated to evaluate the results considering all the queries.

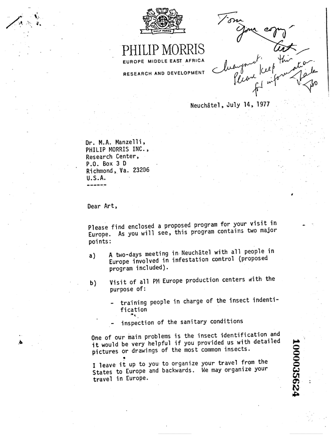
Thus, the performance of feature extractor and the similarity measures are adequately evaluated for the tasks of IR and PS. The results are compared with those found in the literature. Evaluation of the computational time for recovering a query given the candidates and the space necessary to maintain all these results is also being considered.

3.5 Datasets

To evaluate the proposed method, two databases were selected to form the experimental protocol, namely the databases Tobacco800 and DocExplore, which will be described in this section.

3.5.1 Tobacco800

Tobacoo800 is a public data set from the Complex Document Image Processing (*CDIP*) collection developed by Illinois Institute of Technology, containing 42 million pages of documents (in 7 million multi-page TIFF images) released by tobacco companies through the Master Settlement Agreement. The collection is made up of 1290 document images. It is a set of real data for research on document images. These documents were collected and scanned using a wide variety of equipment over time (LEWIS et al., 2006) (AGAM et al., 2006). Figure 3.17 shows some examples of images from this data set.



PHILIP MORRIS
EUROPE MIDDLE EAST AFRICA
RESEARCH AND DEVELOPMENT

[illegible]

Neuchâtel, July 14, 1977

Dr. M.A. Manzelli,
PHILIP MORRIS INC.,
Research Center,
P.O. Box 3 D
Richmond, Va. 23206
U.S.A.

Dear Art,

Please find enclosed a proposed program for your visit in Europe. As you will see, this program contains two major points:

a) A two-days meeting in Neuchâtel with all people in Europe involved in infestation control (proposed program included).

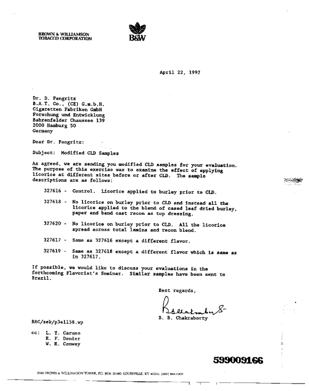
b) Visit of all PM Europe production centers with the purpose of:

- training people in charge of the insect indentification
- inspection of the sanitary conditions

One of our main problems is the insect identification and it would be very helpful if you provided us with detailed pictures or drawings of the most common insects.

I leave it up to you to organize your travel from the States to Europe and backwards. We may organize your travel in Europe.

1000035624



BROWN & WILLIAMSON
TOBACCO CORPORATION

B&W

April 22, 1992

Dr. D. Pangritz
B.A.T. Co., (CE) G.m.b.H.
Cigaretten Fabriken GmbH
Forschung und Entwicklung
Bahrenfelder Chaussee 139
2000 Hamburg 50
Germany

Dear Dr. Pangritz:

Subject: Modified CLD Samples

As agreed, we are sending you modified CLD samples for your evaluation. The purpose of this exercise was to examine the effect of applying licorice at different sites before or after CLD. The sample descriptions are as follows:

327616 -	Control. Licorice applied to burley prior to CLD.
327618 -	No licorice on burley prior to CLD and instead all the licorice applied to the blend of cased leaf dried burley, paper and band cast recon as top dressing.
327620 -	No licorice on burley prior to CLD. All the licorice spread across total lamina and recon blend.
327617 -	Same as 327616 except a different flavor.
327619 -	Same as 327618 except a different flavor which is same as in 327617.

If possible, we would like to discuss your evaluations in the forthcoming Flavorist's Seminar. Similar samples have been sent to Brazil.

Best regards,

[illegible]

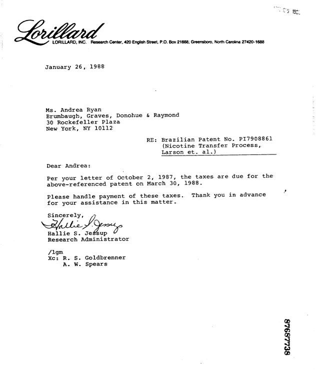
B. B. Chakraborty

BBC/sek/p3e1158.wp

cc: L. T. Caruso
R. F. Denier
W. R. Conway

599009166

1500 BROWN & WILLIAMSON TOWER, P.O. BOX 35090, LOUISVILLE, KY 40232, (502) 568-7000



Lorillard
LORILLARD, INC. Research Center, 420 English Street, P.O. Box 21688, Greensboro, North Carolina 27420-1688

January 26, 1988

Ms. Andrea Ryan
Brumbaugh, Graves, Donohue & Raymond
30 Rockefeller Plaza
New York, NY 10112

RE: Brazilian Patent No. PI7908861
(Nicotine Transfer Process,
Larson et. al.)

Dear Andrea:

Per your letter of October 2, 1987, the taxes are due for the above-referenced patent on March 30, 1988.

Please handle payment of these taxes. Thank you in advance for your assistance in this matter.

Sincerely,

Hallie S. Jessup

Hallie S. Jessup
Research Administrator

/lgm
Xc: R. S. Goldbrenner
A. W. Spears

87687738

Figure 3.17: Examples of documents from the Tobacco800 dataset

The ground truth of the Tobacco800 data set was created by the Language and Media Processing Laboratory at the University of Maryland. The material created includes the ground truth information in logos and signatures, considering the location and dimensions of each visual entity (ZHU et al., 2007)(ZHU; DOER-

MANN, 2007). The data set consists of 412 document images containing logos and another 878 without logos.

This study will use only logos for IR and PS tasks. There are 35 different categories of logos in the galleries, and for each logo, the number of occurrences can vary from 1 to 68. We used 21 categories that have from 2 to 68 occurrences by category totalizing 418 queries. Each occurrence in each category is used as a query, and the remaining occurrences in the same category are kept as good retrieval results. Figure 3.18 shows the classes of the logo in the Tobacco800 dataset used in this study.

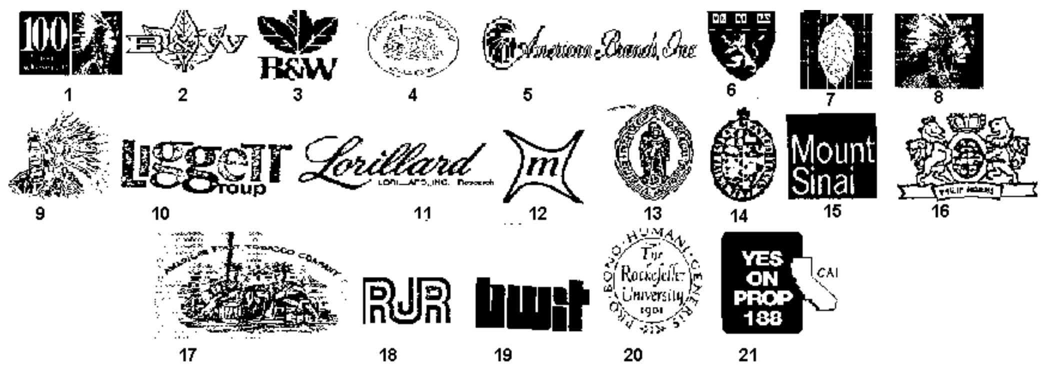


Figure 3.18: The 21 categories of the Tobacco800 dataset used in this work

3.5.2 DocExplore

This database is from the Project DocExplore[1], published in En et al. (2016a). All the manuscripts date from the 10^th through the 16^th century. Each page was scanned using the same high-resolution configuration, and the images were stored with low resolution with 90% compression quality. Examples of manuscripts available in this database can be seen in Figure 3.19. The query images and their corresponding number of occurrences are presented in Figure 3.20.

The 1500 images available in the data set are organized into 35 categories and 1447 objects. The number of occurrences of each object in the collection can

[1]http://spotting.univ-rouen.fr

vary from 2 to more than 100. The objects differ in color, shape, size, distortion and possible degradation of the manuscript.

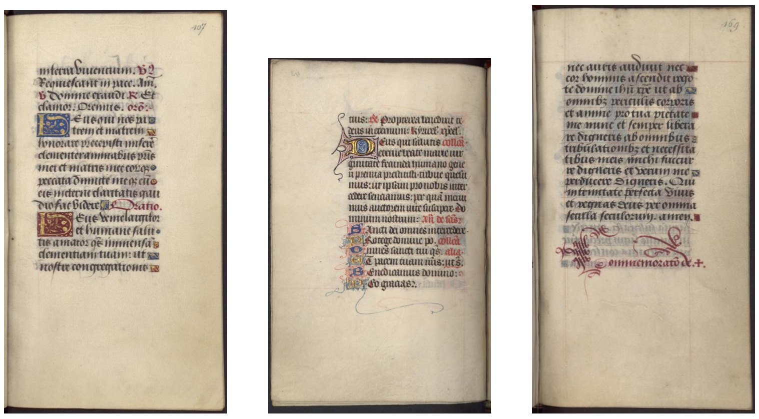


Figure 3.19: Examples of documents from DocExplore dataset

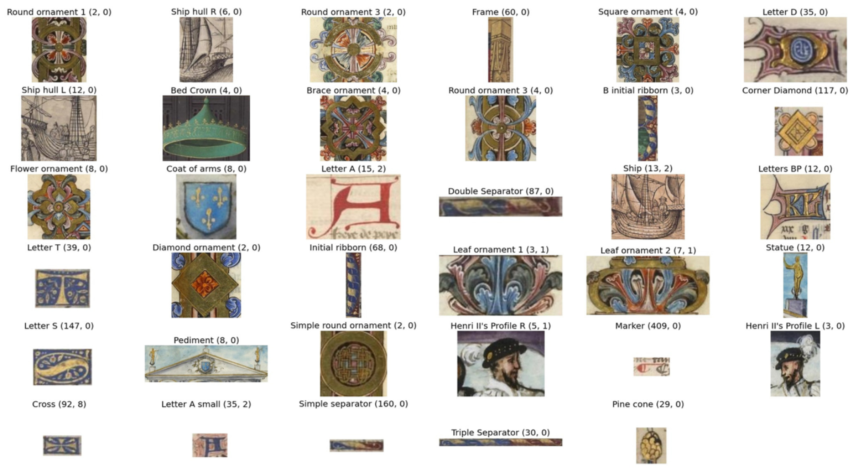


Figure 3.20: The 35 categories of queries from the DocExplore dataset. The images and corresponding number of occurrences

Source: En et al. (2016a)

3.6 Final Considerations

In this chapter, we described the steps for the two proposed approaches, single CNN and SCNN, such as candidate generation, feature extraction, indexing and similarity measure. Thus, we presented the approaches based on deep learning that can be used as a feature extractor to capture different levels of feature abstraction, while the number of feature map was reduced to try to improve the computational cost. We have two public datasets to evaluate our method considering the IR and PS tasks. Thus, in the next chapter (Chapter 4) we present the results obtained in our experiments.

Chapter 4

Experimental Results

This chapter describes the experimental results used to evaluate the proposed approaches for IR and PS tasks in two different datasets. Finally, the results were compared with the state-of-the-art. The final of this chapter, we discuss possible improvements to the method presented.

4.1 Tobacco800 dataset

This section presents the experimental results for the Tobacco800 data set. The pre-processing, image representation, IR and PS tasks are presented. Analysis of the results is based on the experimental protocol described in Section 3.5. The end of this section shows comparisons of the results with the state of the art.

4.1.1 Pre-processing

As we presented in Section 3.1, based on the literature, the Selective Search algorithm proposed by Uijlings et al. (2013) shows good parameters for finding quality candidates. It combines the use of exhaustive search with segmentation methods. In this way, it tries to find all possible object locations, making several partitions in the images to work with all the image conditions.

Let us recall that the experiments were carried out using two versions of SS:

SSv1 and SSv2, where the parameters were not modified in the SSv1 implementation. During a search stage at ÉTS, SSv1 was used to the first approach based on single CNN. In the next implementations, we used SSv2, due to the improvement of objects quality presented as follows.

Considering the variations of the parameters implemented in the SSv2 tool, Table 4.1 shows the number of candidates generated in the Tobacco800 dataset. Thus, for SSv2, we set the adaptive threshold with $block = 241$ and$offset 0.12$, and the parameters $k = 50$ and 100, $color + texture + fill + size$, feature space RGB and normalized RGB.

Table 4.1: Selective Search applied at the Tobacco800 dataset

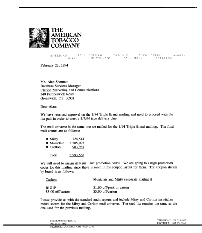
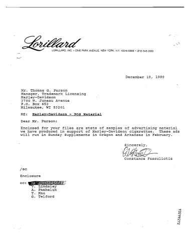
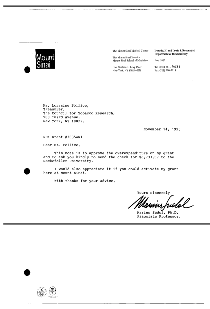
Num of candidates	**SSv1**	**SSv2**
Totals	1816852	1278174
$IOU >= 10$	826	14286
$IOU >= 30$	516	4064
$IOU >= 50$	311	1693
$IOU >= 70$	238	1202
$IOU >= 90$	44	573

In Table 4.1 it is possible to observe the difference between the number of candidates using the SSv1 implementation of Uijlings et al. (2013) and SSv2 with the use of the customized parameters, as well as the implementation of the adaptative threshold. It was observed that the use of the threshold reduced the image noise resulting from document scanning. The quality of the objects in SSv2 was increased in comparison to the SSv1, where 573 candidates have high quality (overlap above 90%) of SSv2 versus 44 candidates of SSv1.

Evaluating other algorithms, only for comparison, Sliding Windows generated $468, 262.125$ with only 12 candidates with IOU at or above 90 % overlap with the queries. In OpenCV, the algorithm *opencv_search_single* was used, generating $1, 869.923$ candidates, with 507 IOU above 90 %.

Therefore, to verify the quality of the candidates generated and compare the algorithms, a method was used to save the information from some randomly selected queries and their position on the page (width and height). So, for each query on the respective page, it can be seen how many candidates with IOU between 10 and 90 the method was able to select. Table 4.2 shows a comparison of the algorithms.

Table 4.2: Number of candidates generated in some pages of Tobacco800

Page	**Query**	**Search**	**IOU**					**Total**
			10	**30**	**50**	**70**	**90**	**regions**
		SSv1	0	1	0	0	0	6,133
		SSv2	28	7	2	2	**3**	552
		Opencv	8	2	0	**3**	0	328
		SW	640	141	25	0	0	75,480
		SSv1	1	0	0	0	0	12,854
		SSv2	13	1	3	0	**3**	320
		Opencv	2	0	0	**1**	0	708
		SW	1680	134	0	0	0	339,433
		SSv1	0	0	0	1	0	7,024
		SSv2	10	1	6	**3**	0	334
		Opencv	1	1	4	2	0	419
		SW	869	255	39	0	0	168,522

We consider an algorithm with quality when the candidates overlap to the query with more than 70%. For example, using the implementation SSv2, in the first page, for the 552 regions found, 3 of them have an overlap above 90% as compared to the query on that page. Compared with the SSv1 or OpenCV implementation and the same page, no region of this quality will be retrieved.

4.1.2 First approach results: single CNN

The selective search algorithm (SSV1) was applied at the pre-processing stage in all the 1,290 documents of the dataset and it generated 1.8M candidate images. All these candidate images are used in the retrieval process. It is worth mentioning that the images used to fine-tune the CNN were generated from the use of data augmentation techniques on the set of all the 432 image queries from Tobacco800 dataset, resulting in 17,240 training images, but the data augmentation does not balance the data. The dataset was split into a training set of 12,960 images (75%) and a validation set of 4,320 images (25%) to fine-tune the CNN. So, only transformed images were used to train the neural network.

Different models based on the Alexnet architecture were evaluated as feature extractors, namely:

- *Single CNN* (SM_{4096}) which is the standard trained model of Alexnet, having a map size of 4096 features, without fine-tuning.
- *Single CNN Fine-tuned model* (SM_{4096} + FT) in which the model of Alexnet that was pre-trained with the Imagenet dataset was used, and it was fine-tuned in the layer with also 4096 features.
- *Single CNN Fine-tuned model with additional fully connected layer* (CM_n) which reduces the size of the feature map and, the model of Alexnet that was pre-trained with the Imagenet dataset was used. In this case, 3 different maps are evaluated, with n = 512, 256 or 128 features (values randomly chosen and evaluated). The values for n were based on Lin et al. (2015a), where binary codes are used as network output (feature map).

The results were evaluated using the exhaustive search, in which a comparison was made between all the features extracted from the candidates and the query image, and then the lowest distance was ranked first. To analyze the results, a threshold k was defined to rank the results, as suggested by Rusinol and Lladós (2010) and Bhattacharjee et al. (2016).

The evaluation metrics were described in detail in Chapter 3. In Pattern Spotting task, which includes pattern localization of query, the values of mAP consider the overlap with IOU for each candidate retrieved. The next sections deal with the results for these two retrieval tasks, IR and PS, respectively.

4.1.2.1 Image Retrieval task

Table 4.3 shows the experimental results comparing the feature map obtained with and without fine-tuned (FT) AlexNet architecture, which has 4,096 elements at the output of the convolutional layer (*fc7*), with the modified model that has an additional layer to reduce the feature map to 512, 256 or 128 elements ($fc8_{new}$). The performance is significantly better using fine-tuning and the reduced feature map when compared to the original one with 4,096 elements. This can be explained by the fact that in a high dimensional feature space, the similarity measure is less discriminant than in a smaller feature space, therefore leading to a decrease of the precision when the dimensionality increases. Besides, the computational cost decreases in this exhaustive search.

Afterward, to analyze more precisely the performance of our system, we computed the mAP within a ranked list made of the top k candidates, for $k = \{10, 25, 50, 75, 100\}$ based on Rusinol and Lladós (2010).

Table 4.3: mAP for cosine distance using Top-k ranking for feature map of 4,096, 128, 256 or 512

Feature Map	top-**k**				
Dimension	**10**	**25**	**50**	**75**	**100**
$SM_{4,096}$ (WFT)	0.43	0.33	0.26	0.21	0.18
$SM_{4,096}$ (FT)	0.68	0.57	0.45	0.38	0.32
CM_{512}	0.53	0.40	0.31	0.27	0.23
CM_{256}	**0.72**	**0.61**	0.50	0.42	**0.35**
CM_{128}	0.69	0.61	**0.51**	**0.42**	0.34

WFT: Without fine-tuning and FT: fine-tuned

As one may see in Table 4.3 considering the top-10 ranking, the best results were achieved using the cosine distance with the modified CNN model. The

AlexNet fine-tuned with 4,096 features resulted in 0.68 of mAP, while the model without fine-tuning resulted in 0.43. Using the modified model with just 256 dimensions, a mAP of 0.72 was achieved. These results show that the use of an additional fully connected layer to reduce the number of features does not decrease the performance when compared to the original AlexNet. In addition, we observed that using a feature map with 512 entries, the results decrease when compared with the other models evaluated. In summary, we achieved an improvement of 4% in mAP, while the number of features was reduced by 16 times. The best recalls were 69% and 74% for the 128 and 256 based model, respectively. These recalls are similar to those that were observed when using the 4,096-dimensional model.

We also can see the results per category of logos using the AlexNet model with 256 features and the top-10 ranking in Figure 4.1. We observe that for some categories no correct images were retrieved, such as categories 5, 10 and 19. This can be explained by the fact that these categories contain too few samples that are moreover of bad quality. However, for seven categories (6, 8, 9, 12, 13, 16 and 20) we achieved an mAP greater than 0.8. The categories 8, 9, and 16, in particular, have many samples in the dataset.

Table 4.4 shows some qualitative results of the retrieved logos. These results are very promising since many correct logos were retrieved in the top-10 ranking. When the search involves a logo with few positive samples or few details, the proposed approach returns some false candidates, as seen in the fifth and sixth rows.

