Efficient and Scalable Handwritten Word Spotting on Historical Documents using Bag of Visual Words

David Aldavert-Miró

Universitat Autònoma de Barcelona

Efficient and Scalable Handwritten Word Spotting on Historical Documents using Bag of Visual Words

A disertation submitted by **David Aldavert-Miró** at Universitat Autònoma de Barcelona, Dept. Ciències de la Computació, to fulfil the degree of **Doctor of Philosophy** in Computer Science.

Bellaterra, February 5th, 2021

Director:	**Dr. Marçal Rossinyol Sanabra**
	Centre de Visió per Computador.
	AllRead Machine Learning Technologies
Tutor:	**Dr. Josep Lladós Canet**
	Universtàt Autònoma de Barcelona
	Dept. Ciències de la Computació & Centre de Visió per Computador.

This document was typeset by the author using $\LaTeX 2_\varepsilon$.

The research described in this book was carried out at the Computer Vision Center, Universitat Autònoma de Barcelona.

Als meus pares **Javier** i **Francesca**

ABSTRACT

Word spotting can be defined as the pattern recognition tasked aimed at locating and retrieving a specific keyword within a document image collection without explicitly transcribing the whole corpus. Its use is particularly interesting when applied in scenarios where Optical Character Recognition performs poorly or can not be used at all. This thesis focuses on such a scenario, word spotting on historical handwritten documents that have been written by a single author or by multiple authors with a similar calligraphy. This problem requires a visual signature that is robust to image artifacts, flexible to accommodate script variations and efficient to retrieve information in a rapid manner. For this, we have developed a set of word spotting methods that on their foundation use the well known Bag-of-Visual-Words (BoVW) representation. This representation has gained popularity among the document image analysis community to characterize handwritten words in unsupervised manner. However, most approaches on this field rely on a basic BoVW configuration and disregard complex encoding and spatial representations. We determine which BoVW configurations provide the best performance boost to the spotting system.

Then, we extend the segmentation-based word spotting, where word candidates are given *a priori*, to segmentation-free spotting. The proposed approach seeds the document images with overlapping word location candidates and characterizes them with a BoVW signature. Retrieval is achieved comparing the query and candidate signatures and returning the locations that provide a higher consensus. This is a simple but powerful approach that requires a more compact signature than in a segmentation-based scenario. We first project the BoVW signature into a reduced semantic topics space and then compress it further using Product Quantizers. The resulting signature only requires a few dozen bytes, allowing us to index thousands of pages on a common desktop computer. The final system still yields a performance comparable to the state-of-the-art despite all the information loss during the compression phases.

We also study how to combine different modalities of information in order to create a query-by-X spotting system where, words are indexed using an information modality and queries are retrieved using another. We consider three different information modalities: visual, textual and audio. Our proposal is to create a latent feature space where features which are semantically related are projected onto the same topics. Creating thus a new feature space where information from different modalities can be compared.

The codebooks used to encode the BoVW signatures are usually created using an unsupervised clustering algorithm and, they require to test multiple parameters to determine which configuration is best for a certain document collection. We propose a semantic clustering algorithm which allows to estimate the best parameter from data. Since gather annotated data is costly, we use synthetically generated word images. The resulting codebook is *database agnostic*, i. e.a codebook that yields a good performance on document collections that use the same script. We also propose the use of an additional codebook to approximate descriptors and reduce the descriptor encoding complexity to sub-linear.

Finally, we focus on the problem of signatures dimensionality. We propose a new symbol probability signature where each bin represents the probability that a certain symbol is present a certain location of the word image. This signature is extremely compact and combined with compression techniques can represent word images with just a few bytes per signature.

RESUM

La localització de paraules en el camp de anàlisis de documents es pot definir com el reconeixement de patrons encarregat de localitzar i recuperar una paraula específica dins d'una col·lecció d'imatges sense transcriure explícitament el corpus sencer. El seu ús és particularment interessant quan s'aplica a escenaris on el reconeixement òptic de caràcters funciona malament o no es pot utilitzar en absolut. Aquesta tesi se centra en un escenari d'aquest tipus, detectar paraules en documents manuscrits històrics que han estat escrits per un sol autor o per diversos autors amb una cal·ligrafia similar. Aquest problema requereix d'una signatura visual que sigui robusta contra artefactes de les imatges, flexible per adaptar-se a les variacions del traç i que sigui eficient per recuperar la informació de manera ràpida. Per a això, hem desenvolupat un conjunt de mètodes de localització de paraules que, en la seva base, utilitzen la coneguda representació Bag-of-Visual-Words (BoVW). Aquesta representació ha guanyat popularitat entre la comunitat d'anàlisi d'imatges de documents per caracteritzar paraules manuscrites en tasques no supervisades. Tanmateix, la majoria d'enfocaments en aquest camp es basen en una configuració bàsica de BoVW i ignoren les codificacions complexes i les representacions espacials. Determinem quines configuracions de BoVW proporcionen el millor increment de rendiment.

A continuació, estenem la localització de paraules de sistemes on aquestes estan pre-segmentades a un on no utilitzem cap tipus de segmentació. L'enfocament proposat selecciona regions sobreposades del document com a candidates i les caracteritza amb una signatura BoVW. La localització s'aconsegueix comparant la imatge de consulta amb les signatures dels candidats i retornant les ubicacions que tenen un consens més alt. Aquest és un enfocament senzill però potent que requereix una signatura compacta. Primer projectem la signatura BoVW en un espai de temes semàntics i després la comprimim encara més mitjançant un producte de quantificadors. La signatura resultant requereix només unes dotzenes de bytes, cosa que ens permet indexar milers de pàgines en un ordinador de sobretaula estàndard.

També estudiem com combinar diferents modalitats d'informació per tal de crear un sistema on les paraules s'indexa mitjançant una modalitat d'informació i les consultes mitjançant una altra. Considerem tres modalitats d'informació diferents: visual, textual i àudio. La nostra proposta és crear un espai de característiques latents on les característiques relacionades semànticament es projectin sobre els mateixos temes latents. Creant així un nou espai on la informació de diferents modalitats es pugui comparar.

Els diccionaris que s'utilitzen per codificar les signatures BoVW es creen generalment mitjançant un algorisme de no supervisat i requereixen provar diversos paràmetres per determinar quina configuració és la millor per a una col·lecció de documents determinada. Proposem un algorisme d'agrupament semàntic que permet estimar els paràmetres a partir de dades. Atès que la recopilació de dades anotades és costosa, fem servir imatges de paraules generades sintèticament. El diccionari resultant proporciona un bon rendiment a les col·leccions de documents que utilitzen el mateix estil de text. També proposem l'ús d'un diccionari addicional per aproximar els descriptors i reduir la complexitat de codificació del descriptor a sub-lineal.

Finalment, ens centrem en el problema de la dimensionalitat de les signatures. Proposem una nova signatura on cada element representa la probabilitat que un determinat símbol tingui una determinada ubicació dins la imatge de la paraula. Aquesta signatura és extremadament compacta i combinada amb tècniques de compressió, pot representar la imatge d'una paraula amb només uns quants bytes.

PUBLICATIONS

The pursuit of this thesis has led to the following heterogeneous set of publications:

JOURNALS

- D. Aldavert, M. Rusiñol, R. Toledo, J. Lladós, **"A Study of Bag-of-Visual-Words Representations for Handwritten Keyword Spotting"**, *International Journal on Document Analysis and Recognition, volume 18, issue 3, pp. 223-234, September, 2015.*
- M. Rusiñol, D. Aldavert, R. Toledo, J. Lladós, **"Efficient Segmentation-free Keyword Spotting in Historical Document Collections"**, *Pattern Recognition, volume 48, issue 2, pp. 545-555, February, 2015.*
- A. Ramisa, D. Aldavert, S. Vasudevan, R. Toledo, R. López de Mántaras, **"Evaluation of Three Vision Based Object Perception Methods for a Mobile Robot"**, *Journal of Intelligent and Robotic Systems, volume 68, issue 2, pp. 185-208, November, 2012.*
- A. Ramisa, A. Goldhoorn, D. Aldavert, R. Toledo, R. López de Mántaras, **"Combining Invariant Features and the ALV Homing Method for Autonomous Robot Navigation Based on Panoramas"**, *Journal of Intelligent and Robotic Systems, volume 64, issue 3-4, pp. 625-649, 2011.*
- A. Ramisa, A. Tapus, D. Aldavert, R. Toledo, R. López de Mántaras, **"Robust Vision-Based Localization using Combinations of Local Feature Regions Detectors"**, *Autonomous Robots Journal, volume 27, number 4, pages 373-385, November, 2009.*

INTERNATIONAL CONFERENCES

- D. Aldavert, M. Rusiñol, **"Manuscript Text Line Detection and Segmentation using Second-order Derivatives Analysis"**, *In Proceedings of the Thirteenth International Workshop on Document Analysis Systems, DAS18.*
- D. Aldavert, M. Rusiñol, **"Synthetically Generated Semantic Codebook for Bag-of-Visual-Words Based Word Spotting"**, *In Proceedings of the Thirteenth International Workshop on Document Analysis Systems, DAS18.*
- D. Aldavert, M. Rusiñol, R. Toledo, **"Automatic Static/Variable Content Separation in Administrative Document Images"**, *In Proceedings of the Fourteenth International Conference on Document Analysis and Recognition, ICDAR17.*
- M. Rusiñol, D. Aldavert, R. Toledo, J. Lladós, **"Towards Query-by-Speech Handwritten Keyword Spotting"**, *In Proceedings of the Thirteenth International Conference on Document Analysis and Recognition, ICDAR15.*

- D. Aldavert, M. Rusiñol, R. Toledo, J. Lladós, **"Integrating Visual and Textual Cues for Query-by-String Word Spotting"**, *In Proceedings of the Twelfth International Conference on Document Analysis and Recognition, ICDAR13*.

- M. Rusiñol, D. Aldavert, R. Toledo, J. Lladós, **"Browsing Heterogeneous Document Collections by a Segmentation-free Word Spotting Method"**, *In Proceedings of the Eleventh International Conference on Document Analysis and Recognition, ICDAR11, pages 63-67*.

- A. Ramisa, D. Aldavert, S. Vasudevan, R. Toledo, R. López de Mántaras, **"The IIIA30 Mobile Robot Object Recognition Dataset"**, *11th International Conference on Mobile Robots and Competitions (ROBOTICA 2011), Lisbon, 2011*.

- M. Rusiñol, D. Aldavert, D. Karatzas, R. Toledo, J. Lladós, **"Interactive Trademark Image Retrieval by Fusing Semantic and Visual Content"**, *33th European Conference on Information Retrieval, LNCS 6611, 314–325, 2011*.

- D. Aldavert, A. Ramisa, R. Toledo, R. López de Mántaras, **"Real-Time Object Segmentation using a Bag of Features Approach"**, *13th International Conference of the ACIA. Frontiers in Artificial Intelligence and Applications, 2010*.

- D. Aldavert, A. Ramisa, R. Toledo, R. López de Mántaras, **"Fast and Robust Object Segmentation with the Integral Linear Classifier"**, *Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), San Francisco, California, USA, 13-18 June 2010*.

- D. Aldavert, A. Ramisa, R. Toledo, R. López de Mántaras, **"Efficient Object Pixel-Level Categorization using Bag of Features"**, *5th International Symposium on Visual Computing, Las Vegas, Nevada, USA, November 30 - December 2, 2009*.

- A. Ramisa, D. Aldavert, S. Vasudevan, R. Toledo, R. López de Mántaras, **"Evaluation of the SIFT Object Recognition Method in Mobile Robots"**, *12th International Conference of the ACIA. Frontiers in Artificial Intelligence and Applications, 2009*.

- D. Aldavert, A. Ramisa, R. Toledo, R. López de Mántaras, **"Visual Registration Method For A Low Cost Robot"**, *7th International Conference on Computer Vision Systems, Liege, Belgium, 13-15 October 2009*.

- D. Aldavert, R. Toledo, **"Stereo Vision Local Map Alignment for Robot Environment Mapping"**, *in Proceedings of the Second International Workshop, RobVis 2008, Auckland, New Zealand, February 18-20, 2008*.

- A. Ramisa, R. López de Mántaras, D. Aldavert, R. Toledo, **"Comparing Combinations of Feature Regions for Panoramic VSLAM"**, *4th International Conference on Informatics in Control, Automation and Robotics, Angers, France, 9-12 May 2007, 292-297*.

internal workshops & technical reports

- A. Ramisa, D. Aldavert, S. Vasudevan, R. Toledo, R. López de Mántaras, **"Evaluation of Three Vision Based Object Perception Methods for a Mobile Robot"**, *arXiv:1102.0454, 2011*.
- D. Aldavert, **"Visual Simultaneous Localization and Mapping"**, MSc Thesis, Computer Vision Center, 2006.
- D. Aldavert, A. Ramisa, R. Toledo, **"Wide Baseline Stereo Matching Using Voting Schemas"**, *1st CVC Internal Workshop, Computer Vision: Progress of Research and Development, CVC(UAB), Bellaterra (Spain), 2006*.

AGRAÏMENTS

Quan vaig començar el doctorat fa molts anys res em podia fer pensar que l'acabaria de la manera en que ho he fet. La malaltia de la mare ha dominat aquests últims anys i a fet que meva memòria de temps anteriors al seu Alzheimer siguin molt difusa. Per tant, he de començar demanant disculpes a la gent que segurament m'oblidaré d'anomenar. Tampoc hagués pogut creure que quan la mare va començar a marxar, el meu director de tesi ens deixaria. En Ricardo Toledo va ser qui em va introduir al mon de la recerca, ja des del projecte de final de carrera, i qui em va introduir al mon de la robòtica i la visió per computador. Ell no va ser mai només un professor o el director de tesi, sinó que va ser un bon amic. Sense ell aquesta tesi no hauria estat possible.

Donar les gràcies a Marçal Rossinyol amb qui em compartit un munt d'hores, ja sigui fent articles, fent docència o al bar, i que va agafar el rol de director de tesi al traspassar Ricardo. Donar gràcies a Josep Lladós per fer de tutor d'aquesta tesi. Evidentment donar les gràcies també l'Arnau Ramisa amb que varem passar moltes hores plegats intentant fer que els robots poguessin veure alguna cosa. Agrair les estones compartides a tots els companys de despatx, a en Mohammad Rouhani, a en David Gerónimo, a l'Ivan Huerta, a n'Alicia Fornes, a en José M. Álvarez, en Joan Mas i a en Javier Vazquez (qui pot oblidar d'en Javier). Als companys de docència, la Carme Julià, l'Anna Salvatella, en Pau Baiget i als responsables de les assignatures, na Gemma Sánchez, en Javier Sánchez, na Petia Radeva i en Fernando Vilariño. A en Raúl Rojas per acollir-me durant tres mesos a la Freie Universität de Berlin. I també especialment a Jordi Gonzàlez qui ha insistit en que presentes aquesta tesi quan jo ja l'havia donat per perduda.

Finalment, donar les gràcies a la meva família pel seu suport incondicional. A la meva mare i al meu pare que sempre m'han animat a que fes el que m'agradi a la vida i a la meva germana que al començar la tesi érem companys de pis i ara el seu fill ja està a punt de començar la escola secundaria.

Segurament m'he deixat molta gent i els torno a demanar disculpes. Moltes gràcies a tothom qui m'ha acompanyat durant tots aquests anys de doctorant.

CONTENTS

1 INTRODUCTION 1
1.1 The word spotting task 1
1.2 Challenges and Objectives 3
1.3 Contributions 4
1.3.1 Study of Bag-of-Visual-Words 4
1.3.2 Segmentation-free Word Spotting 5
1.3.3 Multi-modal Word Spotting 5
1.3.4 Supervised Codebook Generation 5
1.3.5 Compact Character Probability Signatures 6
1.3.6 Document Structure Analysis 7
2 WORD SPOTTING WITH BAG OF VISUAL WORDS 9
2.1 Introduction 9
2.1.1 Keyword Spotting as an Object Recognition Task 10
2.1.2 Contributions and Outline 11
2.2 Bag-of-Visual-Words Representations 11
2.2.1 Sampling 12
2.2.2 Description 12
2.2.3 Encoding 13
2.2.4 Pooling 14
2.3 BoVW Parameter Evaluation 16
2.3.1 Experimental Setup 16
2.3.2 Locality-constrained Linear Coding (LLC) Encoding 17
2.3.3 Spatial Pyramids 17
2.3.4 Power normalization 19
2.3.5 Codebook size 19
2.3.6 Descriptor sampling 20
2.3.7 Summary of the Results 21
2.4 Performance Comparison with the State of the Art 21
2.4.1 George Washington Dataset 22
2.4.2 H-KWS 2014 Competition 23
2.5 Conclusions 24
3 SEGMENTATION FREE WORD SPOTTING 27
3.1 Introduction 27
3.2 Off-line Corpus Representation 29
3.2.1 Local Patch Descriptor 29
3.2.2 Latent Semantic Analysis Transform 31
3.2.3 Product Quantization Indexing 32
3.3 Word Retrieval 32
3.3.1 Candidate Search 33
3.3.2 Candidate Localization 34
3.4 Experimental Results 34
3.4.1 Dataset and Evaluation Measures 34
3.4.2 Results 36
3.5 Conclusions 43
4 DATA FUSION 45
4.1 Introduction 45
4.2 Word Representations 47
4.2.1 Textual Representation 47
4.2.2 Audio Representation 48

4.2.3 Visual Representation 49
4.3 Multi-modal Information Fusion 51
4.4 Retrieval Phase 52
4.4.1 Word snippet retrieval 52
4.4.2 Audio Retrieval 52
4.5 Experimental Results 52
4.5.1 Query-by-String results 53
4.5.2 Query-by-Speech results 55
4.5.3 Query-by-speech with Unheard Voices 57
4.5.4 Handwritten Text-to-speech via Audio Retrieval 58
4.6 Conclusions 58
5 SEMANTIC CODEBOOK 59
5.1 Introduction 59
5.2 Synthetic Codebook Generation 61
5.2.1 Agglomerative Clustering 61
5.2.2 Shannon Entropy 62
5.3 Descriptor Encoding 63
5.3.1 Codeword Encoding 63
5.3.2 Approximate Codebook 64
5.4 Results 64
5.5 Conclusions 66
6 COMPACT SYMBOL PROBABILITY SIGNATURES 67
6.1 Introduction 67
6.2 Visual word generation 68
6.3 Sliding Window Classifier 70
6.3.1 Histogram pooling 70
6.3.2 Normalization and classification 72
6.4 Integral Linear Classifier 73
6.4.1 Algorithm overview 73
6.4.2 Classification score 74
6.4.3 Advantages and limitations 75
6.5 Learning Algorithms 76
6.5.1 Mini-batch First Order Stochastic Gradient Descent 77
6.5.2 FOSGD Semantic Segmentation Learner 78
6.6 Symbol Probability Signature 81
6.7 Experimental Results 82
6.7.1 Datasets 82
6.7.2 System parameters 83
6.7.3 Symbol classification 83
6.8 Spotting Results on the George Washington Dataset 85
6.9 Results on Botany and Konzilsprotokolle 86
6.10 Conclusions 87
7 CONCLUSIONS AND FUTURE WORK 89
A LINE SEGMENTATION 93
A.1 Introduction 93
A.2 Second Order Derivative Analysis 94
A.2.1 Second Order Derivative 95
A.2.2 Scale Selection 96
A.3 Line Segmentation 97
A.3.1 Orientation and Scale Selection 97
A.3.2 Text Region and Line Detection 99
A.3.3 Text Segmentation 100
A.4 Experimental Results 100

A.5 Conclusions and future work 101
B LINE SEGMENTATION 103
B.1 Introduction 103
B.2 Overview 104
B.3 Document Alignment 105
B.4 System Description 107
B.4.1 Document Static Model Generation 107
B.4.2 Dynamic Elements Detection 110
B.5 Experimental Results 110
B.6 Conclusions 112

BIBLIOGRAPHY 113

LIST OF FIGURES

Figure 1.1 Different version of the *Llibre dels fets*. The first three versions are hand written while the last version of 1873 is already typewritten. 2

Figure 1.2 Four samples of dataset [19] showing the effects of desiccant, tea stains, scrapping and heat document degradations. 3

Figure 1.3 Portions of documents that use different supports. Paper belongs to the "*Genealogia regum Navarrae et Aragoniae et comitum Barchinonae*", papyrus belongs to the "*Rhind Mathematical Papyrus*, the parchment is a part of the "*Isaiah Scroll*" and the stone is a portion of the "*Stone cuneiform tablet with inscription of Ashurnasirpal II*". 3

Figure 1.4 Examples of three different types of script: a Greek alphabet, a Cherokee syllabary and a simplified Chinese logogram. 4

Figure 2.1 Norm of the descriptors extracted from regions of 16, 24 and 32 pixels width sampled at each pixel of the image. The bold contours encircle the regions where descriptors which have a large enough norm and are considered reliable. 11

Figure 2.2 Codebook creation and descriptor encoding example: **a)** Descriptors are randomly sampled from the indexed images, **b)** the k-means algorithm is used to build the codebook and **c)** descriptors are encoded using sparse coding with the cluster centroids. 12

Figure 2.3 The bins corresponding to the visual words sampled over the *motorbike* objects increase their contribution linearly to the size of the object. Therefore, the score of a *motorbike* linear classifier will also increase linearly from (a) to (d). A better behavior would be that the score sharply increases from (a) to (b) but keeps a similar score from (b) to (d). 15

Figure 2.4 mean Average Precision (mAP) score obtained using different number of neighbors with LLC. 17

Figure 2.5 Distribution of the spatial bins in the two levels of the spatial pyramid. 18

Figure 2.6 Effect of the power norm to the performance of the word spotting system. 19

Figure 2.7 Evolution of the mAP score while increasing the size of the codebook. 20

Figure 3.1 Local patch descriptor signature: **a)** SIFT descriptors are extracted from the document image. Note that the missing descriptors are filtered out due to their low descriptor normal value. Then, **b)** SIFT descriptors are encoded into visual words and **c)** the visual words which lie within the local patch are accumulated in the local patch signature. 30

Figure 3.2 Example of the voting procedure. **a)** Query, **b)** sample page, **c)** obtained voting space. 33
Figure 3.3 Example of pages from the a) George Washington, b) Barcelona Cathedral and c) Lord Byron's collections. 35
Figure 3.4 9 top-most retrieved images for some queries in the four evaluated collections. 36
Figure 3.5 Mean average precision and recall for different c and m values when using PQ for the a) and b) George Washington, c) and d) Barcelona Cathedral and e) and f) Lord Byron's collections. 39
Figure 4.1 Example of the n-gram textual descriptor. 47
Figure 4.2 Example of utterances of the word ***companies***, their RASTA-Perceptual Linear Prediction (PLP) cepstral coefficients and their BoAW histograms for two different voices. 49
Figure 4.3 Distribution of the spatial bins in the two levels of the spatial pyramid. 50
Figure 4.4 Examples of the 20 most similar word images and their average precision (AP) returned by the system for some queries. 53
Figure 4.5 mAP attained by the system using different number of dimensions in the topic space. 54
Figure 4.6 Different recall values obtained while increasing the number of elements returned by the system. 55
Figure 4.7 Examples of the 20 most similar word images returned by the system for different spoken queries. 56
Figure 4.8 mAP attained by the system using different number of dimensions in the topic space. 56
Figure 5.1 Two examples of the clusters generated by the codebook. CLUSTER 09 contains descriptors from symbols [0, 8, c, g, b, j, h, p, f, x, u], while CLUSTER 10 is formed by descriptors from [h, b, 4, 6, t, k, u, o, f, d, j, l, p, y, w, a, q, i, n]. CLUSTER 10 is represented by 5 centroids because it has at least a nested cluster. 60
Figure 5.2 Schema of the agglomerative tree codewords and centroids. 63
Figure 5.3 Some qualitative results obtained in the George Washington dataset. 65
Figure 6.1 Method overview: We extract a BoVW histogram for each pixel of the image and categorize it with a linear classifier. The probabilities of each symbol category are then grouped to form the final descriptor. 68
Figure 6.2 Sampling regions of the FREAK descriptor. The pattern follows the distribution of the retinal ganglion cells with their corresponding receptive fields. Each circle represents a receptive field where the image is smoothed with its corresponding Gaussian kernel. 69

Figure 6.3 The Huang algorithm [125] obtains the new histogram in 6.3a by subtracting left column values $\{(a:1),(c:2)\}$ and adding the right column values $\{(a:2),(c:1)\}$. The distributive algorithm [126, 127] reduces the computational cost by first calculating each column histogram 6.3b and then following as in Huang but using the already calculated histograms. 72
Figure 6.4 The *weights image* b) is created by replacing visual words from image a) by their corresponding classifier weights. In this example, the classifier values are $\mathbf{c} = [3,5,-5,-4,1,-1,1,4]$. Then, we can create the *integral image* c) from the *weights image* b), so that the classification score of the window can be calculated as $s = 21+5-3-9 = 14$. 74
Figure 6.5 Evolution of the classifier score and loss estimation images at each iteration of the First-Order Stochastic Gradient Descent (SGD) algorithm. 80
Figure 6.6 Probabilities of the classifiers with the strongest response overlapping the word image 81
Figure 6.7 Example pages of the George Washington, Botany and Konzilsprotokolle datasets. 82
Figure 6.8 Example of original text line from the George Washington dataset, the same text line created with a TrueType (TTF) font and its associated annotation. 83
Figure 6.9 Example of the manual annotation on the George Washington dataset. 83
Figure 6.10 Evolution of the best mAP score attained for different number of spatial bins and different normalization power factor. 85
Figure A.1 Blurring the binary document with a large enough Gaussian generates an image where text lines appear as blobs. 95
Figure A.2 Bases used to compute the second order Gaussian derivative at any orientation. 95
Figure A.3 Example of the selected pixels over the original document image. The magnitude of the values is shown using a hue color map. In the orientation images, green pixels correspond to horizontal orientations while red and blue pixels correspond to vertical orientations (orientation is circular). In the first row we show all local maxima, while in the second row we see the local maxima after applying the non-maxima suppression filter. 96
Figure A.4 Text region and line detection on the same image under different transforms. 97
Figure A.5 Histograms obtained for the original image, the image transformed by a rotation and an affine transform. In this histogram images, the x-axis corresponds to the orientation while the y-axis corresponds to the σ of filter. 98
Figure A.6 Final segmentation under different image transforms. 99
Figure B.1 System Overview 104
Figure B.2 Sample training documents for a particular class 108

Figure B.3 Probabilistic model template obtained from the document set shown in Fig. B.2. 109
Figure B.4 Detection results. a) Test images, b) Probability maps of the variable elements after aligning the test documents wih their respective templates, c) Final bounding-box extractions. 111

LIST OF TABLES

Table 2.1 mAP score obtained using different spatial configurations. 18
Table 2.2 mAP scores obtained when modifying the descriptor sampling parameters 20
Table 2.3 Summary of the improvements over the baseline Bag of Visual Words (BoVW) implementation with the gains in performance 21
Table 2.4 Comparison of the performance attained by the system using the baseline and final BoVW configurations against the results reported by each work. The methods in the first half are exemplar-based methods while second half methods are learning-based. 22
Table 2.5 Comparison of the performance attained by the system using the baseline and enhanced BoVW configurations with the methods that participated in the Handwritten Keyword Spotting Competition 2014 (H-KWS 2014) competition. 24
Table 3.1 Local patch and feature geometries parameters used at each database. 35
Table 3.2 Mean average precision and recall at different codebook sizes and amount of topics for the three collections. 37
Table 3.3 Bytes and compression ratio per patch for each PQ setup. 38
Table 3.4 Performance of the proposed baseline method with PQ, multi-length configuration. 40
Table 3.5 Performance of the system in the GW1500 scenario. 41
Table 3.6 Review of performances of the state-of-the-art methods using the GW20 dataset. 42
Table 4.1 State-of-the-art queried-by-string word spotting results for the GW database. 53
Table 4.2 Comparison with our previous work using the same visual description. 57
Table 4.3 Retrieval performance using unheard voices as queries. 57
Table 4.4 Accuracies for the Handwritten Text-to-speech. 58
Table 5.1 mAP scores at the Washington dataset. 65
Table 5.2 Sizes of the evaluated codebooks. 66

Table 6.1 Example of the result of applying the two strategies used to deal with the negative weights. In this example, the methods are applied on the following list of ***(weight, identifier)*** tuples representing the visual words: $[(-3.12, 0), (1.27, 1), (-0.59, 1), (0.79, 2)]$ 75
Table 6.2 Classification results of the Integral Linear Classifier (ILC) classifier for each symbol category. The AGM column corresponds to the geometric mean between the positive and negative accuracies. 84
Table 6.3 mAP score obtained by the best configuration for each descriptor. 85
Table 6.4 mAP precision scores obtained by different methods on the Botany and Konzilsprotokolle datasets 86
Table A.1 Text line segmentation results 101
Table B.1 Contingency matrix 110

LIST OF ALGORITHMS

1 Algorithm to collapse the projection vector so the dot product with the difference pairs can be computed directly over the sampling point values. 69
2 Mini-batch version of the First-Orders Stochastic Gradient Descent algorithm. 78
3 Mini-batch version of the First-Orders Stochastic Gradient Descent algorithm adapted to the semantic segmentation problem. 79

ACRONYMS

BoVW	Bag of Visual Words
BoAW	Bag-of-Audio-Words
HOG	Histogram of Oriented Gradients
FREAK	Fast Retina Keypoint
LLC	Locality-constrained Linear Coding
LBP	Local Binary Pattern
mAP	mean Average Precision
HKM	Hierarchical k-Means
SPM	Spatial Pyramid Matching
HMM	Hidden Markov Model
TTS	Text-to-Speech
TTF	TrueType
ILC	Integral Linear Classifier
SVD	Singular Value Decomposition

SGD	Stochastic Gradient Descent
VLAD	Vector of Locally Aggregated Descriptors
DTW	Dynamic Time Warping
OCR	Optical Character Recognition
PQ	Product Quantizer
LSA	Latent Semantic Analysis
LDA	Linear Discriminant Analysis
PLP	Perceptual Linear Prediction
PHOC	Pyramidal Histogram Of Characters

1 INTRODUCTION

A large amount of information is available about past events is stored and available on archives all over the world. Traditionally this data was only available though physical access, limiting thus its availability. However, during the last decade, we have seen an effort to digitalize them and make them available through Internet, both by public and private institutions [1]. This information does not consist only off published books but also reports and registers from all sorts of institutions. From birth, marriage [2, 3] and death certificates, to traded good or weather reports. The analysis of these documents leads to a better understanding of our past [4] but also to model our future, by for example, studying the propagation of pathogens in medieval Europe [5] or incorporating nineteenth century NOAA weather reports into the historical data to the weather prediction models [6].

Document digitalization allows to researchers and the general public to easily access these reports. However, the sheer amount of archived information makes it virtually inaccessible without automatic data mining systems that are able to search for the desired information. In order to enable users to access these information, the Document Image Analysis research field focuses on problems like, document digitally restoration and enhancement[7, 8], layout document analysis [9, 10], character/word recognition [11, 12, 13], semantic annotation [14, 15] or word/symbol spotting [16, 17, 18], to easily process, locate and extract information from document images.

1.1 THE WORD SPOTTING TASK

Word or symbol spotting is the task of localing and retrieving a keyword of interest within a document image collection without explicitly recognizing or transcribing the whole corpus. The methods employed to solve this task vary depending on the documents they are designed to processes.

Typewritten documents are printed by a machine so, multiple instances of the same symbol are going to have the same shape. In contrast, the shape of the same symbol can greatly vary in handwritten documents, even in documents written by the same author. We can see in Fig. 1.1 a sample of the same book copied at different dates. The first three versions are handwritten and we can appreciate that the style and shape of letters varies quite a bit. The last version is already typewritten and its shape is completely regular. Besides the different shapes of the script symbols, these documents also show differences in the type of content. For example, the use of abbreviations and symbols was common in the older versions of 1314 and 1380. Additionally, the words used in the older document not always are spelled the same. For example, the city of *Zaragoza* is can be written as *Saragossa*, *Çaragoça* or *Saragoça* in the same paragraph. Therefore, we may need that the spotting system is flexible enough to detect the three variants of this word as the same.

The age of the document is an important factor. Modern documents usually have a *clean* background so they can be text can be easily segmented by means of a binarization algorithm [20]. On the other hand, historical documents are likely to contain artifacts due to paper degradation, wrinkles, stains or ink artifacts like faded ink and ink bleed though. In these scenarios the spotting system needs to be more robust to noise. Alternatively, an additional pre-processing step is needed to attempt to clean the document image [21, 22, 19]. For example, Giacometti et al. address the document restoration problem in [19] in a multispectral image setting. They attempt to combine the information extracted at different light frequencies to remove or lessen the effects of 25 of the most common document degradations found in historical documents. In Fig. 1.2, we can see an example of the degraded images and the original images they attempt to recover.

A different source of noise is the support itself as it can also introduce artifacts in the document image. For example, papyrus documents have a background pattern and parchment documents are more likely to contain wrinkles.

Another important aspect is the language used in the document. Different languages use different writing direction conventions and different scripts. A spotting system only needs to be aware of the writing direction as words written left to right, right to left or top to bottom does not substantially change the difficulty of the spotting task. The script type however has a great impact the difficulty of word retrieval as the script type determines the amount of different symbols possible. The language scripts can be basically divided into a alphabet, a syllabary or a logographic system (see Fig. 1.4 for an example of each script type). An alphabet is a standardized set of symbols or graphemes that represent phonemes, a syllabary the symbols represents syllables and a logographic system the symbols represent a word or a morpheme. Alphabets only contain dozens of unique symbols, a syllabary can grow up to hundred symbols while a logographic system can contain thousands of different symbols. Therefore, features extracted by a spotting system specifically designed to characterize words in Latin script may not be representative enough to properly distinguish Chinese symbols.

Summarizing, the difficulty of the word spotting task greatly varies depending on the properties of the documents it has to index. Documents being handwritten or typewritten, the number of writers, document degradations, or language script define the challenges that a spotting system has to handle [23].

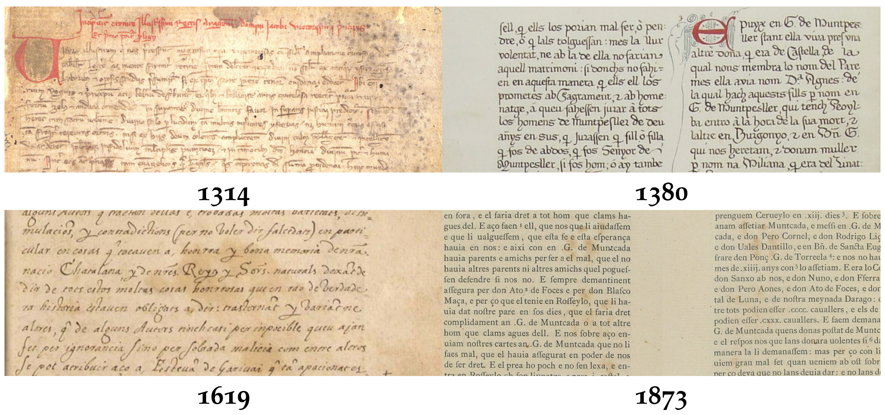


Figure 1.1: Different version of the *Llibre dels fets*. The first three versions are hand written while the last version of 1873 is already typewritten.

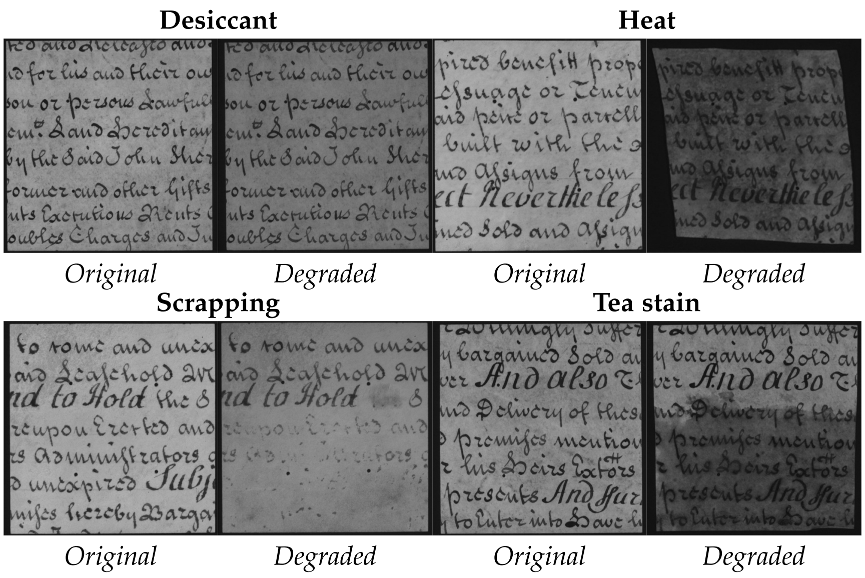

Desiccant
Heat
Original
Degraded
Original
Degraded
Scrapping
Tea stain
Original
Degraded
Original
Degraded

Figure 1.2: Four samples of dataset [19] showing the effects of desiccant, tea stains, scrapping and heat document degradations.

1.2 CHALLENGES AND OBJECTIVES

The objective of these thesis is to develop a word spotting system that is able to handle historical handwritten documents. Specifically, documents written by a single author or by multiple authors that have a similar calligraphy. Therefore, we expect that multiple instances of the same symbol are not going to look exactly the same and, the method used to characterize them needs to be flexible enough to account for these variances. We could use ad-hoc feature for this task, i.e. features that are designed to tackle a specific document type [16, 24]. This approach however limits the usefulness of the system as it is tailored for an exceedingly concrete problem and, we would like a system that does not impose pre-requisites on the type of documents it can process. For example, we are going to mostly process handwritten documents using Latin script. However, we do not want to constrain ourselves to only these kind of documents. For us, a more appealing approach is a system that relies on generic techniques that allows to work out of the box with any sort of script or graphical symbols.

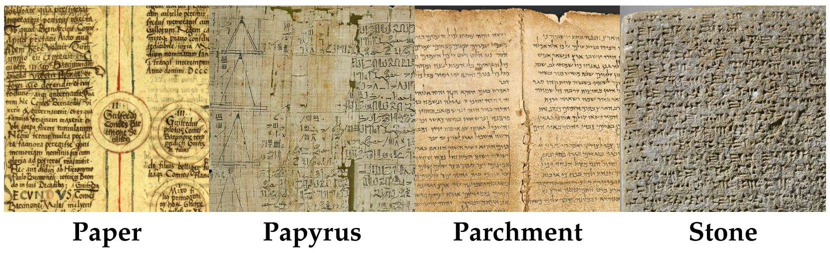


Figure 1.3: Portions of documents that use different supports. Paper belongs to the "*Genealogia regum Navarrae et Aragoniae et comitum Barchinonae*", papyrus belongs to the "*Rhind Mathematical Papyrus*, the parchment is a part of the "*Isaiah Scroll*" and the stone is a portion of the "*Stone cuneiform tablet with inscription of Ashurnasirpal II*".

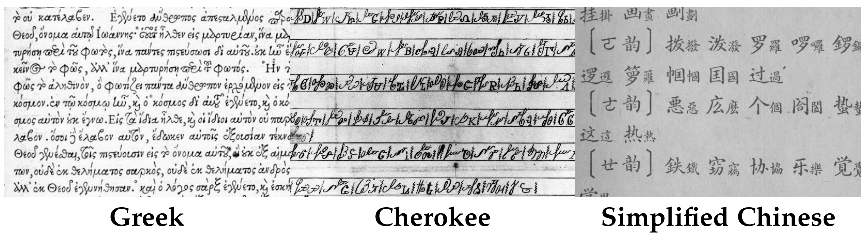

Greek
Cherokee
Simplified Chinese

Figure 1.4: Examples of three different types of script: a Greek alphabet, a Cherokee syllabary and a simplified Chinese logogram.

The features extracted from the image need to be robustness to noise introduced by document artifacts. They also need to be flexible as in handwritten documents there is a high variability in writing style. An image pre-processing step can be used to reduce these effects, e.g. an algorithm to remove document degradation artifacts [21, 22], or to normalize the document words by correcting their slope and slant [25]. Although these techniques are useful to improve the quality of the indexed word snippets, they will also introduce limitations on the scope of our algorithm. Therefore, we believe that a system that is robust and flexible on its own is preferable.

Finally, the system scalability is another aspect that needs our attention. The datasets commonly used to evaluate the performance of a word spotting system are composed of only a few dozen pages which contain several thousands of words. However, document collections in real world scenarios may be composed of thousands of pages and contain millions of words. Therefore, the visual representation of the word snippets needs to be compact and easy to compare in this kind of scenario. A vectorial representation which can be compared with a standard distance measure is suitable as standard machine learning techniques can be used to compress its size and to search for likely matches in sub-linear time.

1.3 CONTRIBUTIONS

The following is the list of contributions that we have made throughout this thesis:

1.3.1 *Study of Bag-of-Visual-Words*

In Chapter 2, we explore the use of the Bag of Visual Words (BoVW) framework to characterize the visual information of the word snippets. This framework has gained popularity among the document image analysis community, specifically as a representation of handwritten words for recognition or spotting purposes. Although in the computer vision field the BoVW method has evolved incorporating many enhancements in its representation, most of the approaches in the document image analysis domain still rely on a basic implementation of the BoVW method. In this thesis, we review different BoVW configurations and its applications to the keyword spotting task. We determine which are the best BoVW configurations and what parameters provide the largest retrieval gain. We demonstrate that a careful design of the BoVW visual signature has a drastic impact on the performance of the spotting system, making it comparable to more complex approaches. We are compare the proposed method against state-of-the-art keyword spotting methods on the well-known George Washington dataset and the Handwritten Keyword Spotting Competition 2014.

1.3.2 *Segmentation-free Word Spotting*

In Chapter 3, we present an efficient segmentation-free word spotting method that follows the query-by-example paradigm applied in the context of historical document collections. Taking into account the insights acquired in Chapter 2, we propose a patch-based framework where local patches are described by a BoVW model. The segmentation-free approach prevents us to infer the putative location of words in the document, so we follow a greedy approach where any possible location where a valid BoVW signature is found is considered. This requires us to reduce the footprint of the visual signature. First, by projecting the patch descriptors to a topic space with the Latent Semantic Analysis (LSA) technique and afterwards, by compressing the visual signature with Product Quantizer (PQ) methods. This allows us to efficiently index the document information both in terms of memory and time. The proposed method is evaluated using four different collections of historical documents achieving good performances both on handwritten and typewritten scenarios. The yielded performances outperform the recent state-of-the-art keyword spotting approaches.

1.3.3 *Multi-modal Word Spotting*

In Chapter 4, we explore the usage of additional information to create a query-by-X spotting system where word snippets are indexed and retrieved using different information modalities. Besides the visual BoVW signature, we consider two additional sources of information. A textual signature where words are represented by a codebook of n-grams and an audio signature where words are represented by a Bag-of-Audio-Words (BoAW) signature. The main idea of our approach is to find a projection that transforms two different signatures into a common feature space, so we can characterize word snippets using either information modality. The goal is to create a word snippet index using only visual information and then query them only with textual or audio information, i.e. obtaining a query-by-string or a query-by-audio word spotting system. Additionally, the process can be used in reverse and retrieve the audio utterance from visual queries. The proposed method generates a vectorial signature that can be used together with state-of-the-art indexation structures can be used in large-scale scenarios. The proposed method is evaluated using a collection of historical documents outperforming state-of-the-art performances.

1.3.4 *Supervised Codebook Generation*

In Chapter 5, we focus on the codebook generation and descriptor encoding steps. The BoVW codebooks used up until now are created in an unsupervised manner. The main advantage of this approach is that no annotated data is needed to generate the BoVW signature, which is preferable as annotating enough information to train a codeword model is a costly process. On the other hand, an unsupervised codebook also has its drawbacks. Without feedback about how good are the codewords of the codebook in representing the word images, we can only rely on retrieval performance measures to determine which parameters are the best for a certain document. Also, unsupervised clustering algorithm does not offer any guarantee on the quality of the clusters so, a codebook created from some document images does not have properly represent a different set of images. This means that we have to retrain our codebook each time we want to index a document corpus. Therefore, in this chapter we propose the use of synthetic data to generate a database agnostic codebook which remove the need to retrain the codebook. The only constrain imposed by the codebook is that the symbols used to generate the codebook match those contained by the document, i.e. that the codebook and the document use the same script type. The use of synthetic data also allows to easily incorporate semantic information in the codebook generation. So, the proposed method is able to determine which set of codewords have a semantic representation of the descriptor feature space. This eliminates the need to generate multiple codebooks with different parameters as the best parameter configuration can be determined automatically.

In this chapter, we also address the computational cost of encoding a descriptor into a visual word. In the methods presented up until now, the encoding process has a linear complexity with respect the amount of codewords of the codebook. This dominates the complexity obtaining BoVW signatures. Therefore, we propose a different approach where we use two codebooks. We keep the traditional BoVW codebook which contains the codewords but, we incorporate a new Hierarchical k-Means (HKM) codebook that is used to approximate the descriptor feature space. This HKM codebook acts as a look up table and its leaves represents an approximated version of the original descriptor instead of codewords. Following this approach, we can pre-compute the encoding of each approximated descriptor and then encode word snippet descriptor in sub-linear time. Experimental results show that the resulting codebook attains a state-of-the-art performance while having a more compact representation.

1.3.5 *Compact Character Probability Signatures*

In this chapter, we focus on the creation of compact visual signature. The BoVW signatures created so far have a high number of dimensions as signature size depends on the codebook size and the number of Spatial Pyramid Matching (SPM) spatial bins. In Chapter 5 we managed to obtain a more compact codebook and in Chapter 3 we use less spatial bins to reduce the signature dimensionality. Still, the dimensionality of the BoVW signature is too big to work with large collection of word images. Therefore, we still need to use data compression techniques like LSA or PQ codes to further reduce the signature dimensionality to an acceptable level. However, following this approach we are trading accuracy for memory.

In this chapter, we follow a different approach. Instead of using the BoVW signature to directly characterize the word snippets, we use it to detect the symbols present in a word image. We do not aim at a perfect symbol recognition but to an approximation that will roughly determine which symbols are present in each snippet. We believe that a rough detection is enough for our purposes as we do not want to transcribe the images but to characterize them. Therefore, classification errors are not going as important for our task as we expect them to be consistent, i. e.the same words is going to generate the same misclassification errors.

The word signature is formed by accumulating the probability of each character at different spatial bins. The signature represents therefore the probability that a character is found at a certain part of the word. The resulting signature is extremely compact signature. For example, it only requires 26 dimensions per spatial bin in a Latin script document. Combined with PQ quantizers, we are able to represent the entire George Washington dataset using only 9 728 bytes while maintaining a performance similar to the BoVW signatures of Chapter 2.

1.3.6 *Document Structure Analysis*

Finally, we present two document structural analysis algorithms that we have developed during technological transfer projects in the two appendixes of the thesis.

In Appendix A, we presents a learning-free segmentation algorithm that examines the structure of the scale space to detect text lines in document images. We take advantage that the second-order derivative gives a minimum response when a dark element (i. e.text) over a bright background has the same orientation as the filter. Since all Gaussian derivatives are steerable, we can easily compute the strength and orientation of the second-order derivative at each pixel of the image and at multiple scales. Line location and scale then can be easily obtained by merging the strongest responses which are near and have a coherent orientation. This approach gives us a performance similar to the state of the art methods in publicly available datasets.

In Appendix B, we present a method to automatically separate static and variable content from administrative document images. We build a probabilistic template by aligning examples of the same document kind and estimating the likelihood that a pixel belongs to the static or variable category. In the extraction phase, the template is aligned with the incoming document and used to determine which zones belong to variable entries. We validate our approach on the public NIST Structured Tax Forms Dataset.

2 WORD SPOTTING WITH BAG OF VISUAL WORDS

The Bag of Visual Words (BoVW) framework has gained popularity among the document image analysis community, specifically as a representation of handwritten words for recognition or spotting purposes. Although in the computer vision field the BoVW method has been greatly improved, most of the approaches in the document image analysis domain still rely on the basic implementation of the BoVW method disregarding such latest refinements. In this chapter we present a review of those improvements and its application to the keyword spotting task. We thoroughly evaluate their impact against a baseline system in the well-known George Washington dataset and compare the obtained results against nine state-of-the-art keyword spotting methods. In addition, we also compare both the baseline and improved systems with the methods presented at the Handwritten Keyword Spotting Competition 2014.

2.1 INTRODUCTION

Keyword spotting can be defined as the pattern recognition task aimed at locating and retrieving a particular keyword within a document image collection without explicitly transcribing the whole corpus. Its use is particularly interesting when applied in scenarios where Optical Character Recognition (OCR) performs poorly or can not be used at all, such as in historical document collections, handwritten documents, etc. Being a mature research problem [16], many different keyword spotting approaches have been proposed thorough the years.

In the document image analysis literature, we can distinguish two different families of keyword spotting methods depending on the representation of the handwritten words [26]. On the one hand, *sequential* word representations [27] describe handwritten words as a time series by using a sliding window in the writing direction. On the other hand, *holistic* word representations [28] extract a single feature vector of fixed dimensionality that characterizes the word as a whole.

Sequential word representations exploit the sequential nature of handwritten words formed by the concatenation of individual characters. However, since the size of the word's descriptors will depend on the width of the word, two different words cannot be directly compared by means of a distance between points, but some sort of alignment technique has to be used instead. The seminal work by Kołcz et al. [29] achieved a breakthrough in the handwritten keyword spotting domain by proposing the use of the Dynamic Time Warping (DTW) method (often used in speech analysis) for nonlinear sequence alignment. The use of DTW together with profile features was popularized by the well-known works by Rath and Manmatha [30, 31] and Rath et al. [32] and many flavors of DTW-based handwritten keyword spotting methods appeared since those publications. Adamek et al. proposed in [33] to use DTW to align convexity and concavity features extracted from contours. Khurshid et al. presented in [34] a method that first aligned features at character level by DTW and then the resulting character prototypes are aligned at word level. Papandreou et al. [35], proposed an adaptive zoning description that can be matched by DTW. Besides direct matching strategies, learning-based methods have also been proposed over the years. Hidden Markov Models are the most widely used techniques to model the keywords' sequential features [36, 37, 38, 17, 39], although other machine learning approaches such as Neural Networks [13] have also been used in the keyword spotting domain.

Holistic word representations have also received some attention thorough the years. Their main advantage is that by representing handwritten words by feature vectors of fixed size, the alignment step (which usually is very time consuming) is bypassed, and thus, two handwritten words can be compared using standard distances, or any statistical pattern recognition technique. We can find many different holistic word descriptions used in the literature for keyword spotting tasks. For example, simplified versions of the shape context descriptor, have been used in example-based keyword spotting architectures by Lladós and Sánchez [40] or by Fernández et al. [41]. Zoning-based characteristics have also been widely used to represent word images holistically, e.g. [42, 43]. A combination of Histogram of Oriented Gradients (HOG) and Local Binary Pattern (LBP) descriptors has been proposed by Kovalchuk et al. in [44] in a segmentation-free keyword spotting scenario. A set of biologically inspired features formed by a cascade of Gabor descriptors was proposed by van der Zant and Schomaker in [45]. The combination of gradient, structural and concavity features was proposed by Srihari and Ball in [46]. All of these word representations present their strengths and weaknesses and is hard to argue that a set of features is steadily better than another. Although in the latest years a trend towards using gradient-based features can be appreciated [47].

2.1.1 *Keyword Spotting as an Object Recognition Task*

Since the publication of the SIFT method [48], the computer vision task of recognizing and finding objects in cluttered scenes has been driven by methods extracting local descriptors that are further matched between the query model and the scene images. Many authors from the document analysis field, understanding keyword spotting as being a particular case of the object recognition task, started to apply such keypoint matching techniques to the problem of keyword spotting [49, 50, 51, 52]. Such matching techniques have been either used to directly estimate similarities between word images, or by searching the query model image within full pages in segmentation-free scenarios. However, the keypoint matching framework presents the same disadvantage than the sequential methods since an alignment between the keypoint sets has to be computed.

In order to avoid exhaustively matching all the keypoints among them, the classic bag-of-words paradigm from the information retrieval field was reformulated as the Bag of Visual Words (BoVW) [53, 54]. Such paradigm yield an holistic and fixed-length image representation while keeping the discriminative power of local descriptors such as SIFT.

Soon enough, researchers from the document image analysis domain adapted such BoVW representations to the keyword spotting problem [55, 56, 57, 58, 59, 39, 60, 61], obtaining very competitive results. However, we have the feeling that although the computer vision community kept proposing improvements on the BoVW framework in the last years, in the document analysis field, such improvements are still scarcely used. As an exception, it is worth to cite the works from Shekhar and Jawahar [58], or our last contribution [62], where more complex BoVW setups are used for the keyword spotting task.

2.1.2 *Contributions and Outline*

In this chapter we are going to review different aspects of the BoVW framework, namely sparse coding, spatial pyramids, and power normalization and its application to the keyword spotting task. We will thoroughly evaluate the impact of such improvements as well as the different parameters of the BoVW method by comparing their performances against a baseline system. We will finally compare the obtained results against nine state of the art segmentation-based keyword spotting methods by using the well-known George Washington dataset. In addition, we also compare both the baseline and improved systems with the methods presented at the Handwritten Keyword Spotting Competition 2014.

The chapter is structured as follows, in Section 2.2, the different parts of the BoVW pipeline used to characterize the word images are presented. Then, the effects that each BoVW enhancement have in the performance of a keyword spotting system are evaluated in Section 2.3 and the results obtained by the system are compared with the state of the art in Section 2.4. Finally, we review the most important conclusions of the chapter in Section 2.5.

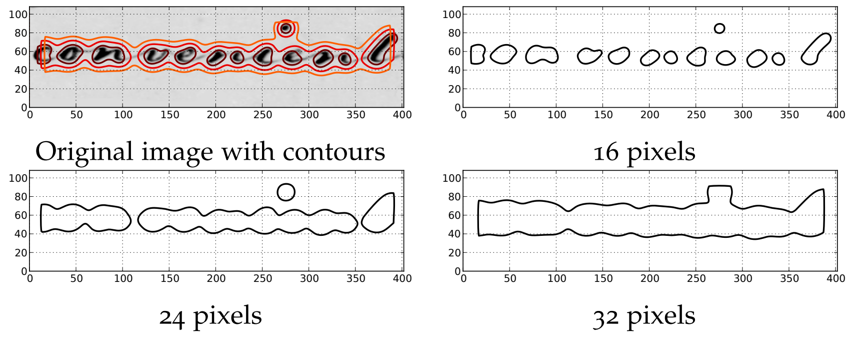


Figure 2.1: Norm of the descriptors extracted from regions of 16, 24 and 32 pixels width sampled at each pixel of the image. The bold contours encircle the regions where descriptors which have a large enough norm and are considered reliable.

2.2 BAG-OF-VISUAL-WORDS REPRESENTATIONS

In order to spot keywords in document images, we start by a layout analysis step devoted to segment the document images into individual words. The interested reader is referred to [10, 63]. Once the words are segmented, a visual signature is computed for each of them. The keyword spotting will be then performed by calculating the similarity between the description of the query word and all the descriptors of the words in the corpus. These visual signatures are created using a BoVW framework which has obtained good performances in keyword spotting tasks [57, 58].

The BoVW framework has many variants in the literature, but all of them can be roughly divided into four basic steps: *sampling*, *description*, *encoding* and *pooling*. In order to increase the retrieval performance of the spotting system, we need to carefully select the methods used at each step. In this chapter, we will mainly focus on the BoVW improvements that bring better word representations for recognition or spotting tasks.

2.2.1 *Sampling*

The first step is to select the regions of the image which contain meaningful information to describe the word snippets. Although covariant or salient region detectors can be used, it has been proven that the performance of BoVW representations is correlated with the number of sampled regions. For instance, Nowak et al. demonstrate in [64] that the larger the number of regions, the better the results. They show that the combination of several region detectors usually improves the performance of the BoVW framework, but this performance gain is related to the number of regions rather than the kind of sampled regions. Therefore, for our baseline implementation we decided to densely sample regions at different scales over the image instead of using a keypoint detector.

Regions are densely sampled using a fixed step and at different scales. The different scales are selected so that words are going to be modeled at different levels of detail: small regions will model portions of characters while large regions will model the relationships between characters.

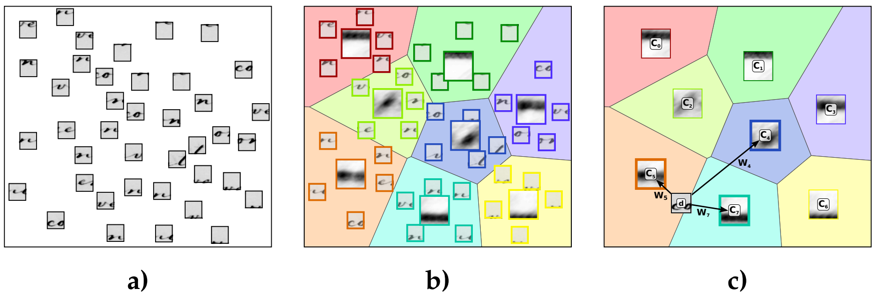


Figure 2.2: Codebook creation and descriptor encoding example: **a)** Descriptors are randomly sampled from the indexed images, **b)** the k-means algorithm is used to build the codebook and **c)** descriptors are encoded using sparse coding with the cluster centroids.

2.2.2 *Description*

Once regions have been sampled, we need to characterize them with a local descriptor. Although descriptors specifically tailored for document analysis can be used, gradient based descriptors have recently shown better performances in keyword spotting tasks [65, 57, 62].

We are going to use the HOG descriptor [66] to characterize the regions. This descriptor is derived from the SIFT descriptor [48], but it is more suited for dense sampling scenarios when rotation invariance is not needed. In our case, it is safe to assume that the orientation of the word images has been corrected by the word segmentation algorithm or intermediate slant correction steps. The HOG algorithm takes advantage of the information redundancy between overlapping regions, so that descriptors can be calculated at a much lower computational cost [67, 68].

Although the dense sampling strategy will generate a large amount of HOG descriptors, only reliable descriptors are eventually accepted. Since

HOG descriptors are based on gradient information, descriptors are more reliable when gradient vectors have a large module. Therefore, the norm of the descriptor can be used as a reliability indicator. For instance, Fig. 2.1 shows the norm of the HOG descriptors calculated at each pixel of the image. It can be appreciated that descriptors calculated near character locations have a high norm while descriptors sampled over other image regions have a low norm. Therefore, the BoVW signature can focus on the visual information from characters by filtering the descriptors depending on the value of their norm. The bold contours in the Fig. 2.1 encircle the zones where the descriptors have a norm higher than the used threshold. Descriptors which have a value lower than this threshold, i.e. descriptors outside the contours, are simply disregarded.

2.2.3 *Encoding*

After calculating the descriptors, we have to encode them into visual words. First, we need a codebook which quantizes the descriptor space into an arbitrary set of m salient regions. This codebook is created by randomly sampling descriptors from the indexed word snippets and using the *k*-Means algorithm to calculate m clusters. Then, a descriptor $\mathbf{d}_i$ is encoded by a vector $\mathbf{W}_i \in \mathbb{R}^m$ which weights the contribution of each codeword (i.e. cluster centroid). The most straightforward method to calculate $\mathbf{W}_i$ is to use hard-assignment [53], i.e. the weight vector has a single non-zero element corresponding to the nearest codeword to the descriptor.

This encoding approach has problems near the boundaries between codewords. Small changes in the descriptor may lead to a completely different visual words vector $\mathbf{W}_i$. This problem can be alleviated by using soft-assignment instead, i.e. encoding a descriptor using a weighted combination of codewords. Besides, combining the information of several codewords also reduces the information loss resulting of the descriptor quantization. Therefore, we decided to encode descriptors using the sparse coding technique proposed in [69], known as Locality-constrained Linear Coding (LLC). This method generates a compact BoVW signature that have a higher discriminative power than more complex representations [70].

Given a descriptor $\mathbf{d}_i$, the LLC method tries to find the linear combination of codewords which better approximates the original descriptor:

$$\mathbf{d}_i \approx \sum_{j=1}^{m} w_j \mathbf{C}_j, \tag{1}$$

where $\mathbf{C}_j$ is the j-th codeword and w_j its associated weight. Unlike other sparse coding algorithms, LLC emphases locality over sparsity and it only uses the t nearest codewords to encode a descriptor. This ensures that the resulting encoding is locally smooth, so that similar descriptors are likely to be encoded using the same codewords. Therefore, the LLC encoding is more robust compared to other sparse coding solutions. Another advantage is that the weights $(w_1, w_2, \dots, w_m)$ can be derived analytically. Hence, the computational cost is drastically reduced compared to other sparse coding algorithms which require computationally demanding optimization procedures to find a solution. Then, a descriptor $\mathbf{d}_i$ is encoded by searching the t nearest codewords and using the LLC algorithm to calculate the weights vector $\mathbf{W}_i = (w_1, w_2, \dots, w_m)$.

An example of the codebook creation and descriptor steps is summarized in Fig. 2.2. The randomly sampled descriptors of Fig. 2.2.a) are clustered into eight clusters in Fig. 2.2.b). In Fig. 2.2.c), we can see that the closest codewords to the descriptors $\mathbf{d}_i$ are $\mathbf{C}_4$, $\mathbf{C}_5$ and $\mathbf{C}_7$. Using hard-assignment, the descriptor will be encoded as $\mathbf{W}_i = (0,0,0,0,0,1,0,0)$ as its nearest centroid is $\mathbf{C}_5$. On the other hand, the LLC algorithm will calculate the weights w_4, w_5 and w_7 so that $\mathbf{d}_i \approx w_4\mathbf{C}_4 + w_5\mathbf{C}_5 + w_7\mathbf{C}_7$ and the resulting encoding will be $\mathbf{W}_i = (0,0,0,0,w_4,w_5,0,w_7)$. Notice that the encoded descriptor is close to a boundary between codewords, so that a small variation of the descriptor can shift the closest codeword from $\mathbf{C}_5$ to $\mathbf{C}_7$. This would result in a completely different encoding when hard-assignment is used. In contrast, the LLC algorithm will generate a similar weight vector $\mathbf{W}_i$ since it still uses the same codewords and the weights w_4, w_5 and w_7 are slightly different.

2.2.4 *Pooling*

Once descriptors are encoded into visual words, the BoVW signature is obtained by simply accumulating the weight vectors $\mathbf{W}_i$:

$$\mathbf{s} = \sum_{i=1}^{N} \mathbf{W}_i, \tag{2}$$

where N is the number of valid descriptors extracted from the word image. In the following, we are going to see how to improve this representation.

2.2.4.1 *Spatial information*

In Eq. 2, visual words are accumulated without taking into account their spatial location, so the signature lacks any spatial information. However, spatial information is quite important in keyword spotting tasks since it helps to reduce the perceptual aliasing problem. Different instances of the same character are expected to be represented by similar visual words. Hence, the obtained BoVW signatures mostly depends on the characters that form the word, and it is possible that dissimilar words are represented by similar signatures when spatial information is not taken into account. For instance, anagrams will obtain a very similar visual signature in this scenario.

This problem can be addressed by using using the Spatial Pyramid Matching (SPM) technique proposed by Lazebnik et al. in [71] in order to add some spatial information into the unstructured BoVW model. This method roughly takes into account the visual word distribution over the image by creating a pyramid of spatial bins.

The spatial pyramid defines an initial set of horizontal P_x^0 and vertical P_y^0 partitions which create $P_x^0 \times P_y^0$ spatial bins. Then, these spatial bins are further divided into P_x horizontal and P_y vertical partitions at each level of the pyramid. Therefore, a spatial pyramid of L levels creates a collection of overlapping D_{sp} spatial bins, where

$$D_{sp} = P_x^0 P_y^0 \sum_{l=0}^{L-1} (P_x P_y)^l. \tag{3}$$

The final BoVW signature $\overline{\mathbf{W}}_i$ is created by independently accumulating the visual words for each spatial bin obtaining a $D_W = mD_{sp}$ dimensions descriptor. The amount of visual words assigned to each bin is lower at

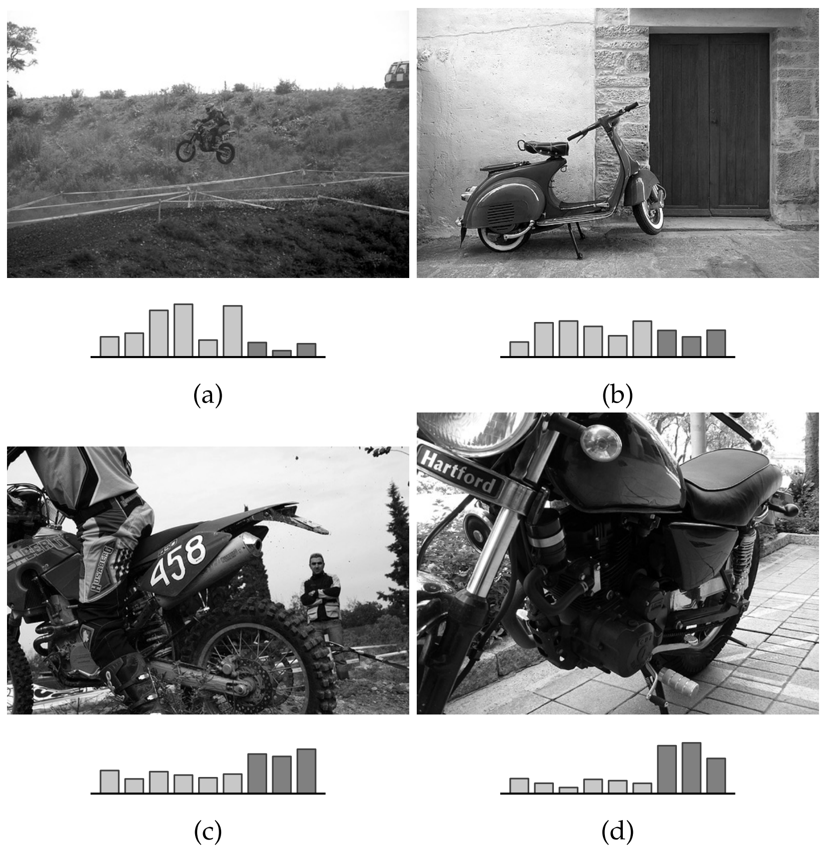


Figure 2.3: The bins corresponding to the visual words sampled over the *motorbike* objects increase their contribution linearly to the size of the object. Therefore, the score of a *motorbike* linear classifier will also increase linearly from (a) to (d). A better behavior would be that the score sharply increases from (a) to (b) but keeps a similar score from (b) to (d).

higher levels of the pyramid, due to the fact that the spatial bins are smaller. This is compensated by multiplying the contribution of each visual word to each spatial bin by the factor $s_l = P_x^0 P_y^0 (P_x P_y)^l$.

2.2.4.2 *Normalization*

Once we have obtained $\overline{\mathbf{W}}_i$, we can normalize the contribution of each visual word in order to obtain a better representation. First, we can reduce the importance of overrepresented visual words by using the method proposed by Perronnin et al. in [72] which applies the following normalization function to each bin of the signature:

$$g(x) = sign(x)|x|^{\alpha}, \tag{4}$$

where $0 < \alpha < 1$ is the power normalization factor. The power normalization improves the BoVW model since it removes the assumption that visual words come from an identically and independently distributed population [73]. The amount of visual words that represent an object depends on the size of that object in the image. For example in Fig. 2.3, we can see that the amount of visual words representing the *motorbikes* varies depending on how close they are to the camera. This may cause that the object of interest is underrepresented in the BoVW signature, resulting in retrieval or classification errors. These problems are less likely to happen in keyword spotting as the scale of the words is similar. However, avoiding the i.i.d. assumption is still important as the frequency of visual words is highly correlated to the characters forming the word. For instance, the visual words modeling the character *e* will be overrepresented in words like *freeze* or *exceed* and hence their visual signature is going to be somehow similar. Therefore, by lessening the contribution of the overrepresented visual words, we are highlighting the other visual words and making both signatures more dissimilar.

Finally, the BoVW signature is ℓ_2-normalized to account that the amount of visual words accumulated in $\overline{\mathbf{W}}_i$ may change between two instances of the same word due to scale difference or image noise.

2.3 BOVW PARAMETER EVALUATION

In order to evaluate the different parameters of the BoVW signature in a keyword spotting framework, we use a straightforward method to index and retrieve the word snippets from a database. The image signatures are indexed using an inverted file structure taking advantage that the BoVW representation is sparse, specially when SPM is used. The system is evaluated by calculating the mean Average Precision (mAP) score from the ranked list obtained by sorting in ascending order the Euclidean distances between the query and the indexed signatures.

2.3.1 *Experimental Setup*

The keyword spotting system is evaluated in the George Washington dataset described in [31]. This dataset consists of 20 handwritten pages with a total of 4860 words written by several Washington's secretaries. Although it was written by several authors, the writing style is pretty uniform and shows less variation than typical multi-writer collections. The database provides a set of word bounding-boxes with their transcription. These bounding-boxes are obtained using the segmentation algorithm proposed in [63] by Manmatha and Rothfeder.

The baseline BoVW configuration densely samples the HOG descriptors at every 5 pixels and at three different scales: 20, 30 and 45 pixel wide regions. The codebook has $m = 1024$ codewords and the histogram is created without using any improvement, i.e. descriptors are encoded using hard-assignment, no spatial information is added and the power normalization is not used (i.e. $\alpha = 1$). At each step of the experimental evaluation, we are going to assess the effects that a single improvement has on the spotting performance of the system. These evaluations are conducted by calculating the mAP score using two different setups:

- **Setup A**: Use as queries all words in the collection which appear at least twice.
- **Setup B**: Use as queries only words which have at least ten occurrences and with 3 or more characters.

The configuration **Setup A** is defined to use all possible word snippets as queries while the configuration **Setup B** cast queries which are more likely to be used in a real world scenario (e.g. avoiding short queries like *"a"* or *"to"*).

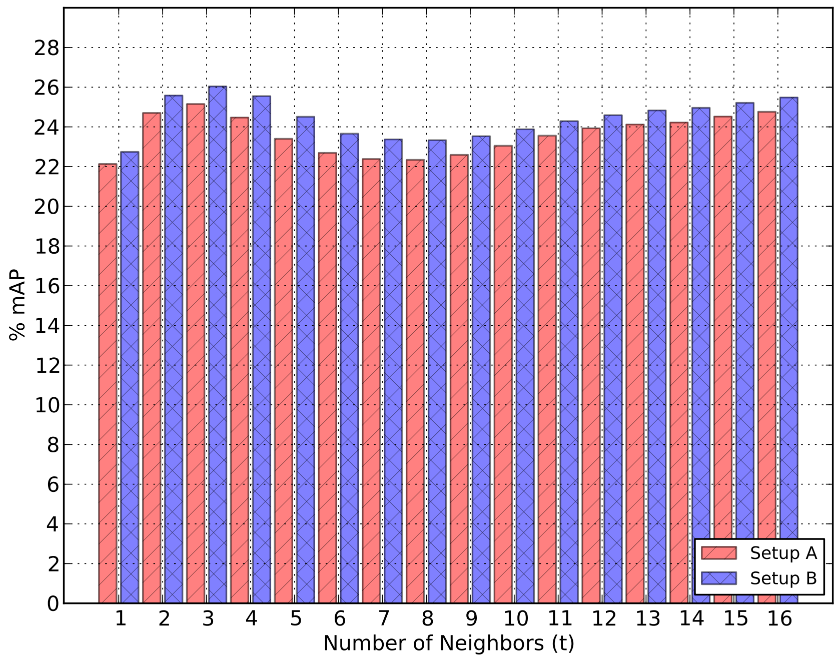


Figure 2.4: mAP score obtained using different number of neighbors with LLC.

In both setups, word snippets which have been discarded as queries are still used as distractors in the database. Therefore, the system has a 100% recall since it always returns a ranked list with all the 4859 elements, corresponding to all indexed images except the query.

2.3.2 *LLC Encoding*

First, we evaluate the effects of using a different amount of nearest neighbors t in the LLC encoding step. The mAP scores obtained while testing from 1 to 16 nearest neighbors are shown in Fig. 2.4. Note that using a single nearest neighbor corresponds to hard-assignment encoding, since only the closest codeword is used.

The results show that using LLC encoding slightly increases the performance of the word spotting system. The best results are obtained when three nearest neighbors are used to encode the descriptors: for **Setup A** the mAP score improves from 22,13% to 25,15% while for **Setup B** the score raises from 22,74% to a 26,04%. Although the selected number of neighbors may seem small, this result is coherent with the results shown in the original LLC paper [69] where using a small number of neighbors results in a better performance than when a large number of neighbors is employed. In the remaining experiments, we are going to use 3-nearest neighbors for the encoding step with LLC.

2.3.3 *Spatial Pyramids*

After evaluating the encoding, we are going to evaluate the importance of spatial information in the BoVW signature. In Table 2.1 we can see that the addition of spatial information greatly increases the performance of the system. In both setups, the mAP score increases two and a half times between the orderless representation and the best spatial pyramid configuration. From

Table 2.1: mAP score obtained using different spatial configurations.

P_x^0	P_y^0	P_x	P_y	L	D_{sp}	D_W	**Setup A**	**Setup B**
1	1	1	1	1	1	1024	25,15%	26,04%
1	1	2	2	2	5	5120	40,96%	43,43%
1	1	2	2	3	21	21504	51,49%	54,03%
1	1	2	2	4	85	87040	57,65%	60,47%
1	1	3	2	2	7	7168	46,45%	48,79%
1	1	3	2	3	43	44032	58,09%	60,91%
1	1	3	2	4	259	265216	61,11%	64,26%
1	1	2	3	2	7	7168	42,45%	45,05%
1	1	2	3	3	43	44032	51,38%	53,91%
2	2	2	2	2	20	20480	55,46%	58,53%
3	2	2	2	2	30	30720	60,32%	63,56%
2	3	2	2	2	30	30720	55,71%	58,80%
2	2	3	3	2	40	40960	59,27%	62,43%
3	3	3	3	2	90	92160	62,01%	65,46%
3	1	2	2	2	15	15360	58,39%	61,43%
1	3	2	2	2	15	15360	43,37%	45,97%
3	1	2	1	2	9	9216	55,32%	58,50%
3	1	2	1	3	21	21504	58,98%	62,27%
3	1	3	2	2	21	21504	60,38%	63,66%
3	**2**	**3**	**1**	**2**	**24**	**24576**	**61,33%**	**64,75%**

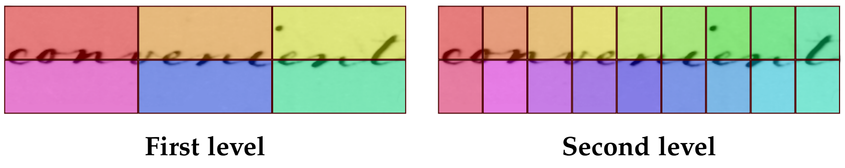


Figure 2.5: Distribution of the spatial bins in the two levels of the spatial pyramid.

the obtained results, we can see that horizontal partitions are more important than vertical partitions. This is to be expected as adding more horizontal partitions helps to increase the representation of the word characters. For instance, in Fig. 2.5 we can see an example of the spatial bins defined by a two level spatial pyramid. In the first level, spatial bins roughly model syllables while in the second level bins are smaller and they model individual characters.

After evaluating the obtained results, we have selected a two level SPM with 3×2 spatial bins in the first level and 9×2 in the second (row in bold in Table 2.1) as the SPM configuration used in the following experiments. With this configuration the retrieval performance grows from $22,15\%$ to $61,33\%$ using **Setup A** and from $26,04\%$ to $64,75\%$ in **Setup B**. Although there is another configuration which obtains better results, the selected configuration offers a better compromise between performance and dimensionality growth.

Additionally, we have re-checked the effects of LLC by disabling it and the performance is slightly reduced to $60,62\%$ and $64,16\%$ respectively.

2.3.4 *Power normalization*

Concerning power normalization, the retrieval performance obtained using different α power values can be found in Fig. 2.6. The results show that the use of power normalization also obtains an important boost of performance of the system. It attains the maximum performance of $68,27\%$ mAP at $\alpha = 0,4$ for **Setup A** and of $72,20\%$ mAP at $\alpha = 0,3$ for **Setup B**. Since the performance

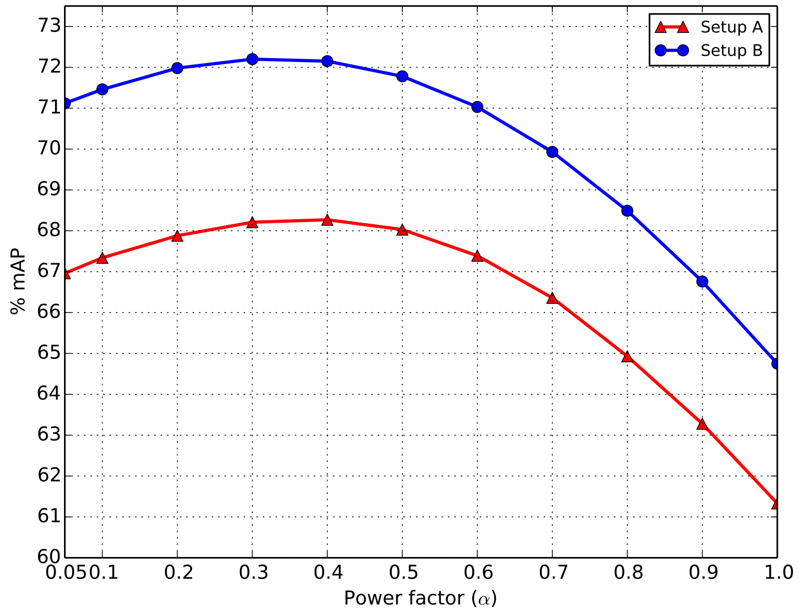


Figure 2.6: Effect of the power norm to the performance of the word spotting system.

is pretty similar for $\alpha = 0,3$ and $\alpha = 0,4$, we are going to use a power normalization of $\alpha = 0,35$ for both setups in the following experiments.

2.3.5 *Codebook size*

All the experiments until now have used a relatively small codebook of 1024 codewords. Since the performance usually increases as larger codebook are used, we compare the effects of different codebook sizes in Fig. 2.7.

The performance of the system keeps improving until it saturates for the $m = 8192$ codebook. For larger codebooks, the performance degrades, because descriptor quantization errors start to be too frequent. Since the mAP score increase is marginal between codebooks of $m = 4096$ and $m = 8192$, we decided to use the 4096-codebook for the last experiment.

It is worth noting that the mAP score attained by the smallest codebook (with $m = 32$ codewords) in Fig. 2.7 doubles the score obtained by the baseline configuration: $45,85\%$ against $22,13\%$ for **Setup A** and $52,07\%$ versus $22,74\%$ for **setup B**. Although the BoVW signature is more compact and it has 768 dimensions compared to the 1024 dimensions of the baseline configuration, the use of LLC, SPM and power normalization greatly increase the spotting capabilities of the system.

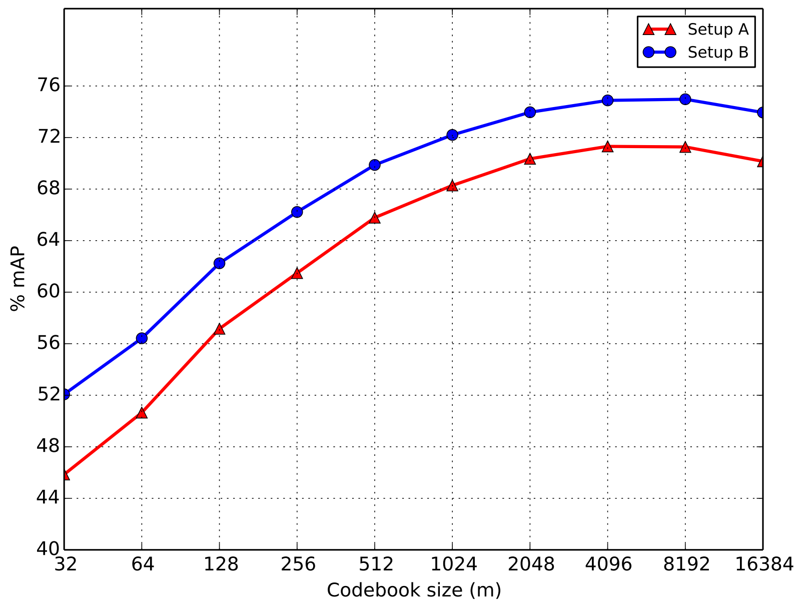


Figure 2.7: Evolution of the mAP score while increasing the size of the codebook.

2.3.6 *Descriptor sampling*

Subsequently, we evaluate in Table 2.2 the effects of using different the descriptor sampling parameters. We have evaluated the use of larger regions to check which information is more important to characterize word images. The results show that it is more important that visual words model character fragments rather than the relationships among them. We have also evaluated the sampling density, observing that the performance increases as the descriptors are sampled more densely. Since the performance gap between the two configurations is quite important, it is safe to assume that works that used larger regions (e.g. our previous segmentation-free keyword spotting method [57]) will improve their performance by simply using smaller regions.

Table 2.2: mAP scores obtained when modifying the descriptor sampling parameters

Region size			**Step**	**Setup A**	**Setup B**
small	*medium*	*large*			
40	60	90	10	39,94%	43,71%
			8	43,37%	47,54%
			5	47,24%	51,61%
			4	47,75%	52,20%
			3	47,90%	52,35%
20	30	45	10	54,23%	58,25%
			8	62,94%	66,85%
			5	71,31%	74,88%
			4	72,35%	75,86%
			3	**72,98%**	**76,45%**

2.3.7 *Summary of the Results*

Table 2.3: Summary of the improvements over the baseline BoVW implementation with the gains in performance

	Setup A	**Setup B**	**Cost**
Baseline	22.13%	22.74%	
LLC	25.15% (↑ 13.65%)	26.04% (↑ 14.51%)	Computational complexity
SPM	61.33% (↑ 177.14%)	64.75% (↑ 184.74%)	Descriptor size
Power normalization	68.27% (↑ 208.50%)	72.20% (↑ 217.50%)	None
Codebook size	71.31% (↑ 222.23%)	74.97% (↑ 229.68%)	Descriptor size
Descriptor sampling	72.98% (↑ 229.78%)	76.45% (↑ 236.19%)	Computational complexity

Finally, we present in Table 2.3 a summary of the results obtained by the different improvements over the baseline BoVW implementation. Besides the performance gains for each of the improvements, we also report the extra cost that each of the different steps might have. Both using sparse coding through LLC and tuning the descriptor sampling stage have a minimal cost in terms of computational complexity. In the encoding step the weights of the LLC have to be calculated instead of just using a hard-assignment strategy. When using denser and smaller HOG descriptors, the amount of descriptors to process per word image is increased, and thus the whole encoding and pooling steps are more complex to compute. When using an SPM configuration, the dimensionality of the word descriptors is exponentially increased, so one has to find a good trade-off between discriminative power and efficiency of the overall system in terms of speed and memory usages. The same goes for the codebook size, although we have seen that in that case, the system's performance degrades when starting to use too large dictionaries. Finally, the use of power normalization has no extra cost with regard to the baseline BoVW implementation. After the final experiment, the performance of the system has increased a 230% (from $22,13\%$ to $72,98\%$) in **Setup A** and a 236% (from $22,74\%$ to $76,45\%$) in **setup B**.

2.4 PERFORMANCE COMPARISON WITH THE STATE OF THE ART

Now that we have shown that the performance of the BoVW model greatly varies depending on the methods used to create the signature, we can compare the baseline and enhanced BoVW implementations with the state of the art. In order to demonstrate that the enhanced BoVW implementation is competitive against most spotting methods, we are going to compare it against method which used the popular George Washington dataset and the H-KWS 2014 Competition benchmark [74] to assess their performance.

2.4.1 *George Washington Dataset*

The George Washington dataset has become a de-facto standard to evaluate handwritten recognition and keyword spotting methods. In order to conduct this comparison, we will only focus on segmentation-based methods to focus only on the performance of the word snippet descriptor. Segmentation-free and line-based methods follow a more general approach that is likely to obtain worse results due to processing a larger amount of information or due to errors introduced while locating words in the document image.

Table 2.4: Comparison of the performance attained by the system using the baseline and final BoVW configurations against the results reported by each work. The methods in the first half are exemplar-based methods while second half methods are learning-based.

Reference	**Experimental Setup**	**Originally Reported**	**Baseline BoVW**	**Enhanced BoVW**	**Measure**
Example-based methods					
Rath and Manmatha [30]	10 good quality pages (2381 queries).	40.9%	28.1%	77.2%	mAP
Rothfeder et al. [75]	10 good quality pages (2381 queries).	36.2%	28.1%	77.2%	mAP
Kovalchuk et al. [44]	Same configuration as **Setup B**	66.3%	22.7%	76.5%	mAP
Wang et al. [52]	Same configuration as **Setup B**	17.5%	22.7%	76.5%	mAP
Howe [76]	4-folds: 3 train and 1 test folds. All non-stop words used as queries.	93.4%	55.0%	91.8%	Mean Precision
		78.9%	19.0%	79.0%	P@R=100%
Learning-based methods					
Howe et al. [77]	20-folds: 19 train and 1 test fold.	79.5%	38.5%	81.9%	mAP
Rodríguez-Serrano and Perronnin [38]	5-folds: 1 train, 1 validation and 3 test folds.	53.1%	23.6%	74.0%	mAP
Liang et al. [78]	5-folds: 4 train and 1 test folds. 38 words are selected as queries.	67.0%	39.9%	84.5%	mAP at rank 10
Almazán et al. [79]	5-folds: 1 train, 1 validation, 3 test folds. Words in the test set are used as queries.	85.7%	24.0%	74.3%	mAP

Although the George Washington dataset is widely used, there is not an standard experimental setup, and each work adapts it to the needs of their proposed algorithm. For instance, learning based algorithm usually use cross-validation to avoid evaluating the method on the same data used to fit their model. This reduces the amount of queries since query words must appear both in train and test folds. Also, the number of distractors is reduced as the number of putative results is trimmed. These changes make that a direct comparison between methods is not possible. Therefore, we have recalculated the results obtained by the proposed method employing the experimental setup used in each paper.

A brief summary of the experimental setup and the performance comparisons are shown in Table 2.4. We can see that all exemplar-based algorithms but the method proposed by Howe [76] do not use cross-validation. In [76], the author compares his method with the learning-based method proposed by Frinken et al. in [13], hence the use of cross-validation. Also, most works use mAP to asses their performance, only Liang et al. [78] and Howe [76] use other measures. In [78] the mAP is calculated only using the ten best results of each query. In [76], the author first calculates the mean of the precision and recall curves for all the queries and then reports the area under this curve and the precision at full recall. Finally, learning-based methods use the training set as queries, except the work by Almazán et al. [79]. In this work, the authors use the test set as a completely new database so that both query and indexed images have not been seen in the training phase of the algorithm.

In the comparison table, we can see that the obtained results using the baseline BoVW implementation are significantly worse than the compared works. Only in Wang et al. [52] the baseline implementation obtains a better result. On the other hand, the results attained by the system when using the enhanced BoVW implementation are significantly better than most of the compared works. The proposed BoVW signature is only outperformed by the method proposed by Almazán et al. [79] while Howe [76] have comparable results. It is worth to note, that the method from [79] use a Canonical Correlation Analysis step over a BoVW signature, aimed at finding correlations between visual words and word transcriptions. Obviously, the integration of machine learning techniques over BoVW representations is expected to produce better results than a simple distance among descriptors [62]. Concerning the method by Howe [76], we have to consider the computational complexity of the keyword spotting system. The vectorial nature of BoVW allows to apply standard indexation techniques for an efficient retrieval. In addition, [76] needs an alignment step to compute the similarity between the query and the document's words.

2.4.2 *H-KWS 2014 Competition*

The H-KWS 2014 [74] is a recently proposed benchmark dataset to compare the advances in keyword spotting. It analyzes both segmentation-based and segmentation-free algorithms using performance measures frequently found in the literature. This benchmark is composed by the Bentham and Modern datasets. The Bentham dataset is a collection of 50 images written by Jeremy Bentham himself as well as his secretarial staff. This collection is similar to the George Washington dataset in the sense that the calligraphic differences between different instances of the same word are minimal. The Modern dataset is a collection of 100 handwritten pages written by several writers. The writers were asked to copy a text written in English, German, French or Greek. Therefore, this dataset has a high calligraphic variety and it uses different scripts.

The comparison between the results obtained by the proposed basic and enhanced configurations and the methods which participated in the segmentation-based track of the H-KWS 2014 competition are shown in Table 2.5. The results of this table have been obtained using the evaluation tool provided with the benchmark[1]. As we have seen in the George Washington comparison, Kovalchuk et al. [44] and Howe [76] are exemplar-based while Almazán et al. [79] is a learning-based algorithm. This algorithm is trained using the annotations of George Washington dataset while creating the model for the Bentham dataset and using the IAM dataset for the Modern dataset.

In Table 2.5, we can see that the baseline configuration obtains rather bad results whereas the enhanced configuration is competitive when compared with the other methods. Specifically, looking at the mAP indicator, the enhanced configuration only obtains slightly better results than Howe [76] in the Bentham dataset while in the Modern dataset it is only surpassed by Almazán et al. [79].

1 H-KWS 2014 competition homepage: http://vc.ee.duth.gr/h-kws2014/

Table 2.5: Comparison of the performance attained by the system using the baseline and enhanced BoVW configurations with the methods that participated in the Handwritten Keyword Spotting Competition 2014 (H-KWS 2014) competition.

	Bentham Dataset			
Method	P@5	MAP	NDCG (Binary)	NDCG
G1 (Kovalchuk et al. [44])	0.738	0.524	0.742	0.762
G2 (Almazán et al. [79])	0.724	0.513	0.744	0.764
G3 (Howe [76])	0.718	0.462	0.638	0.657
Baseline	0.491	0.292	0.565	0.578
Enhanced	0.629	0.465	0.707	0.723
	Modern Dataset			
Method	P@5	MAP	NDCG (Binary)	NDCG
G1 (Kovalchuk et al. [44])	0.588	0.338	0.611	0.612
G2 (Almazán et al. [79])	0.706	0.523	0.757	0.757
G3 (Howe [76])	0.569	0.278	0.484	0.485
Baseline	0.231	0.091	0.349	0.350
Enhanced	0.619	0.389	0.680	0.681

The results obtained in both comparisons stress the fact that the use of simple improvements of the BoVW signatures can lead to a great boost in performance of keyword spotting systems and that it is possible to attain better results than more complex solutions.