The computation time is another very important performance measure for retrieval systems. Table 4.5 compares the number of candidates processed per second by the proposed approaches (256 and 128 features) and the original feature map (4,096 features). As one may see the computational time decreased by approximately 55% and 47% when using the model with 128 and 256 features, respectively. The computational resources consist of 2 Intel (R) Xeon (R) CPUs E5-2620 v3 2.40GHz with 12 nuclei each and 64GB of RAM; 2 NVIDIA GPU Model Tesla K40 with 11 GB of memory and 2,880 CUDA, available during the

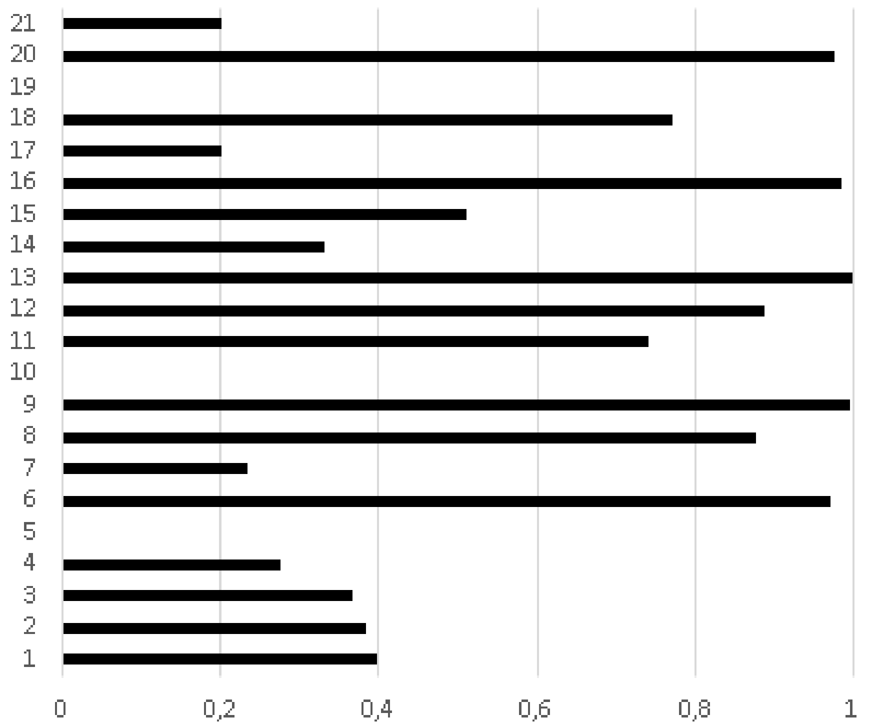
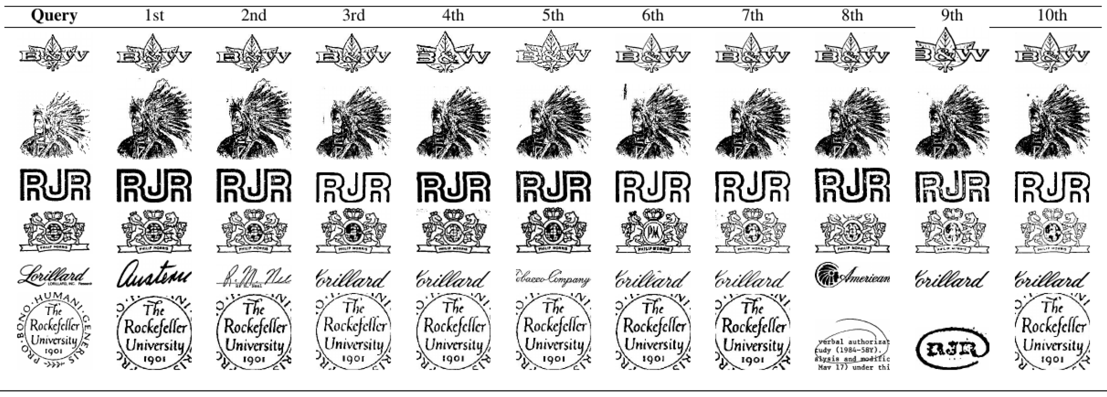


Figure 4.1: mAP per category in Tobacco800 dataset

Table 4.4: Qualitative retrieval results for some queries. Query logo and first ten retrieved logos by similarity and 256 feature map


Query
1st
2nd
3rd
4th
5th
6th
7th
8th
9th
10th

stage at ÉTS in Montreal.

Table 4.5: Computational time for the retrieval task (in number of candidates processed per second) using SSv1

Model	**Feature Extraction (num/sec)**	**Retrieval (num/sec)**
4,096	53.70	25.6
256	114.08	48.3
128	117.20	56.8

4.1.2.2 Pattern Spotting task

According to the PS, the method explained in Chapter 3, given a list of retrieved candidates, each candidate will be considered correct only if it satisfies the overlap value (IoU) established in the experiment. In this case, the results of the values with IoU of 0.1 to 0.7 were analyzed.

Table 4.6 shows a comparison of the performance for the PS task using the settings of the IR method, that is, comparing the feature extraction models, as well as the SSv1 algorithm for candidate generation. It is important to note that in these experiments only models with fine-tuning (FT) were used, according to the best performance obtained in the IR results.

Table 4.6: mAP for pattern spotting considering different values of IoU, top - 10 ranking, Single CNN, fine-tuned model using SSv1

Model	**IoU**			
	0.1	**0.3**	**0.5**	**0.7**
SM_{4096}	0.65	0.65	0.64	0.61
CM_{512}	0.50	0.50	0.50	0.49
CM_{256}	**0.69**	**0.69**	**0.60**	**0.56**
CM_{128}	0.67	0.67	0.60	0.59

Based on the results obtained, the following observations can be made:

- Location precision: There is a small difference in the location performance with the settings IoU = 0.1 and IoU = 0.7. This means that the proposed system was able to recover the candidate and precisely locate the position of good quality objects. For example, SM_4096 returned 0.65 for IoU = 0.1 and 0.61 for IoU = 0.7, then the difference is only 0.04.

- Influence of the new layer: The previous section showed that with the use of the feature map, the IR performance can be significantly improved. The location performance can be increased from 0.65 (IoU = 0.1) to 0.69 (IoU = 0.1). This significant improvement is also found using IoU = 0.3. However, considering IoU = 0.5, which is a fairly standard IoU value, the location also differ a little, we reached 0.60 for the model with 256 features and 0.64 for SM. This probably occurred due to the quality of the regions.

The next section we present the results for the second approach using SCNN, where we show the difference of results, comparing the methods used.

4.1.3 Second approach results: SCNN

This section shows the results obtained using the second approach from Chapter 3, that is, using the SCNN proposed. In the pre-processing stage, the improved Selective Search (SSv2) algorithm produces 1.2M of regions of interest considering the 1,290 documents in the database. We decided to use the SSv2 for the experiments. We have considered the **aspect ratio** of the query image to guide the SS algorithm as presented in Section 3.3.4. The candidate c will only be considered for retrieval if the aspect ratio $\frac{height_c}{width_c}$ has a maximum of 25% difference with the query q. For example, if a query has $\frac{height_q}{width_q}$ =0.5, the candidate similarity calculation will only be performed if the candidate has a $\frac{height_c}{width_c}$ between 0.375 and 0.625. For Tobacco800 dataset, by applying this query-based contextual information, the initial 1.2M of candidates were reduced to 873,876 candidates, improving the quality of the pre-processing stage (candidate generation) The improved version of SS increased in $13\times$ the quantity of objects that overlap more than 90% to the query, when compared to our previous approach.

The SCNN architecture trained with ImageNet dataset provides 4096-dim feature maps corresponding to its original fc_7 layer. We have also investigated the addition of a new layer (fc_{new}) to reduce the feature maps to 512, 256 and 128 dimensions. These values were used in our previous approach to reduce the dimensionality of the feature map of a CNN model. Besides, we can use the SCNN as a feature extractor to avoid spending time to process each candidate image more than once. Table 4.7 compares the time spent in the retrieval task for some queries when using the distance layer implemented inside the SCNN, and the SCNN used as a feature extractor. In the last, the feature map is extracted using the fully-connected layer, then an external Euclidean distance measure is computed. We observed that the computational time to retrieve a single query using the SCNN as a feature extractor is about 11 times lower.

Table 4.7: Comparison of computational time for the retrieval task in some queries-distance layer of SCNN and feature extractor approach. Example with 256 feature map. (time in seconds)

Query Image	**Num. of candidates**	**Siamese (V1) feature extractor**	**Siamese (V2) distance layer**
	210,148	52.08	571.43
	124,060	35.29	427.56
	167,571	54.02	492.16

The evaluation metrics were described in detail in Chapter 3. The next sections deal with the results for these two retrieval tasks, IR and PS, respectively. Then, we compared the final results with our previous approach and the current state-of-the-art.

4.1.3.1 Image Retrieval task

Table 4.8 shows the experimental results of the proposed image retrieval method, comparing the SCNN with a 4096-dim feature map with those with n

entries, where n=$\{512, 256, 128\}$. The similarity between the network feature maps was calculated using the Euclidean distance. Afterward, the Top-k candidates were chosen to generate a ranked list of relevant image candidates according to the mAP, where k=$\{5, 10, 25, 50, 100\}$. As one may see in Table 4.8, the best results were achieved by using the AlexNet with 4096 features resulting in 0.94 and 0.87 of mAP in the Top-5 and Top-10 ranking, respectively. The recall is higher than 90% in Top-5 and Top-10. With the additional layer, the best results are related to 256 features with a 0.74 of mAP for Top-5 ranking. The performance is significantly better using the 4096-dimensional feature map than any version of the additional layer. However, the computational cost to index all the vectors of the dataset is much higher.

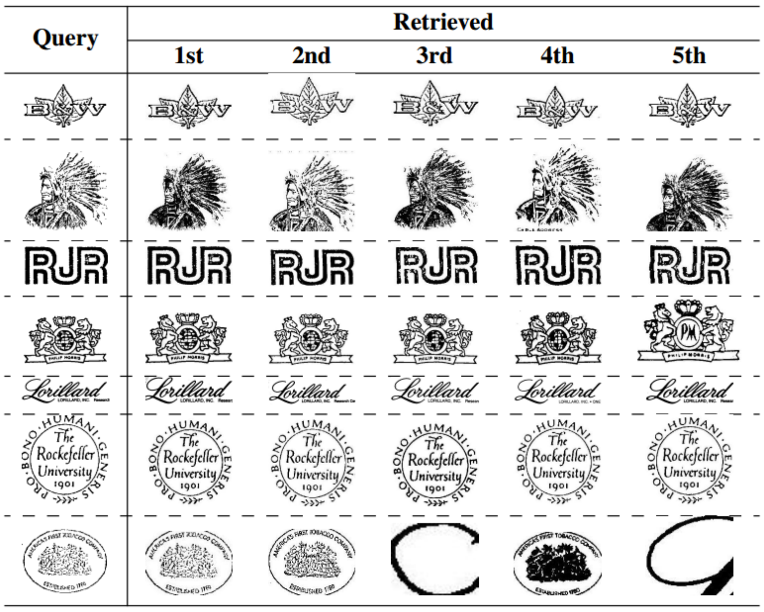
Table 4.8: mAP with Euclidean distance, Top-k ranking, and Siamese with feature map sizes $n = \{4096; 512; 256; 128\}$

Feature Map	**Top-*k***				
	5	**10**	**25**	**50**	**100**
4096	**0.944**	**0.872**	**0.735**	**0.546**	**0.306**
512	0.744	0.640	0.486	0.358	0.220
256	0.746	0.654	0.503	0.368	0.227
128	0.738	0.655	0.506	0.373	0.229

Table 4.9 shows some qualitative results of the logos retrieved using the SCNN with full feature map (4096 dimensional). These results are very promising since many correct logos were retrieved in the Top-5 ranking. We can observe the good performance especially in the fifth row, where the query is very similar to a signature, but we did not have false positives in Top-5 ranking. In the last row, we can see some false candidates, motivated by the presence of very few positive samples in the dataset for this query.

We can also see the results per category in Figure 4.2. We observed in some categories an mAP greater than 0.8. The categories 8, 9, and 16, in particular, have many samples in the dataset. The categories 5, 17, and 19 have very few samples that are moreover, of bad quality. In our previous approach, several queries had

Table 4.9: Qualitative retrieval results. Query and the first five retrieved logos using 4096-dimensional feature map


Query
Retrieved
1st
2nd
3rd
4th
5th

0.0 of mAP in some categories, while in this new approach we do not observe queries with an mAP lower than 0.2.

Therefore, we observed that the results related to the reduced maps are quite competitive. The computation time is another very important performance measure for retrieval systems, then, the computation cost of the proposed method with SCNN Alexnet was evaluated for the two phases (off-line and online). The computational resources consist of 1 AMD Ryzen 5 1600, 32GB of RAM, 1 GPU NVIDIA GTX 1080 with 8GB of memory and 2560 CUDA. Table 4.10 compares the number of candidates processed per second by the proposed approaches (512, 256 and 128 features) and the original feature map (4,096 features). As one may see the number of candidates processed per second increased by approximately 6.1%, 9.2% and 8.4% in feature extraction using the model with 512, 256 and 128 features, while for retrieval method it increased in almost 2, 3.5 and 5 times, respectively.

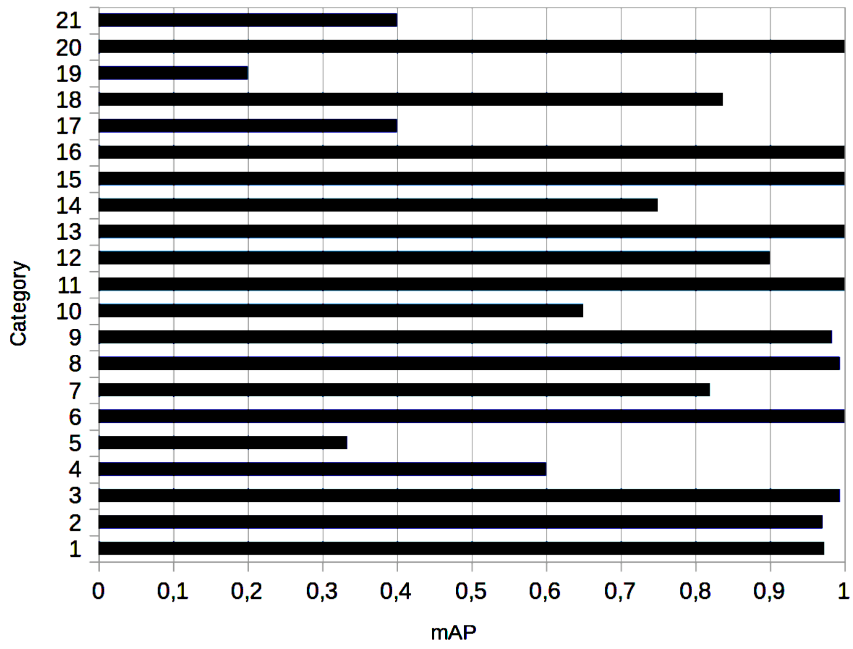


Figure 4.2: Results by category using SCNN 4,096 feature map and top 5 ranking on Tobacco800 dataset

Table 4.10: Computational time for the retrieval task (in number of candidates processed per second)

SCNN	**Feature Extraction** (num/sec)	**Retrieval** (num/sec)
4096	70.38	65.02
512	76.87	233.5
256	76.26	392.22
128	76.34	451.55

Comparison between the two proposed approaches

Table 4.11 shows the experimental results comparing the single CNN and the SCNN approaches. Comparing the results with the single CNN approach in Table 4.11, for the 4096-dimensional and 512-dimensional feature maps the proposed SCNN-based method is 28.23% and 20.75% better for Top-10, respectively. However, considering the 256-dimensional and 128-dimensional feature maps, the results were 9.20% and 5.07% worst, respectively. We observe the use of SCNN is more discriminant with high dimensional features, while the single CNN returns better performance with low dimensional features. Thus, the better choice depends on computational cost and the respective dataset to achieve promising results. However, the difference between the approaches for 128 feature map is only 0.03 of mAP in Top-10.

Table 4.11: Comparison with the two proposed approaches, retrieval task and Top-k ranking for 4096, 512, 256 or 128-dimensional feature map on Tobacco800 dataset.

Method	**Feature Map Dimension**	**Top-k**			
		10	**25**	**50**	**100**
Single CNN	4096	0.68	0.57	0.45	0.32
	512	0.53	0.40	0.31	0.23
	256	**0.72**	**0.61**	0.50	**0.35**
	128	0.69	0.61	**0.51**	0.34
SCNN	4096	**0.87**	**0.73**	**0.54**	**0.30**
	512	0.64	0.48	0.35	0.22
	256	0.65	0.50	0.36	0.22
	128	0.65	0.50	0.37	0.22

In addition, to evaluate the quality of candidates using the improved version of SS (SSv2), we did experiments with single CNN (4096 features) and improved SS of Tobacco800 dataset. Table 4.12 shows these results. We can observe top-10 ranking returned 0.75 of mAP, thus we increase 0.07 mAP (before was 0.68, according to Table 4.11) but still, it is 14% below than improved SS with SCNN (4096). Then the use of the new SS improved the results but we get better results using new SS with SCNN.

Besides, we observed there is not a significant time-consuming difference between the models single CNN and SCNN during retrieval phase since the online phase depends on the feature map already extracted to calculate the distances and perform the ranking list in both models. During the online phase, in both models, the query search time is improved from $\cong$ 7s (4096 features) to $\cong$ 3s (256 features).

Table 4.12: Comparison with the two proposed approaches for SSv2, Top-k ranking for 4096 or 256-dimensional feature map on Tobacco800 dataset

Approach	**Feature Map Dimension**	**Top-k**			
		10	**25**	**50**	**100**
Single CNN	4096	0.75	**0.59**	0.39	0.24
	256	0.86	0.73	**0.57**	0.33
SCNN	4096	**0.87**	**0.73**	0.54	**0.30**
	256	0.65	0.50	0.36	0.22

Then, the computational time decreased when using the model with 256 features, while the approaches return promising results.

4.1.3.2 Pattern Spotting task

Table 4.13 shows the experimental results of the pattern spotting task. Here, the candidate is taken as relevant if it overlaps enough with the image query. We can observe that our best results were achieved using the whole feature map (4096), but the results related to the reduced maps are quite competitive. Despite the reduction of about 0.12 in mAP, they have shown a significant reduction in terms of time-consuming as shown in Table 4.10. Still, in Table 4.13, we can see a small gap in terms of localization performance between IoU=0.1 and IoU=0.7. This means that our system succeeded not only to retrieve the relevant image candidates for each query but also in finding the query position precisely. Qualitative analysis can be seen in Table 4.14, in which we have some queries and the first candidate retrieved with its respective location.

4.1.4 Comparison with the state-of-the-art

Comparison with state-of-the-art - Image Retrieval

The results in IR were compared with the current state-of-the-art, and the results are shown in Table 4.15. In Jain and Doermann (2012), a scalable algo-

Table 4.13: mAP for pattern spotting with euclidean distance considering different values of IoU and top-5 on Tobacco800 dataset

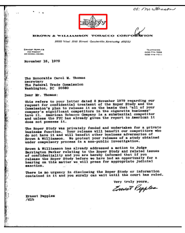
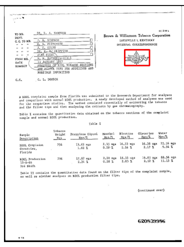
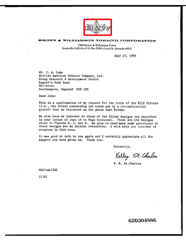
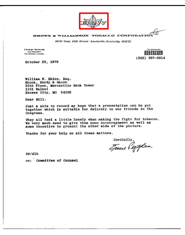
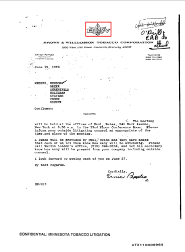
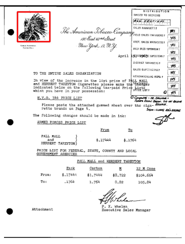
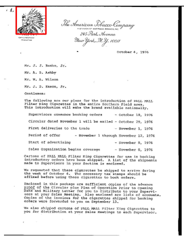
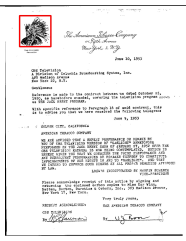
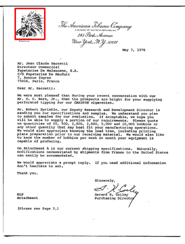
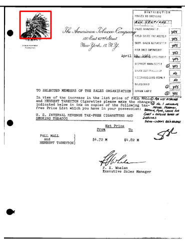
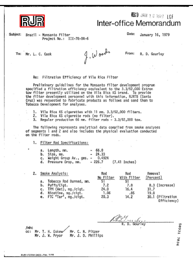
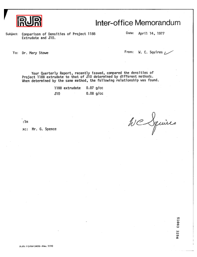
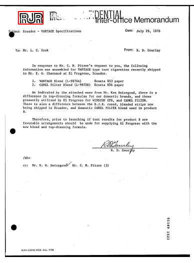
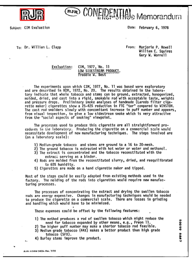
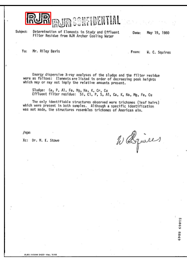
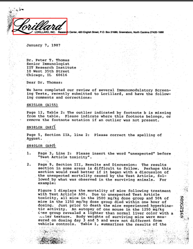
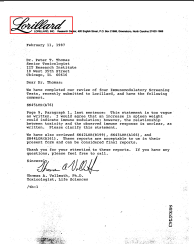
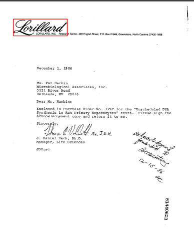
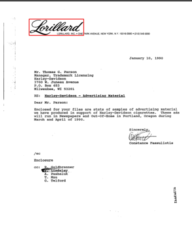
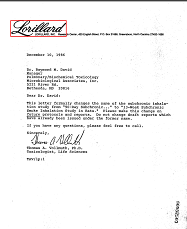
SCNN	**IoU**				
	0.1	**0.3**	**0.5**	**0.7**	**0.9**
4,096	**0.841**	**0.841**	**0.839**	**0.836**	**0.793**
512	0.711	0.707	0.705	0.699	0.607
256	**0.719**	**0.716**	**0.714**	**0.707**	**0.614**
128	0.706	0.704	0.702	0.696	0.616

rithm was used to retrieve logos without segmentation in document images. The proposed CNN approach is 51% and 60% better than the mAP reported by (JAIN; DOERMANN, 2012) considering the 4096 and 256 features, respectively. On the other hand, the mAP achieved by the CNN-256 is 15% below than the approach proposed by (RUSINOL; LLADóS, 2010). This means that the candidates retrieved by the proposed approach are not well ranked in the first positions or that there are too many false positives. The SCNN approach is 93.80% and 45.33% better than the results achieved by (JAIN; DOERMANN, 2012), with 4096 and 256 features, respectively. The SCNN-4096 is 2.08% better the results achieved by (RUSINOL; LLADóS, 2010). The results are almost 30% below for model with 256 features. However, it is important to highlight that in the paper of Rusinol and Lladós (2010), 10% of the logos were randomly chosen as the training set to carry out k-means clustering and the other 90% were considered as a test. In our SCNN approach, we did not require previous knowledge about the images.

Comparison with the state-of-the-art - Pattern Spotting

Table 4.16 shows a comparison with the current state-of-the-art for pattern spotting on the Tobacco800 dataset. We selected our best image retrieval results which have been obtained with the SCNN-4096 and the CNN-256 to evaluate the PS task. For such a comparison we have used the same experimental parameters of (LE et al., 2014; LE et al., 2013), who consider Top-5, IoU$\geq$0.6 and classes with at least three samples per category. The mAP achieved by the CNN-256 did not

Table 4.14: Qualitative spotting results for some queries. Query logo and first five retrieved logos by similarity and 4,096 feature map and location (red square) on Tobacco800 dataset

Query	1st	2nd	3rd	4th	5th
					
					
					
					

outperform the state of the art. However, the mAP achieved by the proposed SCNN-4096 is 1.31% better than mAP achieved by (LE et al., 2014) but it did not outperform the mAP presented in (LE et al., 2013). On the other hand, the recall is 4.44% and 4.86% better than (LE et al., 2014) and (LE et al., 2013) respectively. It is important to highlight that both (LE et al., 2013) and (LE et al., 2014) require the previous knowledge of the logo gallery, which is not necessary for the proposed

Table 4.15: Comparison with the state-of-the-art for retrieval task (mAP). The CNN and the SCNN-based methods, considering Top-10 and Top-25 on Tobacco800 dataset.

Method	**Top-k**	
	10	**25**
(JAIN; DOERMANN, 2012)	0.45	NA
(RUSINOL; LLADóS, 2010)	NA	0.72
CNN-4096	0.68	0.57
SCNN-4096	**0.87**	**0.73**
CNN-256	0.72	0.61
SCNN-256	0.65	0.50

NA: Not Available

method.

Table 4.16: Comparison with the state-of-the-art for Pattern Spotting in terms of mAP. Top-5, IoU$\geq$0.6, and classes with at least three samples of Tobacco800 dataset.

Method	**mAP**	**Recall (%)**
(LE et al., 2013)	**0.970**	88.42
(LE et al., 2014)	0.910	88.78
SCNN-4096	0.922	**92.72**
CNN-256	0.600	63.00

4.2 DocExplore dataset

This section presents the experimental results for the Docexplore dataset. The pre-processing, image representation, image retrieval, and spotting stages are presented. Analysis of the results is based on the experimental protocol described in Section 3.5. The end of this section shows comparisons of the results with the state-of-the-art.

4.2.1 Pre-processing

This section presents the experiments that were carried out using the methods presented in Section 3 used to search for occurrences of query images based on the Selective Search algorithm proposed by Uijlings et al. (2013). We conclude in Section 4.1.1 that the SSv2 presented better results. Thus, we use SSv2 and the same parameters for finding quality candidates. However, none objects were generated in 3 pages for errors not debugged. In this sense, we decide to personalize other parameters, being the third version of SS for DocExplore dataset, named SSv3. The parameters for SSv3 were: adaptative threshold with $block = 209$ and $offset = 0.14$, $k = 50$, 100 and 150; feature space RGB and normalize RGB; *color* + *fill* + *size* + *texture*.

To compare the quality of objects, we implement a Template Matching (*TM*) which saves the approximate location of the queries present in each image, creating our ground truth. Let us recall that the DocExplore dataset's ground truth is not publicly available by the authors. Thus, the only way to check the approximate quality of objects is through a TM. The TM compares a template against overlapped image regions. The implementation is available at a website [1]. We evaluated the performance of our TM at the online system `http://spotting.univ-rouen.fr`.

Table 4.17: Selective Search applied at the DocExplore dataset

Num of candidates	**SSv2**	**SSv3**
Totals	45,152.172	36,159.870
$IOU >= 10$	382,860	294,335
$IOU >= 30$	125,784	109,800
$IOU >= 50$	47,574	50,915
$IOU >= 70$	13,956	17,445
$IOU >= 90$	1,008	695
$IOU >= 99$	0	5

We can observe in Table 4.17, the SSv2 and SSv3 present good candidates for the system, and we reduced the number of candidates from 45M to 36M. The next sections show the results obtained using the approaches from the methods in Chapter 3, that is, Single CNN and SCNN. The results also were evaluated using

[1]`https://docs.opencv.org/3.4/df/dfb/group__imgproc__object.html#gga3a7850640f1fe1f58fe91a2d7583695daf9c3ab9296f597ea71f056399a5831da`

exhaustive search, and then the lowest Euclidean distance was ranked first. To analyze the results, a threshold k was defined to rank the results. We evaluated our results thanks to the evaluation kit provided on-line[2], where the authors included a rule to ignore junk objects and IoU$\geq$0.5 for PS task. The next sections deal with the results for the retrieval and spotting tasks, respectively.

4.2.2 First approach results: single CNN

It is worth mentioning that the images used to fine-tune the CNN were generated from the use of data augmentation techniques on the set of all the 1447 image queries from DocExplore dataset, resulting in 54,644 training images. The dataset was split into a training set of 75% and a validation set of 25% to fine-tune the CNN. So, only transformed images were used to train the neural network. The results were evaluated using the exhaustive search, in which a comparison was made between all the features extracted from the candidates and the query image, and then the lowest distance was ranked first. The next sections present the results for IR and PS tasks.

4.2.2.1 Image Retrieval task

Table 4.18 shows the experimental results comparing the feature map obtained with fine-tuned AlexNet architecture, which has 4,096 elements at the output of the convolutional layer (*fc7*), with the modified model that has an additional layer to reduce the feature map to 256 elements ($fc8_{new}$). As one may see in Table 4.18 the best results were achieved using the top-1000 ranking with 256 features resulted in 0.141 of mAP, while the model with 4096 features resulted in 0.042. In summary, we achieved an improvement of 3.3 times in mAP, while the number of features was reduced by 16 times.

We can observe that our results for IR and PS tasks have shown the same behavior of Tobacco800 dataset. The model trained with 256 features improve the

[2]`http://spotting.univ-rouen.fr/evaluation-kit/`

results while the set of features was reduced. Besides, related to computational time, during the online phase, the query search time is improved from $\cong$ 1.7m (4096 features) to $\cong$ 1.6m (256 features). Then, the computational time decreased when using the model with 256 features, while it returns better results.

Table 4.18: Image retrieval (IR) results with Top-k ranking for Single CNN with 4096 and 256 on DocExplore dataset and SSv3.

Feat. map	**Top-*k***				
	100	**300**	**500**	**700**	**1000**
256	0.052	0.096	0.120	0.133	**0.141**
4096	0.007	0.017	0.025	0.033	**0.042**

We also can see the results per category of queries using the single CNN with 256 features for IR, SSv3 and top-1000 ranking in Figure 4.3. We observe that for 2 categories we achieved a mAP $\cong$ 0.3 and 8 categories above 0.1. Some categories achieved mAP lower than 0.1. This can be explained by the fact that these categories could contain too few samples that are moreover of bad quality.

4.2.2.2 Pattern Spotting task

According to the system at *Rouen* system, the IOU was $IoU \geqslant 0.5$ to consider that a returned bounding box is a good retrieval and the precision/recall is calculated. Table 4.19 shows the experimental results of the proposed PS method, which single CNN-4,096 and 256, and the candidates generated with SSv3. We observe our results are similar to any Top-k ranking, where we achieved 0.001 of mAP in our best results, using 256 features. Thus we improve in 10 times the results for PS using a reduced set of features. However, this approach did not return promising results, since only 0.1% of queries were retrieved with good precise location.

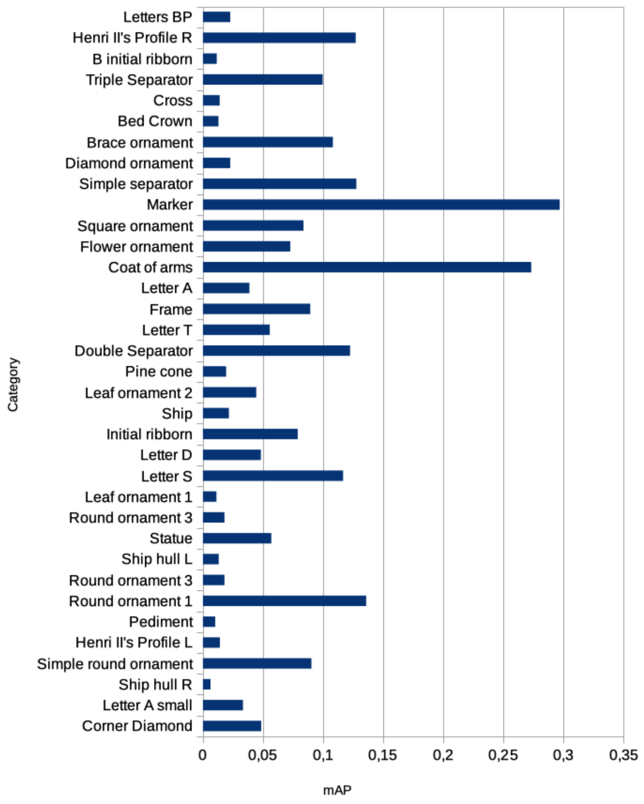


Figure 4.3: Results by category using Single CNN 256 feature map and top 1000 ranking on DocExplore dataset

Table 4.19: Pattern spotting (PS) results with Top-k ranking for Single CNN 4096 and 256 on DocExplore dataset and SSv3

Feat. map	**Top-k**				
	100	**300**	**500**	**700**	**1000**
256	0.0007	0.0010	0.0010	0.0010	0.0010
4096	0.0001	0.0001	0.0001	0.0001	0.0001

4.2.3 Second approach results: SCNN

This section shows the results obtained for Image Retrieval and Pattern Spotting tasks using the SCNN approach. The SCNN architecture trained with ImageNet dataset provides 4096-dim feature maps corresponding to its original fc_7 layer. We have also investigated the addition of a new layer (fc_{new}) to reduce the feature maps to 256 dimensions.

4.2.3.1 Image Retrieval task

We also evaluated our results thanks to the evaluation kit provided online[3]. Table 4.20 shows the experimental results of the proposed image retrieval and pattern spotting methods, using the SCNN 4096 and 256. We selected this feature maps because of its performance in Tobacco800 dataset. Let us recall the authors included a rule to ignore junk objects and IoU$\geq$0.5 for PS task. We can observe that our best result for IR is with Top-1000 in SCNN-4096 and SSv3 with 0.386 for mAP, but for Top-500 the difference is only 3.5%.

Table 4.20: Image retrieval (IR) results with Top-*k* ranking for SCNN 4096 and 256 on DocExplore dataset

Feat. map	**Top-*k***				
	100	**300**	**500**	**700**	**1000**
4096	0.296	0.355	0.373	0.381	**0.386**
256	0.039	0.167	0.184	0.193	0.201

We also can see the results per category of queries using the SCNN with 4,096 features, SSv3 and top-1000 ranking in Figure 4.4. We observe that for 2 categories we achieved mAP 1.0 and 14 categories above 0.4. The categories Pediment, Flower ornament and Henri's II Profile L in particular, we achieved mAP greater than 0.7. Some queries of 6 categories achieved mAP lower than 0.1. This can be explained by the fact that these categories could contain too few samples that are moreover of bad quality. However, for 19 categories we have more than 0.3 in mAP and 10 categories more than 0.5.

Table 4.21 shows some qualitative results of the images retrieved using SCNN with 4096 feature map. These results are very promising since many correct images were retrieved in the top-5 ranking. However, when the search involves an image with few positive samples, the method returns some false candidates, as

[3]`http://spotting.univ-rouen.fr/evaluation-kit/`

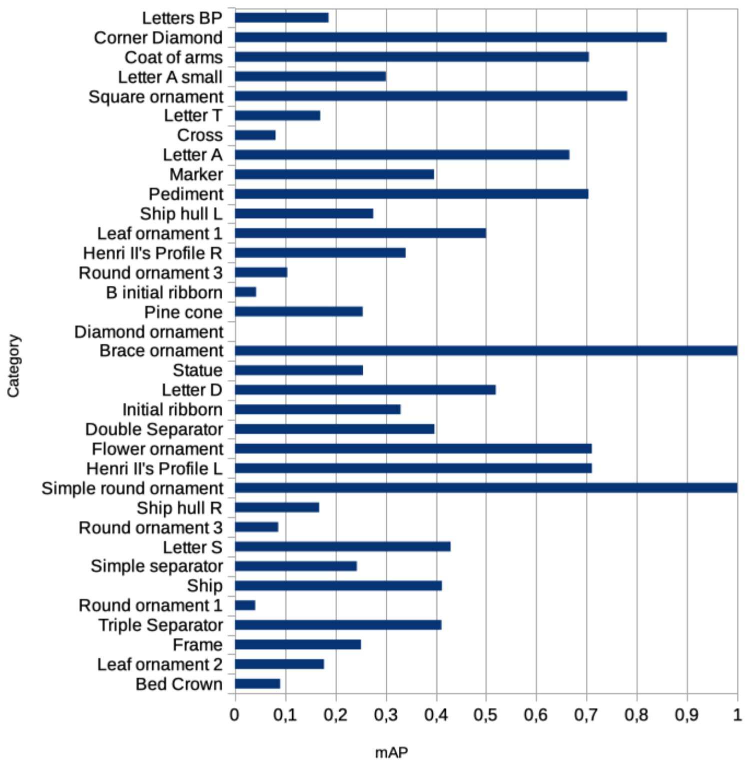


Figure 4.4: Results by category using 4,096 feature map and top 1000 ranking on DocExplore dataset

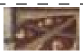
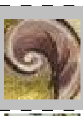
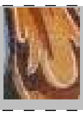
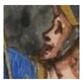
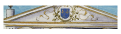
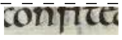
seen in the eighth row (in gray), although the level of detail of the method is represented in Figure 4.5 for this same query (red square).

Besides, in this approach, we also observed the time-consuming. During the online phase, the query search time is improved from ≅ 2.1m (4096 features) to ≅ 1.6m (256 features). Then, the computational time decreased when using the model with 256 features, while it returns promising results.

4.2.3.2 Pattern Spotting task

Table 4.22 shows the experimental results of the proposed PS method, which SCNN-4,096 because of our best results of IR,and the candidates generated with SSv2 and SSv3. The same evaluation system was used as the Single CNN

Table 4.21: Qualitative retrieval results for some queries of DocExplore dataset. Query image and first five retrieved candidates by similarity and 4,096 feature map

Query	Retrieved				
	1st	2nd	3rd	4th	5th
					
					
					
					

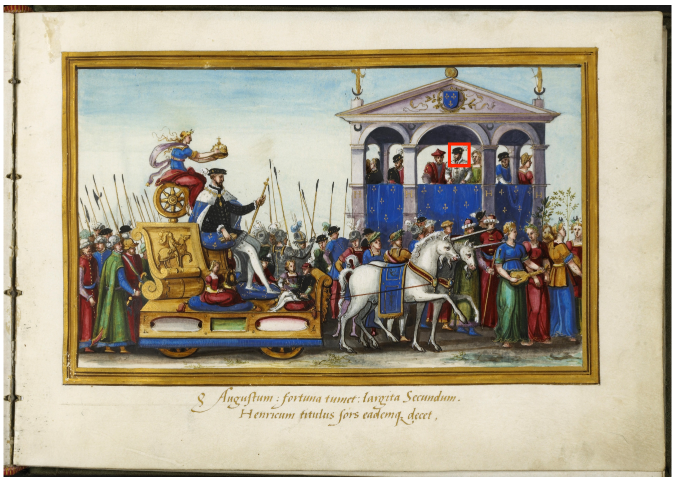

S Augustum : fortuna timet : largita Secundum.
Henricum titulus sors eademq. decet.

Figure 4.5: First candidate found for the query 945 using SCNN$_{4096}$

approach. We observe our results are very similar to any Top-k ranking, but in our best result, we achieved 0.076 of mAP.

As En et al. (2016b), we observed that many candidates often contain only a part of the query or overlap with other ranked candidates, hence reducing the performance of the system. Thus, we propose a post-processing stage to use a "union" of these retained candidates to discover rectangular regions as a way to improve the performance of the localization task. Thus, we selected the first 2000 candidates to apply the union step. After the union, the first 1000 were considered to feed the evaluation system, similar to (EN et al., 2016a) and (EN et al., 2016b). We can observe that our results for PS are approximately 2.3 times better using post-processing.

Table 4.22: Pattern spotting (PS) results with Top-k ranking for SCNN-4096 on DocExplore dataset

Feat. map	**Top-k**				
	100	**300**	**500**	**700**	**1000**
PS	0.073	0.075	0.076	0.076	0.076
PS (PP)	0.174	0.174	0.174	0.174	**0.174**

PP: Post-processing

4.2.4 Comparison with the state-of-the-art

Table 4.23 shows the comparison with the current state-of-the-art and our experimental results for SCNN, because of the best results presented for this approach. It can be observed that we did not outperform (EN et al., 2016a), being almost 37% worse for IR. They used BING to propose regions and the feature extraction is based on VLAD and K-means. In addition, the authors included a post-processing stage using template matching. At each retrieval, the Top-2000 regions similar to the query are kept as inputs of the template matching stage. Finally, only the first 1000 matched regions are used for calculating the mAP. It can be observed that the mAP achieved by the proposed method in the localization task outperforms in 57% the results presented in En et al. (2016a) when considering our post-processing method.