2.5 CONCLUSIONS

In this chapter we have studied the effects of different BoVW representations for a handwritten word spotting task. Although the use of BoVW has gained attention during the course of this thesis as a way to represent segmented handwritten words, most of the literature still uses a basic implementation of the BoVW framework, neglecting the latest improvements of such method.

We have reviewed the improvements that we believe are more suitable for word representation and, we have seen which of them can lead to a huge boost on the spotting performance of the system. Some of these improvements come at a negligible increase on the system's cost, some do not have a noticeable effect while others boost the retrieval performance at the cost of increasing the memory needed to store a signature.

Overall, the most important increase in performance came from the use of spatial pyramids, specifically when selecting a configuration that split the handwritten words across the horizontal axis. We believe that such performance boost comes from the fact that this SPM configuration led the descriptor to encode sequential information of the word, i.e. which character comes before another, mimicking the information that is encoded in sequential word representations, but while preserving the advantage of holistic word representations. This performance boost comes however with a significant increase of the signature's memory footprint. In later chapters, we have to take this factor into account when creating systems that handle document collections with tens- or hundreds of thousands of word images.

3 SEGMENTATION FREE WORD SPOTTING

In this chapter we present an efficient segmentation-free word spotting method, applied in the context of historical document collections, that follows the query-by-example paradigm. We use a patch-based framework where local patches are described by a bag-of-visual-words model powered by SIFT descriptors. By projecting the patch descriptors to a topic space with the Latent Semantic Analysis technique and compressing the descriptors with the Product Quantization method, we are able to efficiently index the document information both in terms of memory and time. The proposed method is evaluated using four different collections of historical documents achieving good performances both on handwritten and typewritten scenarios. The yielded performances outperform the recent state-of-the-art keyword spotting approaches.

3.1 INTRODUCTION

Nowadays, in order to grant access to the contents of digital document collections, their texts are transcribed into electronic format so users can perform textual searches. When dealing with large collections, automatic transcription processes are used since a manual transcription is not a feasible solution. In the context of digital collections of historical documents, handwriting recognition strategies [27] are applied to achieve an automatic transcription since most of those documents are manuscripts. However, handwriting recognition often do not perform satisfactorily enough in the context of historical documents. Documents presenting severe degradations or using ancient glyphs might difficult the task of recognizing individual characters, and the lexicon definition and language modeling steps are not straightforwardly solved in such context. *Keyword spotting* has become a crucial tool to provide accessibility to historical collection's contents. Keyword spotting can be defined as the pattern recognition task aimed at locating and retrieving a particular keyword from a document image collection without explicitly transcribing the whole corpus.

Two different families of keyword spotting methods can be found in the document image analysis literature. On the one hand, *learning-based* methods such as [38, 17, 13], use supervised machine learning techniques to train models of the words the user wants to spot. Those models are then used to classify whether an incoming document image contains or not one of the sought words. On the other hand, *example-based* methods such as [31, 80, 65], receive as input an instance of the keyword the user wants to retrieve from a previously indexed document image collection. Learning-based methods are preferred for applications where the keywords to spot are a priori known and fixed. If the training set is large enough they are usually able to deal with multiple writers. However, the cost of having a useful amount of annotated data available might be unbearable in most scenarios. In that sense methods running with few or none training data are preferred. It is the case of example-based methods, which are specially interesting when it is not feasible to obtain labeled data. They also present the advantage that the user is free to cast whatever query keyword he wants and is not restricted to the set of modeled words.

However, one of the main drawbacks of keyword spotting methods, either learning or example-based, is that they usually need a layout analysis step that segments the document images into words [77, 75, 30, 78] or text lines [13, 80]. But this segmentation step is not always straightforward and might be error prone. In fact, although word and text line segmentation is a quite mature research topic, it is far from being a solved problem in critical scenarios dealing with handwritten text and highly degraded documents [10, 81]. Any segmentation errors affect the subsequent word representations and matching steps. This dependence on a good word segmentation motivated the researchers of the keyword spotting domain to recently move towards complete segmentation-free methods [65, 82, 83, 39, 76]. The literature dealing with segmentation-free keyword spotting methods is rather scarce since it is a relatively new and unexplored research topic. However, we strongly believe that bypassing the segmentation step is a must in the context of historical document collections where achieving a perfect word or text line segmentation might be unfeasible. So, architectures that dismiss the segmentation step present a clear asset in the context of historical documents.

In addition, quite often, keyword spotting methods rely on computing expensive distances exhaustively between the query and the words in the collection such as Dynamic Time Warping (DTW) [31] or learning and applying complex models such as Hidden Markov Model (HMM) [38, 17, 39] or neural networks [13]. In that sense, in large-scale scenarios, the complexity issue should to be taken into account by proposing efficient and scalable methods both in terms of memory usage and response time.

In this chapter we present an efficient segmentation-free keyword spotting method based on a Bag of Visual Words (BoVW) model powered by SIFT descriptors in a patch-based framework. Since an explicit word segmentation is avoided, the proposed method can be applied in scenarios where word segmentation might be problematic such as documents that do not follow a classical Manhattan layout, or even be used to spot handwritten annotations that do not follow a regular text line structure. Other preprocessing steps such as binarization, slant correction, etc. are also avoided, directly processing the raw image. The proposed architecture follows the query-by-example paradigm and do not involve any supervised learning method, thus do not rely on any previous content transcription. Our proposal adapts techniques that have been successfully applied in other computer vision problems to the historical documents context. By using such general representations instead of relying on hand-crafted features, both handwritten and typewritten documents are handled indifferently.

The proposed method includes an indexation scheme aimed to scale the proposed method to handle large datasets. We also use a multi-length patch representation, which increases the retrieval performance by taking into account the different possible lengths of the query words. A thorough analysis and evaluation of all involved parameters of the method is presented in order to assess the configuration maximizing the retrieval performance. Finally, a performance comparison with the recent state-of-the-art literature in keyword spotting is also presented.

The remainder of this chapter is organized as follows. In Section 3.2, we present how the document corpora are constructed and organized. We detail the feature extraction from document pages and the encoding system used in order to efficiently query the collection. Section 3.3 details the retrieval stage. We show how queries are treated and how regions of interest are determined within document pages. Experimental results are presented in Section 3.4. We study the influence of the method's parameters and compare our performance against a number of state-of-the-art keyword spotting approaches. Finally, conclusions and further research lines are drawn in Section 3.5.

3.2 OFF-LINE CORPUS REPRESENTATION

The word spotting problem is addressed by dividing the original document images into a set of densely sampled local patches. These local patches are the basic structure used to spot the words within the document: once a query image is given, the local patches are used to determine the page locations where the query keyword has a greater likelihood to appear. With such a procedure having an explicit word segmentation is avoided as well as any other word pre-processing steps (i.e. binarization, slant correction, etc.). These local patches must roughly match the size of the text in the document. More precisely, the height H of the local patches should roughly match the height of the text in the document. This height parameter H can be either set automatically, by for instance using a projection profile algorithm, or it can by manually set by the user.

Then, for a given height H, four different widths W_ℓ are defined in order to cope with queries of different lengths. Specifically, the geometry of the patches has been set to $H \times H$, $2H \times H$, $3H \times H$ and $4H \times H$ and are densely sampled using a regular grid of $\frac{H}{3} \times \frac{H}{3}$ pixels. The most convenient patch width will be determined at query time. This setup guarantees that there is enough overlapping between the local patches and the document words so that each word in the document is covered by at least a patch. Although a salient patch detection strategy will effectively reduce the amount of patches to be processed [83], by densely sampling them no assumption has been made on which portions of the documents are important to the final user.

3.2.1 *Local Patch Descriptor*

Local patches are described using the BoVW signature so that, first visual words are extracted from the document images. The visual words are obtained by densely sampling SIFT descriptors over the image by using the method proposed by Fulkerson et al. in [68]. The SIFT descriptors are sampled over a regular grid of 5×5 pixels at three different scales: $\frac{H}{2}$, $\frac{3H}{4}$ and H. This multi-scale representation is used to capture from fine to coarse characteristics from the word characters. The finer scale characterizes sub-parts of a character while the coarser scale characterizes whole characters and their surroundings.

The performance of the BoVW model depends on the amount of visual words extracted from the image. In the related literature it has been noted that the larger is the amount of descriptors extracted from an image, the better the performance is [64]. Therefore, a dense sampling strategy has a clear advantage over approaches using interest points. However, a dense sampling over the image results in some SIFT descriptors calculated in low textured areas that are unreliable. In order to avoid this, descriptors having a low gradient magnitude before normalization are directly discarded.

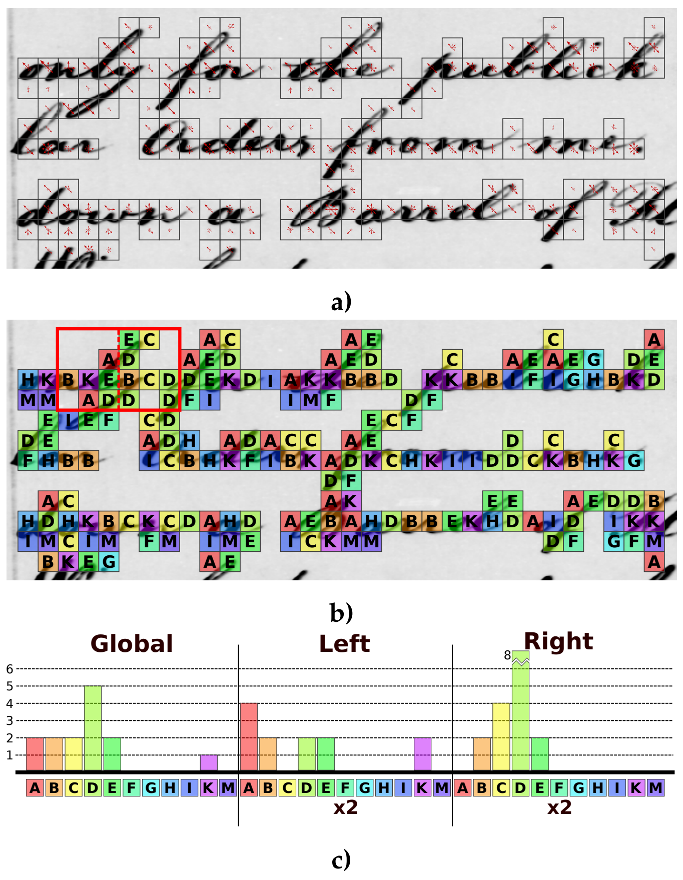


Figure 3.1: Local patch descriptor signature: **a)** SIFT descriptors are extracted from the document image. Note that the missing descriptors are filtered out due to their low descriptor normal value. Then, **b)** SIFT descriptors are encoded into visual words and **c)** the visual words which lie within the local patch are accumulated in the local patch signature.

Once the SIFT descriptors are calculated, a codebook is used to quantize them into visual words. The codebook is obtained by clustering the descriptor feature space into K different clusters by using the k-means algorithm. Then, visual words are obtained by simply assigning to each SIFT descriptor the nearest codeword of the codebook, i.e. the one with the smaller Euclidean distance.

After SIFT descriptors have been encoded into visual words, these visual words are used to create the signatures of the local patches: a local patch p_j of the document is described by a histogram $\mathbf{f}_j = \left[f_j^1, f_j^2, \ldots, f_j^K\right]$ which accumulates the frequencies of each visual word within the local patch. This K-dimensional descriptors do not take into account the spatial distribution of the visual words within the local patch. This is a drawback of the BoVW representation, since words with the same letters but resorted altogether may have a very similar signature. For instance, anagrams are completely indistinguishable using this representation. Therefore, the Spatial Pyramid Matching (SPM) method proposed by Lazebnik et al. in [71] is used in order to add spatial information to the unstructured BoVW model. This method roughly takes into account the distribution of the visual words over the local patches by creating a pyramid of spatial bins.

Different spatial configurations have been evaluated and, a two level SPM with a single vertical partition, differentiating the left and the right parts of the patch, gives the best compromise between the retrieval performance and the number of dimensions of the obtained descriptor. Since the amount of visual words assigned to each bin is lower at higher levels of the pyramid, due to the fact that the spatial bins are smaller, the visual words contribution is weighted according to the spatial coverage. In our case, the visual words assigned to the left and right spatial bins contribute twice to the final histogram. Finally, a local patch is described by a $3 \times K$ dimensions descriptor $\mathbf{f}_j = \left[\mathbf{f}_j^G, \mathbf{f}_j^L, \mathbf{f}_j^R\right]$, where $\mathbf{f}_j^G$, $\mathbf{f}_j^L$, $\mathbf{f}_j^R$ are the patch descriptor sub-vectors corresponding to the global, left and right spatial bins of the spatial pyramid. Finally, all the patch descriptors from the corpus are re-weighted by applying the *tf-idf* model [84] and normalized using the $\mathbf{L}_2$ norm. The different steps used to obtain the signatures of the local patches are summarized in Fig. 3.1. First, in Fig. 3.1.a) SIFT descriptors are calculated from the image. Note that since descriptors with low norm are discarded, only descriptors nearby a character are present. Then, SIFT descriptors are quantized into visual words in Fig. 3.1.b) and these visual words are used to create the descriptor of the local patch in Fig. 3.1.c).

3.2.2 *Latent Semantic Analysis Transform*

Ideally, two instances of the same character are always represented by the same set of visual words. However, the clusters obtained using the k-means algorithm might not be optimal, so that some salient structures in the descriptor space might not be properly represented. Since the number of visual words of the codebook is not inferred from the descriptor space, this space may be under- or over-clustered. Besides, the shape of a character is likely to change from word to word in the context of keyword spotting, specially in handwritten documents. Therefore, the Latent Semantic Analysis (LSA) technique introduced by Deerwester et al. in [85] has been applied to represent the local patch descriptors in a way which eludes unreliability, ambiguity and redundancy of individual visual words.

The LSA technique assumes that exists some underlying semantic structure in the descriptor space. This semantic structure is defined by a set of abstract *topics* where each topic is a representative distribution of visual words. The topics are estimated in an unsupervised way using the Singular Value Decomposition (SVD) algorithm. Then, local patches are represented by a mixture of topics instead of a histogram of visual words. The goal is to obtain a transformed space where patches having similar topics but encoded by different visual words will lie close. In the context of our problem where a document is represented by millions of local patches, the LSA technique has the advantage over similar alternatives that the SVD can be calculated incrementally [86]. This allows to obtain the transformation space matrix processing the whole corpus of local patches in a very efficient way.

In order to obtain the space transformation matrix, the document patches of the global level descriptors $\mathbf{f}_j^G$ are arranged in a visual-word-by-patch matrix $\mathbf{A} \in \mathbb{R}^{K \times M}$, where K is the codebook size and M is the number of patches of the document. The LSA obtains the transformed space by decomposing the visual-words-by-patch matrix in three matrices by a truncated SVD. In order to reduce the descriptor space to T topics, where $T \ll K$, we proceed as follows:

$$\mathbf{A} \simeq \hat{\mathbf{A}} = \mathbf{U}_T \mathbf{S}_T \left(\mathbf{V}_T\right)^\top ,$$

where $\mathbf{U}_T \in \mathbb{R}^{K \times T}$, $\mathbf{S}_T \in \mathbb{R}^{T \times T}$ and $\mathbf{V}_T \in \mathbb{R}^{M \times T}$. Then, a patch descriptor $\mathbf{f}_j$ is projected into the transformed space vector $\hat{\mathbf{f}}_j$ by applying the transformation matrix $\mathbf{X}_T = \mathbf{U}_T \left(\mathbf{S}_T\right)^{-1}$ to each spatial sub-vector $\mathbf{f}_j^G$, $\mathbf{f}_j^L$, $\mathbf{f}_j^R$ separately as follows:

$$\hat{\mathbf{f}}_j = \left[{\mathbf{f}_j^G}^\top \mathbf{X}_T, {\mathbf{f}_j^L}^\top \mathbf{X}_T, {\mathbf{f}_j^R}^\top \mathbf{X}_T \right] .$$

This setup has obtained better experimental results than applying the LSA technique directly to the local patch descriptor $\mathbf{f}_j$. By separately transforming each sub-vector, the spatial information encoded by the SPM scheme is maintained.

3.2.3 *Product Quantization Indexing*

In order to efficiently store and retrieve the patch descriptors, an indexing structure is needed. The Product Quantizer (PQ) indexation framework proposed by Jégou et al. in [87] has been used. This method allows both to reduce the amount of memory needed to store the local patch descriptors by means of binary codes and to reduce the computational cost of searching the nearest neighbors by using a sub-linear approximate distance computation. The method is governed by two parameters m and c that will determine the achieved compression rates. The product quantizers decompose the local patch descriptor space into a Cartesian product of m local sub-vectors. The original $\hat{\mathbf{f}}_j$ descriptors are mapped into the $T^* = 3T/m$ dimensional sub-vectors as

$$\begin{aligned} \hat{\mathbf{f}}_j &= [\underbrace{\hat{f}_j^1, ..., \hat{f}_j^{T^*}}_{u_1(\hat{\mathbf{f}}_j)}, ..., \underbrace{\hat{f}_j^{(m-1)T^*+1}, ..., \hat{f}_j^{3T}}_{u_m(\hat{\mathbf{f}}_j)}] \\ &= \left[u_1(\hat{\mathbf{f}}_j), ..., u_m(\hat{\mathbf{f}}_j) \right] . \end{aligned}$$

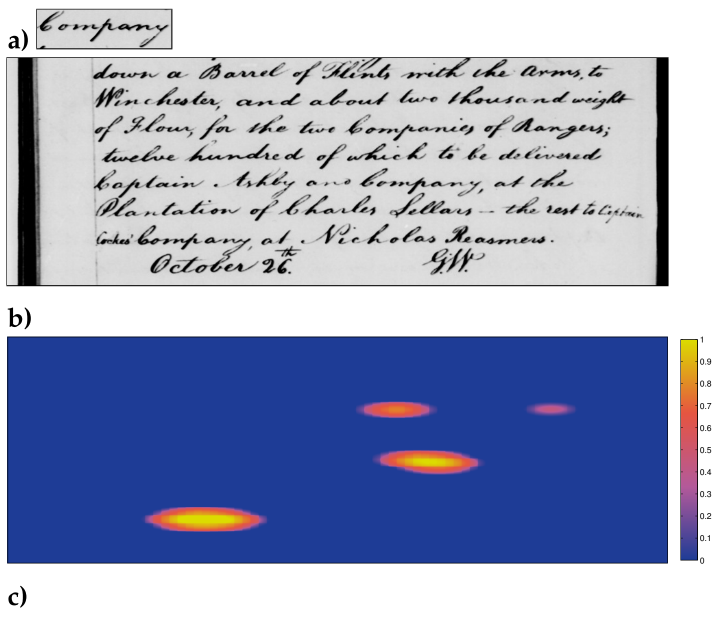
Then, each sub-space is quantized separately using c sub-quantizers. Finally, the descriptors are represented by a short code composed of its subspace quantization indexes calculated as,

$$PQ(\hat{\mathbf{f}}_j) = \left[\kappa_1(u_1(\hat{\mathbf{f}}_j)), ..., \kappa_m(u_m(\hat{\mathbf{f}}_j)) \right] ,$$

where $\kappa_i(\cdot)$ is the index of the sub-quantizer associated with the i-th subvector. For instance, if in the LSA step, the number of topics is set to $T = 512$, then the dimensionality of the patch descriptor is 1536. Given a PQ configuration which divides the original space into $m = 128$ sub-vectors of 12 dimensions and uses $c = 256$ sub-quantizers, the 1536-dimensional local patch descriptor is effectively represented by a 128 bytes code.

3.3 WORD RETRIEVAL

To perform the retrieval, the *query-by-example* paradigm is followed, where the user inputs the system a sample image of the sought word. In our segmentation free approach, a set of putative patches which are visually similar to the given query are first obtained. Then, a voting scheme aims at finding the locations within the document pages with a high likelihood to find the query word.



down a Barrel of Flints with the arms, to
Winchester, and about two thousand weight
of Flour, for the two Companies of Rangers;
twelve hundred of which to be delivered
Captain Ashby and Company, at the
Plantation of Charles Sellars — the rest to Captain
Cockes Company, at Nicholas Reasmers.
October 26th. GW

Figure 3.2: Example of the voting procedure. **a)** Query, **b)** sample page, **c)** obtained voting space.

3.3.1 *Candidate Search*

Given a query image which has been cropped by the user from the images in the collection, the proposed method first densely samples the SIFT descriptors. Then, quantizes them into visual words using the codebook. Afterwards, the patch descriptor is obtained by accumulating the visual words into the different bins of the spatial pyramid histograms. Subsequently, the obtained descriptor is normalized using the *tf-idf* model obtaining $\mathbf{f}_q$ which is projected into the transformed LSA space by

$$\hat{\mathbf{f}_q} = \left[{\mathbf{f}_q^G}^\top \mathbf{X}_T, {\mathbf{f}_q^L}^\top \mathbf{X}_T, {\mathbf{f}_q^R}^\top \mathbf{X}_T \right].$$

Finally, the cosine distance is computed between the query descriptor $\hat{\mathbf{f}_q}$ and the document patch descriptors $\hat{\mathbf{f}_j}$ as a similarity measure to select the patches from the documents where the query keyword is more likely to appear.

The cosine distance is calculated by using the asymmetric distance computation [87] method from the PQ framework. First, the dot product is separately calculated for each m sub-vector between the query and all the c sub-quantizers obtaining a distance matrix $\mathbf{D} \in \mathbb{R}^{m \times c}$, where the i-th row of the matrix is computed as

$$d_i = \left[\langle u_i(\hat{\mathbf{f}}_q), u_i(\hat{\mathbf{f}}_1) \rangle, \ldots, \langle u_i(\hat{\mathbf{f}}_q), u_i(\hat{\mathbf{f}}_c) \rangle \right],$$

where $\langle \cdot, \cdot \rangle$ is the dot product between the two sub-vectors. Then, following the multi-length scheme, the query width determines which local patches agree in terms of word length. According to that, just the most similar width W_ℓ^* to the query is taken into account. Finally, combining the PQ codes $[\kappa_1, \ldots, \kappa_m]$ of the selected local patches and the matrix $\mathbf{D}$, the approximated cosine distances are obtained

$$\delta_{qj} = 1 - \sum_{i=1}^{m} \mathbf{D}_{i, \kappa_i(u_i(\hat{\mathbf{f}}_j))}$$

between the query $\hat{\mathbf{f}_q}$ and the patch descriptors $\hat{\mathbf{f}}_j$ that match the query length.

3.3.2 *Candidate Localization*

Once the most similar local patches have been retrieved, the regions of the document which gather most support have to found and selected as putative retrieved locations.

For each document page image, a 2-D voting space is constructed in where each retrieved local patch will cast its votes. In our implementation each cell of the voting space has a geometry equal to the local patch sampling step ($\frac{H}{3} \times \frac{H}{3}$ pixels). Then, each selected local patch casts a vote to the cell where its geometric center falls, weighted by the approximate distance δ_{qj}. Afterwards, the contribution at each cell of the voting space is smoothed by using an elliptic Gaussian filter $g_{\perp}(x, y; W^*_{\ell}, H)$. For instance, Fig. 3.2 shows an example of the smoothed voting space obtained for a given query. Finally, the retrieved regions of $W^*_{\ell} \times H$ pixels are found by searching the local maxima in the smoothed voting space. The resulting list of putative document regions R_D is obtained by resorting the selected candidates in terms of its local maxima value.

3.4 EXPERIMENTAL RESULTS

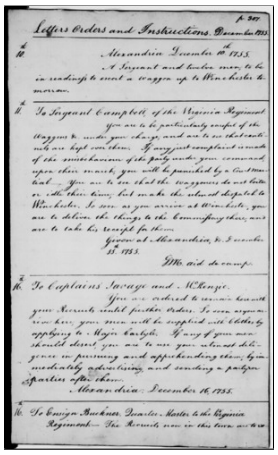
Let us first introduce the datasets and the evaluation measures used to assess the performance of the proposed system and then analyze the obtained results.

3.4.1 *Dataset and Evaluation Measures*

In order to perform the experiments, we have used three datasets of handwritten documents and one dataset of typewritten documents. The first image corpus (GW20 dataset) consists of a set of 20 pages from a collection of letters by George Washington [31] dated 1755. Its ground-truth has a total of 4 860 segmented words with 1 124 different transcriptions. In order to test the scalability of the method, we have used a much larger set of images from the George Washington letters[1] composed of 1 500 pages (GW1500 dataset), however, there is no ground-truth for this dataset. The third evaluation corpus (BCN dataset) contains 50 pages from a collection of handwritten marriage licenses written in 1 617 from the Barcelona Cathedral [88]. In that collection just some words are transcribed. We have 6 735 segmented words corresponding to 21 different transcriptions. Finally, although the main aim of our method is to deal with handwritten documents, for the sake of generality, we also tested a typewritten corpus (LB dataset) consisting of a set of 20 pages from a 1 825 book on Lord Byron's life [57]. In that case we have 4988 segmented words corresponding to 1569 different transcriptions. We can see an example of the four datasets in Fig. 3.3. In terms of the document degradation, the LB collection is the most well-preserved and, since it is typewritten, it is expected to be the less challenging dataset. Between the GW20 and the BCN collections, the handwriting style in the GW20 images is less variable and the image quality is quite good, whereas the BCN collection

1 Library of Congress `http://memory.loc.gov/ammem/gwhtml/`

is the most challenging one since the images present severe degradations and the variability in handwriting style is highly noticeable.



p. 307

Letters Orders and Instructions. December 1755.

10th. Alexandria December 10th 1755.

A Sergeant and twelve men, to be in readiness to escort a waggon up to Winchester to-morrow.

11th. To Sergeant Campbell, of the Virginia Regiment.

You are to be particularly careful of the Waggons &c. under your charge, and are to see that centinels are kept over them. If any just complaint is made of the misbehaviour of the party under your command, upon their march, you will be punished by a Court Martial. You are to see that the waggoners do not loiter or idle their time, but make the utmost dispatch to Winchester. So soon as you arrive at Winchester, you are to deliver the things to the Commissary there, and are to take his receipt for them.

Given at Alexandria, &c. December 11th 1755.

G.W. aid de camp.

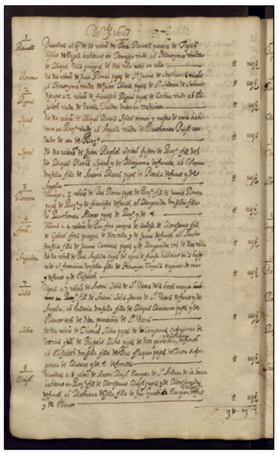
16th. To Captains Savage and McKenzie.

You are ordered to remain here with your Recruits until further Orders. So soon as you arrive here, your men will be supplied with Clothes by applying to Major Carlyle. If any of your men should desert, you are to use your utmost diligence in pursuing and apprehending them, by immediately advertising and sending a party or parties after them.

Alexandria: December 16, 1755.

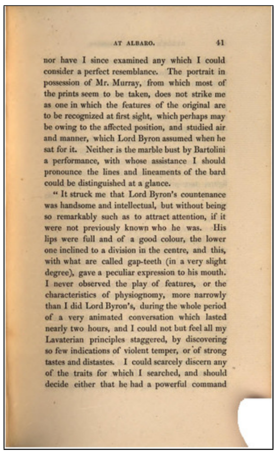
16th. To Ensign Buckner, Quarter Master to the Virginia Regiment. The Recruits now in this town are to

a)



[illegible]

b)



AT ALBARO. 41

nor have I since examined any which I could consider a perfect resemblance. The portrait in possession of Mr. Murray, from which most of the prints seem to be taken, does not strike me as one in which the features of the original are to be recognized at first sight, which perhaps may be owing to the affected position, and studied air and manner, which Lord Byron assumed when he sat for it. Neither is the marble bust by Bartolini a performance, with whose assistance I should pronounce the lines and lineaments of the bard could be distinguished at a glance.

" It struck me that Lord Byron's countenance was handsome and intellectual, but without being so remarkably such as to attract attention, if it were not previously known who he was. His lips were full and of a good colour, the lower one inclined to a division in the centre, and this, with what are called gap-teeth (in a very slight degree), gave a peculiar expression to his mouth. I never observed the play of features, or the characteristics of physiognomy, more narrowly than I did Lord Byron's, during the whole period of a very animated conversation which lasted nearly two hours, and I could not but feel all my Lavaterian principles staggered, by discovering so few indications of violent temper, or of strong tastes and distastes. I could scarcely discern any of the traits for which I searched, and should decide either that he had a powerful command

c)

Figure 3.3: Example of pages from the a) George Washington, b) Barcelona Cathedral and c) Lord Byron's collections.

In order to evaluate the performance of the spotting method we have chosen to report the mean average precision mean Average Precision (mAP) and recall measures. In our case, a returned region from the documents will be considered as relevant when it overlaps at least a 50% of the sought word in the ground-truth.

Table 3.1: Local patch and feature geometries parameters used at each database.

		GW20	BCN	LB
Line Height		80	70	60
Feature Size	Small	40	36	32
	Medium	60	56	48
	Large	80	72	60
Patch Grid		27 × 27	24 × 24	20 × 20
Patch Geometry	Tiny	80 × 80	70 × 70	60 × 60
	Small	160 × 80	140 × 70	120 × 60
	Medium	240 × 80	210 × 70	180 × 60
	Large	320 × 80	280 × 70	240 × 60

For each database, we need to calculate the line height parameter H in order to define the geometry parameters of the local features and local patches. Table 3.1 summarizes the line height and the inferred feature and patch geometries that we have used in the following experiments. Such line height H has been automatically estimated by means of a projection profile analysis [10] over a subset of pages of the document collection. The text line height is obtained by calculating the median separation between peaks of the projection profile. This allows to obtain an accurate H parameter despite the possible errors of the line detection algorithm.

3.4.2 *Results*

In this section, we analyze the performance of the proposed system. We organize the different carried experiments as follows. First, we will present some qualitative results to assess the effectiveness of the method to retrieve visually similar words. Then, we provide an exhaustive study of the effect of the different parameter configurations of the proposed method. Subsequently, we analyze the system's behavior in a large-scale scenario. We finally compare the obtained results with other state-of-the-art methods.

3.4.2.1 *Qualitative Results*

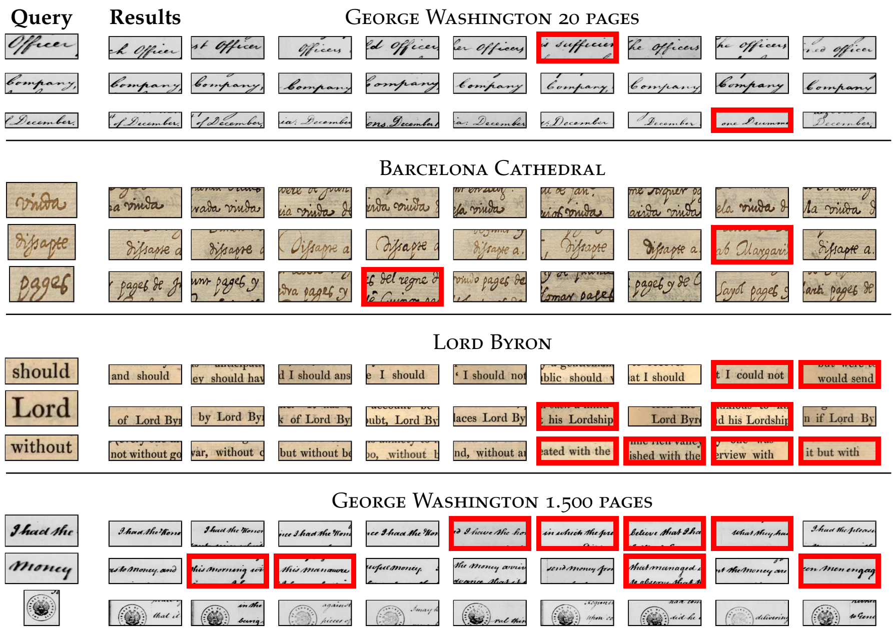


Figure 3.4: 9 top-most retrieved images for some queries in the four evaluated collections.

We present in Fig. 3.4 some qualitative results for the four databases with an SPM-BoVW patch descriptor with a codebook of 2^{15} visual words. In a word spotting application, the chosen word descriptor should agree with the human perception when considering that two words are similar. We report here some queries where the system yields some false positive words in the first ten results. These results show that gradient-based descriptors fulfill the visual requirements in both typewritten and handwritten scenarios since the false positives (framed in red) are visually similar to the queried word. In addition, it is worth to note that even if the method does not entail any segmentation step, the retrieved regions are usually well centered over the text lines.

3.4.2.2 *Baseline*

We can see in Table 3.2 the evolution of the mean average precision and recall indicators for codebook sizes from 2^8 to 2^{15} visual words and amount of topics from 2^6 to 2^9 for the three collections. The system tends to perform better with large codebooks both in terms of ranking and recall abilities. However, we can appreciate that the gain decreases as the codebook grows

Table 3.2: Mean average precision and recall at different codebook sizes and amount of topics for the three collections.

Mean Average Precision

Topics	Codebook size																							
	GW20								BCN								LB							
	256	512	1024	2048	4096	8192	16384	32768	256	512	1024	2048	4096	8192	16384	32768	256	512	1024	2048	4096	8192	16384	32768
Without LSA	14.37	21.09	28.34	35.65	42.47	47.80	52.63	56.32	54.87	62.30	70.18	76.74	82.85	85.76	88.38	90.31	12.92	21.58	36.17	49.99	65.88	74.78	80.69	83.85
64	14.12	18.64	23.29	30.72	35.99	41.23	45.53	**49.21**	51.67	56.77	65.61	73.61	80.15	82.94	85.68	**87.84**	10.82	18.58	27.51	36.82	46.76	54.82	62.88	**71.22**
128	15.50	20.41	25.58	32.93	39.34	44.84	49.46	**54.06**	53.43	58.81	66.41	74.68	81.26	84.38	87.06	**89.17**	13.10	22.71	34.01	42.52	55.45	65.27	70.56	**79.42**
256	15.88	21.56	26.77	33.65	40.46	46.43	51.72	**56.03**	53.69	59.57	67.25	75.14	81.90	85.25	87.94	**89.93**	13.74	26.33	37.76	47.81	61.55	70.32	75.09	**83.28**
512	15.54	21.92	27.35	34.50	41.07	47.26	52.75	**57.51**	53.49	59.66	67.17	74.75	81.59	85.06	87.96	**90.17**	13.40	26.49	40.53	51.01	63.96	72.11	77.70	**85.16**

Recall

Topics	Codebook size																							
	GW20								BCN								LB							
	256	512	1024	2048	4096	8192	16384	32768	256	512	1024	2048	4096	8192	16384	32768	256	512	1024	2048	4096	8192	16384	32768
Without LSA	77.02	79.43	81.26	83.78	85.75	86.99	87.63	87.72	51.32	54.88	58.97	61.37	63.19	63.68	63.54	63.10	81.72	86.92	91.89	95.06	96.55	96.86	96.97	96.01
64	78.28	80.48	82.15	85.55	87.24	87.76	87.87	**88.44**	59.65	62.39	66.37	69.48	71.75	72.72	**73.12**	72.60	81.41	86.23	88.84	91.91	93.58	94.75	**95.53**	95.18
128	78.39	80.55	82.73	85.65	87.73	89.22	90.04	**90.10**	61.57	64.14	68.12	71.70	73.56	**74.58**	74.33	74.07	83.09	87.33	90.92	93.68	95.74	96.49	**96.66**	96.23
256	78.41	80.45	82.98	85.86	87.99	89.45	90.34	**91.15**	61.95	65.07	69.15	72.42	74.06	**74.74**	74.69	74.50	83.25	88.43	92.01	94.91	96.35	96.77	**96.98**	96.50
512	78.38	80.55	82.86	85.74	87.87	89.51	90.49	**91.22**	61.62	65.19	69.65	72.63	74.30	**74.89**	74.79	74.64	83.04	88.41	92.59	95.35	96.53	96.89	**97.09**	96.48

larger until it might result in a decrease of the system's performance. This fact is emphasized in the BCN collection, which presents more noise. Here, when using medium-sized codebooks, the system generalizes better and is able to absorb the noise whereas when we increase the vocabulary size the patch descriptor becomes more noisy and the recall is affected.

As expected, when using the LSA encoding, the greater the number of topics is, the better the system performs. Looking at the GW20 and LB experiments, the dimensionality reduction produces a small drop-off in terms of mean average precision for small codebooks that is counteracted as we increase the codebook size. However, if we look at the experiments carried with the BCN collection, an interesting phenomenon can be observed. Here, the drastic dimensionality reduction not only does not hinder the performance but provokes a significant improvement in recall against the raw descriptors. This recall increase can be attributed to the original idea of the LSA algorithm, which not only reduces the dimensionality of the descriptors but also finds relationships between different visual words corresponding to the same keyword. In noisy environments the use of LSA results in a more compact representation that in addition generalizes better and thus ameliorates the final performance.

The results of our baseline system with $K = 2^{15}$, $T = 512$ and using the SPM scheme are summarized in the first row of the Table 3.4. The obtained results clearly outperform our previous approach presented in [57] in both the GW20 and LB collections, mainly due to the increase in the codebook size and the amount of topics in the LSA encoding.

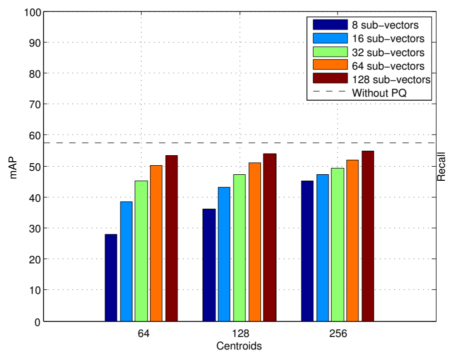
3.4.2.3 *Compressing with Product Quantization*

Each patch from the documents in the baseline system is described by a 1 536-dimensional double-valued feature vector, thus occupying 12 288 bytes in memory. Each page having in average more than 10 000 patches, we need approximately 120MB to store each page from the collection in memory. Since managing such amount of data makes the system not scalable, we have compressed the patch descriptors with the PQ method. We can see in Table 3.3 the details in terms of memory usage per patch and the compression ratios reached for different values of the c and m parameters. We have achieved a lossy patch representation that reduces its size with respect to the baseline by a factor that ranges from 96 to 2048 times. This means that in a Gb of RAM memory, we can fit between 900 to 18 000 pages.

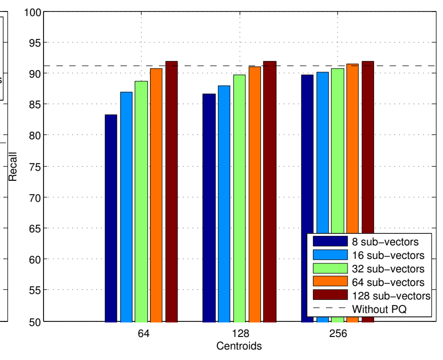
Table 3.3: Bytes and compression ratio per patch for each PQ setup.

c sub-quantizers	m sub-vectors				
	8	16	32	64	128
64	6 (1:2048)	12 (1:1024)	24 (1:512)	48 (1:256)	96 (1:128)
128	7 (1:1755)	14 (1:877)	28 (1:438)	56 (1:219)	112 (1:109)
256	8 (1:1536)	16 (1:768)	32 (1:384)	64 (1:192)	128 (1:96)

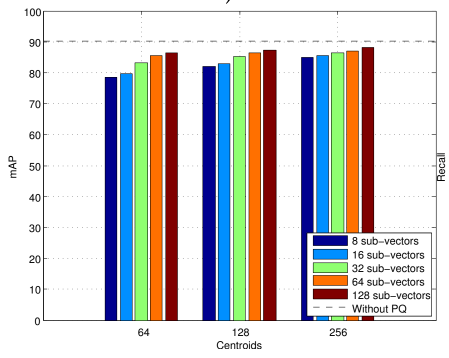
In Fig. 3.5 we present the mAP and recall measures obtained after compressing the patch descriptors with different values of the c and m parameters. As we can appreciate, concerning the mAP, the increase of sub-vectors m enhances the performance of the system whereas no significant improvement is observed when increasing the amount of sub-quantizers c. Regarding the



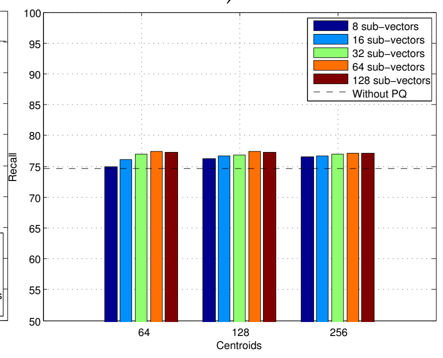
a)



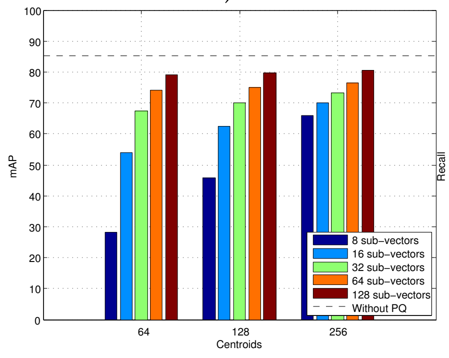
b)



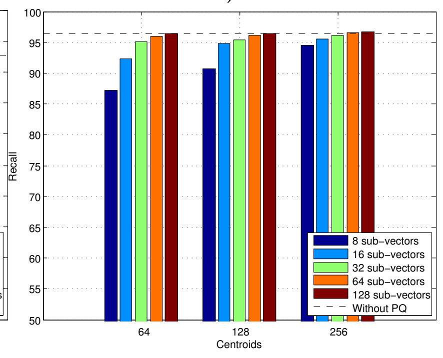
c)



d)



e)



f)

Figure 3.5: Mean average precision and recall for different c and m values when using PQ for the a) and b) George Washington, c) and d) Barcelona Cathedral and e) and f) Lord Byron's collections.

recall indicator, we appreciate the same phenomenon, but in that case the recall values even slightly outperform the baseline system.

These results can be attributed to the quantization step when local patch descriptors are converted to PQ codes. This quantization reduces the discriminative power of the local patch descriptors, resulting in a reduction of the mAP. However, it also reduces the effects of noise, leading to the observed moderate increase of the recall in all databases.

3.4.2.4 *Using Mutli-length Patch Indexation*

So far in the presented experiments the size of the query word has not been taken into account. In our previous work [57], we found that short queries performed worse than larger ones when using a fixed size of the patch.

Table 3.4: Performance of the proposed baseline method with PQ, multi-length configuration.

GW20	mAP(%)	Recall(%)
Baseline ($K = 2^{15}$,$T = 512$)	57.51	91.22
With PQ ($c = 256$, $m = 128$)	54.68	91.88
With PQ and Multi-length	61.35	95.43
BCN	**mAP(%)**	**Recall(%)**
Baseline ($K = 2^{15}$,$T = 512$)	90.17	74.64
With PQ ($c = 256$, $m = 128$)	88.07	77.07
With PQ and Multi-length	88.93	83.21
LB	**mAP(%)**	**Recall(%)**
Baseline ($K = 2^{15}$,$T = 512$)	85.16	96.48
With PQ ($c = 256$, $m = 128$)	80.61	96.71
With PQ and Multi-length	90.38	97.34

Therefore, we have used a multi-length patch representation in order to fix this shortcoming despite the increase in memory requirements. We can see in the third row of Table 3.4 the important gain in both mAP and recall when the patch indexation is adapted to the query width. By looking at the individual performances attained at each patch geometry, we have observed that only the tiny patch configuration performs slightly worse than the fixed length approach. All other scales outperform the fixed approach. For instance in the GW20 dataset we obtained a 48.94%, 54.55%, 67.03% and 79.17% mAP for the tiny, small, medium and large patch configurations respectively. While the fixed approach reach a 54.68% mAP. A similar phenomenon is observed for the recall value. The increase in performance as the patch geometry grows can be explained due to the perceptual aliasing. Short queries are more likely to obtain false positives since matching to a sub-strings is not penalized in our method. Even though this behavior penalizes the performance of the method, this is not an undesired conduct in the query-by-example setup.

3.4.2.5 *Large scale evaluation*

The vectorial representation of our spotting method allows to efficiently index large collection of pages. Unfortunately, publicly available databases only have tens of annotated pages since creating the ground-truth for large collections is a tedious tasks. Therefore, in our large scale experiments using the GW1500 dataset, we had to calculate the retrieval score manually. Query images were generated by randomly selecting 50 word images and the retrieval score was obtained by manually annotating the correct matches for the 50 top-most results. We have used up to 1 500 document images which required up to 60 million patches to represent the whole collection. Therefore, when using the multi-length representation, the whole document collection needed about 7.5 Gb of memory with the best PQ configuration.

The mAP and time needed to process a single page for different sized collections is shown in Table 3.5. As expected, the mAP score of the system slowly decreases as more pages are indexed because of the amount of dis-

Table 3.5: Performance of the system in the GW1500 scenario.

Num. of pages	mAP (%)	time (ms.)
100	57.73	3.01
500	56.83	2.85
1000	55.98	2.84
1500	55.69	2.87

tractors and false positive visually similar words also increases. Regarding the computational cost, the time required to process a page remains constant as more pages are added. The PQ framework allows to retrieve approximate nearest neighbors sub-linearly so that the candidate search time actually decreases when adding more pages to the collection. However, the more pages we add to the collection, the more the voting scheme from the candidate localization step increases its computational cost. Leading to a nearly constant time as more pages are added since both steps compensate each other. In addition, these results do not take into account that pages can be processed independently. Therefore, an straightforward modification to speed up retrieval speed is to process document pages concurrently.

3.4.2.6 *Comparison with related literature*

The George Washington collection has been used in many word spotting works and has become a de-facto dataset used to benchmark different systems. However, the lack of a standard evaluation protocol and the different taxonomies of word spotting methods provokes that achieving a direct comparison among methods is not straightforward. Not all the authors use the same set of pages, query words and even evaluation measures. We present in Table 3.6 a review of the achieved performances of several state-of-the-art methods. Only Almazan et al. [65] used the same evaluation methodology than us. For the sake of comparison, we have evaluated our method using each of the different experimental setups and evaluation measures proposed by the authors in the original papers. We can see that in equal conditions, the proposed method outperforms all state-of-the-art but the method proposed by Frinken et al. in [13]. It is also worth to mention that some segmentation-based methods present their results by using a manual segmentation of the images avoiding thus the problems derived from any erroneous segmentation artifacts. Lets now discuss the different results obtained using the different configurations.

In some methods we have to take into account the reported recall measure. In Rath and Manmatha [30] and Rothfeder et al. [75], they use a pruning step to remove unlikely correspondences and also to speed up the retrieval process. Likewise, we revoked retrieved images with low score until a similar recall value is attained. This pruning increases the mAP score as we can observe in Table 3.6.

Other methods use a selected set of queries [78, 80, 82] avoiding stop words. Since stop words have few characters and are difficult to distinguish visually, by not considering them our method increased its mAP score. Similarly, methods which use cross-validation only cast the query words appearing in all fold sets. The configuration used by Rodriguez-Serrano and Perronnin in [38] divides the database in 5 folds: a fold is used to create the queries,

Table 3.6: Review of performances of the state-of-the-art methods using the GW20 dataset.

Method	Reference	Segmentation	Learning	Dataset	Original Results	Proposed method
Semi-continuous HMMs	Rodríguez-Serrano and Perronnin [38]	Words	Yes, 5-fold cross validation: 1 fold train, 1 fold validation, 3 folds test	20 pages, all words in train set as queries	53.1% mAP	**57.9% mAP**
Boosted decision trees	Howe et al. [77]	Words	Yes: 20-fold cross validation: 19 folds train, 1 fold test	20 pages, all words in train set as queries	79.5% mAP	**87.4% mAP**
Dynamic time warping	Rath and Manmatha [30]	Words	No	10 good quality pages, 2381 queries	40.9% mAP	**64.7% mAP**
Corner feature correspondences	Rothfeder et al. [75]	Words	No	10 good quality pages, 2381 queries	36.2% mAP	**64.7% mAP**
Synthesized words	Liang et al. [78]	Words	Yes: 5-fold cross validation: 4 folds train, 1 fold test	20 pages, 38 queries	67% mAP at rank 10	**79.2% mAP at rank 10**
Recurrent Neural Networks	Frinken et al. [13]	Lines	Yes, 4-fold cross validation: 2 folds train, 1 fold validation, 1 fold test	20 pages, all words of the training set appearing in all 4 folds as queries	**71% mean prec.**	67.1% mean prec.
Character HMMs	Fischer et al. [17]	Lines	Yes, 4-fold cross validation: 2 folds train, 1 fold validation, 1 fold test	20 pages, all words of the training set appearing in all 4 folds as queries	62% mean prec.	**67.1% mean prec.**
Slit style HOG features	Terasawa and Tanaka [80]	Lines	No	20 pages, 15 queries	79.1% mAP	**87% mAP**
Gradient features with elastic distance	Leydier et al. [82]	None	No	20 pages, 15 queries	60% P=R	**71.2% P=R**
Exemplar SVM	Almazán et al. [65]	None	No	20 pages, all 4856 words as queries	54.5% mAP	**61.35% mAP**

another fold is used for validation purposes and the last 3 folds are used as test collection. Since the *GW* dietaries explain facts temporally, some words just appearing in specific page ranges are not considered using this configuration. However, stop words are kept as queries, resulting in a mAP score decrease. However, the leave-one-out configuration used in [77], just use a single page as test leading to drastic reduction both in the number of queries and the possible retrieved results facilitating a steep increase of the mAP.

Finally, Fischer et al. [17] and Frinken et al. [13] use cross-validation with an evaluation framework performed at line level, i.e. a whole line is assessed as relevant when it contains a single instance of the query word. In order to compare their results with our method, we followed a procedure similar to the one defined in both papers when comparing to DTW. The retrieved images are first projected to the closest text line. The score of a whole text line corresponds to the highest score of the projected words. By following such evaluation procedure, we obtain 100 line results for a given query, since each page contains about 20 lines and only 5 pages are indexed per fold. Consequently, the results obtained using this evaluation procedure have to be taken cautiously when compared with word-level evaluations. When using this configuration the mAP tends to increase since the effect of false-positives and lowly ranked true-positives is lessened.

Still, Frinken et al. [13] outperforms our method. This is not surprising since their method is learning-based. Besides using statistical machine learning models that cope with the handwritten word variations, these methods also integrate language models to further reduce the effects of visual ambiguities. However, example-based system are more flexible as they can be used to spot any kind of word or symbol present in the document image. For instance, our system was also able to spot the graphical stamp of the Library of the Congress as shown in the last row of Fig. 3.4.

Finally, another important aspect when evaluating spotting systems is the computational cost. From all the methods reviewed in Table 3.6, just Almazan et al. [65] present an efficient implementation that can be scalable to large environments, requiring 15 ms. per indexed page. Approaches based on sliding-windows [13, 80] are not scalable, since they have to process the whole document corpus each time that a query is cast. Other holistic word signatures like [30, 75] require a complex alignment processes which can not be effectively indexed. By contrast, the vector representation used by our method can naturally handle larger amount of data by efficiently storing indexed information and obtaining results in sub-linear complexity.

3.5 CONCLUSIONS

In this chapter we have presented an efficient keyword spotting method for historical collections that does not involve any segmentation stage. Thus the proposed method presents a clear advantage over segmentation-based methods which are likely to fail in challenging scenarios. A patch-based framework has been used where each patch is described by means of a bag-of-visual-words model, where the visual words encode gradient information. The proposed architecture follows a query-by-example paradigm and thus do not need any supervised learning step. Our method has been tested with three different collections of historical documents either handwritten and typewritten. The proposed method yields a very compact, efficient and discriminative representation thanks to the LSA technique and the PQ compression step. Such representation is able to efficiently index the document information both in terms of memory and computational cost, resulting in a suitable method for large-scale scenarios. By introducing a multi-length patch representation, we have increased the retrieval performance when querying small words. In addition, we have presented a thorough analysis and evaluation of all the involved parameters of the method in order to assess the configuration maximizing the retrieval performance.

Finally, we have presented an exhaustive comparison with state-of-the-art word-spotting methods. We evaluated our method using the experimental setup of each compared method, concluding that our method outperforms all them but Frinken et al. [13]. However, our example-based method is more flexible since it does not rely on any segmentation method nor any image pre-processing step and it does not take advantage of a language model, so that, it is not limited to search words in the document but it can retrieve any kind of symbol present in the images. This is a feature which can be useful for historical documents where symbols used to abbreviate common words and illustrations commonly appear. Nonetheless, adding either a language model or some statistical machine learning steps to the proposed architecture is straightforward. For instance in [62], we added a model of the character distribution over words in a query-by-string framework.

4 DATA FUSION

In this chapter, we explore the introduction other information modalities to create a query-by-X spotting system where word snippets are indexed and retrieved using different information modalities. Besides the visual information characterized with a Bag of Visual Words (BoVW) signature, we consider two additional sources of information. A textual signature where words are represented by a codebook of n-grams and an audio signature where words are represented by a Bag-of-Audio-Words (BoAW) signature. The main idea of our approach is to find a projection that transforms two different signatures into a common feature space, so word snippets can be instinctively characterized using either information modality. The goal is to create a word snippet index using only visual information and then query them using only textual or audio information, i.e. obtaining a query-by-string or a query-by-audio word spotting system. Additionally, the process can be used in reverse and retrieve the audio utterance from visual queries. The proposed method generates a vectorial signature that can be used together with state-of-the-art indexation structures can be used in large-scale scenarios. The proposed method is evaluated using a collection of historical documents outperforming state-of-the-art performances.

4.1 INTRODUCTION

Handwritten keyword spotting can be defined as the problem of locating within a collection of documents, the zones of interest where a particular queried word is likely to appear, without the explicit transcription of all the contents of the collection. Being a mature enough research topic, many different approaches have been presented in the literature aiming at the keyword spotting problem. In particular, one of the most prominent approaches is to follow a ***query-by-example*** paradigm. That is, the user provides a snippet image of the sought handwritten word that serves as example and the system retrieves a ranked list of words from the collection that are similar to the query. The simplicity of such approaches is quite attractive since by only using an adequate handwritten word description and similarity measure, the system can already deliver good retrieval performances. In addition, such spotting approaches can usually be pipelined to any indexing mechanism yielding efficient response times in large-scale scenarios. However, such simplicity hides an important usability flaw. In order to spot a word, the user has to browse the collection looking for an instance of the sought word. But forcing the user to manually extract a template word in large collections might be a really tedious task. Such approaches are thus unusable in real scenarios for plain users that might not be willing to make such an effort to just cast a query.

To overcome such limitations, more complex learning-based techniques have been proposed to bypass this burden. These techniques allow to query the system by just typing the query string, which is known as the ***query-by-***

string paradigm. The first query-by-string proposed attempts such as [89, 90, 42, 50, 78] relied on the extraction of letter or glyph templates. Such character extraction was manually done [42, 50] or by means of some clustering scheme [90, 78]. When the user typed the query, such character templates were put together in order to synthetically generate an example of the sought word. Although such methods proved to be effective and user-friendly, they can just be applied in scenarios where individual characters can be easily segmented and put together again to generate the queries. In fact, they were applied to either typewritten documents [90, 42] or documents having writing styles without many typographic ligatures [89, 50].

In order to propose more generic solutions, some learning-based methods aimed at working with the query-by-string paradigm have emerged. Instead of learning a model for whole words, the main idea is to learn models for individual characters and the relationships among them. Such models, however, are trained on the whole word or even on complete text lines without needing an explicit character segmentation. Fischer et al. proposed in [17] to use Hidden Markov Model (HMM) character models trained over complete text lines whereas Frinken et al. proposed in [13] the use of bidirectional long short-term memory neural networks to infer individual character presence. An usual inconvenient of learning-based methods is that they often require an important amount of annotated data in order to perform well. However, Rodriguez-Serrano and Perronnin recently proved in [91] that the amount of training data needed can be drastically reduced by combining handwritten examples of the word to model together with synthetic samples generated by different computer fonts. However, we believe that such approaches still present an important drawback. Learning character models with either an HMM [17] or a neural network [13] allows to *on-the-fly* generate a word model from the query the user typed in the keyboard. But this on-line generated model has to be used to process the whole collection at query time. When dealing with larger and larger datasets, the computational cost quickly becomes excessive, making such approaches not scalable at all. In contrast, example-based approaches are easily scalable to large collections since they holistically represent handwritten words by numeric feature vectors, and thus word spotting frameworks can be used together with state-of-the-art indexation strategies. Therefore, recent query-by-string methods aimed at representing words in a numerical n-dimensional space, which can be efficiently indexed, have been proposed [62, 92]. Such approaches find a common subspace between textual and visual descriptions of the words, allowing the user to cast a textual query and retrieve words that were just described visually.

In this chapter, we propose a ***query-by-string*** and a ***query-by-speech*** word spotting methods where word images in the training set are represented both by a textual and visual representations or by audio and visual representation. This allows the user to cast spoken or written queries. Such paradigm provides several benefits. First, it produces a more user-friendly query experience than classical query-by-example methods. Second, since the final representation is a numeric feature vector, the solution is scalable to large collections, providing sub-linear query times when used in combination with off-the-shelf indexing strategies. Finally, we believe that the query-by-speech paradigm is even more ergonomic than query-by-string approaches, since we get rid of the keyboard, making it more easily integrable in some specific scenarios like museum exhibitions or keyword search engines integrated in smart-phones.

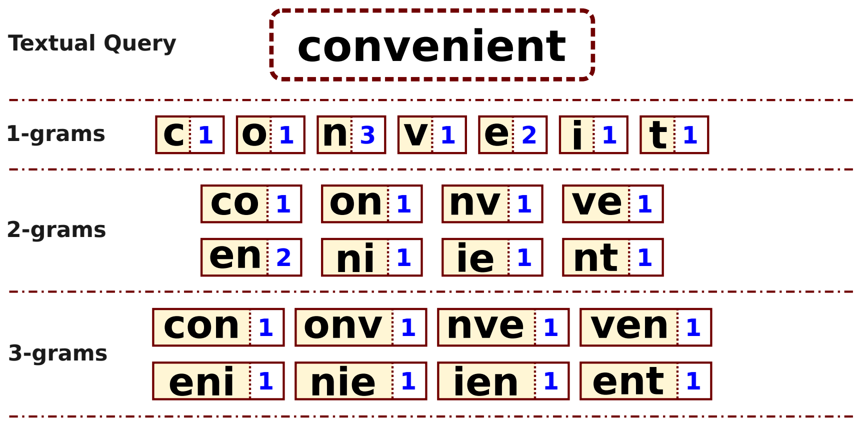


Figure 4.1: Example of the n-gram textual descriptor.

Additionally, the proposed framework presents the advantage that without any additional cost, it can be used backwards. Once we have the trained model, the same system can be used either to cast spoken queries to retrieve handwritten words that were just represented visually, but also given an image of the word, it can retrieve the most similar utterance that we have indexed. Our proposed system can thus be seen either as a query-by-speech handwritten keyword spotting, or as a basic ***handwritten*** Text-to-Speech (TTS) system.

The remainder of the chapter is organized as follows. The different word representations are described in Section 4.2. Subsequently, in Section 4.3 we detail how to different representations can be combined together though the use of latent semantic analysis. In Section 4.4, we overview the retrieval step and how from a query represented with one modality we are able to retrieve word instances that are only represented with the other modality. The experimental results are presented in Section 4.5. We finally draw our conclusion remarks in Section 4.6.

4.2 WORD REPRESENTATIONS

As previously stated, words are represented by three different cues, relying on textual information, visual information or audio information. In this chapter, we only focus on the word spotting problem assuming the words are already segmented by a layout analysis step. Here, segmentation errors are not considered and the word segmentation is manually generated. In this section, we will present the work-flow used to generate the word snippet representations.

4.2.1 *Textual Representation*

The basic block used to represent word transcriptions are n-grams of characters. The transcriptions are divided into a set of consecutive and overlapping blocks of unigrams, bigrams and trigrams respectively. This simple representation allows to extract information from which characters compile a word and encode some neighborhood information. An example of the n-gram frequencies generated by a transcription is shown in Fig. 4.1.

The textual descriptor $\mathbf{f}_i^t$ is obtained by simply accumulating the occurrences of each n-gram into a histogram and normalizing this histogram by its L_2-norm. The number of different n-grams available is obtained by generating a codebook which maps each n-gram into a dimension of the textual descriptor. This codebook is generated from the training set where the textual information is available. Then, in the retrieval phase, the n-grams which do not appear in the codebook are simply ignored.

4.2.2 *Audio Representation*

Let us first detail how we create the audio signals of each document word and how we describe these audio signals.

4.2.2.1 *Synthetic Speech*

In order to carry out our experiments, we need recordings of the words appearing in a collection of handwritten documents. To produce a large enough set of audio records both for the train and the test phases of the algorithm, we decided to use synthetic voices rather than actually recording audio clips. We have used three different TTS engines, namely the Festival Speech Synthesis System [93], the Google TTS API[1] and the AT&T's Natural Voices ™ software[2]. The Google TTS engine yields an utterance with a single voice for each word while Festival and AT&T TTS can create utterances with five different voices for the same word. The Google and AT&T software create audio records of a higher quality than Festival software obtaining in general audio recordings closer to natural voices. Finally, the AT&T software is only used to synthesize a selected set of words that we utilize as queries to evaluate the algorithm's robustness against unheard speakers in the training phase.

4.2.2.2 *Bag-of-Audio-Words*

In order to extract a feature vector from audio signals, we have chosen the Perceptual Linear Prediction (PLP) framework [94]. After applying an overlapped hamming window to the speech signal, PLP features are extracted on the short-term spectrum of speech. PLP uses several psychophysically based transformations to extract a set of cepstral coefficients. A subsequent filtering step named RASTA, proposed by Hermansky et al. in [95] is applied in order to make PLP analysis more robust to spectral distortions.

However, the RASTA-PLP method still outputs a time signal that depends on the length of the pronounced utterance. In order to have a fixed-length feature vector, we have applied the BoAW framework over these time series features. From a set of utterances represented by their RASTA-PLP cepstral coefficients, we create a codebook by clustering the cepstral coefficients using the k-means algorithm. In the experiments presented in is chapter, we use a codebook of 8 192 audio words. A spoken word is then represented by an histogram which accumulates the frequencies of each audio word. We can see an example of a couple of utterances, their RASTA-PLP cepstral features and a simplified BoAW representation (just 10 codewords) in Fig. 4.2. In order to encode some sequential information, we divide the audio information into different temporal bins and the histogram of audio words are independently accumulated for each of them. These temporal bins are created using a pyramid structure similar to the Spatial Pyramid Matching method [71] typically used for visual information. We employ a two level pyramid which

1 http://www.translate.google.com/translate_tts?tl=en&q=Hello
2 http://www2.research.att.com/~ttsweb/tts/demo.php

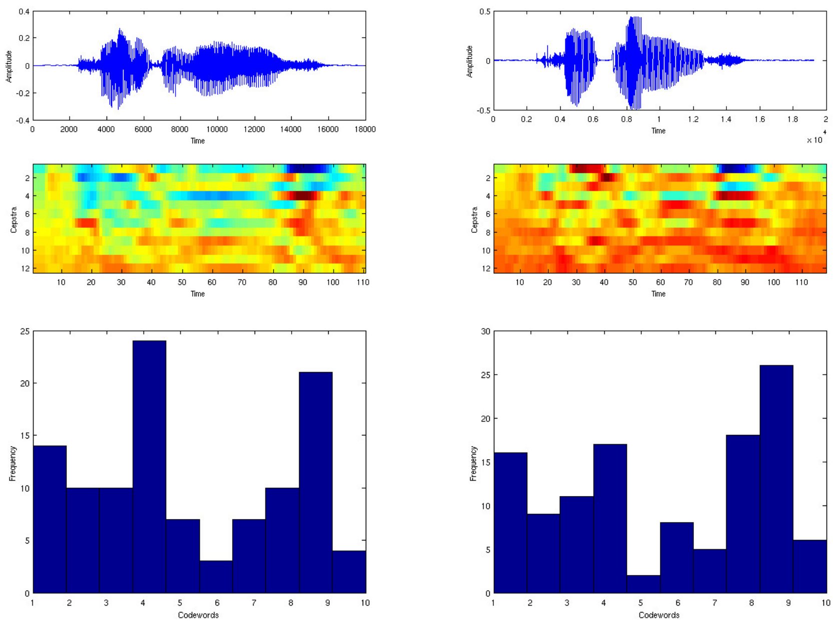


Figure 4.2: Example of utterances of the word ***companies***, their RASTA-PLP cepstral coefficients and their BoAW histograms for two different voices.

halves the temporal bins at each new level, resulting in seven different temporal bins. These temporal bins are concatenated resulting in a 57 344-dimensional audio descriptor which is L2-normalized to obtain the final audio representation $\mathbf{f}^a$.

4.2.3 *Visual Representation*

Word image snippets are represented by a descriptor obtained using the BoVW framework. This framework has obtained a good performance in heterogeneous problems in the computer vision albeit its simplicity and recently has shown to be a powerful tool to describe handwritten words in a word spotting task [57]. Using the evaluation of the performance of the different parameters of the BoVW model of Chapter 2 as reference, we have carefully tuned the methods used at the different steps of the BoVW model generation so that we obtain good performances when visually computing similarities among words. Let us detail the followed steps.

4.2.3.1 *Region sampling*

In order to create a histogram of visual words, we first need to extract local information from the image. We densely sample local regions over the image at different scales. The sampled local regions consist in squared regions having sizes of 20, 30 and 40 pixels which are sampled at a constant step of 5 pixels. Since we do not expect to face rotation distortions of the handwritten text, the dominant orientation of the local regions is ignored. Then, the local regions are characterized by the Histogram of Gradients local descriptor [67] using its standard configuration. Local regions having an accumulated gradient module lower than a certain threshold are rejected.

4.2.3.2 *Encoding*

Once descriptors have been sampled from the image, we need to convert them into visual words by means of a codebook. To generate the codebook, we have randomly sampled two million descriptors of the document images and used the k-means algorithm to create a codebook with 4 096 codewords. Then, the local descriptors are encoded into visual words using soft-assignment which lessen the effects of encoding errors. For each descriptor, we select the three nearest codewords and weight the contribution of the selected codewords by means of the Locality-constrained Linear Coding (LLC) algorithm [69]. The number of nearest neighbors has been validated empirically using different parameters in the BoVW technique and, this value has consistently obtained better retrieval performances. Although the number of neighbors can seem small, this result is coherent with the results shown in the original paper [69] where, when using a small number of neighbors, a better performance is reached than when using a larger number neighbors.

4.2.3.3 *Spatial Representation*

The BoVW model discards all the spatial information of the local regions where the descriptors have been sampled resulting in a orderless representation. However, spatial information can greatly increase the representativity of the visual descriptor. During the creation of the visual descriptor, we have experimented using different spatial configurations and concluded that adding spatial information to the BoVW representation of handwritten words always boosts the retrieval performance of the system.

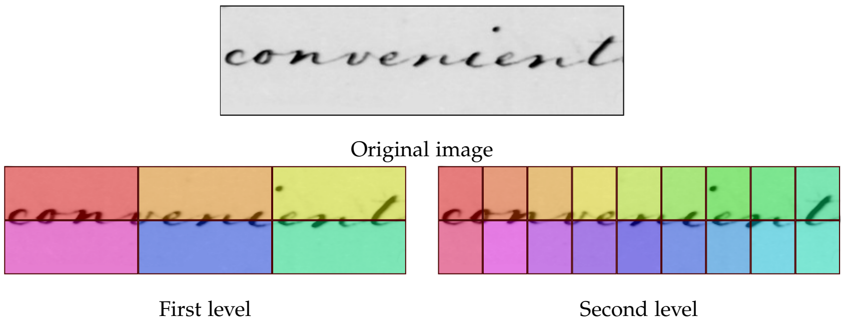


Figure 4.3: Distribution of the spatial bins in the two levels of the spatial pyramid.

We have added spatial information by means of the spatial pyramid proposed by Lazebnik et al. in [71]. Experimental results show that partitions in the abscissa have a greater discriminative power than partitions in the ordinate. Therefore, we created a two level pyramid, were the first level has 3 partitions in the X axis and 2 partitions in the Y axis and the second level tripled the number of divisions in the X axis while keeping the same amount of partitions in the Y axis. Fig. 4.3 shows an example of the prosed configuration. Then, the histogram of visual words is generated by pooling the visual words which fall at each spatial bin in a different histogram. The histograms of visual words of each level of the pyramid are independently concatenated and normalized using the L_2-norm. This ensures that the histograms of visual words of each level of the pyramid have the same contribution to the final descriptor of visual words. Finally, the visual descriptor $\mathbf{f}_i^v$ is obtained concatenating the histograms of visual words of each pyramid level. This configuration results in 24 different spatial bins which combined with the 4 096 codebook results in a 98 304-dimensional visual descriptor.

4.2.3.4 *Normalization*

We use the power normalization proposed by Perronnin et al. in [72]. This method applies the following normalization function at each bin of the visual words descriptor

$$g(x) = sign(x)|x|^{\alpha},$$

where $0 < \alpha < 1$ is the power normalization factor. Finally, an L_2-normalization is applied to the whole descriptor in order to obtain the final visual representation.

4.3 MULTI-MODAL INFORMATION FUSION

Until now, word snippets are independently represented by three information modalities: a textual descriptor, an audio descriptor extracted from different utterances of the word and a visual representation generated from the visual words extracted from the document images. Our proposal is to bring together the textual and visual descriptors or the audio and visual descriptors into a common representation space, so that we are able to use queries from one modality to retrieve words described using the other modality. We want to be able to use textual queries to retrieve word snippets which are described solely by visual descriptors or following the opposite path, to retrieve word utterances given an image of a handwritten word. This is achieved by searching a transform which projects both the visual and audio information into a common feature space, so that ideally the audio and visual descriptors of the same word will be similar in the transformed space.

This transform is obtained by assuming that both visual and audio features will co-occur for two different instances of the same word. Therefore, we can find a set of abstract topics that represent distributions of different types of features associated to some underlying semantics of the indexed words. We calculate these abstract topics with the Latent Semantic Analysis (LSA) algorithm [85] which uses the Singular Value Decomposition (SVD) step to find a linear transform which projects both audio and visual information into a set of abstract topics in an unsupervised way.

In order to calculate the linear projection matrix, we first arrange the word descriptors of the training set in a descriptor-by-word matrix $\mathbf{A} = [\mathbf{f}_1 \ldots \mathbf{f}_i \ldots \mathbf{f}_M]$, where M is the number of train samples and $\mathbf{f}_i = \left[\mathbf{f}_i^a, \mathbf{f}_i^v\right]$ is obtained by concatenating the audio $\mathbf{f}_i^a$ and the visual $\mathbf{f}_i^v$ descriptors of the i-th training sample. Since we can generate multiple utterances for each word, the number of train samples M corresponds to the number of word snippet images multiplied by the number of voices used to create the audio descriptors. The LSA algorithm obtains the linear projection by decomposing this descriptor-by-word matrix in three matrices using a truncated SVD:

$$\mathbf{A} \simeq \hat{\mathbf{A}} = \mathbf{U}_T \mathbf{S}_T \left(\mathbf{V}_T\right)^{\top},$$

where T is the number of abstract topics (i.e., the dimensionality of the transformed space) and $\mathbf{U}_T \in \mathbb{R}^{D \times T}$, $\mathbf{S}_T \in \mathbb{R}^{T \times T}$ and $\mathbf{V}_T \in \mathbb{R}^{M \times T}$. Finally, the transformation matrix $\mathbf{X}_T$ is calculated as

$$\mathbf{X}_T = \mathbf{U}_T \left(\mathbf{S}_T\right)^{-1},$$

so that the descriptor of i-th word snippet $\mathbf{f}_i$ is projected into the new feature space by simply $\hat{\mathbf{f}}_i = \mathbf{f}_i^\top \mathbf{X}_T$.

4.4 RETRIEVAL PHASE

The LSA model has been calculated from word descriptors where the visual and one of the other two modalities (i.e. textual or audio) were available. However, in the corpus indexing and the retrieval phases only a single source of information is present. Only visual information is available when creating the corpus index. In this phase, the visual descriptor of each image word snipped $\mathbf{f}_i^v$ is calculated and projected into the transformed space by

$$\hat{\mathbf{f}}_i = \left[\mathbf{o}_x^\top, \mathbf{f}_i^{v\top}\right] \mathbf{X}_T,$$

where $\mathbf{o}_x$ is a zeros vector with the same dimensionality than the codebook of the other modality. The projected vectors $\hat{\mathbf{f}}_i$ are the descriptors that are actually used to represent the word instances in the document.

4.4.1 *Word snippet retrieval*

In the query phase, an user inputs the non-visual query which is projected into the transformed space by

$$\hat{\mathbf{f}}^q = \left[\mathbf{f}_i^{x\top}, \mathbf{o}_v^\top\right] \mathbf{X}_T,$$

where $\mathbf{o}_v$ is a zeros vector which has the same dimensionality as the visual descriptor. Then, the cosine distance between the projected query $\hat{\mathbf{f}}^q$ and the projected visual descriptors $\hat{\mathbf{f}}_i$ is used as a similarity measure and generate a ranked list.

This procedure allows to retrieve word snippets that have been described using only visual information using queries from another modality. This is possible since the LSA algorithm has found relationships between the visual words and the features of the other modality in the training phase. Then, even if a source of information is not present in one of the descriptors, we are still able to rank and find relevant instances in the indexed documents.

4.4.2 *Audio Retrieval*

In addition, without any additional cost, we can use the proposed method backwards to reverse the retrieval task. That is to index audio information and use handwritten word images as queries. Such audio retrieval can be seen as a handwritten text-to-speech task which does not have an implicit recognition step. Furthermore, we generate multiple ranked lists for a single visual query by indexing the audios of the different available voices separately. These output lists are re-ranked into a single result using the Borda count algorithm to obtain a better retrieval performance. Finally, when only returning the first element of the ranked list to the user, such audio retrieval method can be seen as a handwritten text-to-speech system.

4.5 EXPERIMENTAL RESULTS

The proposed query-by-X word spotting method has been evaluated in the George Washington (GW) database [96, 31] consisting of 4 864 segmented

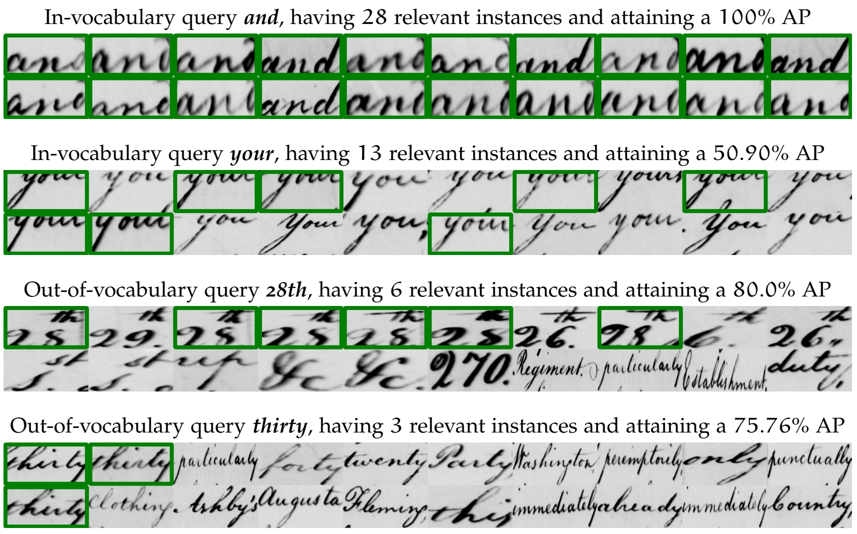

In-vocabulary query ***and***, having 28 relevant instances and attaining a 100% AP
In-vocabulary query ***your***, having 13 relevant instances and attaining a 50.90% AP
Out-of-vocabulary query ***28th***, having 6 relevant instances and attaining a 80.0% AP
Out-of-vocabulary query ***thirty***, having 3 relevant instances and attaining a 75.76% AP

Figure 4.4: Examples of the 20 most similar word images and their average precision (AP) returned by the system for some queries.

Table 4.1: State-of-the-art queried-by-string word spotting results for the GW database.

Method	Segmentation	Cross-validation	Queries	mAP
Proposed	Word-level	4 folds	All words	56.54%
			In-vocabulary	76.2%
Liang et al. [78]	Word-level	5 folds	38 queries	67% at rank 10
Fischer et al. [17]	Line-level	4 folds	In-vocabulary	62.08%
Frinken et al. [13]	Line-level	4 folds	In-vocabulary	71%

words. In order to train the LSA model that brings the visual information and another modality together, we need a portion of the database to be annotated. Therefore, the database is divided into four different folds. Then, the system is trained using three of these folds and evaluated in the remaining one. At query time, all the words in the test set can be used. However, not all the words appearing in the test fold might be present in the train set. We will therefore report in our experiments two different measures, the performances reached when considering all 1 829 words as queries and the performances reached when not considering out-of-vocabulary words, i.e. just casting the queries that are present in both train and test sets, that is 1 090 queries.

4.5.1 *Query-by-String results*

First, we are going to discuss the retrieval performance of the system when visual and textual information is merged by the LSA model.

4.5.1.1 *Qualitative results*

We present in Fig. 4.4 some qualitative results of the system. Framed in green appear the words that are considered relevant in our ground-truth. Note that in our experiments we have not filtered stop-words nor removed words with few appearances. We neither applied any stemming process, so for instance when querying the word ***your***, results as ***you*** or ***yours*** are accounted as negatives. We can also appreciate the performance difference when querying in and out vocabulary items.

4.5.1.2 *Quantitative results*

We report the obtained mean Average Precision (mAP) results of our system in Fig. 4.5 depending on the amount of topics T.

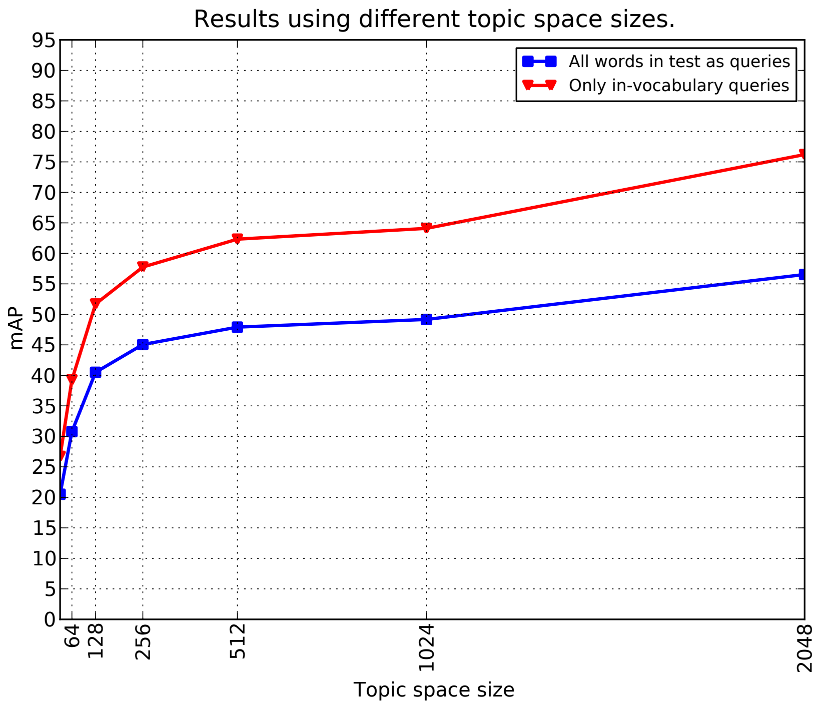


Figure 4.5: mAP attained by the system using different number of dimensions in the topic space.

The resulting mAP shows that the larger the dimensionality of the topic space, the better the retrieval performance of the system. For the largest topic space size, we reached a 76.20% mAP when using only in-vocabulary queries. When considering all the queries in the test set, the performance drops to a 56.54% mAP. This is due to the out-of-vocabulary words. In the worst-case scenario in which just out-of-vocabulary terms are queried, the system yields a 27.08% mAP.

It is worth noting that out-of-vocabulary words have a small number of instances and therefore are often penalized when computing the mAP measure if not ranked perfectly. Yet, the system still manages to retrieve them within the upper part of the rank. Therefore, we decided to report the recall at different ranks in Fig. 4.6. We can appreciate that even when just looking at the 10 first returned elements, we already achieve a 48.84% recall in the out-of-vocabulary scenario and a 68.5% recall in the in-vocabulary setup.

4.5.1.3 *Comparison with the state of the art*

Although it is not straightforward to provide a fair comparison between different methods, we report in Table 4.1 some state-of-the-art performances

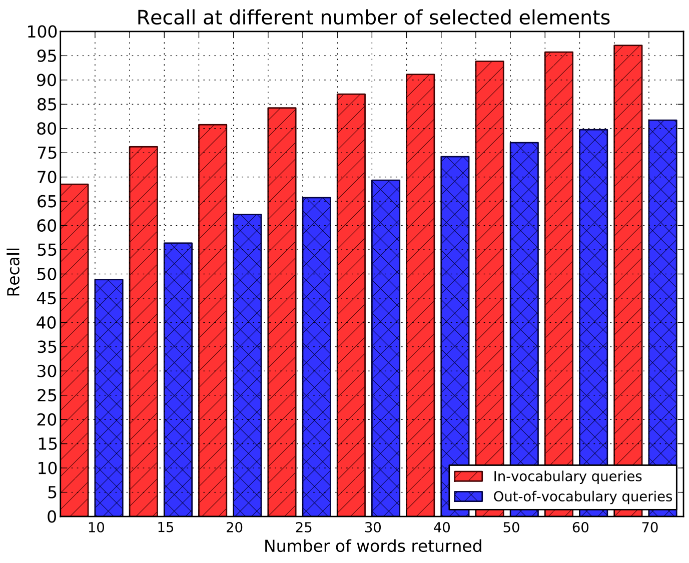


Figure 4.6: Different recall values obtained while increasing the number of elements returned by the system.

of query-by-string methods that also used the GW dataset. Both [17] and [13] use a very similar experimental setup like ours although they evaluate the average precisions at line level. In [78] Liang et al. used a selected set of queries to obtain the mAP just considering the first 10 elements in return. Using only these queries to evaluate the proposed method, our performance increases to a 83.12% mAP. Note as well that none of the considered methods can be easily extended to work in large-scale environments whereas the retrieval in our vector representation can be performed sub-linearly using standard indexing structures.

4.5.2 *Query-by-Speech results*

The proposed method attains a good performance in a query-by-string scenario, now we are going to show it's performance when audio information is used instead of textual as an alternative query modality.

4.5.2.1 *Query-by-speech Qualitative Results*

We present in Fig. 4.7 some qualitative results of the system when casting queries pronounced by either Google's voice (used as well in training) or an AT&T voice, which has not been heard by the system in the training phase. Framed in green appear the words that are considered relevant in our ground-truth. Note that in our experiments we have not filtered stop-words nor removed words with few appearances. We neither applied any stemming process, so for instance when querying the word ***orders***, results as ***order*** are accounted as negatives.

4.5.2.2 *Query-by-speech Quantitative Results*

We report the obtained mAP results of our system in Fig. 4.8 depending on the amount of topics T when using Festival's and Google's voices as queries.

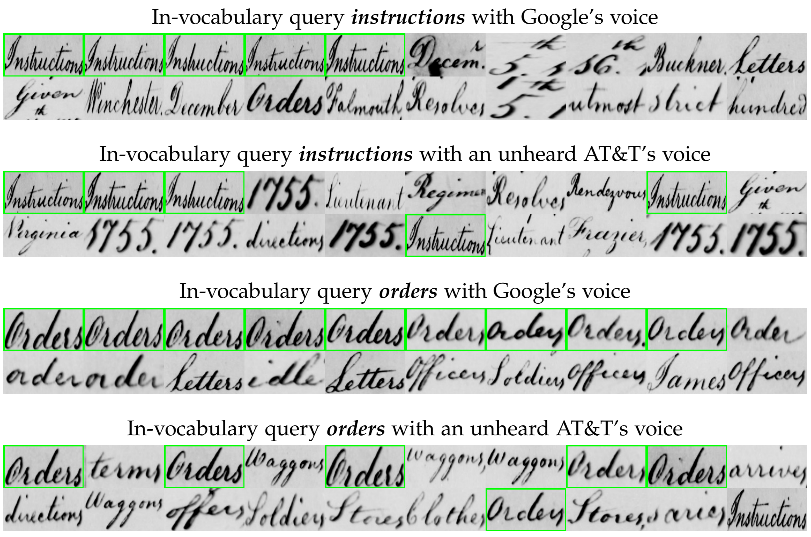

In-vocabulary query ***instructions*** with Google's voice
In-vocabulary query ***instructions*** with an unheard AT&T's voice
In-vocabulary query ***orders*** with Google's voice
In-vocabulary query ***orders*** with an unheard AT&T's voice

Figure 4.7: Examples of the 20 most similar word images returned by the system for different spoken queries.

The resulting mAP shows that the larger the dimensionality of the topic space is, the better the retrieval performance of the system. For the largest topic space size, we reached a 78.38% mAP when using only in-vocabulary queries. When considering all the queries in the test set, the performance drops to a 51.24% mAP.

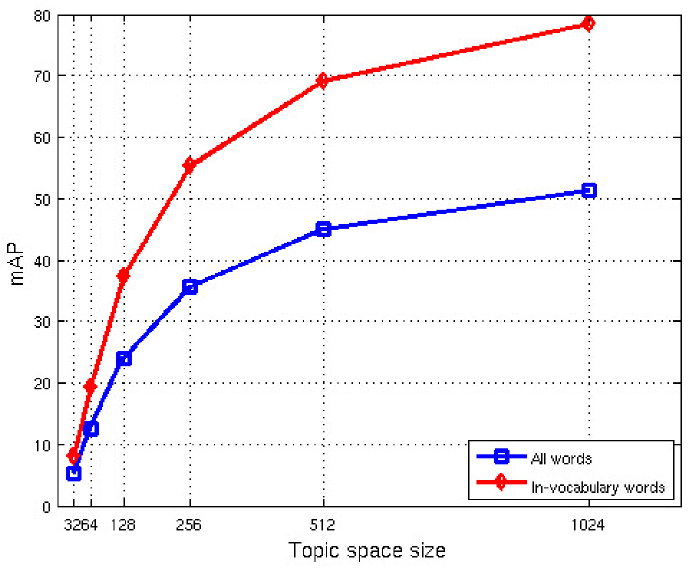


Figure 4.8: mAP attained by the system using different number of dimensions in the topic space.

We report in Table 4.2 a performance comparison of the proposed query-by-speech method against the previous query-by-string presented in section 4.5.1.2 and the query-by-example set up presented in Chapter 2. All three methods use exactly the same visual descriptors and both query-by-speech and query-by-string evaluation protocols are exactly the same. Obviously,

Table 4.2: Comparison with our previous work using the same visual description.

Method	Evaluation	Queries	mAP
Query-by-speech (proposed)	4 folds	All words	51.24%
Query-by-speech (proposed)	4 folds	In-vocabulary words	78.38%
Query-by-string [62]	4 folds	All words	56.54%
Query-by-string [62]	4 folds	In-vocabulary words	76.2%
Query-by-example [18]	1 fold	All words	72.98%

Table 4.3: Retrieval performance using unheard voices as queries.

Voice	mAP all words from [78]	mAP for in-vocabulary words from [78]
AT&T's voice *Claire*	15.46%	16.12%
AT&T's voice *Crystal*	14.72%	15.27%
AT&T's voice *Lauren*	10.69%	11.13%
AT&T's voice *Mike*	15.98%	16.74%
AT&T's voice *Rich*	13.01%	13.64%
Average	13.97%	14.58%

the query-by-example experiment do not require any fold partition and each word snippet is used as query in a leave-one-out fashion. It is interesting to see that even if the audio queries are noisy while our previous string queries were not, the reached performances are quite comparable, even delivering better results in the in-vocabulary query setup. Obviously the reached performances in the case of multi-modal representations do not reach the performance levels of just using a visual descriptor in a query-by-example fashion, but we have to take into account that both query-by-string and query-by-speech do not require a manual search of the query template.

4.5.3 *Query-by-speech with Unheard Voices*

In the previous experiment, we used as queries word utterances pronounced by the same voices that were using in the training phase. We report in Table 4.3 the obtained results by the proposed system with $T = 1024$ when the casted queries are pronounced by a different voice than the ones used to train the LSA model. Here we have used the five different AT&T's voices to pronounce a small set of 38 queries, which are the ones used by Liang et al. in [78]. Here we can observe an important performance drop when compared with our previous experiment. However, such low mAP values might still be acceptable for plain users in certain scenarios, since despite the low mAP scores, usually some positive word instances are well ranked in the top positions as we have seen in Fig. 4.7.

Table 4.4: Accuracies for the Handwritten Text-to-speech.

Voice	Accuracy all words	Accuracy in-vocabulary words
Google voice	31.13%	37.86%
Festival voice 1	55.68%	67.71%
Festival voice 2	58.40%	71.06%
Festival voice 3	55.90%	67.75%
Festival voice 4	61.04%	74.18%
Festival voice 5	56.84%	68.91%
Borda Count	70.00%	85.24%

4.5.4 *Handwritten Text-to-speech via Audio Retrieval*

Finally, we account the recognition accuracies of the handwritten text-to-speech task in Table 4.4. The table reports the percentage of queries in which we retrieved the correct utterance at the first rank. As expected, we obtain a significant improvement when just querying in-vocabulary words in comparison to using the whole corpus. We also observe that the performance reached among different voices is quite disperse. Since our approach is not a TTS engine per se, but an audio retrieval system, given an image query we can combine several retrieval outputs from different voices to overcome such diversity and obtain better performances. In our case, the Borda count combination of the ranks obtained with each voice leads to an important improvement over the indexation of individual voices. The method yields promising results despite its simplicity and the fact that it does not entail an explicit recognition of the handwritten words.

4.6 CONCLUSIONS

In this chapter, we have proposed a method that enables to fuse two different types of information modalities into a single feature space in a simple and efficient manner. The proposed method has been used both in a query-by-string and in a query-by-speech spotting system for historical handwritten document images. The system reaches state of the art word spotting performance when using in-vocabulary queries, i.e. when the both modalities of information are present in the training set. However, there is still room for improvement in the case that the system faces out-of-vocabulary queries. Additionally, we have demonstrated that the system is bi-directional and it can be used to retrieve the other modality only from visual cues. We have shown that the query-by-speech model can also be used as a simple handwritten text-to-speech system. Instead of indexing visual signatures, we can index the utterance signatures of words generated by multiple voices. Then, the speech of a word image is generated by querying the image signature into the index and selecting the utterance with the most consensus.