The mAP for IR achieved by the proposed SCNN is similar to the mAP

Table 4.23: Image retrieval (IR) and pattern spotting (PS) results on DocExplore dataset. Values refer to mAP on Top-1000 candidates for SCNN-4096

Method	**IR**	**PS**
(EN et al., 2016a)	**0.613**	0.111
(ÚBEDA et al., 2019) ES	0.286	0.139
(ÚBEDA et al., 2019) PP	0.386	0.173
SCNN-4096 PP	0.386	**0.174**

ES: exhaustive search, PP: Post-processing

achieved by Úbeda et al. (2019) with post-processing (PP). (ÚBEDA et al., 2019) used RetinaNet as a feature extractor and added a post-processing step where they discard the localization if the bounding box is not entirely contained in the original page. In addition, they trained a classifier with a set of pages of DocExplore and use it to predict non-text regions in all pages to generate region proposals. It is important to notice that we did not perform any training using images of DocExplore for both the region proposal and the feature extractor. However, whereas the authors used an exhaustive search, our results outperformed in approximately 35% in IR and 25% for PS. Therefore, our approach is better at retrieving patterns and it does not need any additional classifier to improve the performance.

4.3 Final considerations

In this chapter, our objective was to evaluated IR and PS tasks of the graphical objects on two public datasets (Tobacco800 and DocExplore). To achieve this objective, firstly we investigated the use SS to produce a reasonable amount of higher-quality candidates. Thus, we evaluated the two proposed approaches based on CNN architecture. Based on the experimental results, the proposed single CNN approach can provide a discriminant representation of queries and image candidates for IR and PS tasks applied in document image collections and it outperformed the state-of-the-art on Tobacco800 dataset. In this case, we prove our first hypothesis proposed in Chapter 1. In SCNN proposed approach, in addition to providing discriminant representation, it can be used to construct independent-dataset solutions for IR and PS tasks in the context of document image collections.

Thus, we prove our second hypothesis developing a generic solution that is independent of the query images to work on different document collections. However, in DocExplore experiments, although the results were very similar to the literature, we believe that there is still a large room for improvement. Given such big difference in IR and PS tasks, a sanity check is necessary to understand if the regions generated could be improved. One simple solution would be to take into account some filter algorithm to reduce the candidate dataset, but it should be discussed with care.

Chapter 5

Conclusions and Future Work

In this work, we presented two approaches for IR and PS constructed to minimize or even avoid the prior knowledge dependence of the dataset to be used. In the first approach, we presented a promising Single CNN for IR and PS in document image collections. The candidate images were indexed and represented by a reduced set of deep features. An additional layer was added to the AlexNet architecture with the aim of reducing the dimension of the feature map and to improve the efficiency of the search process.

However, one main disadvantage of this approach is that the queries are used for training (fine-tuning) the CNN. Therefore, previous knowledge of the queries is needed a priori making this approach problem (or dataset) dependent. The experimental results for this approach were very promising. It was possible to observe an increase in the average precision while reducing considerably the time consumed by the retrieval task. Therefore, we proved that the first hypothesis is valid, which a pre-trained Convolutional Neural Network (CNN) model constructed to deal with the lack of training data can provide a discriminant representation of queries and image candidates for IR and PS tasks applied in document image collections.

In the second approach, we presented a novel approach for IR and PS in document image collections where the images are represented using a Siamese CNN model trained on the ImageNet dataset. We evaluated the results of conventional AlexNet architecture and also the modified version with an additional layer with the aim of reducing the dimension of the feature map. Our experimental results were very promising. It was possible to observe an increase in the mAP since the features generalize well and improve the matching performance compared to features obtained with networks trained for image classification.

The main advantage over the previous approach is that we do not need to know the queries in advance, i.e. during the off-line phase (training), thus being independent of the queries. Therefore, the SCNN based approach may be viewed as a problem (or dataset) independent approach for retrieving document images and spotting patterns, as we demonstrated in the experiments (Section 4) by using the same trained SCNN to retrieve images from two very different datasets. Thus, we proved that the second hypothesis is valid, in which deep metric based on Siamese Convolutional Neural Network (SCNN) trained on a general image dataset can be used to construct independent-dataset solutions for IR and PS tasks in the context of document image collections. This trained model also can be available to other researchers use in different solutions of IR and PS, and, also, to perform online the fine-tuning where the model can learns based on different query images.

Besides, we highlight the main difficulties of these approaches. In data augmentation step in the single CNN because we need to define the processing to generate an adequate number of images for each class which can influence in training step. Some experiments with different values of data augmentation or to generate a training set based on document images to include a bigger variability of images could be evaluated. The object proposal algorithm, where some parameters were chosen and a limited number of candidates were generated. Some comparisons with other algorithms based on segmentation could be interesting and maybe candidates with more quality will be generated. Besides, the chose of the architecture of deep learning as a feature extractor able to return promising results and that spent good computational time was a challenge because of the limits of computational resources.

Further work must be carried out to improve the SCNN model by using other architectures (such as Resnet , Retina network and, VGG). Aside from IR and PS in document images, the transfer learning works in different datasets, we could use our system for other spotting-related problem to evaluate the proposed SCNN model, such as in Paris[1], INRIA Holidays[2], Washington Database[3] . Finally, given that we can turn our system into a word spotting system, it would be interesting to design a system that can spot words and patterns, making it a generic system.

[1]https://www.robots.ox.ac.uk/ vgg/data/parisbuildings/

[2]https://lear.inrialpes.fr/ jegou/data.php

[3]http://www.fki.inf.unibe.ch/databases/iam-historical-document-database/washington-database

Bibliography

AGAM, G. et al. *The Complex Document Image Processing (CDIP) Test Collection Project*. http://ir.iit.edu/projects/CDIP.html, 2006.

AKÇAY, S. et al. Transfer learning using convolutional neural networks for object classification within x-ray baggage security imagery. In: *2016 IEEE International Conference on Image Processing (ICIP)*. [S.l.: s.n.], 2016. p. 1057–1061.

ALAEI, A.; DELALANDRE, M.; GIRARD, N. Logo detection using painting based representation and probability features. In: *12th International Conference on Document Analysis and Recognition*. [S.l.: s.n.], 2013. v. 1236-1239.

ALEXE, B.; DESELAERS, T.; FERRARI, V. What is an object? In: *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. [S.l.: s.n.], 2010.

ALMAZÁN, J. et al. Word spotting and recognition with embedded attributes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 36, p. 2552–2566, 2014.

ARBELÁEZ, P. et al. Multiscale combinatorial grouping. In: *2014 IEEE Conference on Computer Vision and Pattern Recognition*. [S.l.: s.n.], 2014. p. 328–335. ISSN 1063-6919.

BABENKO, A.; LEMPITSKY, V. Aggregating local deep features for image retrieval. In: *The IEEE International Conference on Computer Vision (ICCV)*. [S.l.: s.n.], 2015.

BABENKO, A. et al. Neural codes for image retrieval. In: FLEET, D. et al. (Ed.). *Computer Vision – ECCV 2014*. Cham: Springer International Publishing, 2014. p. 584–599. ISBN 978-3-319-10590-1.

BARNEY, B. *Introduction to Parallel Computing*. June 2018.

BAY, H. et al. Speeded-up robust features (surf). *Computer Vision and Image Understanding*, v. 110, n. 3, p. 346 – 359, 2008.

BEKAS, C. *Introduction to Parallel Computing: The Message Passing, Shared Memory and Hybrid paradigms*. Switzerland, 2009.

BENGIO, Y. Learning deep architectures for ai. *Found. Trends Mach. Learn.*, Now Publishers Inc., Hanover, MA, USA, v. 2, n. 1, p. 1–127, 2009. ISSN 1935-8237.

BENGIO, Y.; LECUN, Y. Scaling learning algorithms towards AI. In: BOTTOU, L. et al. (Ed.). *Large Scale Kernel Machines*. [S.l.]: MIT Press, 2007.

BERLEMONT, S. et al. Class-balanced siamese neural networks. *Neurocomputing*, v. 273, p. 47 – 56, 2018. ISSN 0925-2312.

BHATTACHARJEE, S. D. et al. Query-adaptive small object search using object proposals and shape-aware descriptors. *IEEE Transactions on Multimedia*, v. 18, n. 4, p. 726–737, 2016.

BISHOP, C. M. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006. ISBN 0387310738.

BOSCH, E. F. *Region-oriented Convolutional Networks for Object Retrieval*. Dissertação (Mestrado), 2015.

BROMLEY, J. et al. Signature verification using a "siamese" time delay neural network. In: *Proceedings of the 6th International Conference on Neural Information Processing Systems*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1993. p. 737–744.

CARREIRA, J.; SMINCHISESCU, C. Cpmc: Automatic object segmentation using constrained parametric min-cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 34, n. 7, p. 1312–1328, 2012.

CHATBRI, H.; KWAN, P.; KAMEYAMA, K. A modular approach for query spotting in document images and its optimization using genetic algorithms. In: *Proceedings of the 2014 IEEE Congress on Evolutionary Computation, CEC 2014*. [S.l.: s.n.], 2014.

CHEN, J. et al. Image retrieval based on image-to-class similarity. *Pattern Recognition Letters*, v. 83, Part 3, p. 379 – 387, 2016.

CHENG, M.-M. et al. BING: Binarized normed gradients for objectness estimation at 300fps. In: *IEEE CVPR*. [S.l.: s.n.], 2014.

CHOPRA, S.; HADSELL, R.; LECUN, Y. Learning a similarity metric discriminatively, with application to face verification. In: *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*. [S.l.: s.n.], 2005. v. 1, p. 539–546 vol. 1. ISSN 1063-6919.

CHUNG, Y.-A.; WENG, W.-H. Learning deep representations of medical images using siamese cnns with application to content-based image retrieval. 11 2017.

DALAL, N.; TRIGGS, B. Histograms of oriented gradients for human detection. In: *In CVPR*. [S.l.: s.n.], 2005. p. 886–893.

DATTA, R. et al. Image retrieval: Ideas, influences, and trends of the new age. *ACM Comput. Surv.*, ACM, New York, NY, USA, v. 40, n. 2, p. 5:1–5:60, maio 2008. ISSN 0360-0300. Disponível em: <http://doi.acm.org/10.1145/1348246-.1348248>.

DELALANDRE, M.; OGIER, J.-M.; LLADÓS, J. A fast cbir system of old ornamental letter. In: LIU, W.; LLADÓS, J.; OGIER, J.-M. (Ed.). *Recent Advances and New Opportunities: 7th International Workshop, GREC 2007, Curitiba, Brazil, September 20-21, 2007*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008. p. 135–144.

DOERMANN, D. The indexing and retrieval of document images: A survey. *Computer Vision and Image Understanding*, v. 70, n. 3, p. 287 – 298, 1998.

DOSOVITSKIY, A. et al. Discriminative unsupervised feature learning with convolutional neural networks. In: *Advances in Neural Information Processing Systems 27 (NIPS)*. [S.l.: s.n.], 2014.

DOVGALECS, V. et al. Spot it! finding words and patterns in historical documents. In: *2013 12th International Conference on Document Analysis and Recognition*. [S.l.: s.n.], 2013. p. 1039–1043.

DRUZHKOV, P. N.; KUSTIKOVA, V. D. A survey of deep learning methods and software tools for image classification and object detection. *Pattern Recognition and Image Analysis*, v. 26, n. 1, p. 9–15, 2016. ISSN 1555-6212.

EN, S. et al. New public dataset for spotting patterns in medieval document images. *Journal of Electronic Imaging*, v. 26, n. 1, 2016.

EN, S. et al. Pattern localization in historical document images via template matching. In: *2016 23rd International Conference on Pattern Recognition (ICPR)*. [S.l.: s.n.], 2016. p. 2054–2059.

EN, S. et al. A scalable pattern spotting system for historical documents. *Pattern Recognition*, v. 54, p. 149–161, 2016.

ENDRES, I.; HOIEM, D. Category independent object proposals. In: DANIILIDIS, K.; MARAGOS, P.; PARAGIOS, N. (Ed.). *Computer Vision – ECCV 2010: 11th European Conference on Computer Vision, Heraklion, Crete, Greece,* *September 5-11, 2010, Proceedings, Part V*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010. p. 575–588.

FAKTOR, A.; IRANI, M. Co-segmentation by composition. In: *IEEE International Conference on Computer Vision, ICCV 2013*. Sydney, Australia: [s.n.], 2013. p. 1297–1304.

FANG, Z. et al. A comprehensive analysis and parallelization of an image retrieval algorithm. In: *(IEEE ISPASS) IEEE International Symposium on Performance Analysis of Systems and Software*. [S.l.: s.n.], 2011. p. 154–164.

FORSYTH, D. A.; PONCE, J. *Computer Vision: A Modern Approach*. [S.l.]: Prentice Hall Professional Technical Reference, 2002. ISBN 0130851981.

GIONIS, A.; INDYK, P.; MOTWANI, R. Similarity search in high dimensions via hashing. In: *Proceedings of the 25th International Conference on Very Large Data Bases*. [S.l.: s.n.], 1999. p. 518–529.

GIRSHICK, R. et al. Rich feature hierarchies for accurate object detection and semantic segmentation. In: *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition*. [S.l.: s.n.], 2014.

GLOROT, X.; BORDES, A.; BENGIO, Y. Deep sparse rectifier neural networks. In: GORDON, G. J.; DUNSON, D. B. (Ed.). *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics (AISTATS-11)*. [S.l.]: Journal of Machine Learning Research - Workshop and Conference Proceedings, 2011. v. 15, p. 315–323.

GONG, Y. et al. Multi-scale orderless pooling of deep convolutional activation features. In: FLEET, D. et al. (Ed.). *Computer Vision ECCV 2014*. [S.l.]: Springer International Publishing, 2014, (Lecture Notes in Computer Science, v. 8695). p. 392–407.

GONZALEZ, R. C.; WOODS, R. E. *Digital Image Processing (3rd Edition)*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 2006. ISBN 013168728X.

GORDO, A. et al. Deep image retrieval: Learning global representations for image search. In: LEIBE, B. et al. (Ed.). *Computer Vision ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part VI*. Cham: Springer International Publishing, 2016. p. 241–257. ISBN 978-3-319-46466-4.

GRAMA, A. et al. *Introduction to Parallel Computing*. 2nd. ed. [S.l.]: Pearson Education Limited, 2003.

GUO, J.; ZHANG, S.; LI, J. Hash learning with convolutional neural networks for semantic based image retrieval. In: ______. [S.l.]: Springer International Publishing, 2016. p. 227–238.

HANGARGE, M. et al. Gabor wavelets based word retrieval from kannada documents. *Procedia Computer Science*, v. 79, p. 441 – 448, 2016.

HAYKIN, S. *Neural Networks: A comprehensive foundation*. 2nd. ed. Porto Alegre: Bookman, 2001.

HE, S.; LAU, R. W. H. Oriented object proposals. In: *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*. Washington, DC, USA: IEEE Computer Society, 2015. (ICCV '15), p. 280–288.

HE, S. et al. Historical manuscript dating based on temporal pattern codebook. *Computer Vision and Image Understanding*, v. 152, p. 167 – 175, 2016.

HINTON, G. E.; OSINDERO, S.; TEH, Y.-W. A fast learning algorithm for deep belief nets. *Neural Computation*, Massachusetts Institute of Technology, v. 18, n. 7, p. 1527–1554, 2006.

HU, B. et al. Image mining of historical manuscripts to establish provenance. In: *In SDM*. [S.l.: s.n.], 2012.

HUBEL, D. H.; WIESEL, T. N. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of physiology*, v. 160, p.

106–154, jan. 1962. ISSN 0022-3751. Disponível em: <http://www.ncbi.nlm.nih-.gov/pmc/articles/PMC1359523/>.

INDYK, P.; MOTWANI, R. Approximate nearest neighbors: Towards removing the curse of dimensionality. In: *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*. New York, NY, USA: ACM, 1998. (STOC '98), p. 604–613.

IQBAL, K.; ODETAYO, M. O.; JAMES, A. Content-based image retrieval approach for biometric security using colour, texture and shape features controlled by fuzzy heuristics. *Journal of Computer and System Sciences*, v. 78, n. 4, p. 1258 – 1277, 2012. ISSN 0022-0000. Disponível em: <http://www.sciencedirect.com-/science/article/pii/S002200001100119X>.

JADERBERG, M.; VEDALDI, A.; ZISSERMAN, A. Deep features for text spotting. In: FLEET, D. et al. (Ed.). *Computer Vision – ECCV 2014*. Cham: Springer International Publishing, 2014. p. 512–528. ISBN 978-3-319-10593-2.

JAIN, A. K.; DUIN, R. P. W.; MAO, J. Statistical pattern recognition: A review. *IEEE Trans. Pattern Anal. Mach. Intell.*, IEEE Computer Society, Washington, DC, USA, v. 22, n. 1, p. 4–37, jan. 2000. ISSN 0162-8828.

JAIN, R.; DOERMANN, D. Logo retrieval in document images. In: *2012 10th IAPR International Workshop on Document Analysis Systems*. [S.l.: s.n.], 2012. p. 135–139.

JIA, Y. et al. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.

JING, L. et al. A convolutional neural network based feature learning and fault diagnosis method for the condition monitoring of gearbox. *Measurement*, v. 111, 07 2017.

KALANTIDIS, Y.; MELLINA, C.; OSINDERO, S. Cross-dimensional weighting for aggregated deep convolutional features. In: HUA, G.; JÉGOU, H. (Ed.). *Computer Vision ECCV 2016 Workshops: Amsterdam, The Netherlands, October 8-10 and 15-16, 2016, Proceedings, Part I*. Cham: Springer International Publishing, 2016. p. 685–701. ISBN 978-3-319-46604-0.

KANADJE, M. et al. Assisted keyword indexing for lecture videos using unsupervised keyword spotting. *Pattern Recognition Letters*, v. 71, p. 8–15, 2016.

KAO, O.; STEINERT, G.; DREWS, F. Scheduling aspects for image retrieval in cluster-based image databases. In: *Proceedings First IEEE/ACM International Symposium on Cluster Computing and the Grid*. [S.l.: s.n.], 2001. p. 329–336.

KAVITHA, C. et al. Image retrieval based on local histogram and texture features. *International Journal of Computer Science and Information Technologies*, v. 2, n. 2, p. 741–746, 2011.

KOCH, G.; ZEMEL, R.; SALAKHUTDINOV, R. Siamese neural networks for one-shot image recognition. In: *ICML 2015 Deep Learning Workshop*. [S.l.: s.n.], 2015.

KRIZHEVSKY, A.; HINTON, G. E. Using very deep autoencoders for content-based image retrieval. In: *ESANN*. [S.l.: s.n.], 2011.

KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. E. Imagenet classification with deep convolutional neural networks. In: *Advances in Neural Information Processing Systems*. [S.l.: s.n.], 2012.

KULIS, B.; GRAUMAN, K. Kernelized locality-sensitive hashing for scalable image search. In: *2009 IEEE 12th International Conference on Computer Vision*. [S.l.: s.n.], 2009. p. 2130–2137.

KUMAR, G.; GOVINDARAJU, V. Bayesian background models for keyword spotting in handwritten documents. *Pattern Recognition*, 2016.

KUO, C.-H.; CHOU, Y.-H.; CHANG, P.-C. Using deep convolutional neural networks for image retrieval. In: *International Symposium on Electronic Imaging 2016*. [S.l.]: Society for Imaging Science and Technology, 2016.

KUO, S.-S.; AGAZZI, O. E. Keyword spotting in poorly printed documents using pseudo 2-d hidden markov models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 16, n. 8, p. 842–848, Aug 1994. ISSN 0162-8828.

KUO, W.; HARIHARAN, B.; MALIK, J. Deepbox: Learning objectness with convolutional networks. In: *2015 IEEE International Conference on Computer Vision (ICCV)*. [S.l.: s.n.], 2015. p. 2479–2487.

LE, V. P. et al. Document retrieval based on logo spotting using key-point matching. In: *2014 22nd International Conference on Pattern Recognition*. [S.l.: s.n.], 2014. p. 3056–3061.

LE, V. P. et al. Improving logo spotting and matching for document categorization by a post-filter based on homography. In: *2013 12th International Conference on Document Analysis and Recognition*. [S.l.: s.n.], 2013. p. 270–274.

LECUN, Y. et al. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, v. 1, n. 4, p. 541–551, Dec 1989. ISSN 0899-7667.

LECUN, Y. et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, v. 86, n. 11, p. 2278–2324, Nov 1998. ISSN 0018-9219.

LECUN, Y.; KAVAKCUOGLU, K.; FARABET, C. Convolutional networks and applications in vision. In: *Proceedings of ISCAS*. Paris: [s.n.], 2010. p. 253–256.

LEWIS, D. et al. Building a test collection for complex document information processing. In: *Proc. 29th Annual Int. ACM SIGIR Conference (SIGIR 2006)*. [S.l.: s.n.], 2006. p. 665–666.

LIN, K. et al. Deep learning of binary hash codes for fast image retrieval. In: *2015 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. [S.l.: s.n.], 2015. p. 27–35. ISSN 2160-7508.

LIN, T. et al. Learning deep representations for ground-to-aerial geolocalization. In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. [S.l.: s.n.], 2015. p. 5007–5015. ISSN 1063-6919.

LIU, L.; QI, H. Learning effective binary descriptors via cross entropy. In: *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*. [S.l.: s.n.], 2017. p. 1251–1258.

LIU, S.; LU, C.; JIA, J. Box aggregation for proposal decimation: Last mile of object detection. In: *2015 IEEE International Conference on Computer Vision (ICCV)*. [S.l.: s.n.], 2015. p. 2569–2577.

LOWE, D. G. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, v. 60, n. 2, p. 91–110, 2004. ISSN 1573-1405. Disponível em: <http://dx.doi.org/10.1023/B:VISI.0000029664.99615.94>.

LU, X.; ZHENG, X.; LI, X. Latent semantic minimal hashing for image retrieval. *IEEE Transactions on Image Processing*, v. 26, n. 1, p. 355–368, 2017.

LU, Y. et al. Study of content-based image retrieval using parallel computing technique. In: *Proceedings of the 2007 Asian Technology Information Program's (ATIP's) 3rd Workshop on High Performance Computing in China: Solution Approaches to Impediments for High Performance Computing*. New York, NY, USA: ACM, 2007. (CHINA HPC '07), p. 186–191. ISBN 978-1-59593-903-6. Disponível em: <http://doi.acm.org/10.1145/1375783.1375820>.

MAAS, A. L.; HANNUN, A. Y.; NG, A. Y. Rectifier nonlinearities improve neural network acoustic models. In: *Proc. ICML*. [S.l.: s.n.], 2013. v. 30, n. 1.

MACHADO, F. B.; MAIA, L. P. *Arquitetura de sistemas operacionais*. 4th. ed. Rio de Janeiro: LTC - Livros Técnicos e Científicos Editora S.A., 1999.

MAESSCHALCK, R. D.; JOUAN-RIMBAUD, D.; MASSART, D. The mahalanobis distance. *Chemometrics and Intelligent Laboratory Systems*, v. 50, n. 1, p. 1 – 18, 2000. ISSN 0169-7439. Disponível em: <http://www.sciencedirect.com-/science/article/pii/S0169743999000477>.

MALIK, F.; BAHARUDIN, B. Analysis of distance metrics in content-based image retrieval using statistical quantized histogram texture features in the {DCT} domain. *Journal of King Saud University - Computer and Information Sciences*, v. 25, n. 2, p. 207 – 218, 2013.

MANEN, S.; GUILLAUMIN, M.; GOOL, L. V. Prime object proposals with randomized prim's algorithm. In: *Computer Vision (ICCV), 2013 IEEE International Conference on*. [S.l.: s.n.], 2013. p. 2536–2543.

MANMATHA, R.; HAN, C.; RISEMAN, E. M. Word spotting: a new approach to indexing handwriting. In: *Computer Vision and Pattern Recognition, 1996. Proceedings CVPR '96, 1996 IEEE Computer Society Conference on*. [S.l.: s.n.], 1996. p. 631–637. ISSN 1063-6919.

MARINAI, S.; MIOTTI, B.; SODA, G. Digital libraries and document image retrieval techniques: A survey. In: BIBA, M.; XHAFA, F. (Ed.). *Learning Structure and Schemas from Documents*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011. p. 181–204.

MELEKHOV, I.; KANNALA, J.; RAHTU, E. Siamese network features for image matching. In: *2016 23rd International Conference on Pattern Recognition (ICPR)*. [S.l.: s.n.], 2016. p. 378–383.

MENON, A. et al. Characterization of a class of sigmoid functions with applications to neural networks. *Neural Networks*, v. 9, n. 5, p. 819 – 835, 1996. ISSN 0893-6080. Disponível em: <http://www.sciencedirect.com/science/article-/pii/0893608095001077>.

MITCHELL, T. M. *Machine Learning*. 1. ed. New York, NY, USA: McGraw-Hill, Inc., 1997. ISBN 0070428077, 9780070428072.

MONDAL, T. et al. Flexible sequence matching tecnique: An effective learning-free approach for word sppoting. *Pattern Recognition*, p. 596–612, 2016.

NOWOZIN, S. Optimal decisions from probabilistic models: the intersection-over-union case. In: *Computer Vision and Pattern Recognition (CVPR 2014)*. [S.l.]: IEEE Computer Society, 2014.

OLIVARES, J. et al. Minimum sum of absolute differences implementation in a single fpga device. In: *Field Programmable Logic and Application: 14th International Conference, FPL 2004, Leuven, Belgium, August 30-September 1, 2004. Proceedings*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004. p. 986–990.

ONG, E.-J.; HUSAIN, S.; BOBER, M. Siamese network of deep fisher-vector descriptors for image retrieval. 02 2017.

OQUAB, M. et al. Learning and transferring mid-level image representations using convolutional neural networks. In: *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition*. Washington, DC, USA: IEEE Computer Society, 2014. (CVPR '14), p. 1717–1724.

OU, X. et al. Inductive transfer deep hashing for image retrieval. In: *Proceedings of the 22Nd ACM International Conference on Multimedia*. New York, NY, USA: ACM, 2014. (MM '14), p. 969–972.

PAN, S. J.; YANG, Q. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, v. 22, n. 10, p. 1345–1359, 2010.

POWERS, D. M. W. Evaluation: From precision, recall and f-measure to roc., informedness, markedness & correlation. *Journal of Machine Learning Technologies*, v. 2, n. 1, p. 37–63, 2011.

QI, Y. et al. Sketch-based image retrieval via siamese convolutional neural network. In: *2016 IEEE International Conference on Image Processing (ICIP)*. [S.l.: s.n.], 2016. p. 2460–2464. ISSN 2381-8549.

RADENOVIC, F.; TOLIAS, G.; CHUM, O. Cnn image retrieval learns from bovw: Unsupervised fine-tuning with hard examples. In: *Conference Computer Vision ECCV*. [S.l.: s.n.], 2016.

RAKTHANMANON, T.; ZHU, Q.; KEOGH, E. J. Mining historical documents for near-duplicate figures. In: *2011 11th IEEE International Conference on Data Mining*. [S.l.: s.n.], 2011. p. 557–566.

RATH, T. M.; MANMATHA, R. Word spotting for historical documents. *International Journal of Document Analysis and Recognition (IJDAR)*, v. 9, n. 2, p. 139–152, 2007. ISSN 1433-2825. Disponível em: <http://dx.doi.org/10.1007-/s10032-006-0027-8>.

RAZAVIAN, A. S. et al. Cnn features off-the-shelf: An astounding baseline for recognition. In: *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops*. Washington, DC, USA: IEEE Computer Society, 2014. (CVPRW '14), p. 512–519. ISBN 978-1-4799-4308-1.

RAZAVIAN, A. S. et al. A baseline for visual instance retrieval with deep convolutional networks. *CoRR*, 2014.

RECHT, B. et al. Hogwild: A lock-free approach to parallelizing stochastic gradient descent. In: SHAWE-TAYLOR, J. et al. (Ed.). *Advances in Neural Information Processing Systems 24*. [S.l.: s.n.], 2011. p. 693–701.

RIBA, P. et al. Large-scale graph indexing using binary embeddings of node contexts for information spotting in document image databases. *Pattern Recognition Letters*, v. 48, p. 545–555, 2015.

ROY, P. P.; RAYAR, F.; RAMEL, J. Y. Word spotting in historical documents using primitive codebook and dynamic programming. *Image and Vision Computing*, v. 44, p. 15–28, 2015.

RUSIÑOL, M. et al. Efficient segmentation-free keyword spotting in historical document collections. *Pattern Recognition*, v. 48, p. 545–555, 2015.

RUSINOL, M.; LLADóS, J. Efficient logo retrieval through hashing shape context descriptors. In: *Proceedings of the 9th IAPR International Workshop on Document Analysis Systems*. [S.l.: s.n.], 2010. p. 215–222.

RUSSAKOVSKY, O. et al. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, v. 115, n. 3, p. 211–252, 2015.

SAIKRISHNA, T. V. et al. A novel image retrieval method using segmentation and color moments. *Advanced Computing: An International Journal (ACIJ)*, v. 3, n. 1, 2012.

SANDE, K. E. A. van de et al. Segmentation as selective search for object recognition. In: *2011 International Conference on Computer Vision*. [S.l.: s.n.], 2011. p. 1879–1886. ISSN 2380-7504.

SCHROFF DMITRY KALENICHENKO, J. P. F. Facenet: A unified embedding for face recognition and clustering. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. [S.l.: s.n.], 2015. p. 815–823.

SERMANET, P. et al. Overfeat: Integrated recognition, localization and detection using convolutional networks. In: *International Conference on Learning Representations*. [S.l.: s.n.], 2014.

SHIRDHONKAR, M. S.; KOKARE, M. B. Writer based handwritten document image retrieval using contourlet transform. In: _____. *Advances in Digital Image Processing and Information Technology: First International Conference on Digital Image Processing and Pattern Recognition, DPPR 2011, Tirunelveli, Tamil Nadu,* *India, September 23-25, 2011. Proceedings*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011. p. 108–117. ISBN 978-3-642-24055-3.

SIMONYAN, K.; ZISSERMAN., A. Very deep convolutional networks for large-scale image recognition. In: *5th International Conference on Learning Representations*. [S.l.: s.n.], 2015.

SIVIC; ZISSERMAN. Video google: a text retrieval approach to object matching in videos. In: *Proceedings Ninth IEEE International Conference on Computer Vision*. [S.l.: s.n.], 2003. p. 1470–1477 vol.2.

SZEGEDY, C. et al. Going depper with convolutions. In: *Computer Vision and Pattern Recognition 2015*. [S.l.: s.n.], 2015.

TABIBIAN, S.; AKBARI, A.; NASERSHARIF, B. A fast hierarchical search algorithm for discriminative keyword spotting. *Information Sciences*, v. 336, p. 45–59, 2016.

TAN, P.-N.; STEINBACH, M.; KUMAR, V. *Introduction to Data Mining, (First Edition)*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2005. ISBN 0321321367.

TANENBAUM, H. B. A. S. *Modern Operating Systems*. 4th. ed. [S.l.]: Prentice Hal, 2014.

TERBOVEN, C. et al. Shared-memory parallelization for content-based image retrieval. In: SPRINGER (Ed.). *Computer vision approaches to medical image analysis : second international ECCV workshop*. Austria: [s.n.], 2006.

THOMAS, S. et al. A deep hmm model for multiple keywords spotting in handwritten documents. *Pattern Analysis and Applications*, v. 18, n. 4, p. 1003–1015, 2015.

THUY, Q. D. T. et al. An efficient semantic – related image retrieval method. *Expert Systems with Applications*, v. 72, p. 30 – 41, 2017.

TIELEMAN, T.; HINTON, G. *Lecture 6.5—RmsProp: Divide the gradient by a running average of its recent magnitude.* 2012. COURSERA: Neural Networks for Machine Learning.

TOLIAS, G.; SICRE, R.; JÉGOU, H. Particular object retrieval with integral max-pooling of CNN activations. In: *ICLR 2015.* [S.l.: s.n.], 2015.

TORRES, R. da S.; FALCÃO, A. X. Content-based image retrieval: Theory and applications. *RITA*, v. 13, n. 2, p. 161–185, 2006.

TORREY, L.; SHAVLIK, J. *Transfer Learning.* 2009. GI Global.

TOSELLIA, A. H. et al. Hmm word graph based keyword spotting in handwritten document images. *Information Sciences*, p. 497–518, 2016.

TRANOUEZ, P. et al. Docexplore overcoming cultural and physical barriers to access ancient documents. *ACM Document Engineering*, 2012.

TUNGKASTHAN, A.; PREMCHAISWADI, W. A parallel processing framework using mapreduce for content-based image retrieval. In: *2013 Eleventh International Conference on ICT and Knowledge Engineering.* [S.l.: s.n.], 2013. p. 1–6. ISSN 2157-099X.

ÚBEDA, I. et al. Pattern spotting in historical documents using convolutional models. In: *arXiv:1906.08580.* [S.l.: s.n.], 2019.

UIJLINGS, J. R. R. et al. Selective search for object recognition. *International Journal of Computer Vision*, v. 104, n. 2, p. 154–171, 2013. ISSN 1573-1405.

VINCENT, P. et al. *Extracting and Composing Robust Features with Denoising Autoencoders.* [S.l.], fev. 2008.

WAN, J. et al. Deep learning for content-based image retrieval: A comprehensive study. In: *Proceedings of the 22Nd ACM International Conference on Multimedia.* New York, NY, USA: ACM, 2014. (MM '14), p. 157–166. ISBN 978-1-4503-3063-3. Disponível em: <http://doi.acm.org/10.1145/2647868.2654948>.

WANG, X. Deep learning in object recognition, detection, and segmentation. v. 8, n. 4, p. 217–382, 2016. ISSN 1932-8346. Disponível em: <http://dx.doi.org/10-.1561/2000000071>.

WEISS, Y.; TORRALBA, A.; FERGUS, R. Spectral hashing. In: KOLLER, D. et al. (Ed.). *Advances in Neural Information Processing Systems 21*. [S.l.]: Curran Associates, Inc., 2009. p. 1753–1760.

WU, H. et al. Face recognition based on convolution siamese networks. In: *2017 10th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*. [S.l.: s.n.], 2017. p. 1–5.

WU, S. et al. Deep binary codes for large scale image retrieval. *Neurocomputing*, 2017.

XIA, R. et al. Supervised hashing for image retrieval via image representation learning. In: *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*. [S.l.]: AAAI Press, 2014. p. 2156–2162.

XU, Y. et al. Large-scale image retrieval with supervised sparse hashing. *Neurocomputing*, v. 229, p. 45 – 53, 2017.

YARLAGADDA, P. et al. Recognition and analysis of objects in medieval images. In: KOCH, R.; HUANG, F. (Ed.). *Computer Vision – ACCV 2010 Workshops: ACCV 2010 International Workshops, Queenstown, New Zealand, November 8-9, 2010, Revised Selected Papers, Part II*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011. p. 296–305.

YOSINSKI, J. et al. How transferable are features in deep neural networks? In: GHAHRAMANI, Z. et al. (Ed.). *Advances in Neural Information Processing Systems 27*. [S.l.]: Curran Associates, Inc., 2014.

YUE-HEI, J.; YANG, N. F.; DAVIS, L. S. Exploiting local features from deep networks for image retrieval. In: *2015 IEEE Conference on Computer Vision and Pattern Recognition*. [S.l.: s.n.], 2015. p. 53–61.

ZEILER, M.; FERGUS, R. Stochastic pooling for regularization of deep convolutional neural networks. In: *Proceedings of the International Conference on Learning Representation (ICLR)*. [S.l.: s.n.], 2013.

ZEILER, M. D. Adadelta: An adaptive learning rate method. *CoRR*, abs/1212.5701, 2012.

ZEILER, M. D.; FERGUS, R. Visualizing and understanding convolutional networks. In: FLEET, D. et al. (Ed.). *Computer Vision ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, PartI*. Cham: Springer International Publishing, 2014. p. 818–833. ISBN 978-3-319-10590-1.

ZHANG, Y. et al. Generating discriminative object proposals via submodular ranking. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. [S.l.: s.n.], 2016.

ZHU, G.; DOERMANN, D. Automatic document logo detection. In: *In Proc. 9th Int. Conf. Document Analysis and Recognition (ICDAR 2007)*. [S.l.: s.n.], 2007. p. 864–868.

ZHU, G. et al. Multi-scale structural saliency for signature detection. In: *In Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR 2007)*. [S.l.: s.n.], 2007. p. 1–8.

ZHU, Q.; KEOGH, E. Mother fugger: Mining historical manuscripts with local color patches. In: *2010 IEEE International Conference on Data Mining*. [S.l.: s.n.], 2010. p. 699–708. ISSN 1550-4786.

ZHUANG, F. et al. Supervised representation learning: Transfer learning with deep autoencoders. In: *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*. [S.l.: s.n.], 2015. p. 4119–4125.

ZITNICK, C. L.; DOLLÁR, P. Edge boxes: Locating object proposals from edges. In: *ECCV*. [S.l.: s.n.], 2014.

Appendices

Appendix A

Tobacco800 dataset - additional experiments

We decide to evaluate the proposed SCNN based on Alexnet with other popular architecture in the state-of-the-art. Table A.1 shows a comparison using SCNN implemented with Alexnet and Googlenet. We implement and train a SCNN Googlenet using the same parameters of SCNN Alexnet. However, the Googlenet architecture is bigger than Alexnet, then we cannot use the batch size = 128 as used in Alexnet. We needed to define a batch size = 16 because of the memory available. The standard Googlenet has 1024-dimensional feature map, but the additional layer with reduced feature map was also included to compare results.

Table A.1: Comparison using SCNN AlexNet and Googlenet with top-k ranking

SCNN	**Feature map**	top-k				
		5	**10**	**25**	**50**	**100**
Alexnet	1,024	0.577	0.480	0.342	0.250	0.170
	512	0.744	0.640	0.486	0.358	0.220
	256	**0.746**	0.654	0.503	0.368	0.227
	128	0.738	**0.655**	**0.506**	**0.373**	**0.229**
Googlenet	1,024	**0.585**	0.453	0.300	0.210	0.141
	512	0.360	0.284	0.201	0.152	0.111
	256	0.254	0.210	0.160	0.125	0.096
	128	0.299	0.245	0.168	0.128	0.098

According to the experiments presented in Table A.1, the performance of

Top-5 is only 1.3% better using standard SCNN Googlenet than AlexNet with feature maps of 1024. Using other values of the feature map, the Alexnet architecture returns better results and it is much faster.

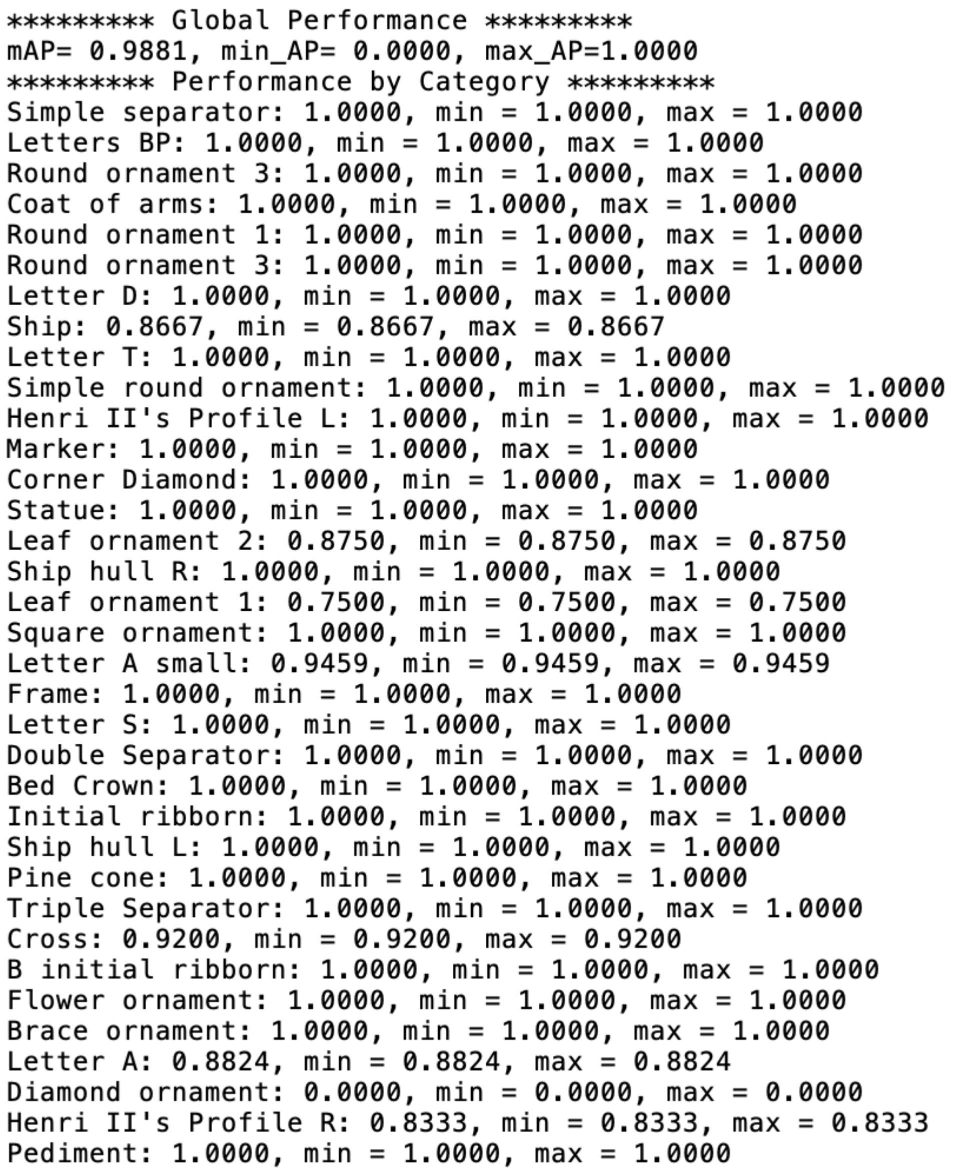
Appendix B

DocExplore - additional experiments

B.1 DocExplore Template Matching

In Chapter 4, experimentation showed the performance for IR and PS using DocExplore dataset and the online system *Rouen*. In this appendix, we show the results verified at the online system with our ground truth generated by TM. Figure B.1 presents the results evaluated at *Rouen* system using a TM for PS. These experiments were performed to evaluate the quality of our TM and use it as an approximate ground truth. Using this ground truth, we could evaluate the values of Table 4.17 presented in Section 4.2.