5 SEMANTIC CODEBOOK

Word-spotting methods based on the Bag-of-Visual-Words framework have demonstrated a good retrieval performance even when used in a completely unsupervised manner. Although unsupervised approaches are suitable for large document collections due to the cost of acquiring labeled data, these methods also present some drawbacks. For instance, having to train a suitable "codebook" for a certain dataset has a high computational cost. Therefore, in this chapter we present a database agnostic codebook which is trained from synthetic data. The aim of the proposed approach is to generate a codebook where the only information required is the type of script used in the document. The use of synthetic data also allows to easily incorporate semantic information in the codebook generation. So, the proposed method is able to determine which set of codewords have a semantic representation of the descriptor feature space. Experimental results show that the resulting codebook attains a state-of-the-art performance while having a more compact representation.

5.1 INTRODUCTION

Handwritten keyword spotting is the document image retrieval task devoted to obtain a ranked list of words that are relevant to a user's cast query. In its most simple formulation, document images are already pre-processed and segmented into individual words. The user casts a query in form of an example of the keyword he wants to retrieve, to then obtain a ranked list in which desirably the words having the same transcription are ranked better than the rest of the words. This paradigm is known as segmentation-based query-by-example keyword spotting, which is the scenario in which we are centered in this chapter.

Since the seminal papers of Manmatha et al. [97, 16] that introduced the problematic of handwritten keyword spotting more than twenty years ago, many advances have been proposed. Performances reached on public datasets have been steadily increasing with the proposal of better feature representations and retrieval strategies. In addition to the overall retrieval accuracy, many other advances have been made as well. Segmentation-free methods have been proposed [61, 98, 39, 44, 99], query-by-string techniques have emerged [62, 100, 92, 101, 102], and different methods have incorporated techniques from the information retrieval field such as relevance feedback [103], re-ranking [98] or query expansion [98].

Although systems which incorporate a learning step to improve the retrieval accuracy obtain a better performance than systems purely based on visual information [39, 92, 104, 100], unsupervised methods are more desirable in certain scenarios. For example, in large document collection with hundreds of pages and without any annotation, an unsupervised method can be used directly without manually annotate a subset of pages. Also, an unsupervised method can be used for instance to group similar looking word snippets into clusters [31]. This word clusters then can be used to

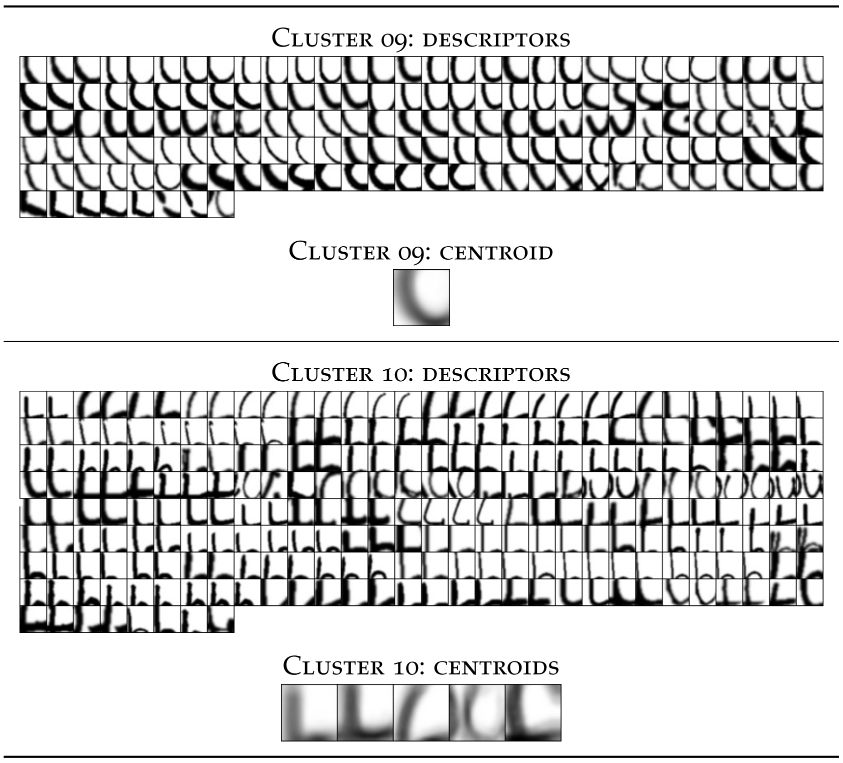


Figure 5.1: Two examples of the clusters generated by the codebook. Cluster 09 contains descriptors from symbols [0, 8, c, g, b, j, h, p, f, x, u], while Cluster 10 is formed by descriptors from [h, b, 4, 6, t, k, u, o, f, d, j, l, p, y, w, a, q, i, n]. Cluster 10 is represented by 5 centroids because it has at least a nested cluster.

simply accelerate the retrieval system but also to propagate the annotations provided by the user or to search consensus to the annotations given by a text recognition system.

Unsupervised word spotting methods based on the Bag-of-Visual-Words paradigm can attain a high retrieval performance when the methods used at each step are selected carefully [18]. Besides its retrieval accuracy these methods have the advantage that words are represented by a fixed-length vector, so standard dimensionality reduction techniques have been used to efficiently store and index large collections of documents [61, 105]. However, these methods require the use of a codebook to encode locally extracted descriptors into codewords. The performance of the system is dependent on the quality of the codebook and the number of codewords which yields a better trade-off between dimensionality (i.e. memory usage) and performance has to be found. On small datasets, creating a codebook does not have a high cost, but, in large collections, with hundreds of thousands of words written by multiple writers the computational cost of generating the codebook might be prohibitive. A straightforward solution is to randomly sample a subset of word snippets to generate the codebook. However, this approach has the drawback that certain characters and writing styles may be underrepresented by the codebook. Therefore, we propose a codebook trained from synthetic data which incorporates semantic information in the generation process to determine the optimal size and cardinality of the codewords. The use of synthetic data has several advantages (c.f. [106, 104, 91]): it ensures that all characters are properly represented and it allows to simulate the script variability present in documents written by multiple writers. Since there are many true-type fonts which replicate the human handwritten style it is easy to incorporate many different versions of the same character. Additionally, it also allows to incorporate semantic labels to the information used to create the codebook. This extra information is used to obtain an actual measure of the clustering accuracy of the codebook. Thus, the codebook is able to automatically determine the amount of codewords needed to properly represent the feature space. The main contributions of the chapter are threefold. We present a method to generate a codebook from synthetic data. A new procedure used to encode descriptors into visual words is proposed. Finally, we provide the basis of a method to encode descriptors efficiently.

The rest of the chapter is structured as follows: in Section 5.2 we present the method used to to create the codebook from synthetic data. Then, in Section 5.3, we show how descriptors are encoded into visual words. Finally, in Section 5.4, we present the spotting performance attained by the proposed codebook and, in Section 5.5, we discuss the main contributions of the chapter.

5.2 SYNTHETIC CODEBOOK GENERATION

The codebook is trained with Histogram of Oriented Gradients (HOG) descriptors [67] extracted from characters generated by true-type fonts that replicate the human handwriting style. Training samples are then pairs $\mathbf{x} = (\mathbf{d}, c)$, where $\mathbf{d}$ is the HOG descriptor and $c \in \mathcal{C}$ is the semantic label of the character. Therefore, the training set is formed by the training samples extracted from all the considered characters. Additionally, we also incorporate training samples that cover multiple characters (i.e. bigrams). We use the statistical data reported by Jones and Mewhort to select the bigrams most common in the English language [107]. Therefore, the training set is generated from 62 different characters and 1874 character bigrams, and has 36 different semantic labels as we do not differentiate between upper and lower case characters.

We generate the codebook by fist grouping the training samples using agglomerative clustering and then using the Shannon entropy to partition these tree into multiple clusters.

5.2.1 *Agglomerative Clustering*

Agglomerative clustering is a bottom-up hierarchical clustering algorithm that recursively groups the two closest clusters until all samples are grouped together. This procedure generates a binary tree that later has to be partitioned into clusters by using some criteria (e.g. fixed number of clusters, cluster compactness [108], *a contrario* approach [109]). The distance between clusters can be computed in many ways but the most common are the distance between the closest two elements (i.e. single-linkage), the distance between the two further away elements (i.e. complete linkage) and the average distance between all elements of the cluster. The estimation of these distances though limit the practical usage of the method as the complexity of the standard algorithms have a $O(N^2)$ complexity both in terms of memory and runtime. For average distances, Leibe et al. proposed the average-link clustering with nearest neighbor chains [108] which reduces the memory complexity to $O(N)$. However, their algorithm can only be used when the dot-product or the Euclidean distance are used as similarity measure between clusters. Therefore, we decide to use the dot-product as similarity measure as HOG descriptors are L_2-normalized so in this case the dot-product is equivalent to the Euclidean distance. Furthermore, the dot-product also allow us to use other distance measures via explicit feature maps [110, 111]. Hence, we are also able to compare HOG descriptors using the Histogram intersection and the χ^2 similarity measures.

Finally, we need to reduce the number of samples used to create the codebook. Although the memory complexity has been reduced to $O(N)$ the temporal complexity remains $O(N^2)$. In order to improve the algorithm runtime, we reduce the number of samples extracted at each character. Instead of using random sampling, we apply the agglomerative clustering at each character independently and then we generate clusters by selecting the sub-trees that have at least R samples. These clusters are then fed to the general agglomerative tree to generate the final codebook.

5.2.2 *Shannon Entropy*

Once we have generated the binary tree, we need a method to partition the nodes in order to obtain the clusters. We want that the partitions are created automatically from data so the user does not need to tweak another parameter. Since the samples have the character label besides the descriptor, we can use this information to partition the tree into semantically meaningful clusters. Therefore, we calculate at each node the Shannon entropy, as in [112]:

$$S_c(L,T) = \frac{2I_{c,t}(L)}{H_c(L) + H_t(L)}$$

where H_c is the class entropy of the samples at the node, H_t is the entropy of the samples division at the two children and $I_{c,t}(L)$ is the mutual information of the split:

$$\begin{aligned}
H_c(L) &= -\sum_{c\in\mathcal{C}} \frac{n_c}{n}\log_2\frac{n_c}{n},\\
H_t(L) &= -\frac{n_l}{n}\log_2\frac{n_l}{n} - \frac{n_r}{n}\log_2\frac{n_r}{n}\\
I_{c,t}(L) &= H_c(L) - \frac{n_l}{n}H_c(L_l) - \frac{n_r}{n}H_c(L_r)
\end{aligned}$$

Here, L, L_l and L_r respectively denote the set of samples at the current node, the left child and the right child and, n, n_l, n_r and n_c denote the cardinality of these sets with n_c being the number of samples within category c.

Using this measure, the higher the Shannon entropy the better are the categories distributed between the two descending nodes. Then, we compute this measure at each node and we partition the trees at the nodes where the Shannon entropy attains a local maxima, i.e. the nodes where it is higher than its direct ascendants and descendants. In order to avoid generating small clusters, nodes which do not have at least 50 samples are not considered. By following this procedure, we are able to generate clusters automatically and these clusters have some semantic significance. In Fig. 5.1, we see an example of the descriptors grouped in two different clusters. In this example, we can see that clusters contain elements from multiple characters as the features sampled from the image are too small and do not contain enough information to perfectly discriminate between the different characters. Although a perfect semantic separation is more desirable, it is not possible to achieve without a more complex descriptor or kernels (e.g. χ^2-Radial Basis Function kernel). Finally, applying the measure locally results in clusters being nested. This means that a cluster can be a sub-tree from another larger cluster so, we may need multiple centroids to represent them properly. For example, the second cluster in Fig. 5.1 has a nested cluster and thus it is represented by multiple centroids.

5.3 DESCRIPTOR ENCODING

Once we have created the codebook we need to define how descriptors are going to be represented as visual words.

5.3.1 *Codeword Encoding*

We are going to represent descriptors using first derivative encoding [113, 114], i.e. descriptors are represented as the residual between the encoded descriptor and a selected codeword. So we need to represent that codewords can be represented by a centroid, but our agglomerative codebook can generate nested codewords. Therefore, codewords are represented by as many centroids as necessary to ensure that no overlapping exist. In Fig. 5.2, we can

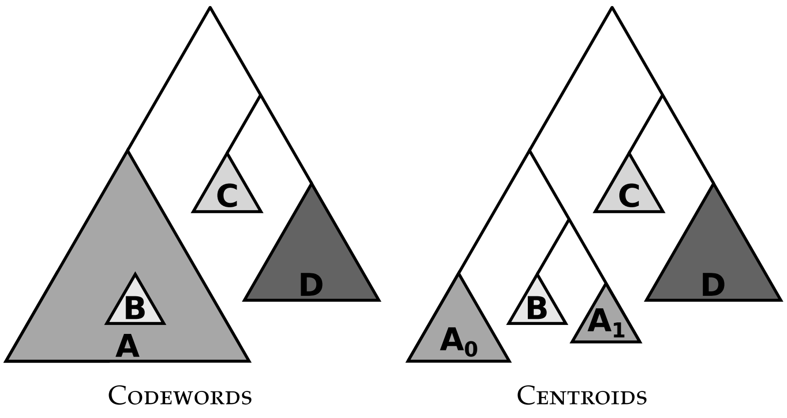


Figure 5.2: Schema of the agglomerative tree codewords and centroids.

see a simplified representation of an agglomerative tree where codeword B is nested inside codeword A. Here, codeword A is represented by two centroids, A_0 and A_1, instead of the centroid at the highest level of the sub-tree so there is no overlapping with centroid B. In Fig. 5.1, we can see a real example of the centroids representing two different clusters.

Then, the contribution of each centroid $\mathbf{C}_i$ of the codebook to encode a given descriptor $\mathbf{d}$ is given by

$$\mathbf{w}_i = W(\mathbf{d}, \mathbf{C}_i) = \exp\left(-\frac{(1 - sim(\mathbf{d}, \mathbf{C}_i))^2}{2\sigma_i^2}\right)$$

where $sim(\mathbf{d}, \mathbf{C}_i)$ is the similarity measure between the descriptor and centroid and, σ_i is the standard deviation of the similarity of the elements within the centroid. In order to increase the sparseness of the encoding, we set to zero the different weights $\mathbf{w}_i$ that satisfy that $\mathbf{w}_i/\mathbf{w}_{max} < t$ where $\mathbf{w}_{max}$ is the maximum weight and $t \in [0, 1]$ is a threshold. When $t = 0$ we use all centroids to encode a descriptor while when $t = 1$ we only use the most similar centroid. Finally, weights are normalized to ensure that the sum of all weights is 1.

These weights multiplied by the residuals between the descriptor and the centroids are the resulting encoding. Since a codeword may be represented by several centroids, the contributions of all its centroids are accumulated together. For instance, in the example at Fig. 5.2 the codebook only has 4 codewords thus the histogram of visual words has $4d$ dimensions where d is the dimensionality of the descriptor. Then, the weighted residuals from A_0 and A_1 are both accumulated in the first d dimensions of the histogram.

5.3.2 *Approximate Codebook*

The encoding method previously described is computationally intensive as it requires computing multiple exponential weights to encode a single descriptor. In order to reduce the computational cost of the descriptor encoding step, we propose the use of an additional codebook which is used to approximate the descriptors. We use a Hierarchical k-Means (HKM) similar to the Vocabulary Tree [115] to approximate the descriptors. The codebook has degree 10, we limit it at a maximum depth of 8 levels and one million leafs. It is build using a priority queue that prioritize nodes which are more populated. In this codebook, we use the Euclidean distance to compare the descriptors regardless of the distance measure used by the agglomerative codebook.

The main idea is to then use the leafs of this codebook as an approximation of the encoded descriptors. Then, we pre-compute the encoding of the leaf descriptors since we know them *a priori*. Thus, instead of computing the weight of each centroid of the agglomerative tree for a given descriptor, we only need to traverse the HKM tree and use the weights stored at the leaf. Although we are adding quantization errors when following this procedure, we are also greatly reducing the encoding computational cost which may be a desirable trade-off when dealing with large collections.

5.4 RESULTS

We generate the codebook using ten different true type fonts which generate between 4000 and 7000 descriptors per character. We group these descriptors in clusters of at least ten descriptors (i.e. $R = 10$) reducing the contribution of each character to 450-800 descriptors. Therefore, the algorithm only needs to aggregate around 50000 descriptors in each evaluated configuration. In all experiments, the Bag of Visual Words (BoVW) signature is generated by densely sampling HOG descriptors each 4 pixels from squared regions of 16, 24, 32 and 40 pixels, a spatial pyramid with 5 horizontal partitions and power factorization at 0.5. We have evaluated the codebooks obtained using different descriptor dimensionality, filter ratio and similarity measures on the George Washington dataset [96, 31]. The dataset consist of 20 pages with 4 860 segmented words. The performance of the retrieval system is evaluated computing the mean Average Precision (mAP) score for any word snippet that appears at least twice in the dataset and returning the overall performance of the system as the mean of mAP scores. In Fig. 5.3, we plot the results obtained by five different queries. All results have been obtained on a Linux box with an Intel® Xeon® E5-1620 CPU running at 3.50GHz and 16 Gb of RAM.

In Table 5.1, we show the mAP score obtained when generating the codebook with different configurations. The dimensionality column (*Dim.*) speci-

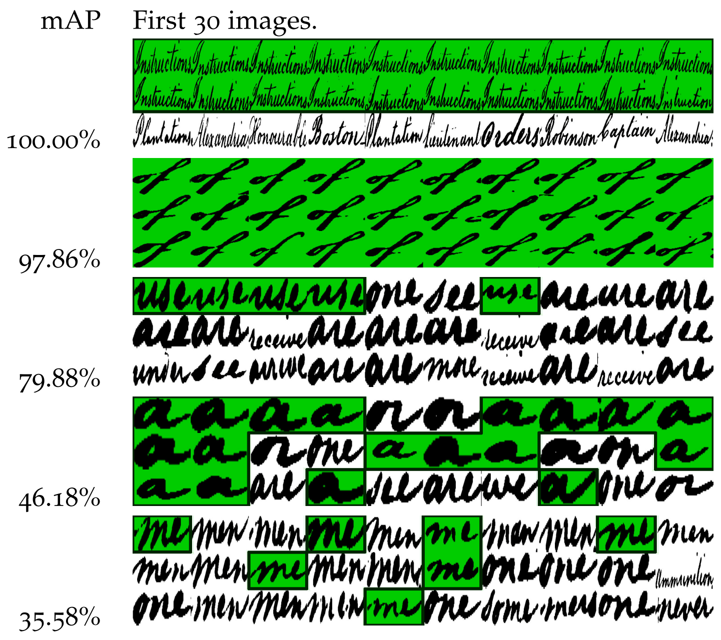


Figure 5.3: Some qualitative results obtained in the George Washington dataset.

		r ⩾ 0%		r ⩾ 80%		r ⩾ 90%		r ⩾ 95%	
Dim.	Simil.	EUC	HIS	EUC	HIS	EUC	HIS	EUC	HIS
32	CHI	61.6	61.6	68.7	65.1	68.5	65.0	68.2	64.4
	EUC	40.1	41.9	52.3	55.8	52.7	54.4	54.6	56.6
	HIS	50.1	52.3	67.3	65.9	68.2	66.5	68.2	66.0
128	CHI	47.7	50.6	69.7	67.4	70.9	68.0	71.0	68.1
	EUC	40.5	42.2	54.1	58.2	57.7	62.1	59.8	63.3
	HIS	43.0	45.3	68.1	66.3	70.5	67.7	70.7	67.5

Table 5.1: mAP scores at the Washington dataset.

fies the dimensionality of the HOG descriptors. In this experiment, we have tested them using 2×2 and 4×4 spatial bins resulting in descriptors of 32 and 128 dimensions. The similarity column (*Simil.*) indicates which similarity measure has been used to compare the descriptors when creating the agglomerative tree. The abbreviations *CHI*, *EUC* and *HIS* correspond to χ^2, Euclidean and Histogram intersection respectively. The mAP scores are divided into the ratio used to filter weights $w_i/w_{max} \geqslant t$ where $t \in \{0\%, 80\%, 90\%, 95\%\}$. When $r \geqslant 0\%$ we accept all weights while we filter all weights which are smaller than 0.95 of the maximum weight in when $r \geqslant 95\%$. The similarity between the histograms of visual words is calculated using both Euclidean distance and Histogram Intersection similarity measures. The obtained results show that creating a more sparse encoding by filtering out small weights improves the performance of the algorithm. It also shows that using χ^2 or Histogram intersection to compare descriptors consistently gives a better codebook while the Euclidean distance is better when comparing the histograms of visual words. Comparing both descriptors, we see that the higher dimensional descriptor provides a better accuracy. However, the performance

Dim.	32			128		
Sim.	CHI	EUC	HIS	CHI	EUC	HIS
	823	741	788	852	787	849

Table 5.2: Sizes of the evaluated codebooks.

increase is modest so depending on the application a smaller descriptor may be more suitable. Comparing these results with other word spotting methods, we can observe that proposed algorithm outperforms most unsupervised spotting methods [18]. We can also see that the proposed codebook shows a similar performance to a carefully crafted BoVW. For example, we reach a 71.0% mAP score while standard BoVW reaches 72, 35% mAP. However, our BoVW signature is more compact as we use less spatial bins (5 vs. 24 spatial divisions) and the codebook is much smaller. The codebooks generated by our method have between 750 and 850 codewords (see table5.2) while a standard k-means codebook uses 4 096 codewords (i.e. the k-means codebook is between 4,8 and 5,4 larger).

Finally, the codebooks using HOG-32 descriptors need on average 7 minutes to be created while HOG-128 require around 40 minutes on average. The encoding runtime for HOG-128 descriptors takes around 490 ms on average to encode a word snippets. This runtime can be reduced by more than an order of magnitude when the descriptors are approximated by a HKM codebook. In this case, encoding takes around 9.2 ms an average per word snippet. However, approximating the descriptors have the drawback that the mAP score consistently drops a 3-5% in all configurations.

5.5 CONCLUSIONS

In this chapter, we have proposed a method to automatically generate a codebook from synthetic data. The main idea is to create a codebook which is database agnostic, i.e. a codebook which has a good performance independently from the data which is used to create it. This is important when processing large collections of documents as creating a codebook can be extremely time consuming. Thus, our algorithm is able automatically determine the amount and size of the clusters by incorporating semantic information into the codebook generation process. Besides, we have proposed the use of an additional codebook to approximate the descriptors and greatly reduce the descriptor encoding computational cost. The experimental results show that the codebook attains a similar performance to other unsupervised bag-of-visual-words spotting algorithms.

6 COMPACT SYMBOL PROBABILITY SIGNATURES

The dimensionality of the Bag of Visual Words signatures depends on the codebook size and number of spatial pyramid bins. Traditional signatures tend to be memory heavy as they contain from thousands to ten of thousands of elements. This can be alleviated by the use of data compression techniques like Latent Semantic Analysis and Product Quantizer codes. In Chapter 3, we have reduced the number of spatial bins and used both techniques to be able to compress the signature enough so it could be used in a large scale indexing scenario. Although successful, we are trading performance for memory using this approach.

In this chapter, we follow a different approach. Instead of characterizing word snippets with a Bag of Visual Words signature, we are going to create a signature where each bin represents the probability that a script symbol is present in the word image. This results in an extremely compact signature. For example, it only requires 26 dimensions per spatial bin in Latin script documents.

We obtain the symbol probabilities using a classifier that characterizes the image with a Bag of Visual Words signature. We follow a simple sliding window approach, where a window scans the image and at each contiguous location it extract the visual signature and categorizes it with a linear classifier. Since this approach is extremely computationally intensive, we develop a series of refinements that allow us computing the symbols probability maps of the whole document image without a noticeable increase in the computational cost of the spotting algorithm.

6.1 INTRODUCTION

The Bag of Visual Words (BoVW) signatures that we have been using up until now have a quite high number of dimensions. This is due to the fact that the signature dimensionality depends on the codebook size and the number of Spatial Pyramid Matching (SPM) bins. As we have seen from the parameter evaluation of Chapter 2, the best results are obtained by codebooks with thousands of codewords and SPM with tens of spatial bins. Thus, the resulting signature has tens of thousands of dimensions. Although this works with small collections of segmented words, we need to reduce the signature's dimensionality in a large scale scenario. For example, in Chapter 3 we have reduced the number of spatial bins to just 2 and used Latent Semantic Analysis (LSA) and Product Quantizer (PQ) codes to reduce the dimensionality from 65 536 dimensions (i. e. 524 288 bytes in memory) to just 128 bytes per patch. Following this approach however we are trading retrieval accuracy for memory usage. The discrepancy between the original amount of dimensions and the final dimensionality, means that there has to be some precision loss. We want to prevent this by using a signature that is compact by design.

The images that we are indexing are word snippets which are actually composed by a small set of symbols. Specifically, we are working with documents that use an alphabet type script so, we can assume that the

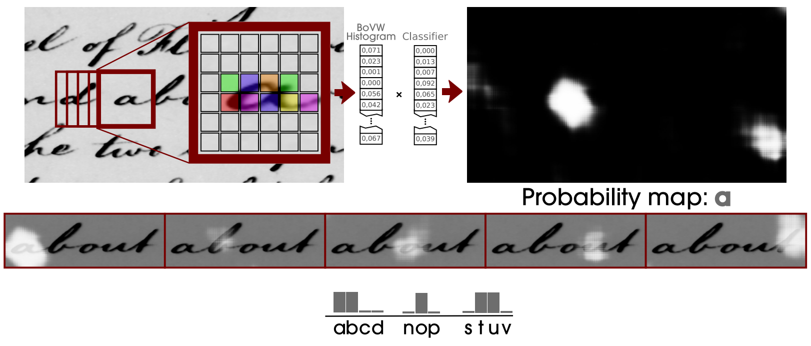


Figure 6.1: Method overview: We extract a BoVW histogram for each pixel of the image and categorize it with a linear classifier. The probabilities of each symbol category are then grouped to form the final descriptor.

script only contains a few dozens of different symbols. Moreover, some of these symbols may have a different shape but share the same meaning (e. g.upper- and lowercase characters). Therefore, a compact signature can obtained by creating a vector where each bin represents the probability that a certain meaningful symbol is present in the word image. This means that in a document written using a Latin script, the signature only requires 26 dimensions per spatial bin.

The method we propose to compute the symbol probability signatures is quite straightforward. First, we extract the codewords from the whole document image. Then, we use a sliding window approach to accumulate them into a local BoVW histogram. The BoVW histograms are next converted into symbol probabilities by means of a linear classifier followed by Platt's algorithm [116]. Finally, the probability maps are pooled together for each word image forming the symbol probability signatures. For example, in Fig. 6.1 we can see the probability of symbol *"a"* and the probabilities for the word *"about"*.

The remainder of the chapter is organized as follows. The descriptor and dictionary used to generate the codewords are shown in Section 6.2. Then, we explore the problem of computing the probability that any pixel image belongs to a certain symbol category using a sliding window approach in Section 6.3. Next, we propose an efficient method to calculate the score of a linear classier of a local BoVW histogram in Section 6.4 and the algorithms needed to train them in Section 6.5. After that, we present the symbol probability signature in Section 6.6. Finally, we display the attained results in Section 6.7 and the conclusions in Section 6.10.

6.2 VISUAL WORD GENERATION

The codewords are generated following the same approach as in Chapter 5. We use agglomerative clustering [108] to group the descriptors and Shannon entropy [112] as the criteria to automatically split the tree into codewords. Like in Chapter 5, we also a Hierarchical k-Means (HKM) [115] to approximate the descriptors and, use it as a lookup table which stores the pre-computed descriptors encodings.

In the previous chapters, we have mainly used the SIFT descriptor [48] or its fast approximation, the Integral Histogram of Oriented Gradients (HOG) descriptor [67]. In our current setting however these two descriptors are too computationally expensive. Instead, we are going to use a modified version

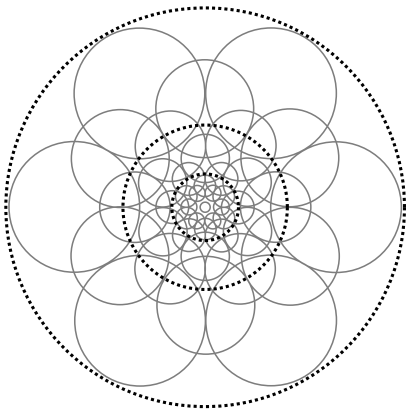


Figure 6.2: Sampling regions of the FREAK descriptor. The pattern follows the distribution of the retinal ganglion cells with their corresponding receptive fields. Each circle represents a receptive field where the image is smoothed with its corresponding Gaussian kernel.

of the lightweight Fast Retina Keypoint (FREAK) descriptor [117]. The FREAK descriptor is a biologically inspired system. It uses the sampling pattern shown in Fig. 6.2 to gather image information imitating the human retina. Each circle represents a receptive field where the image is smoothed by with its corresponding Gaussian kernel. The descriptor is then constructed by thresholding the differences between pairs of receptive fields, i. e.the descriptor is a binary string formed by a sequence of one-bit Difference of Gaussians. In the standard configuration, the FREAK descriptor has 43 receptive fields, so the descriptor has 903 possible differences. The final descriptor is created by the 128 difference pairs (i. e. 16 bytes) that exhibit a higher variance on a image training set.

Instead of binarizing and truncating the descriptor, we prefer to use a linear projection to reduce the FREAK descriptor dimensionality. Although a non-binary descriptor has a higher dimensionality, it also keeps more information about the local regions it describes. The linear projection matrix is a $\mathbf{P} \in \mathbb{R}^{M \times N}$ matrix, where M is the number of difference pairs and N is the final number of projected dimensions. This matrix can be compressed into a $\mathbf{C} \in \mathbb{R}^{D \times N}$ matrix, where D is the number of receptive fields, by applying Algorithm 1 on each column of $\mathbf{P}$. Matrix $\mathbf{C}$ can be used then to

```
input  : $D$, $\mathbf{p}_t$
output: $\mathbf{c}_t$

1 $\mathbf{c}_t \leftarrow \mathbf{0}$;
2 $k \leftarrow 0$;
3 for $i \leftarrow 1$ to $D$ do
4     for $j \leftarrow (i+1)$ to $D$ do
5         $c_{i,t} = c_{i,t} + p_{k,t}$;
6         $c_{j,t} = c_{j,t} - p_{k,t}$;
7         $k \leftarrow k + 1$;
8     end
9 end
```

Algorithm 1: Algorithm to collapse the projection vector so the dot product with the difference pairs can be computed directly over the sampling point values.

compute the projection directly over the receptive fields values as $\mathbf{d} = \mathbf{C}^\top \mathbf{s}$, where $\mathbf{s}$ is the vector containing the receptive fields values. Likewise, the ℓ_2-norm can also be computed without expanding the difference pairs as,

$$\ell_2^2 = (D-1)\sum_{i=1}^{D} s_i^2 - 2\sum_{i=2}^{D} s_{(D-i)} \sum_{j=1}^{i-1} s_{(D-j)}.$$

This greatly reduces the amount of operations needed to compute the descriptor.

We consider two different algorithms to project the difference pairs: the LSA algorithm and the Linear Discriminant Analysis (LDA) algorithm. The LSA algorithm is unsupervised and to obtain a set of topics that properly represent the word snippet information, we train it over the document images that we want to index. The LDA algorithm on the other hand is a supervised algorithm that attempts to increase the separability between classes while reducing the dimensionality. Since we want descriptors that are easy to group into semantic classes by the codebook, we train the LDA algorithm over symbol synthetic images used to train the agglomerative tree (see Chapter 5).

6.3 SLIDING WINDOW CLASSIFIER

Once we have extracted the visual words from the document image, we first pool them locally and then use a classifier to determine which symbol category, if any, the local region belongs to. We compute the symbols probabilities at each pixel of the image so, the computational cost of these two operations will depend on the strategy used to gather the local BoVW signature: process each pixel independently or group them into natural coherent image regions (e. g.superpixels). When processed independently, each local region is characterized by a BoVW histogram that models the pixel and its neighborhood [68, 118]. Therefore, this is an intensive approach where the pooling and classification steps have a relevant impact on the overall computational cost of the algorithm. Alternatively, pixels are grouped into regions that have similar color or texture using an unsupervised segmentation algorithm [119, 120]. These regions can be further combined to obtain image segments which are likely to belong to a certain category [121, 122]. In this strategy, the pooling and classification algorithms are only applied on a few hundreds of regions so their cost is irrelevant to the overall computational cost of the algorithm.

Therefore, following a region-based strategy seems to be more advantageous than a pixel-based when trying to reduce the runtime of the algorithm. However, a region-based strategy requires the use of an unsupervised segmentation algorithm which introduces a significant additional cost. On the other hand, pixel-based strategies can reduce the runtime of both pooling and classification steps to just few milliseconds when implemented carefully [68]. This leads to a solution which is faster than the efficient unsupervised segmentation algorithms [123, 124] typically used in region-based strategies. Therefore in this chapter, we propose an efficient histogram pooling, normalization and classification algorithm that follows a pixel-based strategy.

6.3.1 *Histogram pooling*

In our segmentation approach, the pooling algorithm obtains the BoVW histogram of a pixel by simply accumulating the contributions of the visual words which lie within a square region defined around that pixel. Therefore, we need an efficient method to retrieve all the visual words within an image region since most of the computational cost of the pooling algorithm corresponds to this step.

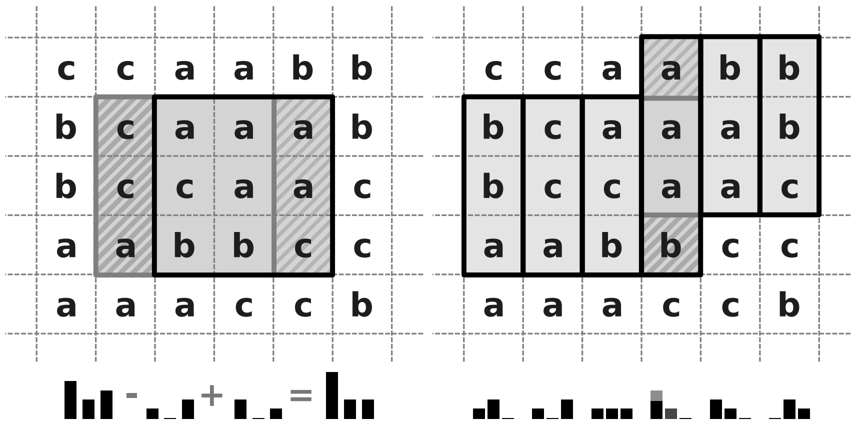
A straightforward solution is to store the visual words into an image structure where each pixel has a list with the visual words at that location, and use an standard pooling algorithm to obtain the BoVW histogram. Then, following a naive implementation which simply look for the visual words within the region at the visual words image, the resulting algorithm has a complexity of $\mathcal{O}\left(r^2wh\right)$, where r is the size of the square region, w is the image width and h is the image height. This cost can be greatly reduced by taking advantage of the overlap between the regions of adjacent pixels.

In the early work presented by Huang et al. in [125], they use a sliding-window approach that updates the histogram as it traverses the image. The algorithm uses the histogram of the previous pixel as base, and it only updates the *elements* which lie outside the intersection between the adjacent regions. This method reduces the pooling complexity to $\mathcal{O}(rwh)$. For examples, the new histogram $\{(a:5),(b:2),(c:2)\}$ in Fig. 6.3a is obtained by subtracting the left column $\{(a:1),(c:2)\}$ and adding the right column $\{(a:2),(c:1)\}$ to the histogram of the previous pixel $\{(a:4),(b:2),(c:3)\}$. This only requires to access to the 6 *elements* of both columns instead the 9 *elements* within the whole region.

Perreault and Hebert proposed in [126] the Distributive pooling algorithm, which Sizintsev et al. extended in [127], to improve the Huang's algorithm. The new algorithm calculates the intersection between windows more efficiently by keeping a partial histogram of a $1 \times r$ window for each column of the image. These column histograms are easily updated each time that the sliding-window moves to the next row by simply subtracting the top *element* and adding the bottom *element* (see Fig. 6.3b). This reduces the complexity of the algorithm to $\mathcal{O}(wh)$ since updating the histograms only requires adding and subtracting the pre-calculated column histograms. However, it also increases the memory usage as it has to maintain an extra array of $d \times w$ elements, where d is the dimensionality of the histogram, to keep the column histograms.

A different approach is to accumulate histogram bins separately. For instance, Porikli proposed the Integral Histogram algorithm [128] which uses an integral image to independently accumulate each bin of the histogram. This algorithm also has complexity of $\mathcal{O}(wh)$ but it is more flexible than the Distributive algorithm. Since it does not exploit adjacency between regions to calculate the histogram, the geometry of the regions can vary and regions can be randomly sampled over the image. However, this flexibility comes at the cost of increasing the memory usage since the algorithm has to maintain an extra array of $w \times h$ elements to keep the integral image.

These methods focus on reducing the cost of obtaining the visual words but ignore the dimensionality of the accumulated histograms. Originally, these algorithms were designed for tasks where the histogram had a relatively small dimensionality, e. g.to calculate the local median value or to calculate Hue histograms in video tracking problems [127]. However, BoVW histograms usually have between thousands to hundreds of thousands dimensions, so the managing such highly dimensional histograms has an important impact on the overall complexity of the pooling algorithm. In order to overcome this problem, Wei and Tao proposed in [129] a variation of the Distributive algorithm [126, 127] specifically tailored for BoVW histograms.



(a) Huang algorithm

(b) Distributive algorithm

Figure 6.3: The Huang algorithm [125] obtains the new histogram in 6.3a by subtracting left column values $\{(a : 1), (c : 2)\}$ and adding the right column values $\{(a : 2), (c : 1)\}$. The distributive algorithm [126, 127] reduces the computational cost by first calculating each column histogram 6.3b and then following as in Huang but using the already calculated histograms.

The authors assumed that the sparsity of the column histograms will be high since the number of visual words within the column window will be much smaller than the dimensionality of the histogram. Therefore, they proposed a new hashing table specifically designed for this problem which has a low editing cost (i. e.the cost of adding and removing elements from the table) while greatly reducing the amount of memory needed to store the column histogram. However, the algorithm still depends on the number of elements stored at each column so its computational cost will increase when complex visual word encodings like first- or second-order encodings [113, 114, 130, 131] are used.

Another way to tackle the problem is to reduce the dimensionality of the histogram. For instance, Fulkerson et al. [68] use the Agglomerative Information Bottleneck algorithm [132] to reduce the number of visual words of the codebook and then use the Integral Histogram algorithm [128] to accumulate the BoVW histograms. Although this methods solves the dimensionality problem, it also reduces the discriminative power of the BoVW histograms.

6.3.2 *Normalization and classification*

After accumulating the BoVW histograms, we have to normalize them to account that different amounts of visual words have contributed to the histograms. For example, the length of the word images determines the amount of visual words available to represent them. So, word snippets containing the same element but different length will never match if not normalized. The BoVW histograms are typically normalized using the Euclidean or the Manhattan norms.

The histogram can be further normalized to account that visual words are not independent and identically distributed on the image by means of the power normalization $g(x) = sign(x)\|x\|^{\alpha}$, where $0 < \alpha < 1$, to each bin of the histogram (see Chapter 2.2.4.2).

Once the BoVW histograms have been normalized, we use a classifier to categorize them, followed by a normalization algorithm that converts the classification scores into probabilities [116, 133, 134, 135]. The classifier must have a low computational cost because we have to we need to categorize as many histograms as pixels the image has. Moreover, the learning algorithm also needs to be efficient since in a pixel-based strategy even a fairly small set of annotated images result in a training set of millions of samples. To deal with this amounts of information, the learning algorithm has to be optimized to greatly reduce the memory usage or we have to use a feature selection algorithm to select a reduced subset of *good* training samples. Therefore, linear classifiers appear as a good choice since they are both efficient in the prediction as well as in the training phase [136, 137, 138]. Alternatively, we can use a non-linear classifier with an additive kernel, e.g. the χ^2 or the intersection kernel, which obtain a better classification accuracy [139, 70] and can be linearized by explicitly mapping the BoVW histograms to the kernel feature space [110, 140, 111].

6.4 INTEGRAL LINEAR CLASSIFIER

The main problem we are facing in our symbol classification approach is the high dimensionality of the BoVW histograms. Therefore, it would be advisable to have a method which is able to directly obtain the classification score avoiding to explicitly accumulate the histograms. In this section, we are review our Integral Linear Classifier (ILC) algorithm [141, 142] which fulfills this condition. The algorithm has a $\mathcal{O}(wh)$ computational complexity which is independent of the dimensionality of the BoVW histograms and the size of the regions.

6.4.1 *Algorithm overview*

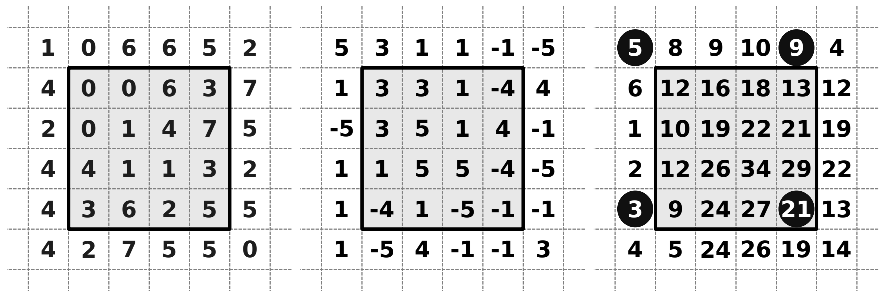
The core idea in our approach is to first calculate the individual contribution of each visual word to the classification score and then aggregate these partial contributions to obtain the scores of each pixel. This is easily accomplished with a linear classifier since the classification score is calculated as the linear combination between the BoVW histogram and the classifier weight vector. Therefore, we first create a *weights image* by replacing the visual words by the product between their weights and their associated classifier weight. Then, the classification score of any pixel is calculated as the sum of the values of the *weights image* within the pixel's region. Following this simple procedure, we avoid accumulating the BoVW histogram altogether.

The algorithm can be further optimized by using *integral images* [143, 144], which is a data structure designed to calculate the sum of an image area in constant time. An *integral image* is a lookup table that at the coordinates $\langle x, y \rangle$ contains the sum of all the values above and to the left of that coordinate,

$$I_{x,y} = \sum_{i \leqslant x; j \leqslant y} i_{i,j},$$

where $I_{m,n}$ and $i_{m,n}$ are respectively the values of the *integral image* and the original image at the coordinates $\langle m, n \rangle$. The values of the *integral image* can be efficiently calculated with a single pass over the original image as,

$$I_{i,j} = i_{i,j} + I_{i-1,j} + I_{i,j-1} - I_{i-1,j-1} \, .$$



(a) Visual words image (b) Weights image (c) Integral image

Figure 6.4: The *weights image* b) is created by replacing visual words from image a) by their corresponding classifier weights. In this example, the classifier values are $\mathbf{c} = [3,5,-5,-4,1,-1,1,4]$. Then, we can create the *integral image* c) from the *weights image* b), so that the classification score of the window can be calculated as $s = 21+5-3-9 = 14$.

Then, given the region $R = [x_0, y_0, x_1, y_1]$ where $\langle x_0, y_0 \rangle$ and $\langle x_1, y_1 \rangle$ are the top-left and bottom-right corner coordinates of the region, the sum of the values within the region is calculated as

$$I(R) = I_{x_0-1,y_0-1} + I_{x_1,y_1} - I_{x_1,y_0-1} - I_{x_0-1,y_1} \, .$$

By following this procedure, we avoid using a sliding-window technique to obtain the symbol classification scores and use instead simple pixel-based image operations. The main advantage of this procedure is that the complexity of the resulting algorithm is independent of the region size and the dimensionality of the BoVW histograms. Moreover, it reduces the pooling and classification steps to just a product for each visual word to create the *weight image*, 4 additions per pixel to create the *integral image* and 4 additions more to calculate the classification score of each region. This computational cost is generally lower than the cost of calculating the dot product between the classifier and the BoVW histograms. For example, given a $w \times h$ image where we have extracted N visual words, the algorithm requires N products and $w \times h \times 8$ additions to calculate the classification score of the whole image. Whereas, to calculate the dot product between the linear classifier and the BoVW histogram of each pixel, we require $w \times h \times s$ products and $w \times h \times (s-1)$ additions, where s is the average number of non-zero bins in the BoVW histogram. Therefore, unless the BoVW histograms are supported on average by an extremely low number of visual words, i. e. with $s \leqslant 8$ on average, the computational cost of just calculating the dot product is going to be higher than the cost of the whole proposed algorithm.

Fig. 6.4 illustrates the steps followed by the proposed method when categorizing the highlighted area with the linear classifier $\mathbf{c} = [3,5,-5,-4,1,-1,1,4]$. First, the visual words are used to obtain the *weights image* of Fig. 6.4b. In this case, the visual words from Fig. 6.4a results from encoding descriptors using a single codeword with hard-assignment. Consequently, the *weight image* is obtained by simply replacing the visual words by their associated weight in the linear classifier. Then, the *weights image* is used to create the *integral image* Fig. 6.4c. Finally, the classification score of the region is calculated as $\text{score} = 21+5-3-9 = 14$.

Table 6.1: Example of the result of applying the two strategies used to deal with the negative weights. In this example, the methods are applied on the following list of ***(weight, identifier)*** tuples representing the visual words: $[(-3.12, 0), (1.27, 1), (-0.59, 1), (0.79, 2)]$

Method	Visual words	Histogram
Absolute	$[(3.12, 0), (1.27, 1), (0.59, 1), (0.79, 2)]$	$\mathbf{h}_{abs} = [3.12, 1.86, 0.79]$
Separate	$[(3.12, 1), (1.27, 2), (0.59, 3), (0.79, 4)]$	$\mathbf{h}_{sep} = [0, 3.12, 1.27, 0.59, 0.79, 0]$

6.4.2 *Classification score*

The classification score is calculated as the dot product between the classifier weight vector and the normalized BoVW histogram. However, the method described previously does not take into account the normalization of histogram. Thus, the actual classification score is calculated as,

$$\texttt{score} = \langle \mathbf{h}, \mathbf{c} \rangle = \sum_{i=0}^{D} h_i c_i = \sum_{i=0}^{D} \frac{\hat{h}_i}{|\mathbf{h}|} c_i = \frac{1}{|\mathbf{h}|} \sum_{i=0}^{D} \hat{h}_i c_i = \frac{\langle \hat{\mathbf{h}}, \mathbf{c} \rangle}{|\mathbf{h}|}$$

where $\hat{\mathbf{h}}$ is the unnormalized histogram and $|\mathbf{h}|$ is the norm of $\hat{\mathbf{h}}$. We also need an efficient method to calculate the norm $|\mathbf{h}|$. This method should be similar to the method used to calculate the dot product, so the classification score of a region R can be calculated as $\texttt{score}(R) = I_{dot}(R)/I_{norm}(R)$. This requires the norm operator to be linear, thus our only choice is the L_1-norm. Although L_p-norms are nonlinear, L_1-norm can be linearized when the bins of the BoVW histograms are all positive since $|h_i| = h_i$ so that $|\hat{\mathbf{h}}|_1 = \sum_{i=0}^{D} h_i$.

However, we cannot ensure that the bins of the BoVW histogram are always going to be positive, since visual words may have negative weights depending on the method used to represent them. Methods like hard-assignment which give a unit weight to all visual words or soft-encoding methods which weights the visual words according to a Gaussian function [145], generate histograms with only positive bins. On the other hand, methods like Locality-constrained Linear Coding (LLC) [69] or first-order encoding [114] may give negative weights to some visual words. Therefore, we use two simple strategies to deal with these negative contributions: accumulate the absolute value of the weights and differentiate between positive and negative contributions. The second strategy doubles the number of bins of the BoVW histogram since a visual word with a positive weight contributes to a different bin than the same visual word with a negative weight. A simple example of the modifications done by the two strategies to the visual words is shown in Table 6.1. Alternatively, we can also set $|\hat{\mathbf{h}}|$ as the number of valid visual words within the region, so that each valid visual word will simply contribute 1 to the norm. These strategies give to each visual word a *norm weight* which corresponds to the partial contribution of the visual word to the BoVW histogram. Then, the norm of the BoVW histogram of each image pixel is calculated similar to the dot product but creating the weight image by simply replacing the visual words by their associated *norm weight*.

Finally, the classification runtime is further reduced by quantizing the weights of visual words and the linear classifier, so that the *weight images* and their corresponding *integral images* are calculated using an integer data type. This reduces the runtime of the algorithm while keeping a similar classification accuracy. In our experiments, the quantization step is only applied while testing the symbol classification algorithm.

6.4.3 *Advantages and limitations*

The main advantage of the ILC algorithm is its reduced computational cost. The complexity of the algorithm only depends on the number of visual words and the size of the image, and it reduces the runtime by at least an order of magnitude when compared to other methods. However, not accumulating BoVW histogram also reduces the number of methods which can be applied to improve its discriminative power. For instance, our algorithm is only able to use the L_1-norm while other normalization techniques must be discarded, e.g. max-pooling [69] or the Log-Euclidean Tangent Space Mapping [122].

A normalization step that we have been consistently using is the power normalization step [72] (see Chapter 2.2.4.2). However, in this formulation it cannot be used as it cannot be linearized. This normalization has always increased the retrieval performance of the spotting systems that we have presented so far. However, we believe that in our current formulation the positive effects would be more limited. The ILC classifiers are applied over a local region of the image so the visual words accumulated are going to mostly belong to the symbol of interest and maybe its neighbors. So the effects of having a large amount of background visual words (i. e.visual words that are not from the same symbol category as the classifier) that *confuse* the classifier is going lower than in the previous word spotting settings. The difference of size between symbols is however something that power normalization would help to lessen.

As we have seen in Chapter 2.2.4.1, another successful enhancement to the BoVW representation is the addition of spatial information. The methods used to add spatial information [71, 146, 147, 148] typically divide the region into a set of overlapping subregion, accumulate the BoVW histogram for each subregion independently and then concatenate the each subregion histogram to obtain the final representation. In our case, the ILC algorithm can be easily adapted to use this methods by calculating the partial dot-product and norm for each subregion, i. e.calculating a different *weight* and *norm* image for each subregion, and then merging the results to obtain the final score. Although spatial information has consistently improved the retrieval performance of the spotting systems up until now, results on semantic segmentation problems which are more related to the current symbol classification problem that word spotting tasks show that the classification enhancement provided by SPM are quited limited [142]. Since this step just provides little classification gain while increase the computational complexity of the algorithm, we decided to not use it in our current formulation.

6.5 LEARNING ALGORITHMS

In the previous section, we focused our attention on the *testing* phase of the classifier, i. e.using the classifier to obtain the classification scores of each local region of the image. Now, we are going to focus on the *training* phase of the classifier where the main problem is the large amount of training samples available. In our symbol classification approach each pixel of the image is represented by a BoVW histogram, thus even a fairly small set of annotated images generates millions of training samples. We can tackle this problem by selecting a *good* subset of samples to train the classifier.

However, this approach introduces an additional step which has a direct impact on the performance of the classifier, as the obtained classification accuracy will depend on how well the selected samples represent each category. Alternatively, we can calculate the BoVW histograms of the samples when needed, or have them pre-calculated and stored on disk, but this approaches have a significant impact on the algorithm's runtime.

A different approach is to adapt the learning algorithm to the current problem and exploit the properties of the BoVW to reduce its computational complexity. Using an approach similar to the ILC algorithm, we adapt some learning algorithms to update the classifier model directly from the visual word without explicitly accumulating the BoVW histograms. Consequently, the resulting algorithm greatly reduces the amount of memory needed to store the training information. Additionally, the adapted algorithms have a smaller runtime as the number of operations is reduced by exploiting the redundancies between neighboring BoVW histograms. Although the adapted algorithm modify the way that training information is processed, both the original and adapted versions of an algorithm are equivalent and both obtain exactly the same classifier given the same set of training samples. The only constrains of the adapted algorithms is that all the samples of an image must be processed together. Therefore, the learning algorithm has to be able to process batches of samples to update the classifier model, and algorithms which update the model using a training sample at a time [149, 150] cannot be used.

We have adapted two batch and three online learning algorithms. The batch algorithms are the logistic- and l_2-regularized variants of the trust region newton solvers [151, 138], while the online algorithms are the Pegasos [136], the First-Order Stochastic Gradient Descent (SGD) and Quasi-Newton SGD algorithms [137, 152]. Since all five algorithms are modified in a similar way, we only present the First-Order SGD algorithm. This algorithm has a lower complexity than the other methods, allowing us to better illustrate the proposed modifications.

6.5.1 *Mini-batch First Order Stochastic Gradient Descent*

The First-Order SGD algorithm was proposed together with the Quasi-Newton SGD algorithm in [137, 152] as online linear classifiers specifically designed for large-scale problems, aiming to obtain an algorithm with a low computational cost and fast convergence rates. Both algorithms are binary (i.e. samples have the form $z = (\mathbf{x}, y) \in \mathbb{R}^d \times \{-1, +1\}$) and minimize the following primal cost function,

$$\min_{\mathbf{w}} f(\mathbf{w}) \equiv \frac{\lambda}{2}\mathbf{w}^\top \mathbf{w} + \frac{1}{n}\sum_{i=1}^{n} \ell(y_i \mathbf{w}^\top \mathbf{x}_i)$$

where $\mathbf{w}$ is the classifier weight vector, the parameter $\lambda > 0$ controls the strength of the regularization term and $\ell(s)$ is the loss function which is convex and twice differentiable with continuous derivatives. At each iteration, the algorithms randomly draws a training example $(\mathbf{x}_t, y_t)$ and updates the classifier weight vector $\mathbf{w}$ by

$$w_{t+1} = w_t - \underbrace{\frac{1}{t+t_0}\mathbf{B}\lambda\mathbf{w}_t}_{\text{regularization update}} + \underbrace{\frac{1}{t+t_0}\mathbf{B}\ell'(y_t\mathbf{w_t}^\top\mathbf{x}_i)y_t\mathbf{x}_t}_{\text{pattern update}} \qquad (5)$$

where the matrix $\mathbf{B}$ is positive semidefinite and t_0 is an heuristically chosen positive constant used to ensure that the updates at the first steps does not have an implausibly large norm. The algorithms apply the *pattern* update at each iteration but the *regularization* update is only applied every few iterations. This reduces the computational cost of the algorithm while it keeps a similar classification accuracy.

```
input  : $\lambda$, $t_0$, skip, $S$, $k$, $T$
output: $\mathbf{w}$

1 $\mathbf{w} = \mathbf{0}$;
2 for $t \leftarrow 0$ to $T$ do
3     Choose $A_t \subseteq S$, where $|A_t| = k$, uniformly at random;
4     $\Delta\mathbf{w} = \sum_{i \in A_t} \ell'(y_i \mathbf{w}_t^\top \mathbf{x}_i) y_i \mathbf{x}_i$;
5     $\mathbf{w}_{t+1} = \mathbf{w}_t - \frac{1}{k\lambda(t+t_0)} \Delta\mathbf{w}$;
6     if $(t+1) \bmod \mathrm{skip} = 0$ then
7         $\mathbf{w}_{t+1} = \left(1 - \frac{\mathrm{skip}}{(t+t_0)}\right) \mathbf{w}_{t+1}$;
8     end
9 end
```

Algorithm 2: Mini-batch version of the First-Orders Stochastic Gradient Descent algorithm.

The First Order SGD algorithm is obtained by setting $\mathbf{B} = \lambda^{-1}\mathbf{I}$ in Eq. 5. The pseudo-code in Algorithm 2 show mini-batch version of the algorithm where the parameter k selects the number of sample used to calculate the *pattern* update at each iteration. When $k = 1$, we have the original stochastic sub-gradient algorithm form [137], while when $k = n$, where n is the total number of training samples, we obtain a deterministic sub-gradient descent method [153]. The parameter skip is an heuristically chosen positive constant used to selected the frequency of the *regularization* updates.

6.5.2 *FOSGD Semantic Segmentation Learner*

In order to adapt this algorithm to the symbol classification problem, we only have to modify the two steps where training samples are used: the random selection step at line 3 and the weight vector update step at line 4. In the random selection step, we are going to work at image level instead of sample level. Therefore k is the number of randomly selected images, n is the number of training images and A_t is the subset of images used to update the classifier. Consequently the weight vector update step at line 4 is rewritten as

$$\Delta\mathbf{w} = \sum_{i \in A_t} \sum_{j \in S_i} \ell'(y_j \mathbf{w}_t^\top \mathbf{x}_j) y_j \mathbf{x}_j, \qquad (6)$$

where S_i are the samples of the i-th training image. Now, we are going to reformulate this equation to obtain which is the update equation for each dimension of the vector. The samples $\mathbf{x}$ in our problem are BoVW histograms so the k-th bin of the j-th sample is

$$\mathbf{x}_{jk} = |\mathbf{x_j}|^{-1} \sum_{m \in \mathbf{x}_{jk}} \nu_m,$$

where $|\mathbf{x_j}|$ is the norm of the sample and the ν_m's are the weights of the visual words accumulated at the bin $\mathbf{x}_{jk}$. Then, the k-th bin of the update vector $\Delta\mathbf{w}$ is calculated as

$$\begin{aligned}\Delta w_k &= \sum_{i\in A_t}\sum_{j\in S_i} \ell'(y_j\mathbf{w}_t^\top\mathbf{x}_j)y_j x_{jk}\\ &= \sum_{i\in A_t}\sum_{j\in S_i} \ell'(y_j\mathbf{w}_t^\top\mathbf{x}_j)y_j \frac{1}{|\mathbf{x_j}|}\sum_{m\in\mathbf{x}_{jk}} \nu_m\\ &= \sum_{i\in A_t}\sum_{j\in S_i}\sum_{m\in\mathbf{x}_{jk}} \frac{\ell'(y_j\mathbf{w}_t^\top\mathbf{x}_j)y_j}{|\mathbf{x}_j|}\nu_m.\end{aligned}$$

Finally, grouping the terms which have the same visual word weight ν_m, we can rearrange the previous equation into

$$\Delta w_k = \sum_{i\in A_t}\sum_{m\in V_i}\nu_m \sum_{j\in R(\nu_m)} \frac{\ell'(y_j\mathbf{w}_t^\top\mathbf{x}_j)y_j}{|\mathbf{x}_j|} \qquad (7)$$

where V_i are the visual words extracted from the i-th image and $R(\nu_m)$ are the samples which have the visual word ν_m accumulated in their BoVW histogram. The last sum of Eq. 7 can be calculated using pixel-wise image operators so we can calculate the update vector $\Delta\mathbf{w}$ directly from the visual words information without having to explicitly accumulate the BoVW histograms.

```
input : $\lambda$, $t_0$, skip, S, k, T
output: $\mathbf{w}$
1  $\mathbf{w} = \mathbf{0}$;
2  for $t \leftarrow 0$ to $T$ do
3      Choose $A_t \subseteq S$, where $|A_t| = k$, uniformly at random;
4      $\Delta\mathbf{w} = \mathbf{0}$;
5      for $i \in A_t$ do
6          $\kappa_i, \rho_i = \mathrm{ILC}(i)$;
7          $\psi_i = \ell'(\mathbf{y}_i \odot \kappa_i) \odot \mathbf{y}_i \div \rho_i$;
8          $\Psi_i = \mathrm{IntegralImage}(\psi_i)$;
9          for $\nu_m \in i$ do
10             $\Delta\mathbf{w}_{\mathrm{idx}(\nu_m)} = \Delta\mathbf{w}_{\mathrm{idx}(\nu_m)} + \nu_m\Psi_i(R(\nu_m))$;
11         end
12     end
13     $\mathbf{w}_{t+1} = \mathbf{w}_t - \frac{1}{k\lambda(t+t_0)}\Delta\mathbf{w}$;
14     if $(t+1) \bmod \mathrm{skip} = 0$ then
15         $\mathbf{w}_{t+1} = \left(1 - \frac{\mathrm{skip}}{(t+t_0)}\right)\mathbf{w}_{t+1}$;
16     end
17 end
```

Algorithm 3: Mini-batch version of the First-Orders Stochastic Gradient Descent algorithm adapted to the semantic segmentation problem.

The modification necessary to adapt the algorithm to our semantic segmentation problem are shown in Algorithm 3. The main differences with the original algorithm are the steps added to replace Eq. 6 with Eq. 7 when calculating the update vector $\Delta\mathbf{w}$.

For each selected image i in A_t, we first use the ILC algorithm to obtain the images κ_i and ρ_i with the prediction scores and histogram norms of each

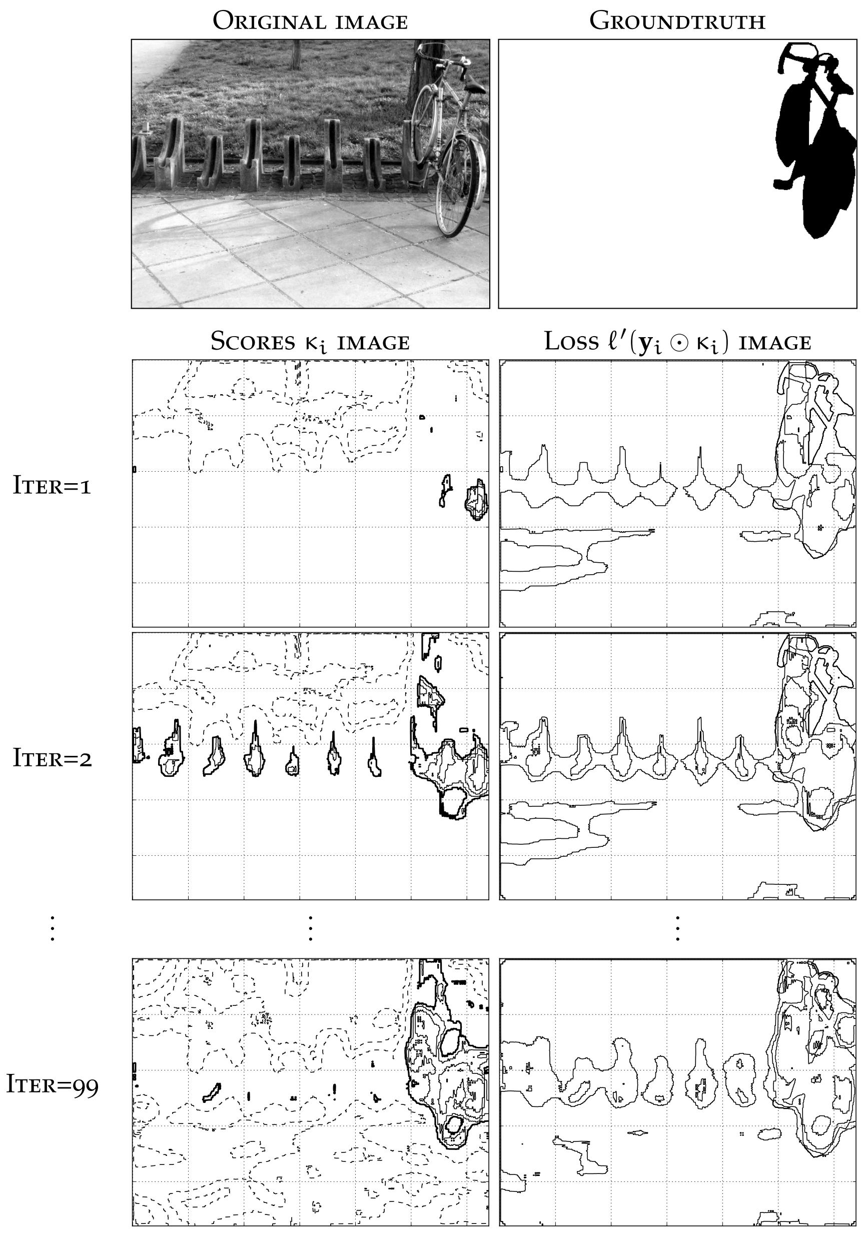


Figure 6.5: Evolution of the classifier score and loss estimation images at each iteration of the First-Order SGD algorithm.

pixel, respectively. Then, we create the image ψ_i with the contributions of each sample to the update vector $\Delta\mathbf{w}$, i.e. the term in the last sum of Eq. 7. The operations to calculate ψ_i at line 7 are pixel-wise, so the value at pixel (u, v) is calculated as

$$\psi_i(u,v) = \ell'(y_i(u,v)\kappa_i(u,v))y_i(u,v)/\rho_i(u,v).$$

Finally, we update the $\Delta\mathbf{w}$ vector with the contributions of the visual words extracted from the image. Each visual word contributes its weight ν_m multiplied by the sum of the ψ_i within the region of the pixels that have accumulated the visual word into their histogram. Since a BoVW histogram of a pixel accumulates all the visual words which lie within a square region defined around it, a visual word contributes to all the BoVW histograms of the pixels which lie within the same square region defined around the visual word.

Therefore, the sum of the values within the region $R(w_m)$ can be efficiently calculated creating the integral image Ψ_i from the weights image ψ_i. Then, the bin $\text{idx}(v_m)$ of the $\Delta\mathbf{w}$ vector is updated simply as $v_m \Psi_i(R(v_m))$.

The adapted algorithm is able to calculate the new weight vector $\mathbf{w}_{t+1}$ only using pixel-wise operations and the visual words. For instance, Fig. 6.5 shows the evolution at each iteration of some of these image *operations*. The images on the left column show the scores images κ_i while the right column images correspond to the derivative of the loss function resulting of applying $\ell'(y_i(u,v)\kappa_i(u,v))$ to each pixel of κ_i. These operations does not require the use of BoVW histograms, so the algorithm only needs the visual words and the image annotations to training the classifier. Therefore, the memory needed by the adapted algorithms is much lower than its original counterpart since storing the visual words require far less memory than storing the BoVW histograms.

6.6 SYMBOL PROBABILITY SIGNATURE

In the previous sections, we have defined how to generate a classification score map for the whole document image. Now, we are going to briefly review how do we build the symbol probability signature. First, we have to convert those scores into probabilities by means of a regression [116, 133, 134, 135]. We have experimented several regression functions and the best results where obtained by fitting a sigmoid function with Platt's method [116] and by using isotonic regression with Niculescu-Mizil and Caruana's method [134]. In order to reduce the computational cost of computing the probabilities, we approximate the regression functions with a lookup table.

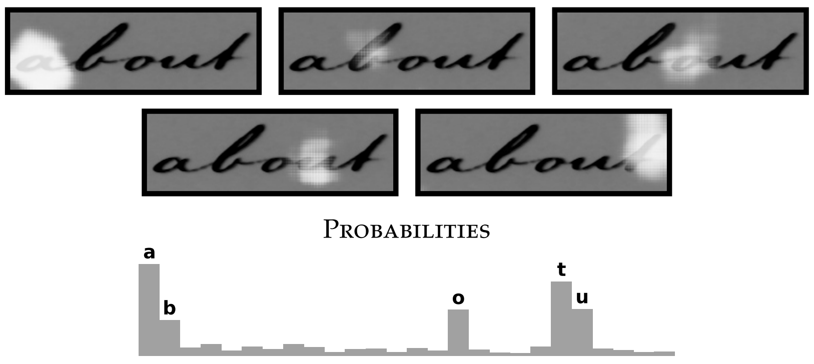


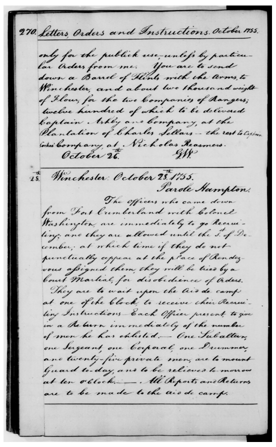
Figure 6.6: Probabilities of the classifiers with the strongest response overlapping the word image

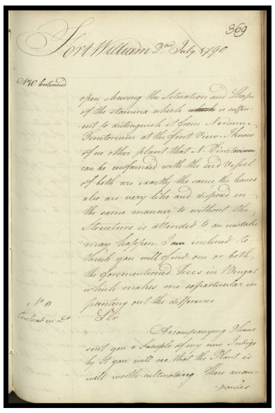
Word images are then characterized by accumulating the symbol probabilities into a histogram. For example, we can see in Fig. 6.6 the probabilities of the classifiers for the symbols *"a"*, *"b"*, *"o"*, *"t"* and *"u"* overlapping the word image and the histogram associated to them. In order to increase the discriminative power of the signature, we add spatial information using SPM (see Chapter 2.2.4.1). Finally, we follow a normalization similar to the BoVW signatures (see Chapter 2.2.4.2): normalize the descriptor with Euclidean or Manhattan norm, apply power factor normalization, and re-normalize with the norm. The use of power normalization is also quite important on this signature, not only because all symbols are detected with the same strength (e. g.*"a"* and *"t"* get a stronger response than the other symbols in Fig. 6.6), but also because the size of the probability *blob* of each image symbol depends on the size of the symbol and symbols display substantials size differences (e. g.uppercase vs lowercase characters).

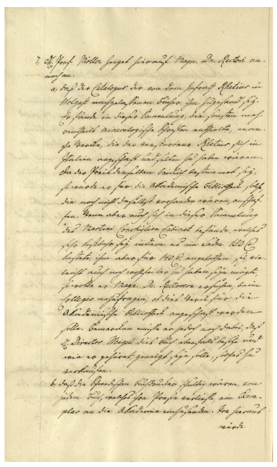
The resulting symbol probability signature is a compact vectorial signature. On documents written using Latin script, it only requires 26 bins per spatial bin, so a typical signature may have a dimensionality of few hundred bins. Since the signature is vectorial, we can apply the same dimensionality reduction techniques that we have been using on the BoVW signature, LSA projections and specially PQ codes, to further reduce the dimensionality so the signature ends up only using a few bytes of memory.

6.7 EXPERIMENTAL RESULTS

In this section, we want to experimentally to determine how dependent is the probability symbol signature on the quality of the ILC symbol classifier, what is the retrieval performance of the signature and how this results vary on different datasets. Let us first introduce the datasets used to test the symbol probability signature. Then, we are going to show the symbol classification accuracy on the George Washington dataset and compare the retrieval performance of the signature using different parameters. Finally, we report the results of applying the same ILC classifiers on two different datasets and determine how adaptable the algorithm is.







GEORGE WASHINGTON BOTANY KONZILSPROTOKOLLE

Figure 6.7: Example pages of the George Washington, Botany and Konzilsprotokolle datasets.

6.7.1 *Datasets*

Like in the previous chapters, we primarily test the different parameters of the signature on the George Washington dataset. Additionally, we test the retrieval capabilities of best signature obtained in George Washington on the Botany and Konzilsprotokolle collections. These two datasets were introduced in the 2016 Handwritten Keyword Spotting Competition [154]. The test set of the Botany dataset contains 3 230 word images with 150 query word images while, Konzilsprotokolle contains 3 533 word images with 200 query word images. Examples of the document images of the three tested databases can be seen in Fig. 6.7.

6.7.2 *System parameters*

The ILC classifiers have been trained using true type fonts that *resemble* the writing of the George Washington dataset. In Fig. 6.8, we can see an example of an original text line image from the dataset, the same text line generated by a TrueType (TTF) font and its associated symbol level annotation. The local regions used to extract the descriptors are 40 pixels wide square regions. The HOG descriptors have only two spatial partitions and eight orientation bins, resulting in a 32 dimensional descriptor. The FREAK descriptor projects the 903 dimensional descriptor into 32 dimensions both for LSA and LDA variants, so it has the same dimensionality as the HOG descriptors. For the LDA projection, we first perform an initial projection using LSA and then compute the LDA projection. We do it this way to avoid numerical instabilities of the LDA algorithm [155]. The agglomerative codebook is trained using the intersection distance and it generates a codebook slightly above 4 000 codewords. Finally, the local regions categorized by the ILC classifier are 80 pixels wide square regions.

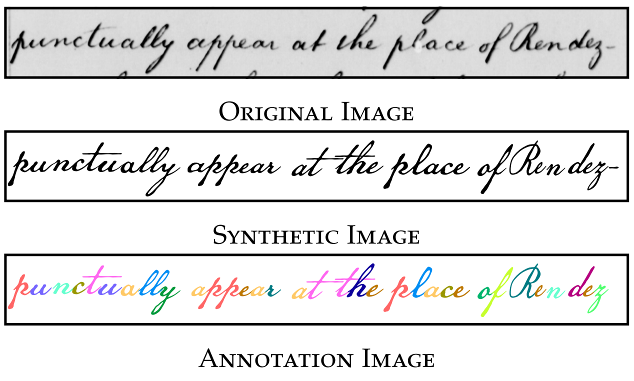

punctually appear at the place of Rendez-
Original Image
punctually appear at the place of Rendez-
Synthetic Image
punctually appear at the place of Rendez
Annotation Image

Figure 6.8: Example of original text line from the George Washington dataset, the same text line created with a TTF font and its associated annotation.

6.7.3 *Symbol classification*

First we want to measure how good is the ILC classifier to determine the actual probability of the symbols on the image. Therefore, we have manually annotated the symbol location in the George Washington dataset so we are able to compute the classifier accuracy for the ILC classifiers of each symbol category.

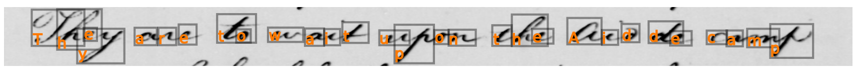


Figure 6.9: Example of the manual annotation on the George Washington dataset.

The classifier has been trained on a synthetic dataset consisting of 50 training and 10 validation pages with random text generated with multiple TTF fonts. First, we have tested the five solvers considered in Section 6.5 on a small subset of train images to determine which solver has the best

Table 6.2: Classification results of the ILC classifier for each symbol category. The AGM column corresponds to the geometric mean between the positive and negative accuracies.

Symbol	Frequency	Positive	Negative	AGM
a	1510	72,56%	99,05%	84,78%
b	301	56,85%	99,71%	75,29%
c	551	50,09%	99,79%	70,70%
d	728	85,29%	99,46%	92,10%
e	2708	51,81%	98,31%	71,37%
f	397	78,27%	99,61%	88,30%
g	363	67,38%	99,44%	81,85%
h	1069	81,32%	98,30%	89,41%
i	1279	37,46%	99,10%	60,93%
j	17	52,06%	99,33%	71,91%
k	78	88,24%	99,74%	93,81%
l	667	84,35%	99,98%	91,83%
m	581	75,39%	99,46%	86,59%
n	1361	62,87%	98,62%	78,74%
o	1660	57,54%	98,94%	75,45%
p	394	89,72%	99,70%	94,58%
q	16	87,62%	99,89%	93,55%
r	1544	41,92%	99,18%	64,48%
s	1066	65,11%	99,50%	80,49%
t	1943	73,77%	98,24%	85,13%
u	646	48,72%	99,42%	69,59%
v	215	51,85%	99,91%	71,97%
w	319	48,11%	99,89%	69,33%
x	45	89,92%	99,78%	94,72%
y	450	68,61%	99,42%	82,59%
z	13	89,30%	99,93%	94,47%
	19921	67,54%	99,37%	81,93%

properties. All of them showed a similar classification accuracy but the logistic-regularized trust region newton solver [151, 138] showed a faster convergence rate. This is the solver we used to train the ILC symbol classifiers. We have selected the regularization factor of the classifier from a range between 10^7 to 10^{14} using cross-validation. The runtime of the solver is about a 70 minutes per symbol category on a desktop computer using 4 cores.

The classification results obtained by the ILC classifier using the FREAK descriptor with LDA projection are shown in Table 6.2. As we can see the classifiers have a mixed performance depending on the symbol category. The negative accuracy is always very high but the positive accuracy ranges between 50% and 90%. Although these results may look disheartening, we need to remember that, as long as the classification errors are consistent, their effect on the spotting task are going to be limited.

6.8 spotting results on the george washington dataset

We have evaluate the performance of the symbol probability signature combining the Euclidean and Manhattan distances and norms. The best results are obtained when using the Euclidean distance with the Manhattan normalization. We have also evaluated its performance when using different number of horizontal spatial partitions and power factor ratio. In Fig. 6.10 we can see the evolution of both parameters.

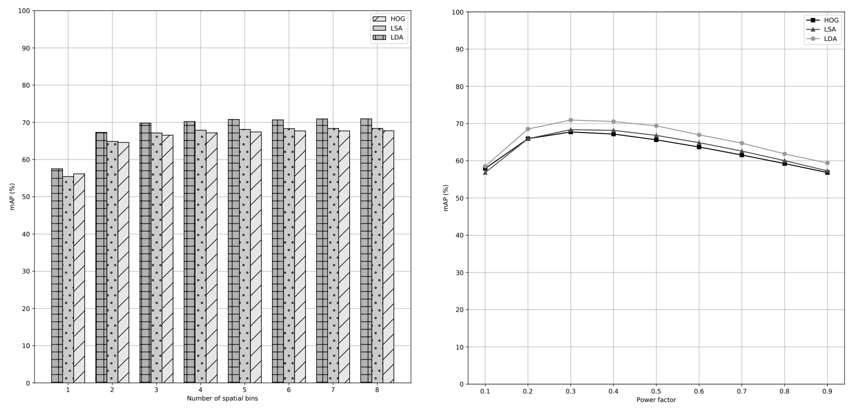


Figure 6.10: Evolution of the best mAP score attained for different number of spatial bins and different normalization power factor.

The usage of spatial bins increases the retrieval performance of the system, like in standard BoVW signatures. The mean Average Precision (mAP) score grows fast up to three spatial bins and after that it still keeps growing but very slowly. The best power factor is 0.3 for all three descriptors. We believe that this value is so low because the size difference between symbols also affects the size of their corresponding probability blobs. A low power factor value reduces the difference between probabilities that are far a part. Therefore, this helps to reduce the effects of symbol size differences.

Table 6.3: mAP score obtained by the best configuration for each descriptor.

Descriptor	Compression	mAP (%)
Simulated		80.01%
HOG		67.75%
LSA		68.36%
LDA		70.95%
LDA	$p = 8$	70.11%
LDA	$p = 4$	68.83%
LDA	$p = 2$	67.36%

Finally, we report in Table 6.3 the mAP scores obtained with different descriptors and compression factors. The *simulated* descriptor corresponds to using the manual annotation to directly generate the probability maps. We wanted to know the mAP score when the best possible symbol probability maps are used. As we can see, the signature has an upper limit around 80% mAP. Comparing the three descriptors, we can see that FREAK descriptors performs slightly better than the HOG descriptor. We can also notice that the LDA projection provides a slight advantage over the LSA. We think that is because LDA helps the semantic codebook find better clusters. The result of the best configuration using the LDA-FREAK descriptor is only 9 points bellow the result obtained with the groundtruth. This demonstrates that although the symbol classification accuracies shown in Table 6.2 where low, the resulting signature still performs well.

The last three results report the mAP score when the signature is compressed using PQ codes. Since the signatures in Table 6.3 use 8 spatial bins, they have 288 dimensions. The PQ codes use 256 quantizers, so each partition needs to be represented by a byte, and divide the signature in 8, 4 and 2 partitions resulting in compressed signatures that require 8, 4 and 2 bytes of memory to represent a single word image.

6.9 RESULTS ON BOTANY AND KONZILSPROTOKOLLE

We compare the mAP attained by our signature with some of the last methods to report on the Botany and Konzilsprotokolle datasets in Table 6.4.

Table 6.4: mAP precision scores obtained by different methods on the Botany and Konzilsprotokolle datasets

Method	Botany	Konzils.	*Notes*
Ours	45.72%	57.13%	
Almazan et al. [92]	75.77%	77.91%	
Sudholt & Fink [100]	89.69%	96.05%	
Wilkinson & Brun [156]	54.95%	82.15%	
Sudholt & Fink [102]	91.23%	97.70%	
Sudholt & Fink [157]	96.05%	98.11%	
Krishnan & Jawahar [104]	84.16%	79.13%	
Krishnan & Jawahar [158]	95.26%	94.27%	
Silberpfennig et al.[159, 44]	50.64%	71.11%	
Retsinas et al. [160]	46.7%	56.5%	
Sfikas et al. [161]	46.5%	59.9%	
Retsinas et al. [162]	53.2%	64.2%	*Holistic-mPOG+Eucl*
	57.0%	71.1%	*PSeq-mPOG+SM*
	58.3%	76.2%	*PSeq-mPOG+MIST*
Stauffer et al. [163, 164]	51.69%	79.72%	
Stauffer et al. [165]	68.88%	84.77%	*Ensemble*
	45.06%	77.24%	*Keypoint*
	49.57%	76.02%	*Projection*
Riba et al. [166]	41.52%	64.42%	*Keypoint*
	42.83%	65.04%	*Projection*

The most successful methods are the learning based method. Almazan et al. [92] encode visual information with Fisher codes [72], combine them with Pyramidal Histogram Of Characters (PHOC) binary labels and learn a SVM-based attribute model which create a common feature space where visual and textual information can be combined. Sudholt and Fink use a Convolutional Neural Network (CovNet) that predict the PHOC binary labels [100] or a real-valued labels [102, 157] of a given word image. Wilkinson and Brun [156] use a CovNet to create image features following a triple network approach [167] that uses the relationship between word images (i. e.which are related and unrelated) to determine which are the best features to represent them. Krishnan and Jawahar [104, 158] propose another CovNet to learn features from word images and they show that incorporating synthetically word images to train the model improves its retrieval performance.

Unsupervised spotting methods show more modest results as visual information alone is not enough overcome the variability of written text. The method proposed by Silberpfennig et al. [159, 44] builds word signature by extracting HOG and LBP descriptors, concatenating them and using maximum pooling over random groups to generate a compact vectorial representation of the image. Retsinas et al. propose in [160] an unsupervised descriptor where first the image is preprocessed to remove skew and normalize the word height. Then, it is described with the Projection of Oriented Gradient (POG) descriptor [168]. Sfikas et al. propose in [161] to divide the word image into zones, then use a pre-trained CovNet to detect characters or bi-grams in these zones and concatenate the classification results of all zones to form image descriptor. In [162], Retinas et al. use the same zones approach but replacing the CovNet with a modified version of their POG descriptor and a sequence matching algorithm. Finally, a different group of unsupervised spotting methods is the graph-based spotting. These methods represent the word in the image as a graph and use a graph matching algorithm to measure the similarity between word snippets. Some of them use a specific edit distance to compare the graphs [163, 164], other combine multiple graphs to obtain a robuster similarity estimation [165] or use deep learning to learn a graph matching metric [166].

Comparing the results obtained by these methods with the results of our signature, we can see that our signature is not performing as good in this datasets as in the George Washington dataset. This is due to the symbol classification accuracy of the ILC algorithm. Although we cannot obtain a quantitative measure of the classification error rate, a visual inspection of the probability maps shows that the strength of the detections is fainter and there are symbols that are not detected at all. We believe that the problem is due to the visual dissimilarity between the images script and the TTF fonts used to train both the codebook and the ILC classifiers.

6.10 CONCLUSIONS

In this chapter, we have presented a signature based on accumulating the probabilities of the symbols appearing on the word image. This results in a compact signature on document images using alphabet type script as, alphabets only contain several dozens of different symbols.

The probabilities are computed by following a very simple procedure where we accumulate BoVW histograms locally and use a linear classifier to obtain the probability of each symbol category. The BoVW codewords are obtained using the semantic codebook presented in Chapter 5 but replacing the HOG descriptor with a modified version of the FREAK descriptors. In our proposal, we use a linear projection to reduce the dimensionality instead of returning the binarized response of the dimensions that display the higher level of variance. Although this modification increases the computational cost with respect to the original implementation, it also increases the discriminative power of the descriptor and it is still faster than HOG descriptors. We have tested two different projections algorithms: LSA and LDA. The unsupervised LSA projection generates descriptors with a discriminative power similar to HOG, while LDA increases the discriminative power of the signature.

The probability maps are the computed by applying the proposed ILC algorithm followed by a regression algorithm that converts the classification scores to probabilities. The ILC algorithm is a simple procedure that combines the accumulation and classification steps. This allows to compute the classification scores on an image rectangular region only using 16 additions. We also proposed a modification to the linear classifier learning algorithms so, they can train the classifier without having to explicitly accumulate the BoVW histograms. The resulting learning algorithm is over an order of magnitude faster than the standard implementation and it uses up to four orders of magnitude less of memory.

We evaluate the different parameters of the proposed signature on the George Washington dataset. The results show that the proposed signature is able to obtain a performance similar to the standard BoVW presented in Chapter 2 but only using 2 bytes of information to represent each word snippet. We have compared the proposed spotting signature against other unsupervised word spotting methods (i. e.methods without a language model incorporated) on the Botany and Konzilsprotokolle datasets. Results show that the signature is comparable to other methods but with a much lower memory footprint.

7 CONCLUSIONS AND FUTURE WORK

In this thesis, we have contributed several methods to the state of the art of word spotting. Our contributions are mainly on handwritten historical documents in documents written by a single writer or multiple writers with a similar calligraphy.

In Chapter 2, we have evaluated the use of the Bag of Visual Words (BoVW) representation for the word spotting task in historical handwritten documents. During the course of this thesis, the BoVW framework has gained attention as a way to represent handwritten words, however there is a great performance discrepancy depending on the parameters used to generate the signature. We study which are the parameters that have the largest impact on the retrieval performance of the system and which have a negligible effect on word spotting systems. For example, the most important increase in performance came from using a Spatial Pyramid Matching (SPM) to divided the visual words contributions into multiple horizontal bins. We believe that such performance boost comes from the fact that this spatial configuration allows the signature to encode the sequential information of the word, i. e.which character comes before another, mimicking the information that is encoded in sequential word representations, but while preserving the advantage of holistic word representations. However, the retrieval performance increase comes also with a substantial increase of the word's signature dimensionality. Therefore, we have to be careful when designing systems that have to index tens- or hundreds of thousands of word images. Fortunately, BoVW signatures have a vectorial representation that allows the usage of standard machine learning techniques for dimensionality reduction and dimensionality compression that have helped us on later chapters. The comparison with other word spotting methods shows us that BoVW signatures provide the best retrieval performance for completely unsupervised systems.

In Chapter 3, we have presented an efficient keyword spotting method that does not involve any segmentation stage. The proposed approach confers a clear advantage over segmentation-based methods as it does not depend on the quality of the putative word locations provided by a segmentation algorithm. We propose a simple patch-based framework where a set of overlapping patches covering the whole document image is used to describe all the possible words locations. Then, a query image is located by comparing its signature with the signatures of these document patches and, returning the locations with the higher number of hits. The patches are described using BoVW signatures that are designed to have low dimensionality so that a document image collection can be indexed by a reasonable amount of memory. Combining these BoVW signatures with the Latent Semantic Analysis (LSA) technique and Product Quantizer (PQ) compression codes, the proposed method yields a very compact, efficient and discriminative representation. The proposed representation is able to efficiently index the document information both in terms of memory and computational cost, resulting in a method suitable for large-scale scenarios. We have also introduced a multi-length patch representation that increases the retrieval performance when querying small words. Finally, we have tested the proposed method on three different collections of historical documents and we have presented an exhaustive comparison with state-of-the-art word-spotting method.

In Chapter 4, we explore the fusion of multiple information modalities into a single feature space so, we can index word images using one modality and cast queries using another. The proposed method has been used both in a query-by-string and in a query-by-speech setting, using both text and audio to retrieve word images from a handwritten historical documents. The proposed approaches reaches state of the art spotting performance for in-vocabulary queries, i. e.queries where both modalities are present while creating the model. However, performance drops for queries that were not present during the training phase. Additionally, we have used the system *in reverse*, i. e.to index words represented by a different modality (utterances in our experiments) while casting queries using visual information. We demonstrated this property by showing a straightforward query-by-speech system that is able, given a query image, generate its audio without actually transcribing its contents. This is accomplished by creating an index of the word audio signatures generated using multiple voices. Then, the audio associated to a word image is created by querying the word visual signature into the index and selecting the audio utterance with the highest consensus.

In Chapter 5, we have proposed a method to automatically generate a codebook from synthetic data. The main idea is to take advantage of the currently existing large corpus of true type fonts emulating human script to generate a codebook that is database agnostic, i. e.a codebook that displays a good performance on any document collection using the same script as the synthetic fonts used to generate the codebook. This allows us to avoid creating a codebook for each document collection that we want to index. Additionally, it also allow us to easily obtain annotated data that can be useful to generate the codebook. As we have seen on Chapter 2, the spotting system performance depends on the number of codewords of the codebook. This number cannot be known *a priori* and it can only be obtained experimentally. Although a rough estimation of the codebook size can be guessed by looking at collections that have similar properties (e. g.script type and writing style), a more convenient approach is to use annotated data to select which is the best set of codewords for a given set of descriptors. Taking advantage that in our setting (i. e.handwritten historical documents using Latin script) words are formed by a relatively small set of different symbols, we wanted to ensure that our codewords will be used to represent a single symbol. Therefore, we propose a supervised codebook that uses Shannon entropy to search partitions on an agglomerative tree so the selected codewords maximize the separability between symbol classes. We train this codebook on synthetic symbol images and test it on handwritten historical document images. Results show that the codebook attains a performance similar to the best configuration of an unsupervised codebooks but with a smaller codebook size. Additionally, we also have proposed the use of an additional Hierarchical k-Means (HKM) codebook to approximate the descriptors instead of encoding them. This codebook is used as a *quantization* codebook where an input descriptor is approximated by the descriptor of its associated leaf (i. e.the one reached after traversing the tree). Since all descriptors are going to be represented by a leaf descriptor, we can pre-compute the leaf descriptor's encoding and use these pre-computed values as an approximation of the actual encoding. This allows us not only to avoid computing the k-Nearest Neighbors when encoding a descriptor but also avoid other steps, like computing its Locality-constrained Linear Coding (LLC) weights, as all these information is already available at its associated tree leaf. The use of the HKM to encode the descriptors reduce the encoding complexity to sub-linear while the spotting system shows a similar retrieval performance.