IIn this appendix, we have proposed additional experiments for PS to validate our TM with SCNN approach. However, we do not have the original ground truth or junk list to observe the results and compare the difference.



```
********* Global Performance *********
mAP= 0.9881, min_AP= 0.0000, max_AP=1.0000
********* Performance by Category *********
Simple separator: 1.0000, min = 1.0000, max = 1.0000
Letters BP: 1.0000, min = 1.0000, max = 1.0000
Round ornament 3: 1.0000, min = 1.0000, max = 1.0000
Coat of arms: 1.0000, min = 1.0000, max = 1.0000
Round ornament 1: 1.0000, min = 1.0000, max = 1.0000
Round ornament 3: 1.0000, min = 1.0000, max = 1.0000
Letter D: 1.0000, min = 1.0000, max = 1.0000
Ship: 0.8667, min = 0.8667, max = 0.8667
Letter T: 1.0000, min = 1.0000, max = 1.0000
Simple round ornament: 1.0000, min = 1.0000, max = 1.0000
Henri II's Profile L: 1.0000, min = 1.0000, max = 1.0000
Marker: 1.0000, min = 1.0000, max = 1.0000
Corner Diamond: 1.0000, min = 1.0000, max = 1.0000
Statue: 1.0000, min = 1.0000, max = 1.0000
Leaf ornament 2: 0.8750, min = 0.8750, max = 0.8750
Ship hull R: 1.0000, min = 1.0000, max = 1.0000
Leaf ornament 1: 0.7500, min = 0.7500, max = 0.7500
Square ornament: 1.0000, min = 1.0000, max = 1.0000
Letter A small: 0.9459, min = 0.9459, max = 0.9459
Frame: 1.0000, min = 1.0000, max = 1.0000
Letter S: 1.0000, min = 1.0000, max = 1.0000
Double Separator: 1.0000, min = 1.0000, max = 1.0000
Bed Crown: 1.0000, min = 1.0000, max = 1.0000
Initial ribborn: 1.0000, min = 1.0000, max = 1.0000
Ship hull L: 1.0000, min = 1.0000, max = 1.0000
Pine cone: 1.0000, min = 1.0000, max = 1.0000
Triple Separator: 1.0000, min = 1.0000, max = 1.0000
Cross: 0.9200, min = 0.9200, max = 0.9200
B initial ribborn: 1.0000, min = 1.0000, max = 1.0000
Flower ornament: 1.0000, min = 1.0000, max = 1.0000
Brace ornament: 1.0000, min = 1.0000, max = 1.0000
Letter A: 0.8824, min = 0.8824, max = 0.8824
Diamond ornament: 0.0000, min = 0.0000, max = 0.0000
Henri II's Profile R: 0.8333, min = 0.8333, max = 0.8333
Pediment: 1.0000, min = 1.0000, max = 1.0000
```

Figure B.1: PS results using TM at Rouen system

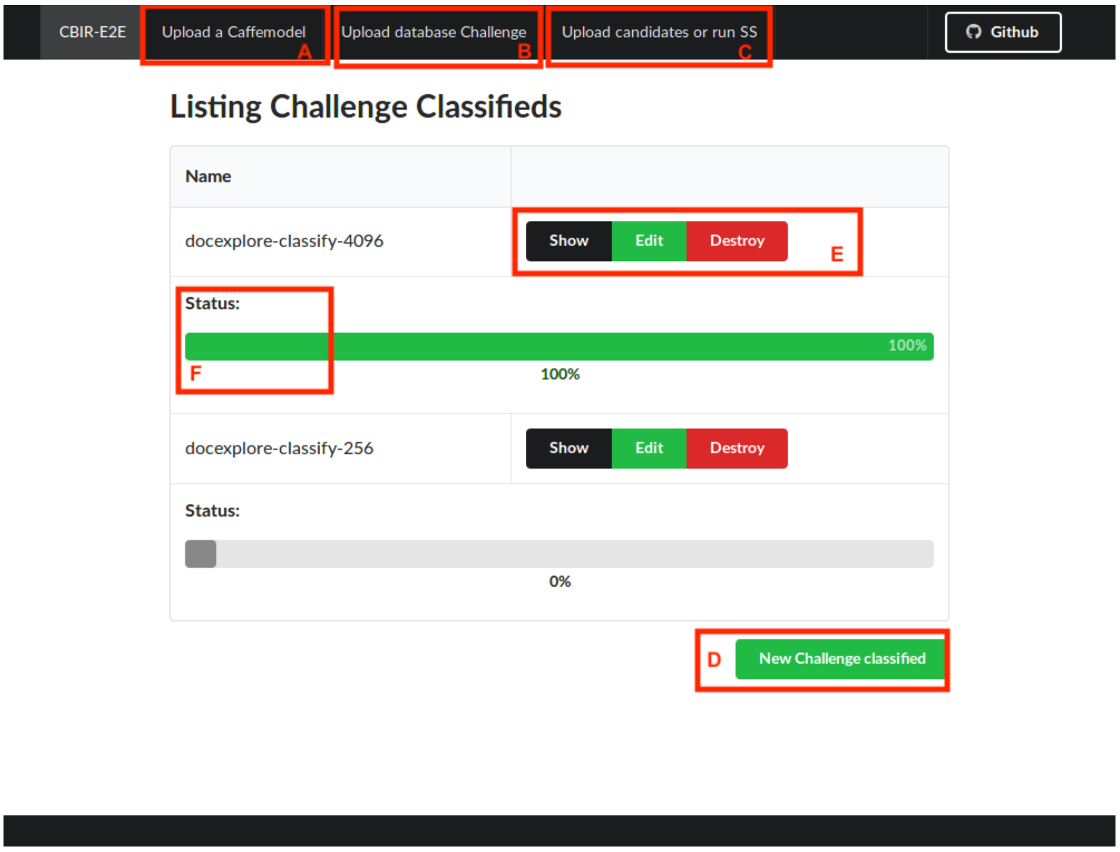
Appendix C

IR and PS system - E2E

In addition to this work, based on the methods investigated previously, we decided to implement an end-to-end system that contains all the necessary steps to evaluate a data set with CBIR approaches. Figure C.1 shows the starting frame. The system was implemented with the ruby on rails backend framework and the frontend framework uses semantic-ui. The system runs all previously deployed scripts, having several options to choose from. Figure C.1) shows the options:

- (A) the possibility to send the own file of weights of a network (.caffemodel file), or the use one of our models;
- (B) The user can choose even the dataset already available or submit a dataset with pages and queries, and, if available, the ground truth.
- (C) Include parameters customized of the SS algorithm or choosing the default.
- (D) A new challenge can be created.
- (E) shows allow to explore details, edit or destroy the challenge.
- The bar progress in (F) shows when the tasks finished.

Figures C.2 to C.11 shows more details about the implemented system and the possibilities to use it. The examples contain some images of DocExplore dataset to validate the system.

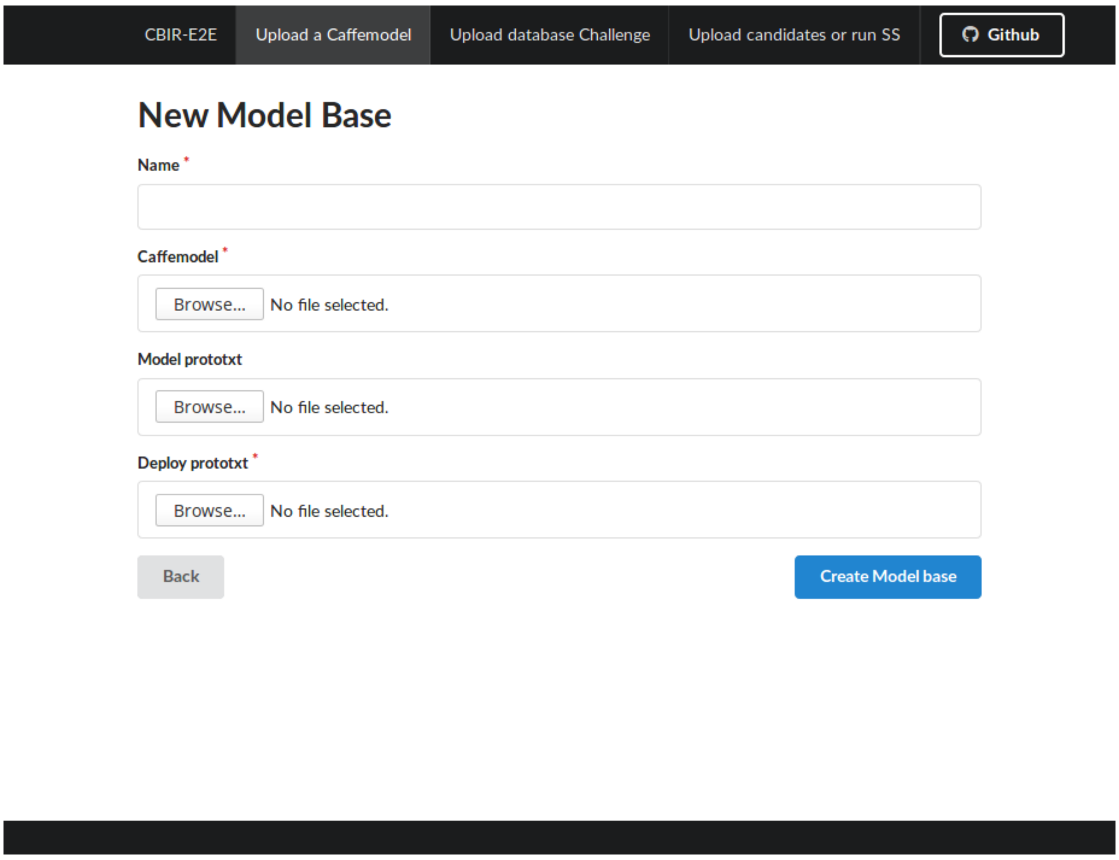


Listing Challenge Classifieds

Name	
docexplore-classify-4096	Show Edit Destroy
Status: 100%	
docexplore-classify-256	Show Edit Destroy
Status: 0%	

Figure C.1: IR and PS system - initial frame with two dataset examples already created

The use of this system can help the researchers to evaluate their datasets, the feature extractor implemented or candidates generate by some algorithm. In the end, the results of IR and PS can be easily explored in the online system.



New Model Base

Name *

Caffemodel *

Browse... No file selected.

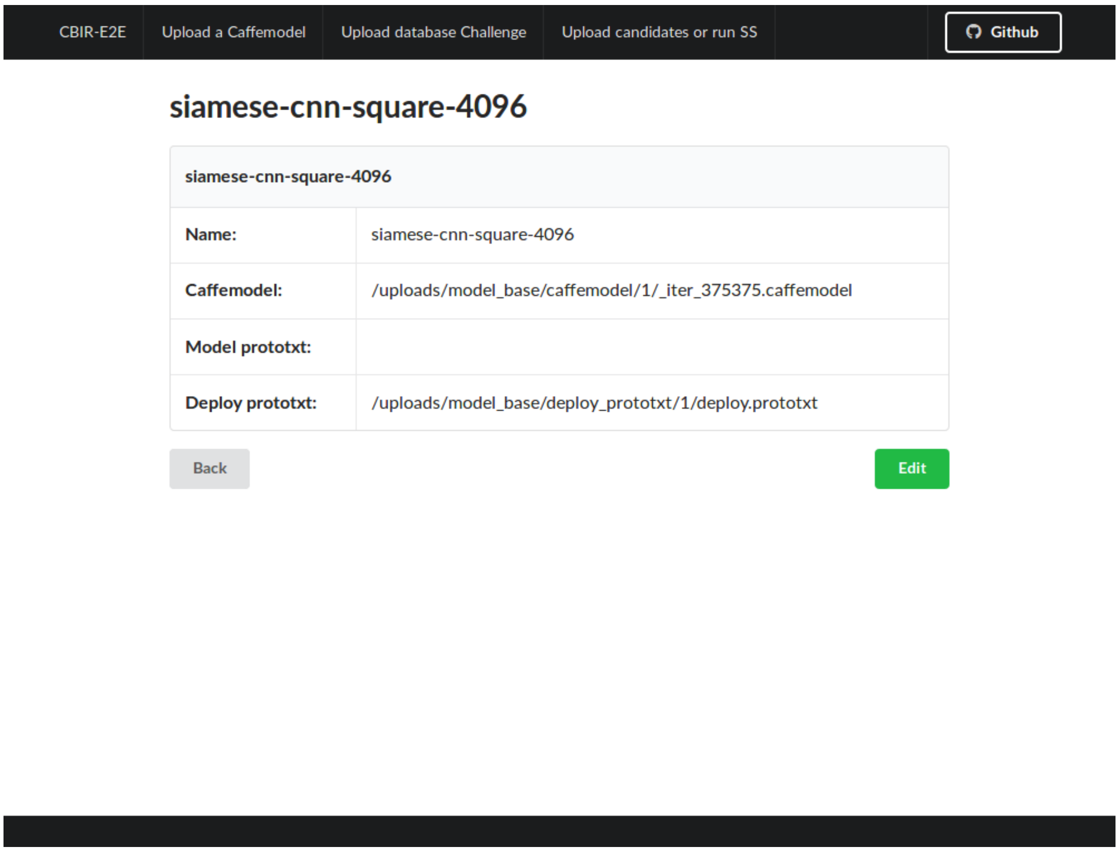
Model prototxt

Browse... No file selected.

Deploy prototxt *

Browse... No file selected.

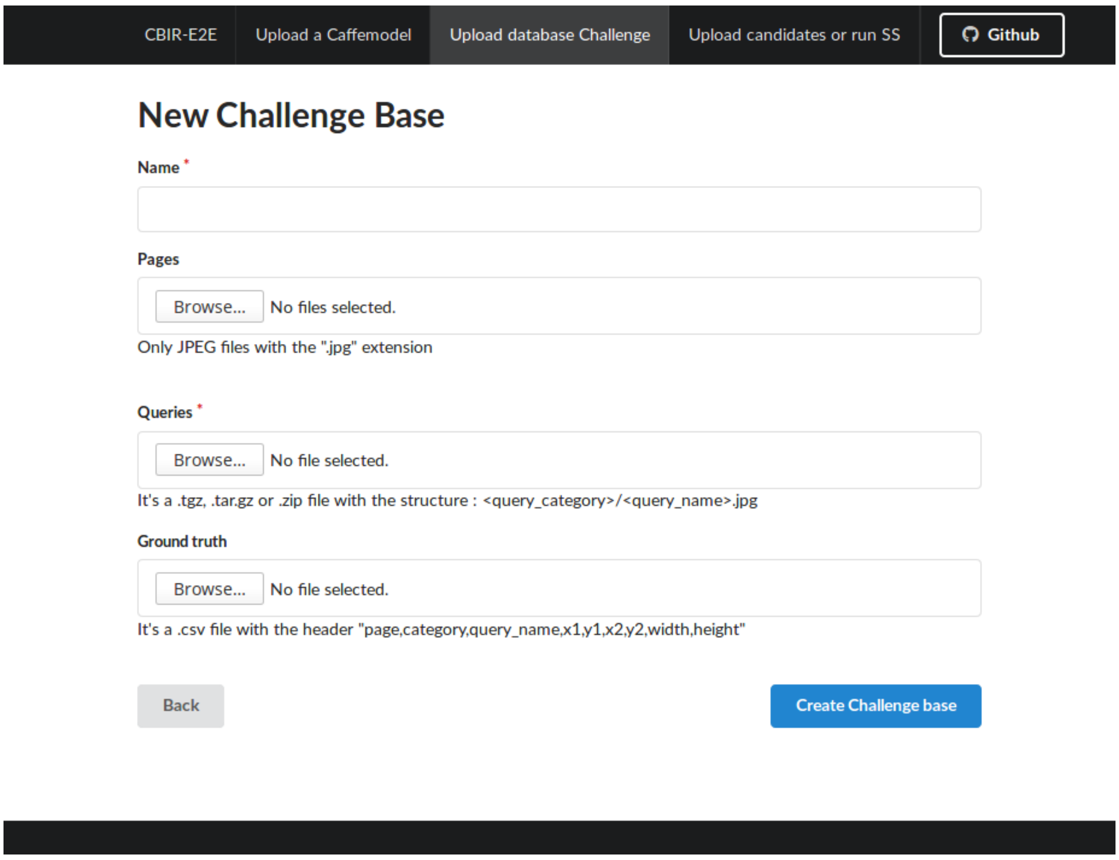
Figure C.2: IR and PS system - (A) Upload a caffemodel



siamese-cnn-square-4096

siamese-cnn-square-4096	
Name:	siamese-cnn-square-4096
Caffemodel:	/uploads/model_base/caffemodel/1/_iter_375375.caffemodel
Model prototxt:	
Deploy prototxt:	/uploads/model_base/deploy_prototxt/1/deploy.prototxt

Figure C.3: IR and PS system - (A) Upload a caffemodel - explore details



New Challenge Base

Name *

Pages

Browse... No files selected.

Only JPEG files with the ".jpg" extension

Queries *

Browse... No file selected.

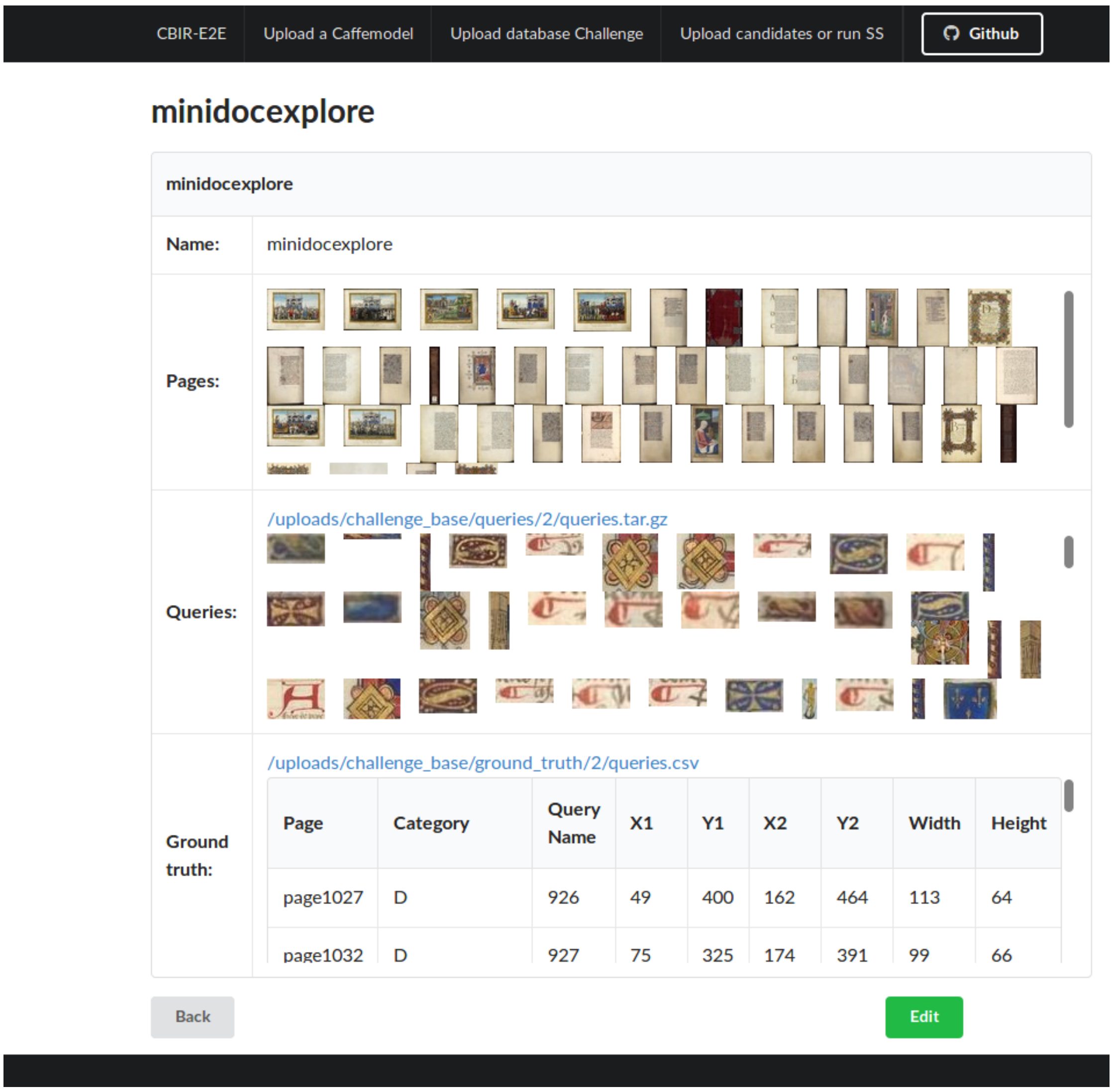
It's a .tgz, .tar.gz or .zip file with the structure : <query_category>/<query_name>.jpg

Ground truth

Browse... No file selected.