In Chapter 6, we expand on the idea of using synthetic data to improve our model and use it to create a very compact visual signature. One of the main drawbacks of the BoVW signatures is their high dimensionality. This limits or prevents the usage of certain enhancements like large codebooks, spatial pyramid or complex encoding (e. g.the Vector of Locally Aggregated Descriptors (VLAD) representation). Additionally, compression techniques like LSA projections and PQ codes are still needed to further reduce the signature's dimensionality. This sharp dimensionality reduction results on a signature that has less characterization power than its initial counterpart. In this chapter, we follow a different approach and we attempt to create a signature that is compact since the beginning. The main idea is that instead of representing visual words, signature bins represent the probability that a certain symbol is present at a certain location of the word image. Such a signature will only have 26 dimensions per spatial bin when used on the document types typically processed on this thesis (i. e.documents that use Latin script). Therefore, in this chapter we use the BoVW framework to compute the symbol probabilities instead of using it to represent word images. We achieve this by following a fairly straightforward approach where a sliding window traverses the image and computes at each pixel the probability that it belongs to a certain category. Such a brute force approach requires certain refinements to the BoVW pipeline so, images can be processed efficiently. Namely, we propose a modified version of the Fast Retina Keypoint (FREAK) descriptor which helps the codebook to increase the separability between classes while at the same time it reduces the computational cost when compared to Histogram of Oriented Gradients (HOG) descriptors. We also propose the Integral Linear Classifier (ILC) algorithm which merges the pooling and classification steps and allows to compute the classification score of any rectangular image region at constant and low computational cost. Finally, we modify the training algorithm of the linear classifier so it does not need to explicitly accumulate BoVW histograms. This reduces its computational complexity by at least an order of magnitude and its spatial complexity by several magnitudes. The resulting signature shows a performance comparable to the best BoVW signature but only using about a hundred dimensions. Combining the new signature with PQ codes we are able to index the Washington dataset using only two bytes of information per word image.

FUTURE WORK

Improve the codeword semantic representation: The semantic codebook from Chapter 5 attains the same retrieval performance than a k-means codebook with the optimal k value. This is an important feature from an practical point of view as we do not need to empirically test multiple codebook sizes to determine which is better for a certain document image collection. However, the codewords obtained do not represent a single codeword but a group of them, introducing non-linearities to the BoVW signature that have to be solved at latter stages. This can be partially alleviated by making our descriptor *aware* of the problem. In chapter Chapter 6, we have used an Linear Discriminant Analysis (LDA) projection to increase the separation between symbol categories. Although LDA slightly increases the classification score, codewords still represent multiple symbols. We can attempt to solve this problem by testing other techniques that offer better properties, e. g.the demixed Principal Component Analysis [169] separates classes while keeping the geometry of the original descriptor space. Or, by exploring completely different approaches to the problem, e. g.compute the projection as the minimum ensemble of linear classifiers needed to categorize the symbols directly using the descriptor [170] However, these approaches can only *help* the codebook but not solve the problem by themselves as symbols categories are not linearly separable at the descriptor level. We believe that a better approach is to reduce the number of descriptors used to train the agglomerative codebook. Instead of training it using descriptors extracted from all possible locations, use only "*representative*" descriptors defined at meaningful symbol locations. The resulting codewords are more likely to belong to a single symbol category as descriptor symbol variability per category will be reduced and descriptors belonging to multiple categories (i. e.descriptors sampled over bi-grams) will not be used. Although the descriptors surrounding the "*representatives*" are not represented by the agglomerative codebook, we can define their relationship directly on the approximate HKM codebook by using the spatial relationship between them.

Generate synthetic Symbol Probability Signatures: The symbol probability signatures presented in Chapter 6 attain a similar performance to BoVW signatures but having a much lower dimensionality. We want to test them on other scenarios like the segmentation-free approach presented in Chapter 3 or in the query-by-X approach proposed in Chapter 3. An additional advantage of this signature compared to a BoVW is its simplicity: the signature it can be easily modeled with the font parameters and the word text. We can use this to analyze the modifications introduced to the signature when different fonts or images distortions are used and use this information to generate a model that will increase the robustness of the signature against them.

A LINE SEGMENTATION

In this appendix, we explore the use of second-order derivatives to detect text lines on handwritten document images. Taking advantage that the second derivative gives a minimum response when a dark linear element over a bright background has the same orientation as the filter, we use this operator to create a map with the local orientation and strength of putative text lines in the document. Then, we detect line segments by selecting and merging the filter responses that have a similar orientation and scale. Finally, text lines are found by merging the segments that are within the same text region. The proposed segmentation algorithm, is learning-free while showing a performance similar to the state of the art methods in publicly available datasets.

A.1 INTRODUCTION

The history of our ancestors is locked in libraries and archives within the vast volumes of preserved historic manuscripts. The accessibility and dissemination of such cultural assets will provide an important impact to our society. However, manually inspecting such huge collections for extracting relevant data is unfeasible and that is why automatic tools for processing such historic data have a paramount importance. But before applying any content extraction method, there is usually a pre-processing step of layout analysis of document images that is needed.

Within the different layout analysis tasks, text line segmentation is one of the pillar stages. Subsequent recognition methods depend on a proper text-line segmentation step. Although text line extraction is somehow an easy step for modern typewritten documents, where a simple algorithm do already perform perfectly, it is not the case for historic handwritten documents. Document degradation, the lack of layout regularity, variability in handwriting styles, text skew and text elements, ascenders and descenders, touching other text lines makes the problem much more difficult.

A proper segmentation of the different text lines that appear within historic manuscripts is a critical point for many document image recognition applications, either because the subsequent algorithms work at line level, or because they can benefit from information extracted from such process such as baseline position, text height, ascenders and descenders localization, etc. Such applications range from document deskewing [171], handwriting recognition[172, 17, 13], or keyword spotting [18].

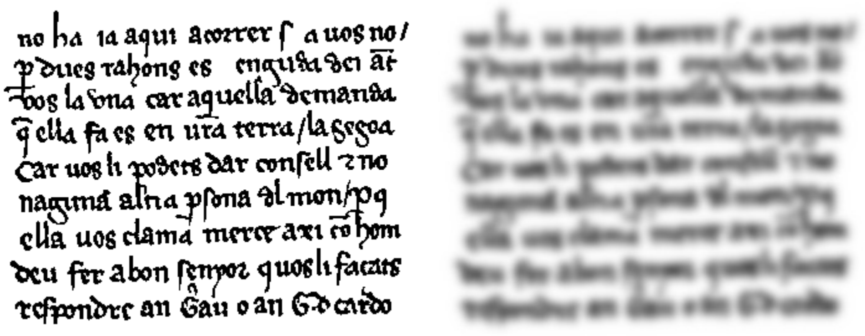
In the literature, there are many text line segmentation algorithms. The two classical approaches are either projection profile-based or rely on the Hough transform. The projection profile approach [173, 174] is based on projecting the document pixels which are relevant (detected through binarization for example) into the Y-axis of the image. Then, the location of the text lines corresponds to the local maxima of the projection histogram. This method requires some pre-processing to ensure that the text lines are aligned with the Y-axis of the image, otherwise the detection is not possible. On the other hand, a different classical approach is to base the text line detection on the use of the Hough transform [175]. These approaches use the Hough transform to find the parameters of the visual linear structures produced by the text in the document. Unlike projection-profile approaches, these approaches can detect text lines which have a different orientation than the dominant one of the text. Both of those approaches process the image in a holistic fashion which might be problematic when the document layout does not follow a classic Manhattan structure, such as when the text is distributed into different columns or does contain tabular structures. Therefore, other approaches try to obtain line information at local level. Some methods group the pixels belonging to the same character into regions and then try to group them into lines. For instance, Cruz and Ramos-Terrades use in [176] the regions detected via connected components analysis to use an expectation-maximization algorithm to detect the text lines. However, connected component-based methods are not particularly interesting when nearby text lines touch each other because of ascender and descender artifacts. Other methods, create an energy map which corresponds to the distribution of the elements over the image and the try segment the lines over this energy map [177, 178] with seam carving-like approaches. Recently, some authors use a learning based methods to detect image regions where text lines are likely to appear [179, 180]. This approach is more robust to image transformations and noise than binarization or filter based methods. However, it has the drawback that a large amount of information is needed to properly train the classifiers. For a detailed explanation of the different approaches used for text line segmentation, we refer the reader to the survey [181].

In this appendix, we use the second-order Gaussian derivatives to find an estimation of the dominant orientation at each pixel. By filtering the image at different scales, we are able to locate also the characteristic scale of the text line, as the second-order Gaussian derivatives give a stronger response at the scale where the blurred text lines is closer to the σ of the Gaussian. Then, these local estimations are accumulated into a histogram to determine which are the most common orientation and scale of the elements present in the image. The pixels belonging to these distributions are processed independently and grouped first into text regions and later text lines. Finally, we use the binarized components within the text region to determine the line segmentation. The main contributions of this appendix are twofold: First, we show that the second-order Gaussian filter can be used as a local operator to determine the orientation and scale of the text lines. Second, we propose a method to group the Gaussian filter estimates into text lines. Although the proposed method is quite straightforward, the obtained results are promising.

A.2 SECOND ORDER DERIVATIVE ANALYSIS

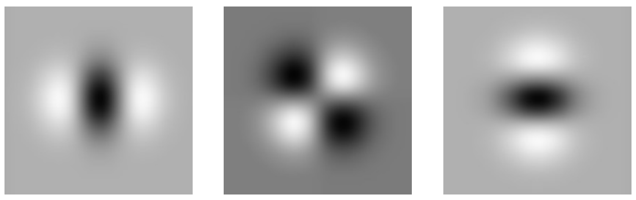
We analyze the output of the second-order Gaussian derivatives of the image to extract which is the orientation and characteristic scale of the text lines present at each pixel of the image. First, we are going to review how we compute the second-order Gaussian derivatives and extract the line orientation efficiently at a given scale. Then, we are going to show how this approach is used at multiple scales to retrieve also the characteristic scales of the text elements in the document.

In order to achieve a certain degree of invariance to illumination and degradation conditions of the document, we first binarize the images with



Binary Blurred.

Figure A.1: Blurring the binary document with a large enough Gaussian generates an image where text lines appear as blobs.



$\frac{\partial}{\partial x^2}\mathbf{G}(x,y)$ $\frac{\partial}{\partial x \partial y}\mathbf{G}(x,y)$ $\frac{\partial}{\partial y^2}\mathbf{G}(x,y)$

Figure A.2: Bases used to compute the second order Gaussian derivative at any orientation.

the Sauvola and Pietikäinen algorithm [20]. Therefore, from now on, all document images are supposed to be binary images.

A.2.1 *Second Order Derivative*

In order to locally estimate the orientation of the text line, we want a filter that yields strong responses at the locations where a text line is present. Therefore, the oriented second-derivative Gaussian function is a good choice, since this operator has strong responses over lines. Since text lines resemble a line when blurred by a Gaussian with a large enough σ, we expect that this operator will have a strong response when the appropriate σ is used. For example, in Fig. A.1 the text lines appear as blobs when they are filtered with a Gaussian filter with $\sigma = 12$. Although we can expect that other operators like Gabor or anisotropic Gaussian filters would give a better response than the second-order derivative, this filter has the advantage that it is steerable, i.e. the response of the filter at any given orientation can be calculated as the combination of base filters. This is a well known property for the first-order Gaussian derivatives, where

$$\mathbf{G}'(i,\theta) = \cos(\theta)\frac{\partial}{\partial x}\mathbf{G}(i) + \sin(\theta)\frac{\partial}{\partial y}\mathbf{G}(i).$$

For higher-order derivatives, Freeman and Adelson present in [182] a method to select the minimum set of bases that better represent the given Gaussian filter. For the second-order Gaussian derivative, they show that it can be computed as a function of $\mathbf{C}_1(i) = 0.921\ {}^{\partial}/_{\partial x^2}\mathbf{G}(i)$, $\mathbf{C}_2(i) = 1.843\ {}^{\partial}/_{\partial x \partial y}\mathbf{G}(i)$ and $\mathbf{C}_3(i) = 0.921\ {}^{\partial}/_{\partial y^2}\mathbf{G}(i)$ (see Fig. A.2):

$$\mathbf{G}''(i,\theta) = \cos^2(\theta)\mathbf{C}_1(i) - \cos(\theta)\sin(\theta)\mathbf{C}_2(i) + \sin^2(\theta)\mathbf{C}_3(i). \quad (8)$$

Then, by setting the derivative of Eq. 8 to zero and solving for θ, we find the angles where the filter attains a maximum or minimum value for

$$\theta_a(i) = \frac{1}{2}\arctan\left(\frac{2\mathbf{C}_2(i)}{\mathbf{C}_3(i) - \mathbf{C}_1(i)}\right),$$

and $\theta_b(i) = \theta_a(i) + {}^{\pi}/_{2}$. We choose the angle $\theta_a(i)$ or $\theta_b(i)$ using Eq. 8 and looking at the orientation which gives the strongest response.

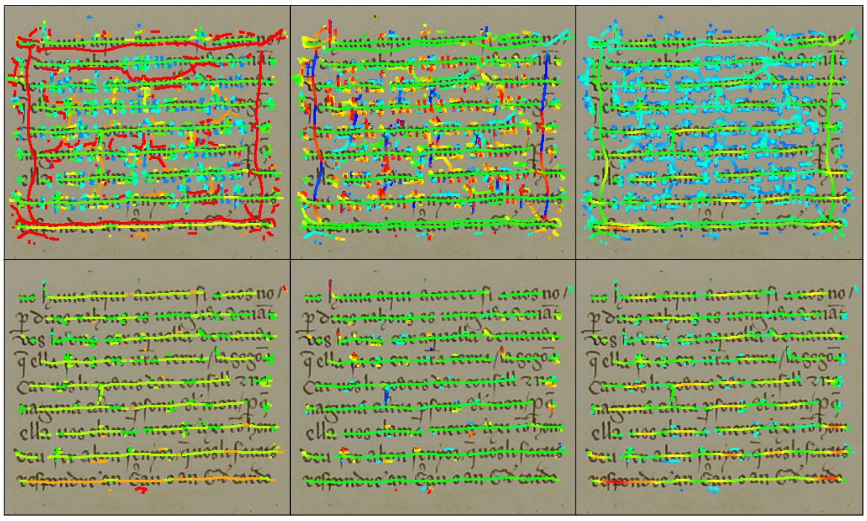


Figure A.3: Example of the selected pixels over the original document image. The magnitude of the values is shown using a hue color map. In the orientation images, green pixels correspond to horizontal orientations while red and blue pixels correspond to vertical orientations (orientation is circular). In the first row we show all local maxima, while in the second row we see the local maxima after applying the non-maxima suppression filter.

A.2.2 *Scale Selection*

The Gaussian filter has a strong response over the text lines when the σ of the Gaussian is similar to the height of the text line. When this happens, the details of the text line characters' have been blurred enough so the line appears as a blob (see Fig. A.1). Therefore, selecting the appropriate σ value for each text line is important to detect line elements independently of the size of the text. The best σ value for each pixel can be automatically calculated by selecting the σ that its second-order Gaussian derivative scaled by $\sqrt{\sigma}$ gives the strongest response. We use the scale factor $\sqrt{\sigma}$ to increase the response at larger scales. Otherwise, we would only consider the smaller scales as document details dilute as we increase the σ of the Gaussian filter. Once we have processed all the scales, we have an image with the line orientation, scale and strength for each pixel. Although this is a computationally expensive approach, it allows to obtain the scale parameter automatically and is efficiently computed by employing a spatial pyramid and recursive Gaussian filters [183] that present a computational cost that is independent of the size of the Gaussian.

Finally, we only consider the pixels that have the strongest response within their line. Therefore, we discard all the pixels that do not have a local maximum when compared to the two neighbors at the perpendicular of the selected orientation. By comparing the neighbors, we assume that two neighboring pixels may belong to the same line when they have a small difference between their orientations and scales. This is similar to extract the ridges of the Gaussian filter response. As a further filtering step, we apply a non-maxima suppression approach where any selected ridge is removed if it falls within the area influence of another local maxima that has a higher filter response. In Fig. A.3, we show the selected orientation, value and scale of the selected pixels for the image in Fig. A.1. In the scale image, we see that the ascenders and descenders have a low value (shown in blue) as they are detected by filters with a smaller σ value. The pixels with a mid-low value (marked as cyan) correspond to the center of the text lines and the pixels marked as red correspond to filters with a large σ. In this example, most red pixels are at the margins of the image and they correspond to the margins of the text paragraph. In the orientation image, we can see that pixels corresponding to horizontal structures all have a similar color (they are green) while elements corresponding to vertical elements (e.g. ascenders and descenders) are marked as red and blue. In this example, we can also see that the non-maxima suppression filters most of the undesired responses.

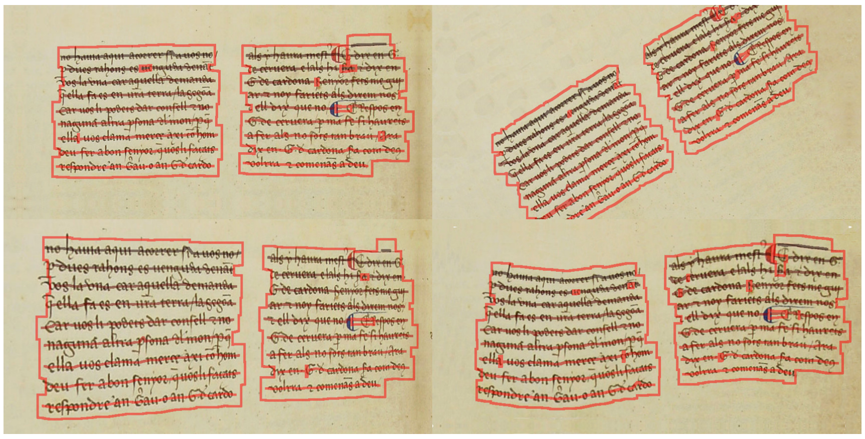


Figure A.4: Text region and line detection on the same image under different transforms.

A.3 LINE SEGMENTATION

Once we have obtained the image with the local orientation, scale and value of the filter, we aggregate them to select the most common orientation and scale of the text lines, detect the text regions, the text lines and finally, obtain the text line segmentation.

A.3.1 *Orientation and Scale Selection*

The first step we take is to obtain which are the most common orientation and scale of the detected line segments. To do so, we accumulate all orientation and scale values of the selected pixels into a histogram. The contribution of each pixel is weighted by the strength of the response of the Gaussian filter. In Fig. A.5, we present an example of the different orientation-scale

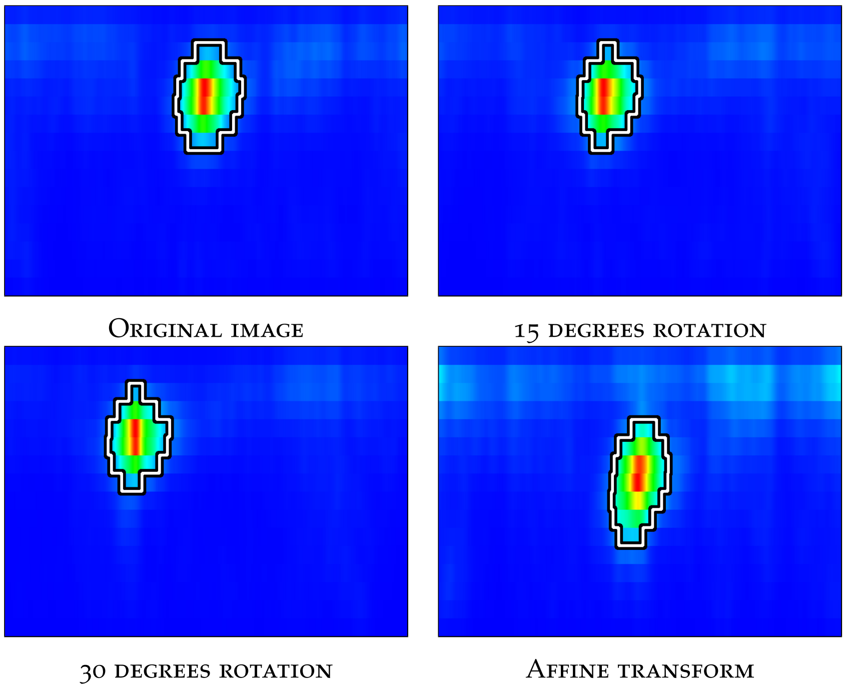


Figure A.5: Histograms obtained for the original image, the image transformed by a rotation and an affine transform. In this histogram images, the x-axis corresponds to the orientation while the y-axis corresponds to the σ of filter.

histogram changes obtained when the image from Fig. A.1 is wrapped by a rotation or an affine transform (c.f. Fig. A.4). Comparing histograms of the original and rotated images, we see that the histogram are simply shifted in the x-axis depending on the rotation applied to the image. The peak at the affine histogram does not move but the values around the peak show a higher *dispersion* as the scale and orientation is slightly different at the margins of the paragraph.

The histograms of Fig. A.5 show that selecting only the pixels which have the same orientation and scale as the histogram's local maxima position is going to filter out too many filter responses. Also, selecting the values which lay within a small window defined around the local maxima is likewise too restrictive (e.g. the window in the affine case has to be larger than in the rigid transform images). Therefore, we apply a watershed algorithm [184] on the histogram in order to assign each histogram value to the local maximum that we will reach while following the gradient direction. Following this procedure, a set of orientation and scale pairs are associated to each local extrema of the histogram. Finally, we discard values which are lower than a ratio of the global maximum value of the histogram. In Fig. A.5, these regions are delimited by the black and white border defined around each local extrema.

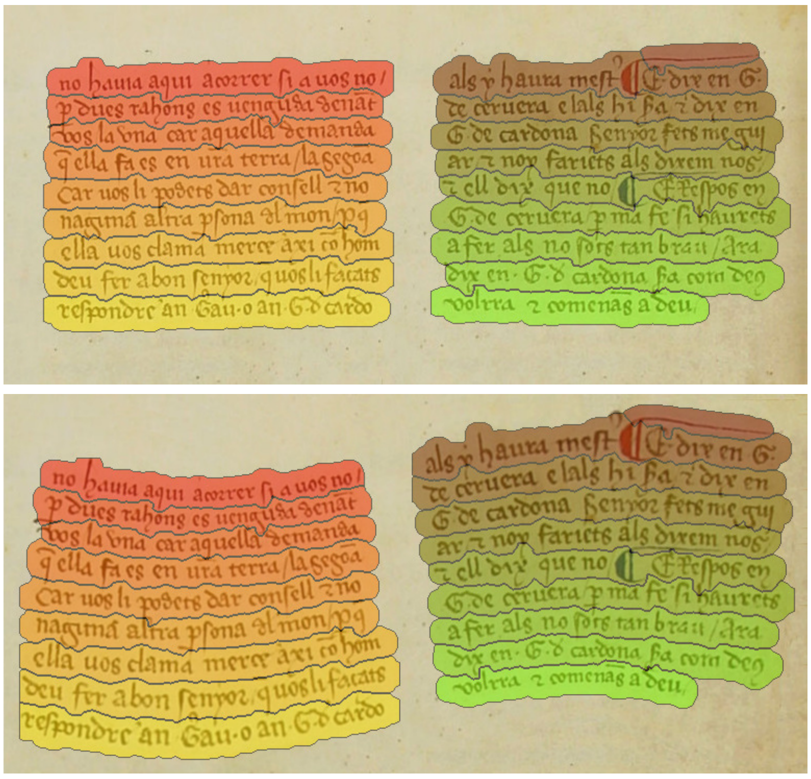


Figure A.6: Final segmentation under different image transforms.

A.3.2 *Text Region and Line Detection*

Once we have selected a scale and an orientation, we use the responses of the Gaussian filter at these parameters to detect the text regions and lines. First, we group the ridges into line segments using connected components. Then, we compute the median separation between consecutive overlapping segments to have an estimation of the separation between text lines L_s. This measure is used to set the maximum distance between neighboring segments. So, by grouping all segments that are within this maximum distance, we obtain regions of the image where text lines are likely to appear. Regions which are too small (a 10% of the largest region) are filtered out. In Fig. A.4, the contour around the text show the two text regions detected following this procedure.

The text regions are processed independently to search for text lines. These lines are formed by successively merging the closest segments within the region until their distance exceeds $\sqrt{L_s}L_s$ or only a single segment is left. We compute the distance between two segments as the smallest distance between its end points. In order to avoid merging segments which are in two different lines, we give a higher weight to the height difference than to the separation between lines. So, the distance between the end-points $\mathbf{p}_a = [p_a^x, p_a^y]$ and $\mathbf{p}_b = [p_b^x, p_b^y]$ is computed as

$$d(\mathbf{p}_a, \mathbf{p}_b) = |p_a^x - p_b^x| + \min(|p_a^y - p_b^y|, L_s/3) + \max(0, |p_a^y - p_b^y| - L_s/3)^2.$$

This measure only gives a quadratic weight to the height differences greater than $L_s/3$, so it greatly penalizes segments which are at a different text line but allows small height differences between consecutive segments. Thus, it gives a certain flexibility to adapt to text line curvature. In Fig. A.4, we show detected lines in document images under different transforms. The coordinates of the end-points are rotated according to the text line orientation, so the x-axis represents the separation between segments and y-axis the separation between the text lines.

A.3.3 *Text Segmentation*

The algorithm so far has only detected the center of the lines, a text line height given by the Gaussian filter and the separation between lines. This parameters can be used to obtain a coarse segmentation of the text lines, but to obtain a finer segmentation we employ a segmentation schema similar to the one proposed by Vo and Lee in [185]. We assign the binary connected components to the closest text line. Components which can be assigned to multiple text lines are decomposed into smaller components using line adjacency graphs [186]. Components which are still assigned to multiple lines are assigned to the line closer to the most of its pixels. The final segmentation is obtained by assigning any pixel within L_s pixels distance to the line of its closest component. Fig. A.6 shows the segmentation obtained in the original and deformed document images.

A.4 EXPERIMENTAL RESULTS

We have evaluated the segmentation algorithm on the IAM database [193], the GRPOLY-DB dataset [188] and the Saintgall and Parzival datasets [194]. The performance of the algorithm is evaluated following the ICDAR 2009 [195] competition, i.e. assigning each predicted line to the groundtruth line with the highest intersection over union score of its binarized pixels and only considering the matches that have more than 90% overlap. Then, the performance is simply reported with the attained precision and recall. Additionally, we have tested the line detection algorithm on the simple documents track of the cBAD dataset [192]. In this dataset, results are also given in precision-recall score but instead of evaluating the segmentation they evaluate the baseline detection accuracy. Therefore, we return as baselines the lines detected by the algorithm displaced $\sigma_s/2$ down where σ_s is the σ of the selected Gaussian filter. For all datasets we use the same parameters. The size of the images is reduced by half, the Gaussian scale-space is computed at $\sigma \in \{2, 4, 6, 8, 10, 14, 18, 22, 26, 30\}$, the maximum distance between line segments is set to $1.2L_s$ and the Sauvola-Pietikäinen binarization window is set to 50 pixels. The only exception is the cBAD dataset where we do not down-scale the images so the Gaussian filters are larger with $\sigma \in \{5, 10, 14, 18, 22, 26, 30, 38, 46\}$. The results obtained in all datasets are shown in Table A.1. These results show that the proposed algorithm is comparable to the state of the art algorithm despite not relying on any supervised learning stage to robustly detect the text lines. The algorithm is also fast as it processes an image from the Parzival dataset in around 1.5 seconds only relying on the CPU. Since the most computationally expensive step of the algorithm is the computation of the Gaussian filter derivatives, this runtime can be greatly reduced by the use of GPU acceleration.

Dataset	Method	Precision	Recall	f-Measure
Parzival	CNN [179]	98.9%	98.7%	98.8%
	RLSA [179]	70.2%	50.0%	58.5%
	Ours	*93.3%*	*92.7%*	*93.0%*
	Ours test only	*95.0%*	*95.2%*	*95.1%*
Saintgall	CNN [179]	96.5%	96.4%	96.5%
	RLSA [179]	92.6%	89.6%	91.1%
	Ours	*97.8%*	*97.7%*	*97.8%*
	Ours test only	*98.5%*	*98.5%*	*98.5%*
GRPOLY-DB	Shredding [187, 188]	80.6%	92.4%	86.1%
	Hough [189, 188]	94.2%	96.7%	95.4%
	Ours	*90.2%*	*93.1%*	*91.6%*
IAM	Projection [190]	–	–	37.7%
	Rectangle [191, 190]	–	–	96.7%
	Hough [189, 190]	–	–	92.6%
	Shredding [187, 190]	–	–	36.0%
	Ours	*99.8%*	*99.7%*	*99.7%*
cBAD Track-A	DMRZ [192]	97.3%	97.0%	97.1%
	UPVLC [192]	93.7%	85.5%	89.4%
	BYU [192]	87.8%	90.7%	89.2%
	IRISA [192]	88.3%	87.7%	88.0%
	LITIS [192]	78.0%	83.6%	80.7%
	Ours	*74.7%*	*92.6%*	*82.7%*

Table A.1: Text line segmentation results

A.5 CONCLUSIONS AND FUTURE WORK

In this appendix, we show that the second order derivatives can be used to detect the orientation and scale at pixel level. The main advantage of this operator, when compared to anisotropic Gaussian filters or Gabor filters, is that it is steerable and can be computed very efficiently with a space pyramid and recursive Gaussian filters. Then, we use the output of this operator to select the dominant orientation and scale of the lines present at the image and obtain the detected text regions and lines by simply merging the detected line segments. Finally, the line detection is easily extended to line segmentation by assigning the binary components to the closest detected line and assigning all pixels to the nearest component. Although the algorithm is simple it attains a performance similar to more complex approaches which rely on supervised machine learning strategies.

Future work

The scale space generated by the second order derivatives displays three distinct peaks: a first faint one at the scale of the text line strokes, then a stronger one at the line level and finally another at a later scale. This last response correspond to larger document structures like text columns or paragraphs. We believe it is possible to exploit this information to automatically detect and segment the document into these larger structures. This will allow us to detect text lines written in different orientations or small text annotations more reliably. Additionally, by analysing the variation of the lower scale orientation (i. e.the orientation of text line strokes), we can determine when a line is structural line or is a text line and extract attributes that will allow us to differentiate between different types of scripts (e. g.typewritten vs handwritten).

B LINE SEGMENTATION

In this appendix, we present an automatic method for separating static and variable content from administrative document images. An alignment approach is able to unsupervisedly build probabilistic templates from a set of examples of the same document kind. Such templates define which is the likelihood of every pixel of being either static or variable content. In the extraction step, the same alignment technique is used to match an incoming image with the template and to locate the positions where variable fields appear. We validate our approach on the public NIST Structured Tax Forms Dataset.

B.1 INTRODUCTION

Many efforts have been made by companies and institutions in the digital age to get rid of processing information stored in physical paper by shifting their workflows towards electronic information. A paperless working environment present several advantages such as important economic and storage savings, remote accessibility to information, security and environmental progress. However, the fact is that nowadays most of entities still need to process an important portion of their incoming data in paper, image or in the best case scenario, in electronic document formats. In most of the cases, such information comes in an unstructured way, so that an interpretation step is still needed in order to extract in a structured way such data. Manually processing the bulk of incoming documents is a really costly task and the industry and the market needs have led an important amount of research and development in the context of automatic processing such administrative documents.

In the field of Document Image Analysis, many tasks that fall within the digital mailroom paradigm have been addressed, from document classification [196, 197, 198], document flow segmentation [199], routing [200], and information extraction [201, 202, 203, 204]. In particular, the information extraction step, is often based on the definition of templates that help to point out the locations in which an OCR engine has to read the particular fields to extract. Such templates are usually based on the detection of anchor elements that can be easily and steadily extracted from different instances of the same document kind, i.e. they are based on the detection of *static* content that always appear within the documents under study and that can help to locate the position of *variable* information. Prior work such as the methods proposed by Ishitani [202, 203], Peanho et al. [201], Rusiñol et al. [204], Schuster et al. [205] or Santosh et al. [206] often involve a manual intervention of an expert user that assist the system in building such templates.

In this appendix, we focus in a particular step of the broader information extraction problem which is the task of segmenting which portions of a document image might contain relevant content since they are the ones that change from instance to instance of the same document type. We propose an unsupervised and fully automated process that given several examples of the same document kind is able to produce a probabilistic template that will indicate the likelihood of every pixel of being either static or variable content. In order to do so, we rely on an image alignment algorithm [207, 208] that is able to cope with the deformations that we can find across different document instances. Once this probabilistic template is build, new incoming document images are processed in order to detect the variable zones of those unseen documents. The main advantage of the system is obviously its ability of producing those templates in an automatic fashion, thus eliminating the need of manual intervention. This specially useful in historical collections where the template of the form is usually not available and therefore has to be generated.

Such approach is mainly interesting when dealing with highly structured documents that have a predefined static layout that is later filled by the users with the relevant information. We carried our tests using the public NIST Structured Tax Forms Dataset (SPDB2) [209], but such method could also be applied to other document kinds that also present this particularity of mixing static and variable content such as invoices, contracts, identification documents, and so on...

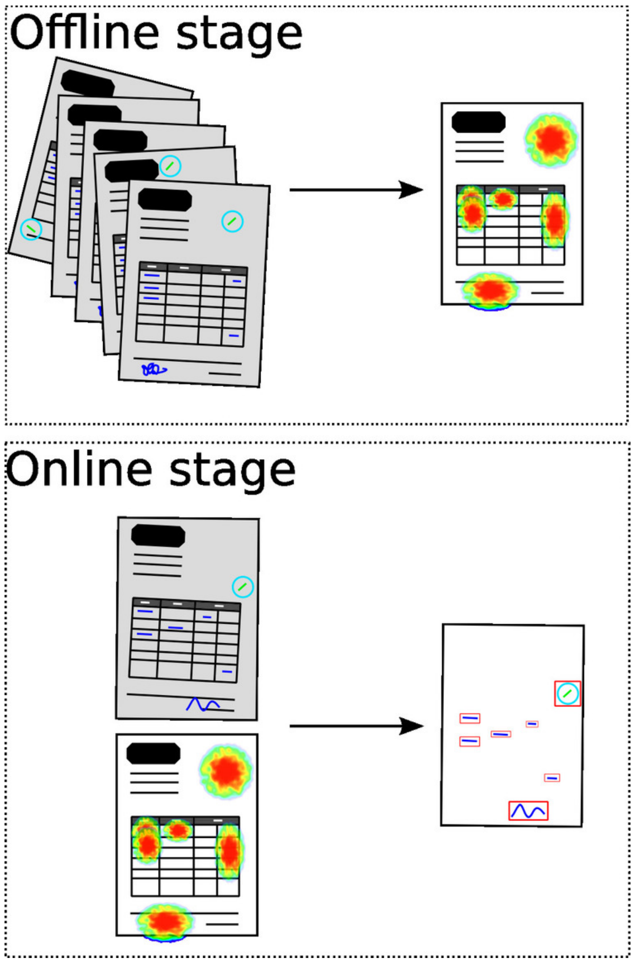


Figure B.1: System Overview

B.2 OVERVIEW

We show in Fig. B.1 a schematic overview of the proposed system. Two different stages are considered. In an offline step, several instances of the same document kind have to be provided to the system. A pairwise alignment step is applied in order to cope with distortions that might appear in the digitization process so that all the images are registered together. Once aligned, pixels that are steadily activated as foreground are considered as most probable to be static content whereas pixels that are foreground in some images but not in the others are considered more likely to be variable content. From this "voting" step, a probabilistic template image is automatically produced.

When a new incoming image arrives, it is then aligned with the probabilistic template which is also used in order to weight the foreground pixels. A simple post-processing step is then used in order to end up either with a binary image that only contains variable content or with a set of bounding-boxes in the original image that locate such variable fields.

Let us continue with the details on how the alignment step is performed.

B.3 DOCUMENT ALIGNMENT

In order to align two document images $\mathbf{I}$ and $\mathbf{T}$, we use the algorithm proposed by Lucas and Kanade [207] to compute the optical flow of the image. This method however has been extensively used in image registration problems, specifically on face registration [210]. The aim of the algorithm is to find the parameters $\mathbf{p}$ that wrap the image $\mathbf{I}$ so it minimizes the differences with the image $\mathbf{T}$,

$$\arg\min_{\mathbf{p}} \sum_{\mathbf{x}} \|\mathbf{I}\left(\mathcal{W}(\mathbf{x},\mathbf{p})\right) - \mathbf{T}(\mathbf{x})\|^2, \tag{9}$$

where $\mathcal{W}(\mathbf{x},\mathbf{p})$ is the wrap function that converts the coordinates $\mathbf{x}$ from the template to the original image reference frame. The complexity of transform $\mathbf{p}$ depends on the model used to relate both images. For example, for face registration problems a non-rigid model like Active Appearance Models [210] is used. However, in our case document images are related at most by an affine transform so $\mathbf{p} = [t_x, t_y, s_x, s_y, s_k, \alpha]$, where t_x and t_y are the translation parameters, s_x and s_y are the scale parameters, s_k is the skew parameter and α is the rotation parameter. Then, the wrap function $\{x', y'\} = \mathcal{W}(\{x,y\},\mathbf{p})$ becomes,

$$\begin{aligned} x' &= s_x \cos(\alpha)\, x + (s_k \cos(\alpha) + s_y \sin(\alpha))\, y + t_x \\ y' &= -s_x \sin(\alpha)\, x + (-s_k \sin(\alpha) + s_y \cos(\alpha))\, y + t_y. \end{aligned}$$

The algorithm finds the solution to Eq. 9 by iteratively computing the parameters increments $\Delta\mathbf{p}$ that solve

$$\arg\min_{\mathbf{p}} \sum_{\mathbf{x}} \|\mathbf{I}\left(\mathcal{W}(\mathbf{x},\mathbf{p}+\Delta\mathbf{p})\right) - \mathbf{T}(\mathbf{x})\|^2, \tag{10}$$

and updating the parameter $\mathbf{p} = \mathbf{p} + \Delta\mathbf{p}$ until the parameters estimate $\mathbf{p}$ converges. The non-linear expression $\mathbf{I}(\mathcal{W}(\mathbf{x},\mathbf{p}+\Delta\mathbf{p}))$ on Eq. 10 is linearized with its first order Taylor expansion,

$$\arg\min_{\mathbf{p}} \sum_{\mathbf{x}} \|\mathbf{I}\left(\mathcal{W}(\mathbf{x},\mathbf{p})\right) + \nabla\mathbf{I}\frac{\partial\mathcal{W}}{\partial\mathbf{p}}\Delta\mathbf{p} - \mathbf{T}(\mathbf{x})\|^2, \tag{11}$$

where $\nabla \mathbf{I} = (\partial \mathbf{I}/\partial x, \partial \mathbf{I}/\partial y)$ is the gradient of $\mathbf{I}$ evaluated at $\mathcal{W}(\mathbf{x}, \mathbf{p})$ and the term $\partial \mathcal{W}/\partial \mathbf{p}$ is the Jacobian of the wrap function which in our case is

$$\frac{\mathcal{W}}{\partial \mathbf{p}} = \begin{bmatrix} 1 & 0 & -xc_\alpha & -ys_\alpha & -yc_\alpha & (y\,s_k + x\,s_x)s_\alpha - ys_y c_\alpha \\ 0 & 1 & xs_\alpha & -yc_\alpha & ys_\alpha & (y\,s_k + x\,s_x)c_\alpha + ys_y s_\alpha \end{bmatrix},$$

where $s_\alpha = \sin(\alpha)$ and $c_\alpha = \cos(\alpha)$. Then, Eq. 11 is solved by computing its partial derivatives and solving the resulting equation, that gives

$$\Delta \mathbf{p} = \mathbf{H}^{-1} \sum_{\mathbf{x}} \left[\nabla \mathbf{I} \frac{\partial \mathcal{W}}{\partial \mathbf{p}} \right]^\top [\mathbf{T}(\mathbf{x}) - \mathbf{I}(\mathcal{W}(\mathbf{x}, \mathbf{p}))], \tag{12}$$

where $\mathbf{H}$ is the Gaussian-Newton approximation to the Hessian matrix

$$\mathbf{H} = \sum_{\mathbf{x}} \left[\nabla \mathbf{I} \frac{\partial \mathcal{W}}{\partial \mathbf{p}} \right]^\top \left[\nabla \mathbf{I} \frac{\partial \mathcal{W}}{\partial \mathbf{p}} \right]. \tag{13}$$

Summarizing, the algorithm starts with an initial estimation of the parameters, in our case $\mathbf{p} = [0, 0, 1, 1, 0, 0]$ and it keeps updating the parameter estimation $\mathbf{p}$ with Eq. 12 until it converges, i.e. until $|\Delta \mathbf{p}| < \epsilon$, where ϵ is a small value.

This algorithm is sensible to local minima, so several authors replaced the sum of squared errors in Eq. 9 by a more robust estimate of the registration error between the two images like the enhanced correlation coefficient [211], Gabor filters computed in the Fourier domain [212], image gradient [213] or feature-based methods [214]. In our case, we use the original algorithm but we take several pre-processing steps before aligning the document images, so the registration algorithm attains a certain degree of robustness. First, the images are binarized using the adaptive binarization algorithm proposed by Sauvola and Pietikäinen [20]. The advantages of working with binarized images are twofold: it lessens the effects of illumination and degradation problems in the document images, and it allows to accelerate the registration algorithm by reducing the amount of points that we need to wrap while estimating $\Delta \mathbf{p}$. Then, a connected component analysis step is used to remove regions which are too small to be relevant and large regions which are touching the image margins and are likely to be marginal artifacts. Finally, the module of the gradient is computed over the image in order to reduce the effects of large *foreground* regions. These regions provide a large contribution to the estimation of $\Delta \mathbf{p}$ in Eq. 12 and they can lead to a misalignment between the images when they belong to a dynamic document structure (e.g. a text written with a larger font or a stamp graphic). Using the module of the gradient, we only keep the contour of the *foreground* elements so the influence of large structures on the $\Delta \mathbf{p}$ is reduced. The obtained image is re-binarized with the Otsu algorithm [215].

As we pointed out before, by using binarized images we only have to take into account the *foreground* pixels while wrapping the original image. Therefore, we can reduce the computational cost of the algorithm by only wrapping those pixels. Moreover, the algorithm can be further speeded up by selecting an small random sub-set of the *foreground* pixels. In our case, images are correctly registered using only a 5% of the *foreground* pixels.

The algorithm is sensible to local minima, so when images are related by a large transform it is very likely that the registration algorithm gets *stuck* at a local minima before obtaining the actual registration parameters. Therefore, we follow a multiscale approach where the parameters $\mathbf{p}$ are initially computed using a large sigma at $\nabla\mathbf{I}$ (e.g. with $\sigma = 20$). Then, the parameters $\mathbf{p}$ are recomputed using a smaller sigma using the previous estimation as a *warm start* until we reach $\sigma = 1$. Although this should increase the computationally cost of the algorithm, the algorithm has the same runtime as we use spatial pyramids and recursive Gaussian derivatives [183] to obtain the $\nabla\mathbf{I}$ at each scale, the algorithm converges faster at coarser scales as there are less details and the *warm initialization* greatly reduces the number of iterations of finer scales.

B.4 SYSTEM DESCRIPTION

We use the image registration algorithm from Section B.3 both to align the document images while creating the model template and also to register the model template with a document image when filtering the static parts to obtain the dynamic content of the document.

B.4.1 *Document Static Model Generation*

We have a set of document images which have the same layout as the sample shown in Fig. B.2 and we want to obtain an image which contains only the document structures which are common in all documents, i.e. the static elements of the document collection. An example of an obtained model template is shown in Fig. B.3.

In order to obtain such a model, we first have to align all the documents of the collection using the algorithm described in Section B.3. We can obtain the model by computing all possible image pairs, however following this approach the number of image pairs to be aligned is quadratic respect the number of image of the set. For example, the collection of Fig. B.2 has 24 images which means that we have to compute 300 relationships to obtain all the image relationships. Instead, we can randomly select N images which in turn are randomly aligned only with M images of the collection. The resulting model is generated only computing $N \times M$ image pair alignments and although it uses less images, the obtained model does not greatly differ from the obtained using the whole collection. For instance, the model in Fig. B.3 has been generated with $N = M = 7$ so only 49 image pairs are aligned.