It's a .csv file with the header "page,category,query_name,x1,y1,x2,y2,width,height"

Figure C.4: IR and PS system - (B) Upload database challenge



minidocexplore

minidocexplore

Name: minidocexplore

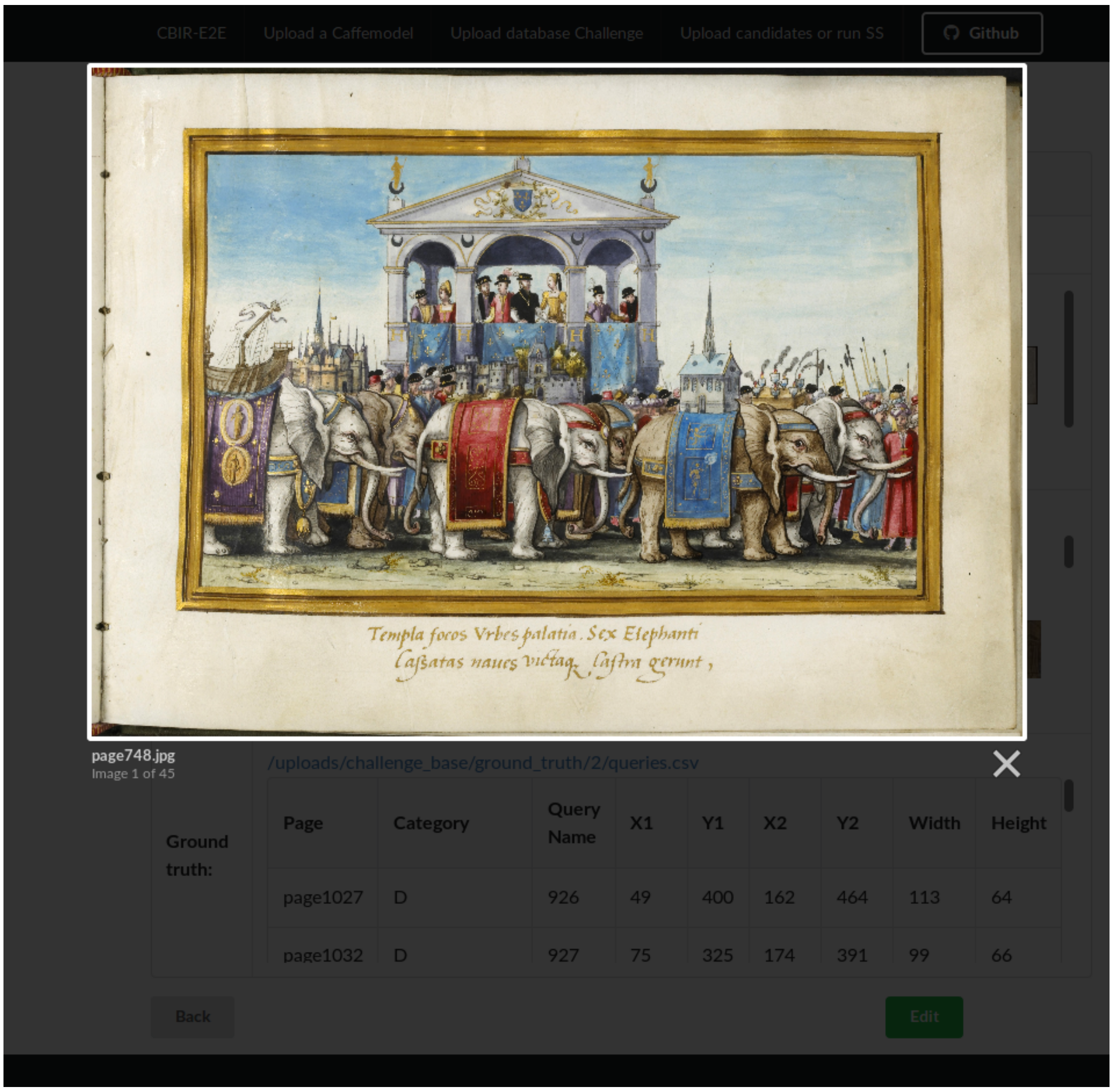
Pages:

Queries: /uploads/challenge_base/queries/2/queries.tar.gz

Ground truth: /uploads/challenge_base/ground_truth/2/queries.csv

Page	Category	Query Name	X1	Y1	X2	Y2	Width	Height
page1027	D	926	49	400	162	464	113	64
page1032	D	927	75	325	174	391	99	66

Figure C.5: IR and PS system - (B) Upload database challenge - details

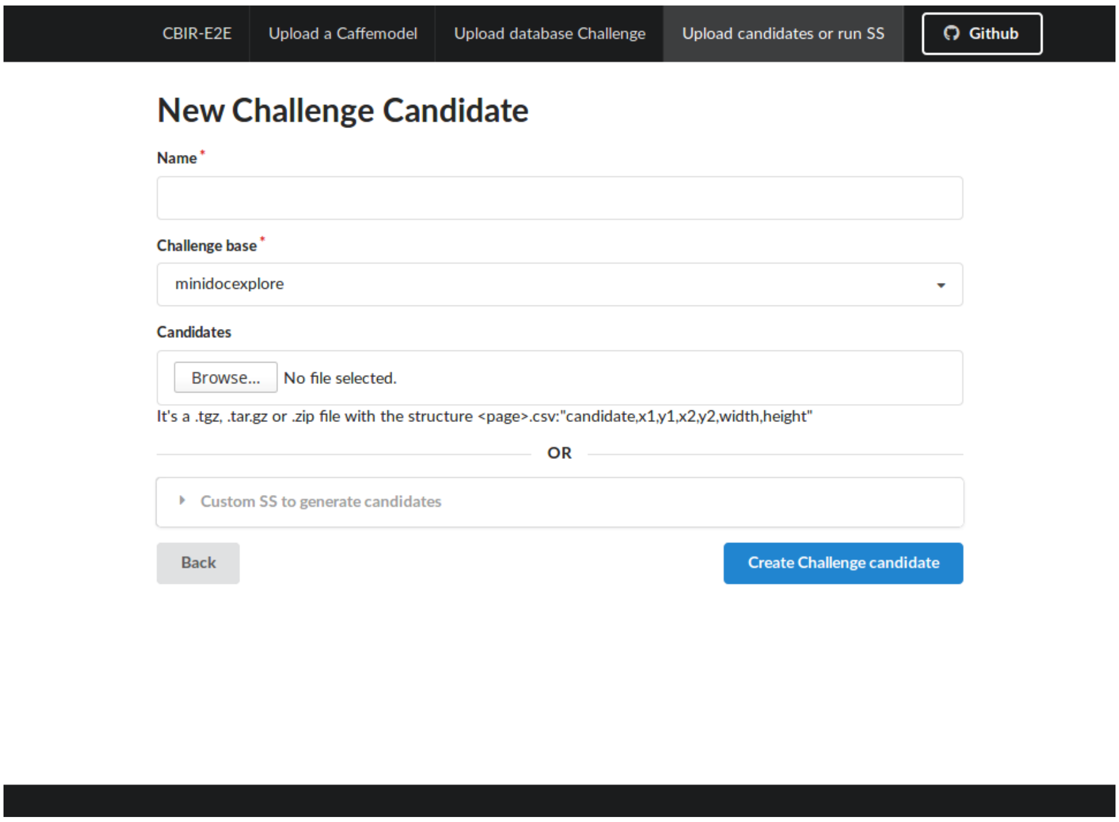

CBIR-E2E
Upload a Caffemodel
Upload database Challenge
Upload candidates or run SS
Github
Templa focos Vrbes palatia. Sex Elephanti
Caſsatas naues victaq. caſtra gerunt,
page748.jpg
Image 1 of 45
Back
Edit

/uploads/challenge_base/ground_truth/2/queries.csv

Ground truth:

Page	Category	Query Name	X1	Y1	X2	Y2	Width	Height
page1027	D	926	49	400	162	464	113	64
page1032	D	927	75	325	174	391	99	66

Figure C.6: IR and PS system - (B) Upload database challenge - explore a page/-query


CBIR-E2E
Upload a Caffemodel
Upload database Challenge
Upload candidates or run SS
Github

New Challenge Candidate

Name *

Challenge base *

minidocexplore

Candidates

Browse... No file selected.

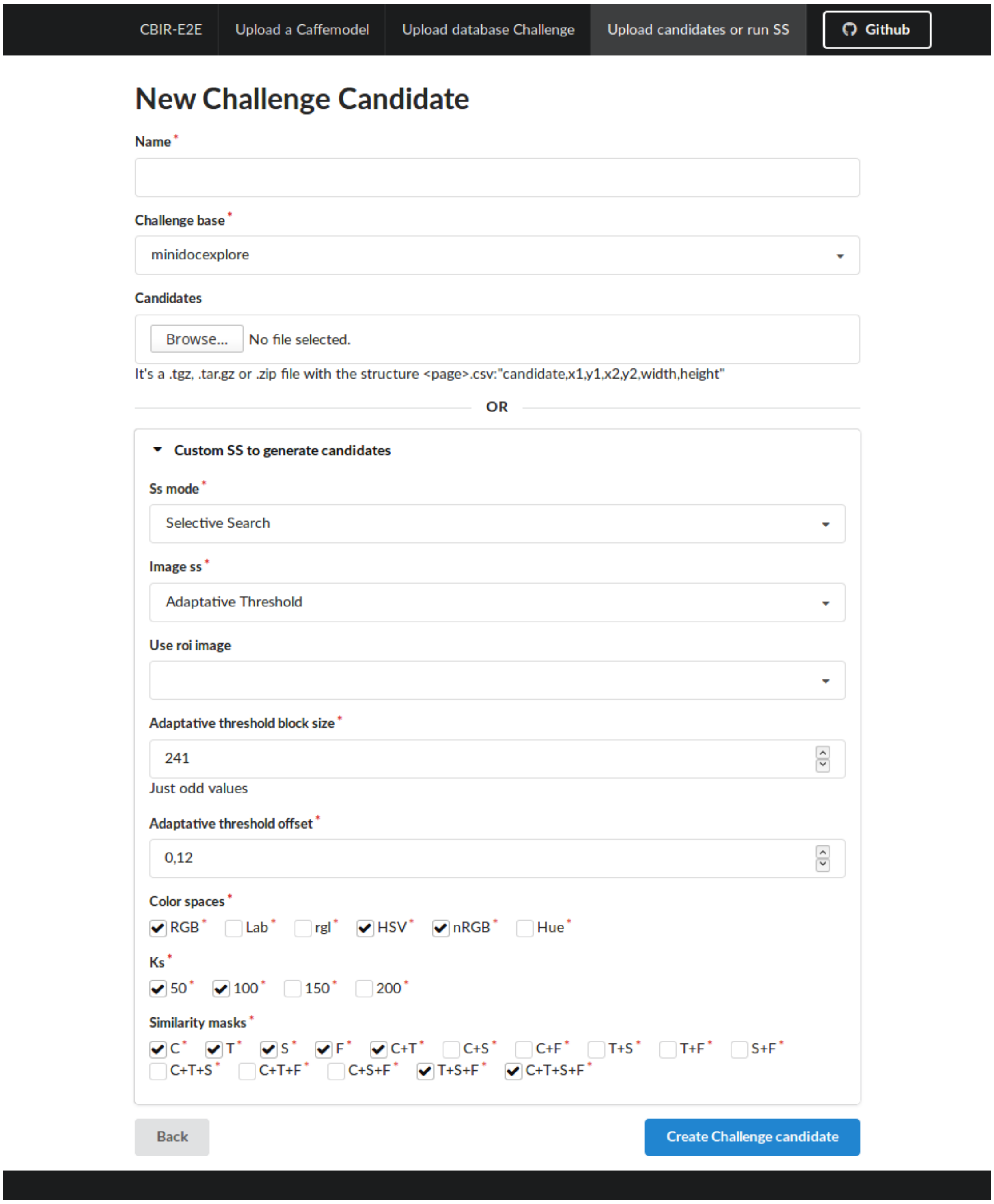
It's a .tgz, .tar.gz or .zip file with the structure <page>.csv:"candidate,x1,y1,x2,y2,width,height"

OR

Custom SS to generate candidates

Back
Create Challenge candidate

Figure C.7: IR and PS system - (C) Upload candidates


CBIR-E2E
Upload a Caffemodel
Upload database Challenge
Upload candidates or run SS
Github
Browse...
Back
Create Challenge candidate

New Challenge Candidate

Name*

Challenge base*
minidocexplore

Candidates
No file selected.
It's a .tgz, .tar.gz or .zip file with the structure <page>.csv:"candidate,x1,y1,x2,y2,width,height"

OR

Custom SS to generate candidates

Ss mode*
Selective Search

Image ss*
Adaptative Threshold

Use roi image

Adaptative threshold block size*
241
Just odd values

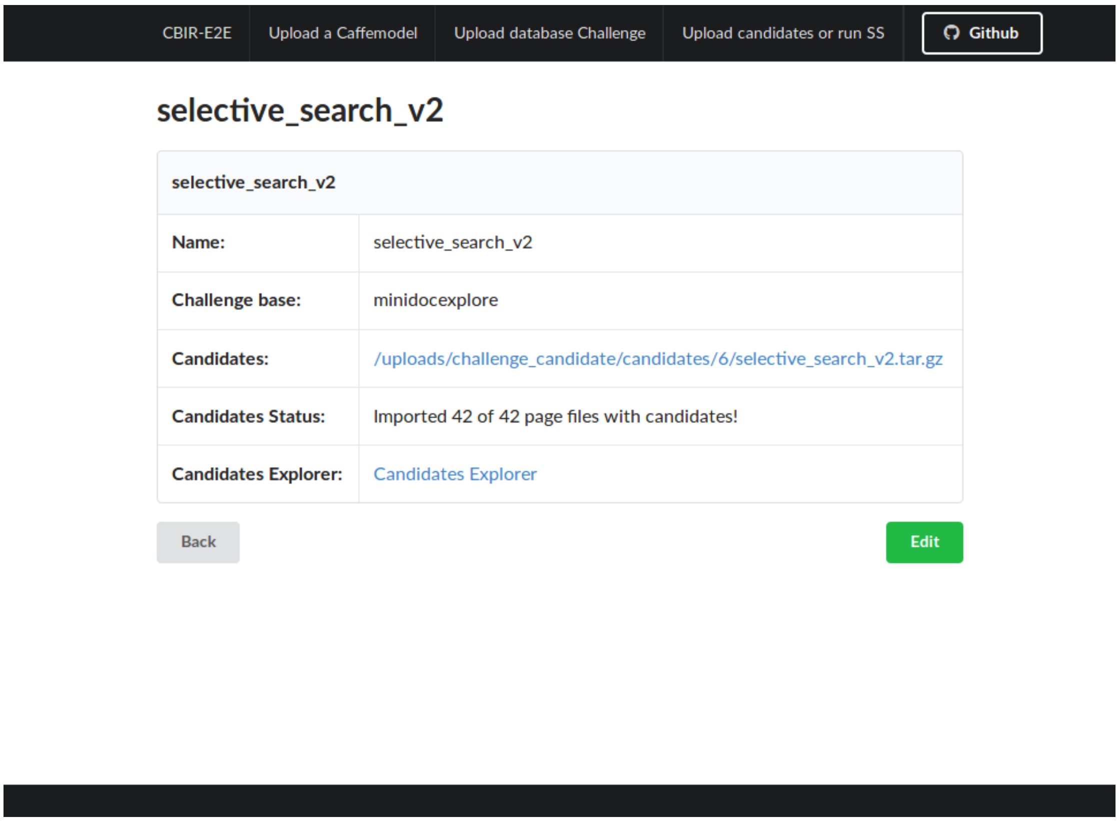
Adaptative threshold offset*
0,12

Color spaces*
[✓ RGB]* Lab* rgl* [✓ HSV]* [✓ nRGB]* Hue*

Ks*
[✓ 50]* [✓ 100]* 150* 200*

Similarity masks*
[✓ C]* [✓ T]* [✓ S]* [✓ F]* [✓ C+T]* C+S* C+F* T+S* T+F* S+F*
C+T+S* C+T+F* C+S+F* [✓ T+S+F]* [✓ C+T+S+F]*

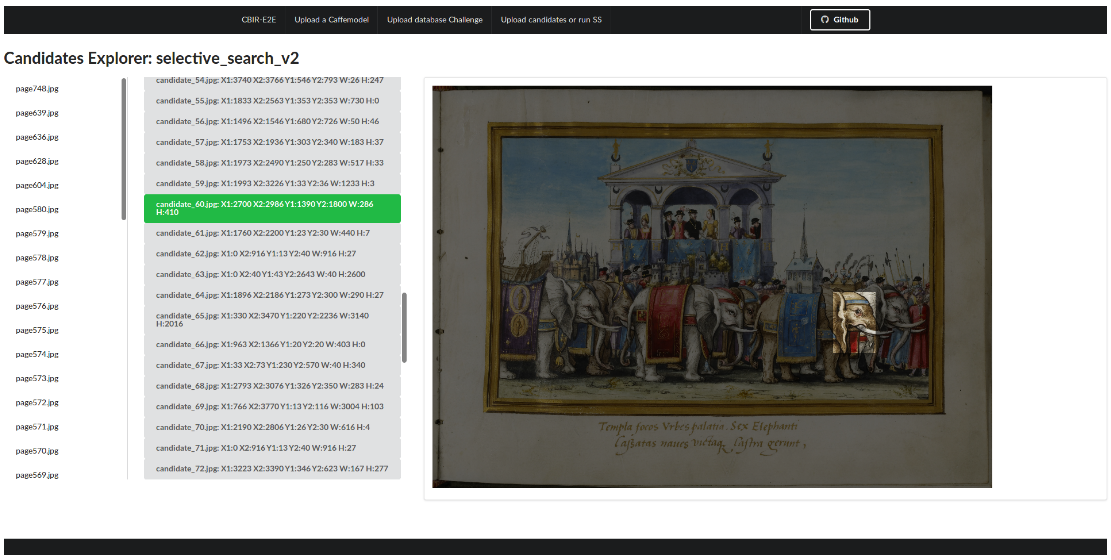
Figure C.8: IR and PS system - (C) Run SS



selective_search_v2

selective_search_v2	
Name:	selective_search_v2
Challenge base:	minidocexplore
Candidates:	/uploads/challenge_candidate/candidates/6/selective_search_v2.tar.gz
Candidates Status:	Imported 42 of 42 page files with candidates!
Candidates Explorer:	Candidates Explorer

Figure C.9: IR and PS system - (C) Candidates details

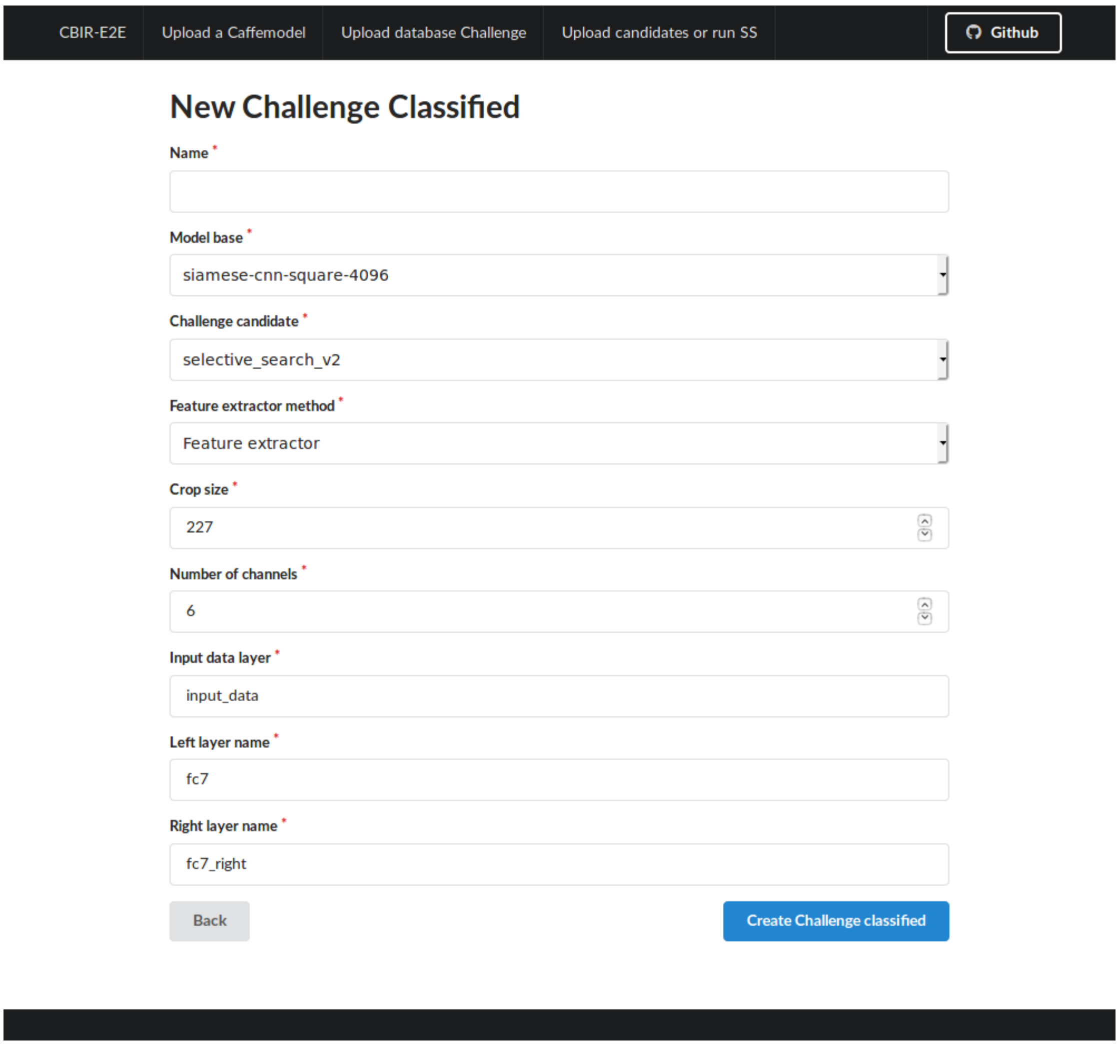


Candidates Explorer: selective_search_v2

page748.jpg
page639.jpg
page636.jpg
page628.jpg
page604.jpg
page580.jpg
page579.jpg
page578.jpg
page577.jpg
page576.jpg
page575.jpg
page574.jpg
page573.jpg
page572.jpg
page571.jpg
page570.jpg
page569.jpg

```
candidate_54.jpg: X1:3740 X2:3766 Y1:546 Y2:793 W:26 H:247
candidate_55.jpg: X1:1833 X2:2563 Y1:353 Y2:353 W:730 H:0
candidate_56.jpg: X1:1496 X2:1546 Y1:680 Y2:726 W:50 H:46
candidate_57.jpg: X1:1753 X2:1936 Y1:303 Y2:340 W:183 H:37
candidate_58.jpg: X1:1973 X2:2490 Y1:250 Y2:283 W:517 H:33
candidate_59.jpg: X1:1993 X2:3226 Y1:33 Y2:36 W:1233 H:3
candidate_60.jpg: X1:2700 X2:2986 Y1:1390 Y2:1800 W:286
H:410
candidate_61.jpg: X1:1760 X2:2200 Y1:23 Y2:30 W:440 H:7
candidate_62.jpg: X1:0 X2:916 Y1:13 Y2:40 W:916 H:27
candidate_63.jpg: X1:0 X2:40 Y1:43 Y2:2643 W:40 H:2600
candidate_64.jpg: X1:1896 X2:2186 Y1:273 Y2:300 W:290 H:27
candidate_65.jpg: X1:330 X2:3470 Y1:220 Y2:2236 W:3140
H:2016
candidate_66.jpg: X1:963 X2:1366 Y1:20 Y2:20 W:403 H:0
candidate_67.jpg: X1:33 X2:73 Y1:230 Y2:570 W:40 H:340
candidate_68.jpg: X1:2793 X2:3076 Y1:326 Y2:350 W:283 H:24
candidate_69.jpg: X1:766 X2:3770 Y1:13 Y2:116 W:3004 H:103
candidate_70.jpg: X1:2190 X2:2806 Y1:26 Y2:30 W:616 H:4
candidate_71.jpg: X1:0 X2:916 Y1:13 Y2:40 W:916 H:27
candidate_72.jpg: X1:3223 X2:3390 Y1:346 Y2:623 W:167 H:277
```

Figure C.10: IR and PS system - (C) Explore candidates


CBIR-E2E
Upload a Caffemodel
Upload database Challenge
Upload candidates or run SS
Github

New Challenge Classified

Name *

Model base *
siamese-cnn-square-4096

Challenge candidate *
selective_search_v2

Feature extractor method *
Feature extractor

Crop size *
227

Number of channels *
6

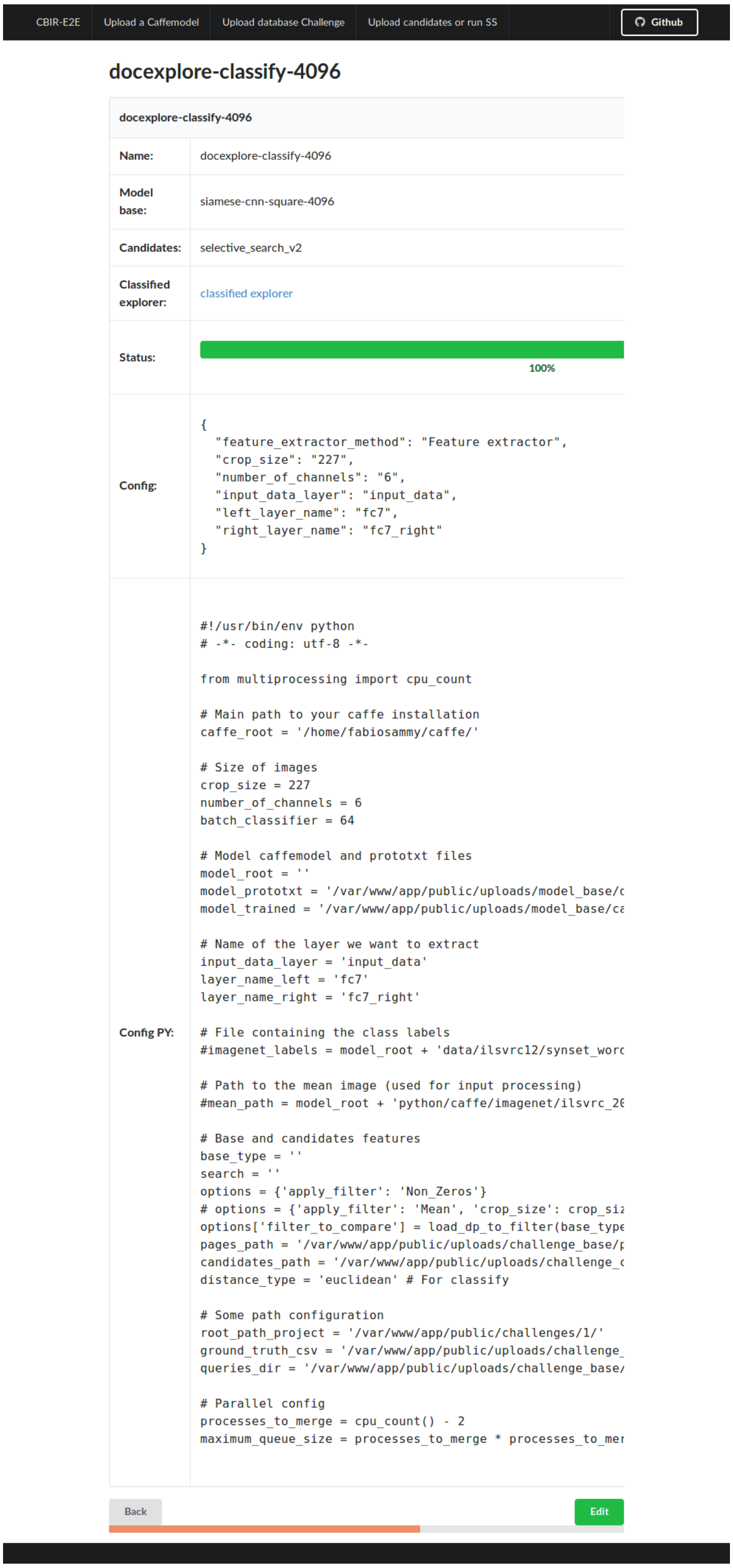
Input data layer *
input_data

Left layer name *
fc7

Right layer name *
fc7_right

Back
Create Challenge classified

Figure C.11: IR and PS system - (D) New challenge classified



docexplore-classify-4096

docexplore-classify-4096

Name:	docexplore-classify-4096
Model base:	siamese-cnn-square-4096
Candidates:	selective_search_v2
Classified explorer:	classified explorer
Status:	100%

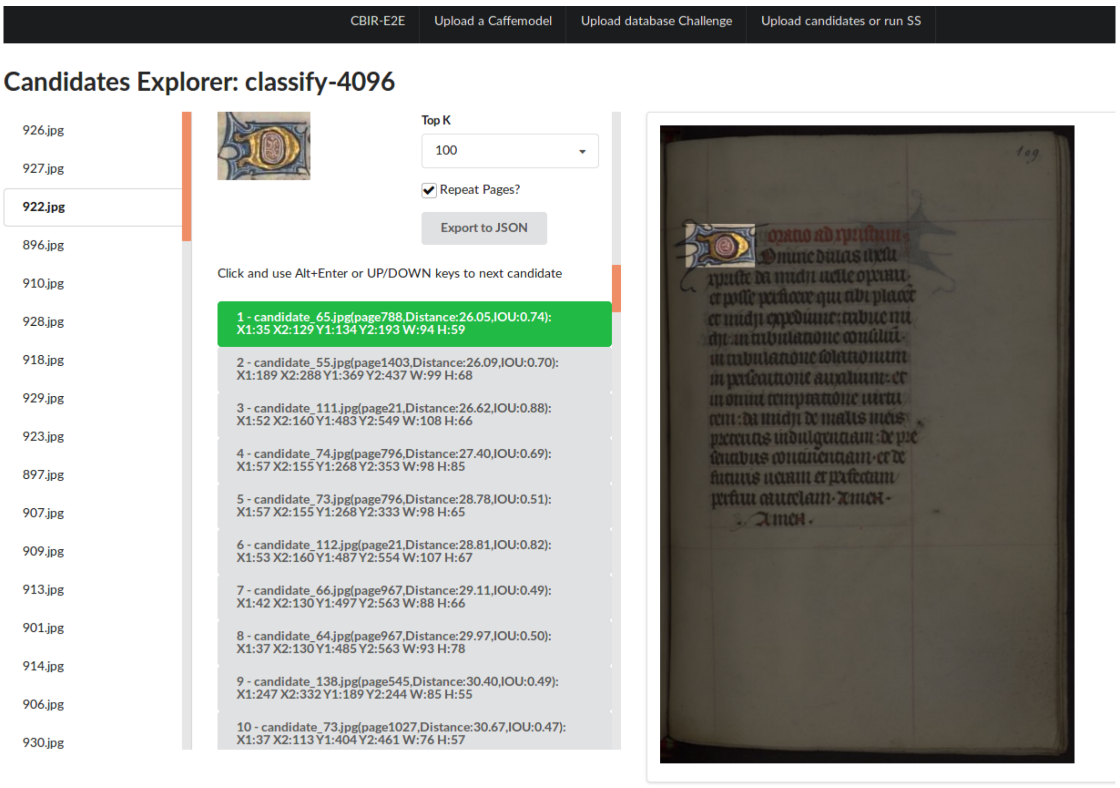
Config:

```
{
  "feature_extractor_method": "Feature extractor",
  "crop_size": "227",
  "number_of_channels": "6",
  "input_data_layer": "input_data",
  "left_layer_name": "fc7",
  "right_layer_name": "fc7_right"
}
```

Config PY:

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-

from multiprocessing import cpu_count

# Main path to your caffe installation
caffe_root = '/home/fabiosammy/caffe/'

# Size of images
crop_size = 227
number_of_channels = 6
batch_classifier = 64

# Model caffemodel and prototxt files
model_root = ''
model_prototxt = '/var/www/app/public/uploads/model_base/c
model_trained = '/var/www/app/public/uploads/model_base/ca

# Name of the layer we want to extract
input_data_layer = 'input_data'
layer_name_left = 'fc7'
layer_name_right = 'fc7_right'

# File containing the class labels
#imagenet_labels = model_root + 'data/ilsvrc12/synset_worc

# Path to the mean image (used for input processing)
#mean_path = model_root + 'python/caffe/imagenet/ilsvrc_20

# Base and candidates features
base_type = ''
search = ''
options = {'apply_filter': 'Non_Zeros'}
# options = {'apply_filter': 'Mean', 'crop_size': crop_siz
options['filter_to_compare'] = load_dp_to_filter(base_type
pages_path = '/var/www/app/public/uploads/challenge_base/p
candidates_path = '/var/www/app/public/uploads/challenge_c
distance_type = 'euclidean' # For classify

# Some path configuration
root_path_project = '/var/www/app/public/challenges/1/'
ground_truth_csv = '/var/www/app/public/uploads/challenge_
queries_dir = '/var/www/app/public/uploads/challenge_base/

# Parallel config
processes_to_merge = cpu_count() - 2
maximum_queue_size = processes_to_merge * processes_to_mer
```

Figure C.12: IR and PS system - (E) New challenge classified - explore classification details



Candidates Explorer: classify-4096

926.jpg
927.jpg
922.jpg
896.jpg
910.jpg
928.jpg
918.jpg
929.jpg
923.jpg
897.jpg
907.jpg
909.jpg
913.jpg
901.jpg
914.jpg
906.jpg
930.jpg

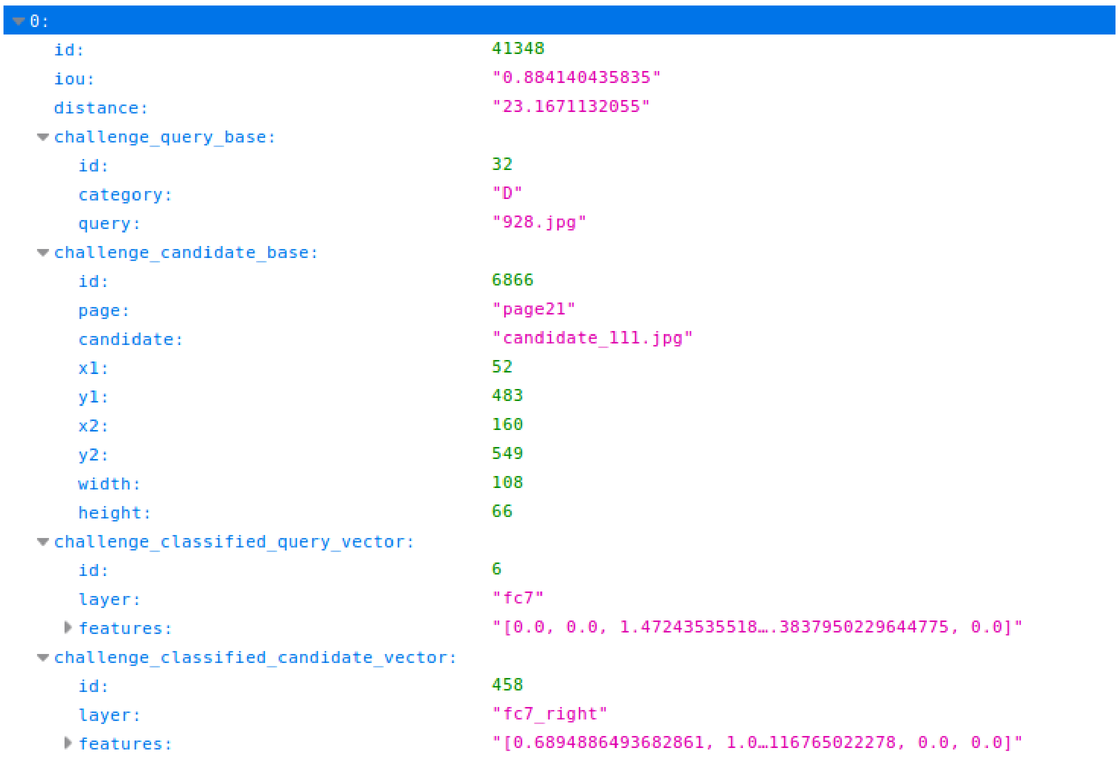
Top K: 100

[✓] Repeat Pages?

Click and use Alt+Enter or UP/DOWN keys to next candidate

```
1 - candidate_65.jpg(page788,Distance:26.05,IOU:0.74):
X1:35 X2:129 Y1:134 Y2:193 W:94 H:59
2 - candidate_55.jpg(page1403,Distance:26.09,IOU:0.70):
X1:189 X2:288 Y1:369 Y2:437 W:99 H:68
3 - candidate_111.jpg(page21,Distance:26.62,IOU:0.88):
X1:52 X2:160 Y1:483 Y2:549 W:108 H:66
4 - candidate_74.jpg(page796,Distance:27.40,IOU:0.69):
X1:57 X2:155 Y1:268 Y2:353 W:98 H:85
5 - candidate_73.jpg(page796,Distance:28.78,IOU:0.51):
X1:57 X2:155 Y1:268 Y2:333 W:98 H:65
6 - candidate_112.jpg(page21,Distance:28.81,IOU:0.82):
X1:53 X2:160 Y1:487 Y2:554 W:107 H:67
7 - candidate_66.jpg(page967,Distance:29.11,IOU:0.49):
X1:42 X2:130 Y1:497 Y2:563 W:88 H:66
8 - candidate_64.jpg(page967,Distance:29.97,IOU:0.50):
X1:37 X2:130 Y1:485 Y2:563 W:93 H:78
9 - candidate_138.jpg(page545,Distance:30.40,IOU:0.49):
X1:247 X2:332 Y1:189 Y2:244 W:85 H:55
10 - candidate_73.jpg(page1027,Distance:30.67,IOU:0.47):
X1:37 X2:113 Y1:404 Y2:461 W:76 H:57
```

Figure C.13: IR and PS system - (E) New challenge classified - explore classification details



```
▼0:
    id:                                 41348
    iou:                                "0.884140435835"
    distance:                           "23.1671132055"
  ▼challenge_query_base:
      id:                               32
      category:                         "D"
      query:                            "928.jpg"
  ▼challenge_candidate_base:
      id:                               6866
      page:                             "page21"
      candidate:                        "candidate_111.jpg"
      x1:                               52
      y1:                               483
      x2:                               160
      y2:                               549
      width:                            108
      height:                           66
  ▼challenge_classified_query_vector:
      id:                               6
      layer:                            "fc7"
    ▶features:                          "[0.0, 0.0, 1.47243535518….3837950229644775, 0.0]"
  ▼challenge_classified_candidate_vector:
      id:                               458
      layer:                            "fc7_right"
    ▶features:                          "[0.6894886493682861, 1.0…116765022278, 0.0, 0.0]"
```

Figure C.14: IR and PS system - (E) New challenge classified - example of JSON object generated to represent the distance between a query and candidate. The user can save these files to calculate the performance