The document static model $\mathcal{M}$ for the collection of document images $\mathcal{C}$ is generated with two steps: first we generate as set of N partial models $\mathcal{P} = \{\mathcal{P}_k \in \mathcal{C} | k \in \{1, .., N\}\}$ and then we group them into the final document static model $\mathcal{M}$. A partial mode $\mathcal{P}_k$ is created by randomly selecting a base image $\mathcal{X}_k \in \mathcal{C}$ which is then intersected with M randomly selected images $\mathcal{Y}_k = \{\mathcal{Y}_{ki} \in \mathcal{C} | i \in \{1, .., M\}\}$. Like in the previous section, we use the Sauvola and Pietikäinen algorithm [20] to remove illumination and degradation artifacts from the images, so $\mathcal{X}_k$ and $\mathcal{Y}_k$ images are binary. The intersection image $\mathcal{I}_{ki}$ between $\mathcal{X}_k$ and $\mathcal{Y}_{ki}$ is computed as

$$\mathcal{I}_{ki} = \mathbf{G}_\sigma(\mathcal{X}_k) \cdot \mathbf{G}_\sigma(\mathcal{W}(\mathcal{Y}_{ki}, \mathbf{p}_{ki})), \tag{14}$$

where $\mathcal{W}(\mathbf{I}, \mathbf{p})$ applies the affine transform $\mathbf{p}$ to the image $\mathbf{I}$, $\mathbf{p}_{ki}$ are the parameters of the transform that align $\mathcal{Y}_{ki}$ to $\mathcal{X}_k$, and $\mathbf{G}_{\sigma=1}$ is a Gaussian

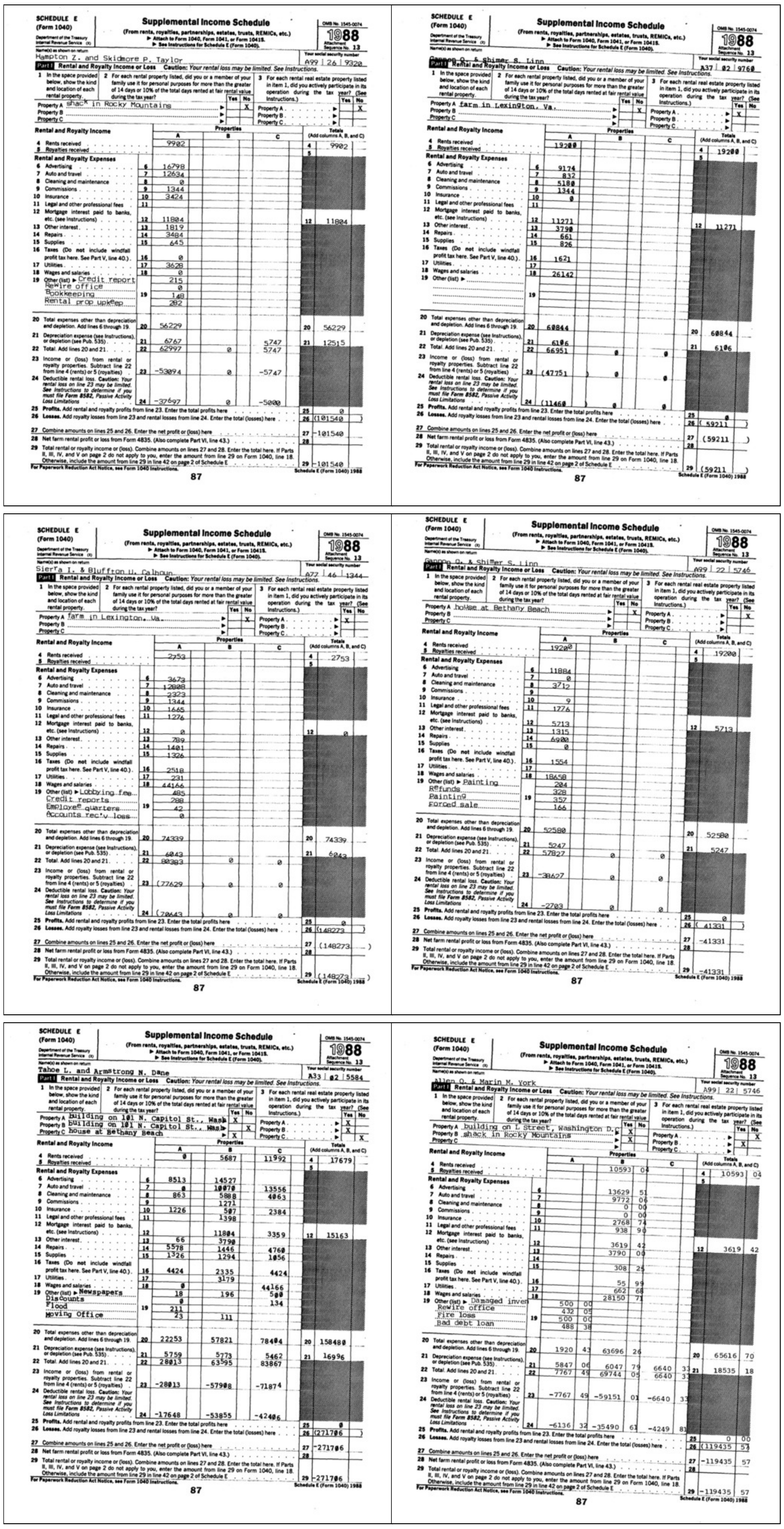
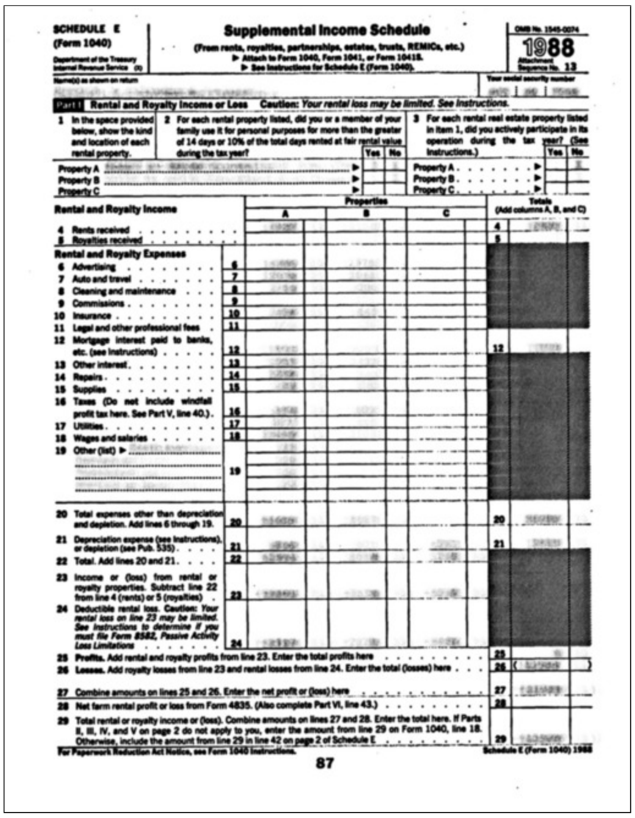


Figure B.2: Sample training documents for a particular class



SCHEDULE E (Form 1040)
Department of the Treasury
Internal Revenue Service (X)

Supplemental Income Schedule
(From rents, royalties, partnerships, estates, trusts, REMICs, etc.)
▶ Attach to Form 1040, Form 1041, or Form 1041S.
▶ See Instructions for Schedule E (Form 1040).

OMB No. 1545-0074
1988
Attachment Sequence No. 13

Name(s) as shown on return
Your social security number

Part I Rental and Royalty Income or Loss **Caution:** *Your rental loss may be limited. See Instructions.*

1 In the space provided below, show the kind and location of each rental property.	2 For each rental property listed, did you or a member of your family use it for personal purposes for more than the greater of 14 days or 10% of the total days rented at fair rental value during the tax year?	Yes	No	3 For each rental real estate property listed in item 1, did you actively participate in its operation during the tax year? (See Instructions.)	Yes	No
Property A	▶			Property A ▶		
Property B	▶			Property B ▶		
Property C	▶			Property C ▶		

		Properties A	Properties B	Properties C	Totals (Add columns A, B, and C)
Rental and Royalty Income					
4 Rents received					4
5 Royalties received					5
Rental and Royalty Expenses					
6 Advertising	6				
7 Auto and travel	7				
8 Cleaning and maintenance	8				
9 Commissions	9				
10 Insurance	10				
11 Legal and other professional fees	11				
12 Mortgage interest paid to banks, etc. (see Instructions)	12				12
13 Other interest	13				
14 Repairs	14				
15 Supplies	15				
16 Taxes (Do not include windfall profit tax here. See Part V, line 40.)	16				
17 Utilities	17				
18 Wages and salaries	18				
19 Other (list) ▶	19				
20 Total expenses other than depreciation and depletion. Add lines 6 through 19.	20				20
21 Depreciation expense (see Instructions), or depletion (see Pub. 535)	21				21
22 Total. Add lines 20 and 21	22				
23 Income or (loss) from rental or royalty properties. Subtract line 22 from line 4 (rents) or 5 (royalties)	23				
24 Deductible rental loss. Caution: *Your rental loss on line 23 may be limited. See Instructions to determine if you must file Form 8582, Passive Activity Loss Limitations*	24				
25 **Profits.** Add rental and royalty profits from line 23. Enter the total profits here					25
26 **Losses.** Add royalty losses from line 23 and rental losses from line 24. Enter the total (losses) here					26 ( )
27 Combine amounts on lines 25 and 26. Enter the net profit or (loss) here					27
28 Net farm rental profit or loss from Form 4835. (Also complete Part VI, line 43.)					28
29 Total rental or royalty income or (loss). Combine amounts on lines 27 and 28. Enter the total here. If Parts II, III, IV, and V on page 2 do not apply to you, enter the amount from line 29 on Form 1040, line 18. Otherwise, include the amount from line 29 in line 42 on page 2 of Schedule E					29

For Paperwork Reduction Act Notice, see Form 1040 Instructions.
Schedule E (Form 1040) 1988
87

Figure B.3: Probabilistic model template obtained from the document set shown in Fig. B.2.

filter used to smooth the binary images in order to account for small binarization and misalignment errors. Then, we compute the average between the intersection images as

$$\widetilde{\mathcal{P}_k} = \frac{1}{M}\sum_{i}^{M}\mathcal{I}_{ki}$$

and obtain the partial model $\mathcal{P}_k$ by applying a pixel-wise sigmoid function over $\widetilde{\mathcal{P}_k}$ to increase its contrast. Instead of just accumulating all the intersection among images to generate the model, we apply a sigmoid function which is a non-linear transform that removes low probability contributions that appear in regions where dynamic structures are commonly present. Therefore, these dynamic structures won't receive enough support in the final static model $\mathcal{M}$.

Finally, the static document model is obtained by accumulating all the partial models $\mathcal{P}$ by

$$\widetilde{\mathcal{M}} = \frac{1}{N}\sum_{k}^{N}\mathcal{W}\left(\mathcal{P}_k, \overline{\mathbf{p}_k}\right), \tag{15}$$

where $\overline{\mathbf{p}_k}$ are the parameters that align the k-th partial model to $\mathcal{P}_1$. The selection of $\mathcal{P}_1$ as reference frame is arbitrary and we can select any other partial model as reference. Alternatively, we could also estimate the location of the reference frame by averaging the transforms that relate all partial models. However, the main goal is just to accumulate all partial models into

Table B.1: Contingency matrix

		True condition		
		Positive	Negative	Total
Predicted	Positive	466 (TP)	12 (FP)	478
	Negative	17 (FN)	–	17
	Total	483	–	

the final model, so the reference frame used is not important. Like with the partial models, the static document model $\mathcal{M}$ is obtained applying a pixel-wide sigmoid function over $\widetilde{\mathcal{M}}$.

B.4.2 *Dynamic Elements Detection*

Once we have generated the static model $\mathcal{M}$ for the collection $\mathcal{C}$, we can remove the static parts of the image $\mathcal{Q} \in \mathcal{C}$ by

$$\mathcal{S} = \mathcal{Q} \cdot \mathbf{D}(\mathcal{W}(\mathcal{M}, \widetilde{\mathbf{p}})), \tag{16}$$

where $\mathbf{D}(\cdot)$ is a dilation operator of 5×5 used to widen the wrapped model and $\widetilde{\mathbf{p}}$ are the parameters of the affine transform that aligns the model $\mathcal{M}$ to the query image $\mathcal{Q}$. The contrast of the resulting image is improved by applying a pixel-wise sigmoid function over $\mathcal{I}$.

In order to find the dynamic element regions, we binarize $\mathcal{S}$ by simply applying a low threshold (e.g. activate the pixels that have a probability above 0.25) and merging the detected regions by applying a morphological opening with a rectangular structuring element. The regions which does not have enough support, have unlikely shapes (e.g. are too thin) or have highly intersect with another region are filtered out.

B.5 EXPERIMENTAL RESULTS

In order to carry out our experiments we used the NIST Structured Tax Forms Dataset (SPDB2) [209]. We have selected a subset of 15 document classes, which are the ones that we have enough document image samples of the same type. For each of those 15 classes, 24 images where selected to build the probabilistic templates in the offline stages. A single example for each class is then used as testing image to assess the quality of the content extraction.

First, we present in Fig. B.4 some qualitative results. We show three different test images along with the produced probability maps for the variable field extraction. Here darker values indicate a higher pixel probability of being a variable field and brighter values indicate a higher probability of being static content. Pixels belonging to the static part are almost imperceptible here. We also show the final segmentations of variable content, and we can appreciate that some false alarms appear either due to binarization noise (which in some sense is a variable element) and to highly textured zones.

In order to quantitatively evaluate our system, we manually groundtruthed all the variable fields in the 15 test images to see at which extend the proposed methodology is able to locate such fields and at which extend it also delivers false alarms. In total the ground-truth is composed of 483 fields

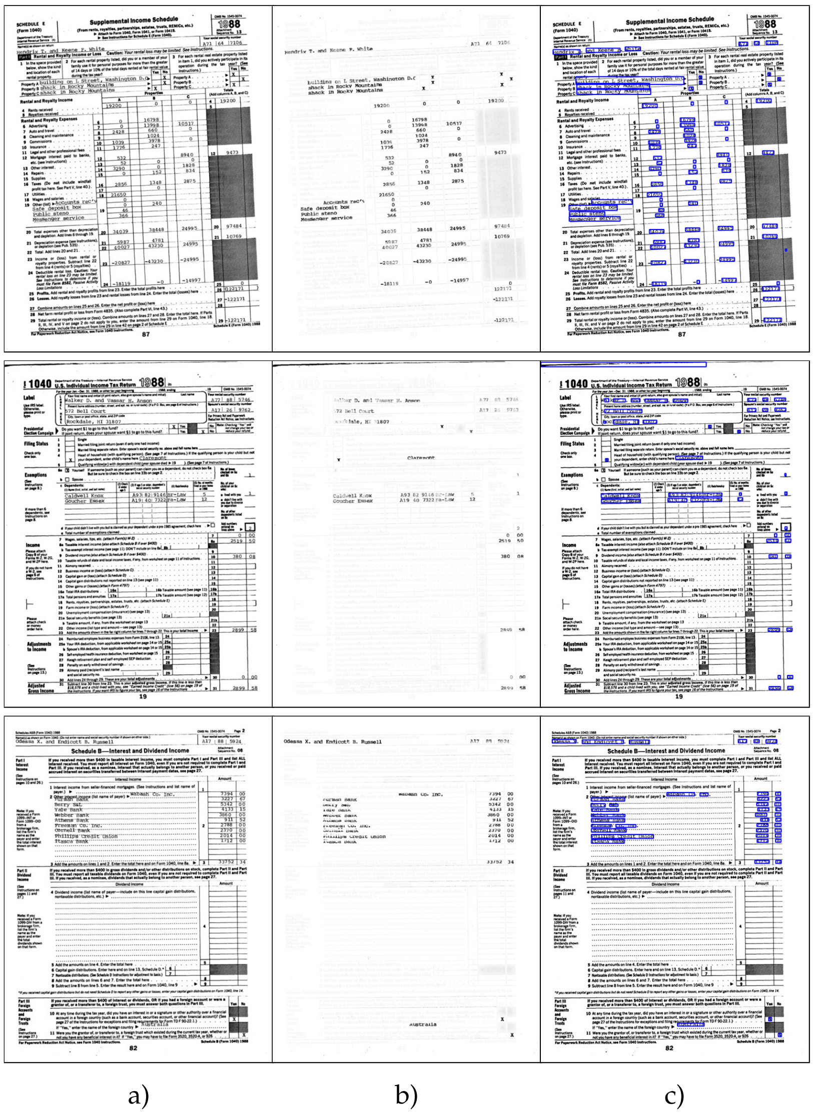


Figure B.4: Detection results. a) Test images, b) Probability maps of the variable elements after aligning the test documents wih their respective templates, c) Final bounding-box extractions.

to extract. We can see the results in form of a contingency matrix in Table B.1. Here, the true positive condition are all the variable fields, whereas the true negative condition is not quantized since it is the rest of the document (static content). When we run our method, we end up correctly retrieving 466 of the variable fields (true positives) while missing 17 of them (false negatives) and providing 12 erroneous segmentations in zones where there are no variable elements (false positives). In summary, the proposed method yielded a precision of 97.49% and a recall of 96.48% in the task of retrieving variable fields.

B.6 CONCLUSIONS

In this appendix, we have presented an automatic method for separating static and dynamic content that appear in administrative document images. Such method allows to define probabilistic templates aimed at locating locations of relevant fields without the need of an expert user intervention. We have validated our approach on the public NIST Structured Tax Forms Dataset, and we plan to make further tests on other administrative documentation such as invoices, contracts, id cards, etc.

Future work

Any iterative alignment algorithm is subject to local minima. By adding attributes to the document images (e. g.pixels belonging to an structural element, to a typewritten text, to a handwritten text, . . .), we can increase the robustness of the alignment algorithm. With attributes, we can ensure that pixels will only be attracted to pixels of a compatible attribute (i. e.lines with lines, text with text). They also help us define the likelihood that an element is will keep the same position between multiple documents (e. g.handwritten elements are likely to move).

BIBLIOGRAPHY

[1] L. Vincent, “Google book search: Document understanding on a massive scale,” in *Proceedings of the International Conference on Documents Analysis and Recognition*, vol. 2, 2007, pp. 819–823. doi:10.1109/ICDAR.2007.4377029

[2] V. Romero, A. Fornés, N. Serrano, J. Sánchez, A. Toselli, V. Frinken, E. Vidal, and J. Lladós, “The esposalles database: An ancient marriage license corpus for off-line handwriting recognition,” *Pattern Recognition*, vol. 46, no. 6, pp. 1658–1669, 2013. doi:10.1016/j.patcog.2012.11.024

[3] D. Fernández-Mota, J. Almazán, N. Cirera, A. Fornés, and J. Lladós, “Bh2m: The barcelona historical, handwritten marriages database,” in *Proceedings of the International Conference on Pattern Recognition*, 2014, pp. 256–261. doi:10.1109/ICPR.2014.53

[4] J. Brodman, “Unequal in charity? women and hospitals in medieval catalonia,” *Medieval Encounters*, vol. 12, no. 1, pp. 26–36, January 2006. doi:10.1163/157006706777502550

[5] J. Gómez and M. Verdú, “Network theory may explain the vulnerability of medieval human settlements to the black death pandemic,” *Scientific Reports*, vol. 7, no. 1, March 2017. doi:10.1038/srep43467

[6] T. Smith, R. Reynolds, T. Peterson, and J. Lawrimore, “Improvements to noaa’s historical merged land–ocean surface temperature analysis (1880–2006),” *Journal of Climate*, vol. 21, no. 10, pp. 2283–2296, May 2008. doi:10.1175/2007JCLI2100.1

[7] R. Hedjam and M. Cheriet, “Historical document image restoration using multispectral imaging system,” *Pattern Recognition*, vol. 46, no. 8, pp. 2297–2312, August 2013. doi:10.1016/j.patcog.2012.12.015

[8] S. He and L. Schomaker, “Deepotsu: Document enhancement and binarization using iterative deep learning,” *Pattern Recognition*, vol. 91, pp. 379–390, July 2019. doi:10.1016/j.patcog.2019.01.025

[9] S. Mao, A. Rosenfeld, and T. Kanungo, “Document structure analysis algorithms: a literature survey,” in *Document Recognition and Retrieval X*, vol. 5010, 2003, pp. 197–207. doi:10.1117/12.476326

[10] L. Likforman-Sulem, A. Zahour, and B. Taconet, “Text line segmentation of historical documents: A survey,” *International Journal on Document Analysis and Recognition*, vol. 9, no. 2–4, pp. 123–138, April 2007. doi:10.1007/s10032-006-0023-z

[11] L. Chen, S. Wang, W. Fan, J. Sun, and S. Naoi, “Beyond human recognition: A cnn-based framework for handwritten character recognition,” in *IAPR Asian Conference on Pattern Recognition (ACPR)*, 2015, pp. 695–699. doi:10.1109/ACPR.2015.7486592

[12] J. Puigcerver, “Are multidimensional recurrent layers really necessary for handwritten text recognition?” in *Proceedings of the International Conference on Documents Analysis and Recognition*, vol. 1, 2017, pp. 67–72. doi:10.1109/ICDAR.2017.20

[13] V. Frinken, A. Fischer, R. Manmatha, and H. Bunke, "A novel word spotting method based on recurrent neural networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 2, pp. 211–224, February 2012. doi:10.1109/TPAMI.2011.113

[14] J. Toledo, M. Carbonell, A. Fornés, and J. Lladós, "Information extraction from historical handwritten document images with a context-aware neural model," *Pattern Recognition*, vol. 86, pp. 27–36, February 2019. doi:10.1016/j.patcog.2018.08.020

[15] A. Gordo, A. Fornés, and E. Valveny, "Writer identification in handwritten musical scores with bags of notes," *Pattern Recognition*, vol. 46, no. 5, pp. 1337–1345, May 2013. doi:10.1016/j.patcog.2012.10.013

[16] R. Manmatha, C. Han, and E. Riseman, "Word spotting: a new approach to indexing handwriting," in *Proceedings of the conference on Computer Vision and Pattern Recognition*, 1996, pp. 631–637. doi:10.1109/CVPR.1996.517139

[17] A. Fischer, A. Keller, V. Frinken, and H. Bunke, "Lexicon-free handwritten word spotting using character HMMs," *Pattern Recognition Letters*, vol. 33, no. 7, pp. 934–942, May 2012. doi:10.1016/j.patrec.2011.09.009

[18] D. Aldavert, M. Rusiñol, R. Toledo, and J. Lladós, "A study of bag-of-visual-words representations for handwritten keyword spottings," *International Journal on Document Analysis and Recognition*, vol. 18, no. 3, pp. 223–234, September 2015. doi:10.1007/s10032-006-0027-8

[19] A. Giacometti, A. Campagnolo, L. MacDonald, S. Mahony, S. Robson, T. Weyrich, M. Terras, and A. Gibson, "The value of critical destruction: Evaluating multispectral image processing methods for the analysis of primary historical texts," *Digital Scholarship in the Humanities*, vol. 32, no. 1, pp. 101–122, April 2017. doi:10.1093/llc/fqv036

[20] J. Sauvola and M. Pietikäinen, "Adaptive document image binarization," *Pattern Recognition*, vol. 33, no. 2, pp. 225–236, February 2000. doi:10.1016/S0031-3203(99)00055-2

[21] B. Gatos, I. Pratikakis, and S. Perantonis, "Adaptive degraded document image binarization," *Pattern Recognition*, vol. 39, no. 3, pp. 317–327, March 2006. doi:10.1016/j.patcog.2005.09.010

[22] ——, "Improved document image binarization by using a combination of multiple binarization techniques and adapted edge information," in *Proceedings of the International Conference on Pattern Recognition*, 2008, pp. 1–4. doi:10.1109/ICPR.2008.4761534

[23] A. Giotis, G. Sfikas, B. Gatos, and C. Nikou, "A survey of document image word spotting techniques," *Pattern Recognition*, vol. 68, no. 8, pp. 310–332, August 2017. doi:10.1016/j.patcog.2017.02.023

[24] F. Chen, L. Wilcox, and D. Bloomberg, "Word spotting in scanned images using hidden markov mode," in *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, vol. 5, 1993, pp. 1–4. doi:10.1109/ICASSP.1993.319732

[25] S. Bera, A. Chakrabarti, S. Lahiri, E. Barney Smith, and R. Sarkar, "Normalization of unconstrained handwritten words in terms of slope

and slant correction," *Pattern Recognition Letters*, vol. 128, pp. 488–495, December 2019. doi:10.1016/j.patrec.2019.10.025

[26] J. Lladós, M. Rusiñol, A. Fornés, D. Fernández, and A. Dutta, "On the influence of word representations for handwritten word spotting in historical documents," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 26, no. 5, pp. 1 263 002.1–1 263 002.25, August 2012. doi:10.1142/S0218001412630025

[27] R. Plamondon and S. Srihari, "Online and off-line handwriting recognition: a comprehensive survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 1, pp. 63–84, January 2000. doi:10.1109/34.824821

[28] S. Madhvanath and V. Govindaraju, "The role of holistic paradigms in handwritten word recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 2, pp. 149–164, February 2001. doi:10.1109/34.908966

[29] A. Kołcz, J. Alspector, M. Augusteijn, R. Carlson, and G. Popescu, "A line-oriented approach to word spotting in handwritten documents," *Pattern Analysis and Applications*, vol. 3, no. 2, pp. 153–168, June 2000. doi:10.1007/s100440070020

[30] T. Rath and R. Manmatha, "Word image matching using dynamic time warping," in *Proceedings of the conference on Computer Vision and Pattern Recognition*, 2003, pp. 521–527. doi:10.1109/CVPR.2003.1211511

[31] ——, "Word spotting for historical documents," *International Journal on Document Analysis and Recognition*, vol. 9, no. 2–4, pp. 139–152, April 2007. doi:10.1007/s10032-006-0027-8

[32] T. Rath, R. Manmatha, and V. Lavrenko, "A search engine for historical manuscript images," in *Proceedings of the Special Interest Group on Information Retrieval (SIGIR)*, 2004, pp. 369–376. doi:10.1145/1008992.1009056

[33] T. Adamek, N. O'Connor, and A. Smeaton, "Word matching using single closed contours for indexing handwritten historical documents," *International Journal on Document Analysis and Recognition*, vol. 9, no. 2–4, pp. 153–165, April 2007. doi:10.1007/s10032-006-0024-y

[34] K. Khurshid, C. Faureb, and N. Vincent, "Word spotting in historical printed documents using shape and sequence comparisons," *Pattern Recognition*, vol. 45, no. 7, pp. 2598–2609, July 2012. doi:10.1016/j.patcog.2011.10.013

[35] A. Papandreou, B. Gatos, and G. Louloudis, "An adaptive zoning technique for efficient word retrieval using dynamic time warping," in *Proceedings of the International Conference on Digital Access to Textual Cultural Heritage*, 2014, pp. 147–152. doi:10.1145/2595188.2595218

[36] J. Rodríguez-Serrano and F. Perronnin, "Handwritten word-spotting using hidden Markov models and universal vocabularies," *Pattern Recognition*, vol. 42, no. 9, pp. 2106–2116, September 2009. doi:10.1016/j.patcog.2009.02.005

[37] J. Rodríguez-Serrano, F. Perronnin, G. Sánchez, and J. Lladós, "Unsupervised writer adaptation of whole-word HMMs with application to word-spotting," *Pattern Recognition Letters*, vol. 31, no. 8, pp. 742–749, June 2010. doi:10.1016/j.patrec.2010.01.007

[38] J. Rodriguez-Serrano and F. Perronnin, "A model-based sequence similarity with application to handwritten word-spotting," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 11, pp. 2108–2120, November 2012. doi:10.1109/TPAMI.2012.25

[39] L. Rothacker, M. Rusiñol, and G. Fink, "Bag-of-features hmms for segmentation-free word spotting in handwritten documents," in *Proceedings of the International Conference on Documents Analysis and Recognition*, 2013, pp. 1305–1309. doi:10.1109/ICDAR.2013.264

[40] J. Lladós and G. Sánchez, "Indexing historical documents by word shape signatures," in *Proceedings of the International Conference on Documents Analysis and Recognition*, 2007, pp. 362–366. doi:10.1109/ICDAR.2007.143

[41] D. Fernández, J. Lladós, and A. Fornés, "Handwritten word spotting in old manuscript images using a pseudo-structural descriptor organized in a hash structure," in *Proceedings of the European Conference on Computer Vision*, ser. Lectures Notes in Computer Science, 2011, vol. 6669, pp. 628–635.

[42] T. Konidaris, B. Gatos, K. Ntzios, I. Pratikakis, S. Theodoridis, and S. Perantonis, "Keyword-guided word spotting in historical printed documents using synthetic data and user feedback," *International Journal on Document Analysis and Recognition*, vol. 9, no. 2–4, pp. 167–177, April 2007. doi:10.1007/s10032-007-0042-4

[43] S. Impedovo, F. Mangini, and G. Pirlo, "A new adaptive zoning technique for handwritten digit recognition," in *Proceedings of the International Conference on Image Analysis and Processing*, 2013, pp. 91–100. doi:10.1007/978-3-642-41181-6_10

[44] A. Kovalchuk, L. Wolf, and N. Dershowitz, "A simple and fast word spotting method," in *Proceedings of the International Conference on Frontiers in Handwriting Recognition*, 2014, pp. 3–8. doi:10.1109/ICFHR.2014.9

[45] T. van der Zant, L. Shoemaker, and K. Haak, "Handwritten-word spotting using biologically inspired features," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 11, pp. 1945–1957, November 2008. doi:10.1109/TPAMI.2008.144

[46] S. Srihari and G. Ball, "Language independent word spotting in scanned documents," in *Digital Libraries: Universal and Ubiquitous Access to Information*, ser. Lectures Notes in Computer Science, 2008, vol. 5362, pp. 134–143.

[47] J. Rodríguez-Serrano and F. Perronnin, "Local gradient histogram features for word spotting in unconstrained handwritten documents," in *Proceedings of the International Conference on Frontiers in Handwriting Recognition*, 2008, pp. 7–12.

[48] D. Lowe, “Distinctive image features from scale-invariant keypoints,” *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, November 2004. doi:10.1023/B:VISI.0000029664.99615.94

[49] M. Rusiñol and J. Lladós, “Word and symbol spotting using spatial organization of local descriptors,” in *IAPR Workshop on Document Analysis Systems (DAS)*, 2008, pp. 489–496. doi:10.1109/DAS.2008.24

[50] Y. Leydier, A. Ouji, F. LeBourgeois, and H. Emptoz, “Towards an omnilingual word retrieval system for ancient manuscripts,” *Pattern Recognition*, vol. 42, no. 9, pp. 2089–2105, September 2009. doi:10.1016/j.patcog.2009.01.026

[51] X. Zhang and C. Tan, “Segmentation-free keyword spotting for handwritten documents based on heat kernel signature,” in *Proceedings of the International Conference on Documents Analysis and Recognition*, 2013, pp. 827–831. doi:10.1109/ICDAR.2013.169

[52] P. Wang, V. Eglin, C. Largeron, J. Lladós, A. Fornés, and C. Garcia, “A novel learning-free word spotting approach based on graph representation,” in *IAPR Workshop on Document Analysis Systems (DAS)*, 2014, pp. 207–211. doi:10.1109/DAS.2014.46

[53] J. Sivic and A. Zisserman, “Video google: A text retrieval approach to object matching in videos,” in *Proceedings of the International Conference on Computer Vision*, 2003, pp. 1470–1477. doi:10.1109/ICCV.2003.1238663

[54] G. Csurka, C. Dance, L. Fan, J. Willamowski, and C. Bray, “Visual categorization with bags of keypoints,” in *Proceeding of the European Conference on Computer Vision Workshops*, 2004, pp. 1–22.

[55] E. Ataer and P. Duygulu, “Matching ottoman words: an image retrieval approach to historical document indexing,” in *Proceedings of the International Conference on Image and Video Retrieval*, 2007, pp. 341–347. doi:10.1145/1282280.1282332

[56] P. Sankar, C. Jawahar, and R. Manmatha, “Nearest neighbor based collection ocr,” in *IAPR Workshop on Document Analysis Systems (DAS)*, 2010, pp. 207–214. doi:10.1145/1815330.1815357

[57] M. Rusiñol, D. Aldavert, R. Toledo, and J. Lladós, “Browsing heterogeneous document collections by a segmentation-free word spotting method,” in *Proceedings of the International Conference on Documents Analysis and Recognition*, 2011, pp. 63–67. doi:10.1109/ICDAR.2011.22

[58] R. Shekhar and C. Jawahar, “Word image retrieval using bag of visual words,” in *IAPR Workshop on Document Analysis Systems (DAS)*, 2012, pp. 297–301. doi:10.1109/DAS.2012.96

[59] V. Dovgalecs, A. Burnett, P. Tranouez, S. Nicolas, and L. Heutte, “Spot it! finding words and patterns in historical documents,” in *Proceedings of the International Conference on Documents Analysis and Recognition*, 2013, pp. 1039–1043. doi:10.1109/ICDAR.2013.208

[60] P. Sankar, R. Manmatha, and C. Jawahar, “Large scale document image retrieval by automatic word annotation,” *International Journal on Document Analysis and Recognition*, vol. 17, no. 1, pp. 1–17, March 2014. doi:10.1007/s10032-013-0207-2

[61] M. Rusiñol, D. Aldavert, R. Toledo, and J. Lladós, "Efficient segmentation-free keyword spotting in historical document collections," *Pattern Recognition*, vol. 48, no. 2, pp. 545–555, February 2015. doi:10.1016/j.patcog.2014.08.021

[62] D. Aldavert, M. Rusiñol, R. Toledo, and J. Lladós, "Integrating visual and textual cues for query-by-string word spotting," in *Proceedings of the International Conference on Documents Analysis and Recognition*, 2013, pp. 511–515. doi:10.1109/ICDAR.2013.108

[63] R. Manmatha and J. Rothfeder, "A scale space approach for automatically segmenting words from historical handwritten documents," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 8, pp. 1212–1225, August 2005. doi:10.1109/TPAMI.2005.150

[64] E. Nowak, F. Jurie, and B. Triggs, "Sampling strategies for bag-of-features image classification," in *Proceedings of the European Conference on Computer Vision*, 2006, pp. 490–503. doi:10.1007/11744085_38

[65] J. Almazán, A. Gordo, A. Fornés, and E. Valveny, "Efficient exemplar word spotting," in *Proceedings of the British Machine Vision Conference*, 2012, pp. 67.1–67.11. doi:10.5244/C.26.67

[66] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proceedings of the conference on Computer Vision and Pattern Recognition*, 2005, pp. 886–893. doi:10.1109/CVPR.2005.177

[67] Q. Zhu, M.-C. Yeh, K.-T. Cheng, and S. Avidan, "Fast human detection using a cascade of histograms of oriented gradients," in *Proceedings of the conference on Computer Vision and Pattern Recognition*, vol. 2, 2006, pp. 1491–1498. doi:10.1109/CVPR.2006.119

[68] B. Fulkerson, A. Vedaldi, and S. Soatto, "Localizing objects with smart dictionaries," in *Proceedings of the European Conference on Computer Vision*, ser. Lecture Notes in Computer Science, vol. 5302, 2008, pp. 179–192. doi:10.1007/978-3-540-88682-2_15

[69] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong, "Locality-constrained linear coding for image classification," in *Proceedings of the conference on Computer Vision and Pattern Recognition*, 2010, pp. 3360–3367. doi:10.1109/CVPR.2010.5540018

[70] K. Chatfield, V. Lempitsky, A. Vedaldi, and A. Zisserman, "The devil is in the details: An evaluation of recent feature encoding methods," in *Proceedings of the British Machine Vision Conference*, 2011, pp. 76.1–76.12. doi:10.5244/C.25.76

[71] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories," in *Proceedings of the conference on Computer Vision and Pattern Recognition*, 2006, pp. 2169–2178. doi:10.1109/CVPR.2006.68

[72] F. Perronnin, J. Sanchez, and T. Mensink, "Improving the fisher kernel for large-scale image classification," in *Proceedings of the European Conference on Computer Vision*, ser. Lecture Notes in Computer Science, vol. 6314, 2010, pp. 143–156. doi:10.1007/978-3-642-15561-1_11

[73] R. Cinbis, J. Verbeek, and C. Schmid, “Image categorization using fisher kernels of non-iid image models,” in *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2012, pp. 2184–2191. doi:10.1109/CVPR.2012.6247926

[74] I. Pratikakis, K. Zagoris, B. Gatos, G. Louloudis, and N. Stamatopoulos, “ICFHR 2014 competition on handwritten keyword spotting (H-KWS 2014),” in *Proceedings of the International Conference on Frontiers in Handwriting Recognition*, 2014, pp. 814–819. doi:10.1109/ICFHR.2014.142

[75] J. Rothfeder, S. Feng, and T. Rath, “Using corner feature correspondences to rank word images by similarity,” in *Proceeding of the Computer Vision and Pattern Recognition Workshops*, vol. 3, 2003, pp. 30–30. doi:10.1109/CVPRW.2003.10021

[76] N. Howe, “Part-structured inkball models for one-shot handwritten word spotting,” in *Proceedings of the International Conference on Documents Analysis and Recognition*, 2013, pp. 582–586. doi:10.1109/ICDAR.2013.121

[77] N. Howe, T. Rath, and R. Manmatha, “Boosted decision trees for word recognition in handwritten document retrieval,” in *Proceedings of the Special Interest Group on Information Retrieval (SIGIR)*, 2005, pp. 377–383. doi:10.1145/1076034.1076099

[78] Y. Liang, M. Fairhurst, and R. Guest, “A synthesised word approach to word retrieval in handwritten documents,” *Pattern Recognition*, vol. 45, no. 12, pp. 4224–4236, December 2012. doi:10.1016/j.patcog.2012.05.024

[79] J. Almazán, A. Gordo, A. Fornés, and E. Valveny, “Handwritten word spotting with corrected attributes,” in *Proceedings of the International Conference on Computer Vision*, December 2013, pp. 1017–1024. doi:10.1109/ICCV.2013.130

[80] K. Terasawa and Y. Tanaka, “Slit style HOG feature for document image word spotting,” in *Proceedings of the International Conference on Documents Analysis and Recognition*, 2009, pp. 116–120. doi:0.1109/ICDAR.2009.118

[81] G. Louloudis, B. Gatos, I. Pratikakis, and C. Halatsis, “Text line and word segmentation of handwritten documents,” *Pattern Recognition*, vol. 42, no. 12, pp. 3169–3183, December 2009. doi:10.1016/j.patcog.2008.12.016

[82] Y. Leydier, F. Lebourgeois, and H. Emptoz, “Text search for medieval manuscript images,” *Pattern Recognition*, vol. 40, no. 12, pp. 3552–3567, December 2007. doi:10.1016/j.patcog.2007.04.024

[83] B. Gatos and I. Pratikakis, “Segmentation-free word spotting in historical printed documents,” in *Proceedings of the International Conference on Documents Analysis and Recognition*, 2009, pp. 271–275. doi:10.1109/ICDAR.2009.236

[84] G. Salton and C. Buckley, “Term-weighting approaches in automatic text retrieval,” *Information Processing & Management*, vol. 24, no. 5, pp. 513–523, 1988. doi:10.1016/0306-4573(88)90021-0

[85] S. Deerwester, S. Dumais, G. Furnas, T. Landauer, and R. Harshman, “Indexing by latent semantic analysis,” *Journal of the American Society for Information Science and Technology*, vol. 41, no. 6, pp. 391–407, September 1990. doi:10.1002/(SICI)1097-4571(199009)41:6<391::AID-ASI1>3.0.CO;2-9

[86] A. Levey and M. Lindenbaum, “Sequential karhunen-loeve basis extraction and its application to images,” *IEEE Transactions on Image Processing*, vol. 9, no. 8, pp. 1371–1374, August 2000. doi:10.1109/83.855432

[87] H. Jégou, M. Douze, and C. Schmid, “Product quantization for nearest neighbor search,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 1, pp. 117–128, January 2011. doi:10.1109/TPAMI.2010.57

[88] J. Almazán, D. Fernández, A. Fornés, J. Lladós, and E. Valveny, “A coarse-to-fine approach for handwritten word spotting in large scale historical documents collection,” in *Proceedings of the International Conference on Frontiers in Handwriting Recognition*, 2012, pp. 455–460. doi:10.1109/ICFHR.2012.151

[89] J. Edwards, Y. Teh, D. Forsyth, R. Bock, M. Maire, and G. Vesom, “Making latin manuscripts searchable using gHMM’s,” in *Proceedings of the Advances in Neural Information Processing Systems*, 2004, pp. 385–392.

[90] S. Marinai, E. Marino, and G. Soda, “Font adaptive word indexing of modern printed documents,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 8, pp. 1187–1199, August 2006. doi:10.1109/TPAMI.2006.162

[91] J. Rodríguez-Serrano and F. Perronnin, “Synthesizing queries for handwritten word image retrieval,” *Pattern Recognition*, vol. 45, no. 9, pp. 3270–3276, September 2012. doi:10.1016/j.patcog.2012.02.015

[92] J. Almazán, A. Gordo, A. Fornés, and E. Valveny, “Word spotting and recognition with embedded attributes,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 12, pp. 2552–2566, July 2014. doi:10.1109/TPAMI.2014.2339814

[93] A. Black and P. Taylor, “The Festival speech synthesis system: System documentation,” Human Communciation Research Centre, University of Edinburgh, Scotland, UK, Tech. Rep. HCRC/TR-83, 1997.

[94] H. Hermansky, “Perceptual linear predictive (PLP) analysis of speech,” *The Journal of the Acoustical Society of America*, vol. 87, no. 4, pp. 1738–1752, April 1990. doi:10.1121/1.399423

[95] H. Hermansky and N. Morgan, “RASTA processing of speech,” *IEEE Transactions on Speech and Audio Processing*, vol. 2, no. 4, pp. 578–589, October 1994. doi:10.1109/89.326616

[96] V. Lavrenko, T. Rath, and R. Manmatha, “Holistic word recognition for handwritten historical documents,” in *Proceedings of IEEE International Workshop on Document Image Analysis for Libraries*, 2004, pp. 278–287. doi:10.1109/DIAL.2004.1263256

[97] R. Manmatha, C. Han, E. Riseman, and W. Croft, “Indexing handwriting using word matching,” in *Proceedings of the ACM International Conference on Digital Libraries*, 1996, pp. 151–159. doi:10.1145/226931.226960

[98] J. Almazán, A. Gordo, A. Fornés, and E. Valveny, “Segmentation-free word spotting with exemplar SVMs,” *Pattern Recognition*, vol. 47, no. 12, pp. 3967–3978, December 2014. doi:10.1016/j.patcog.2014.06.005

[99] L. Rothacker, S. Sudholt, E. Rusakov, M. Kasperidus, and G. Fink, “Word hypotheses for segmentation-free word spotting in historic document images,” in *Proceedings of the International Conference on Documents Analysis and Recognition*, 2017, pp. 1174–1179. doi:10.1109/ICDAR.2017.194

[100] S. Sudholt and G. Fink, “PHOCNet: A deep convolutional neural network for word spotting in handwritten documents,” in *Proceedings of the International Conference on Frontiers in Handwriting Recognition*, 2016, pp. 277–282. doi:10.1109/ICFHR.2016.0060

[101] L. Rothacker and G. Fink, “Segmentation-free query-by-string word spotting with bag-of-features HMMs,” in *Proceedings of the International Conference on Documents Analysis and Recognition*, 2015, pp. 661–665. doi:10.1109/ICDAR.2015.7333844

[102] S. Sudholt and G. Fink, “Evaluating word string embeddings and loss functions for CNN-based word spotting,” in *Proceedings of the International Conference on Documents Analysis and Recognition*, 2017, pp. 493–498. doi:10.1109/ICDAR.2017.87

[103] M. Rusiñol and J. Lladós, “Boosting the handwritten word spotting experience by including the user in the loop,” *Pattern Recognition*, vol. 47, no. 3, pp. 1063–1072, March 2014. doi:10.1016/j.patcog.2013.07.008

[104] P. Krishnan and C. Jawahar, “Matching handwritten document images,” in *Proceeding of the European Conference on Computer Vision Workshops*, 2016, pp. 766–782. doi:10.1007/978-3-319-46448-0_46

[105] S. Sudholt and G. Fink, “A modified isomap approach to manifold learning in word spotting,” in *Proceedings of the German Conference on Pattern Recognition*, 2015, pp. 529–539. doi:10.1007/978-3-319-24947-6_44

[106] N. Gurjar, S. Sudholt, and G. Fink, “Learning deep representations for word spotting under weak supervision,” in *IAPR Workshop on Document Analysis Systems (DAS)*, 2018, pp. 7–12. doi:10.1109/DAS.2018.35

[107] M. Jones and D. Mewhort, “Case-sensitive letter and bigram frequency counts from large-scale english corpora,” *Behavior Research Methods, Instruments and Computers*, vol. 36, no. 3, pp. 388–396, August 2004. doi:10.3758/BF03195586

[108] B. Leibe, A. Leonardis, and B. Schiele, “Robust object detection with interleaved categorization and segmentation,” *International Journal of Computer Vision*, vol. 77, no. 1–3, pp. 259–289, May 2008. doi:10.1007/s11263-007-0095-3

[109] L. Gómez and D. Karatzas, “Textproposals: A text-specific selective search algorithm for word spotting in the wild,” *Pattern Recognition*, vol. 70, no. Supplement C, pp. 60–74, October 2017. doi:10.1016/j.patcog.2017.04.027

[110] A. Vedaldi and A. Zisserman, "Efficient additive kernels via explicit feature maps," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 3, pp. 480–492, March 2012. doi:10.1109/TPAMI.2011.153

[111] O. Chum, "Low dimensional explicit feature maps," in *Proceedings of the International Conference on Computer Vision*, 2015, pp. 4077–4085. doi:10.1109/ICCV.2015.464

[112] F. Moosmann, E. Nowak, and F. Jurie, "Randomized clustering forests for image classification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 3, pp. 1632–1646, March 2008. doi:10.1109/TPAMI.2007.70822

[113] F. Perronnin and C. Dance, "Fisher kernels on visual vocabularies for image categorization," in *Proceedings of the conference on Computer Vision and Pattern Recognition*, 2007, pp. 1–8. doi:10.1109/CVPR.2007.383266

[114] H. Jegou, F. Perronnin, M. Douze, J. Sanchez, P. Perez, and C. Schmid, "Aggregating local image descriptors into compact codes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 9, pp. 1704–1716, September 2012. doi:10.1109/TPAMI.2011.235

[115] D. Nister and H. Stewenius, "Scalable recognition with a vocabulary tree," in *Proceedings of the conference on Computer Vision and Pattern Recognition*, vol. 2, 2006, pp. 2161–2168. doi:10.1109/CVPR.2006.264

[116] J. Platt, "Probabilistic outputs for support vector machines and comparison to regularized likelihood methods," in *Advances In Large-Margin Classifiers*, ser. Neural Information Processing. Cambridge, MA, 1999, vol. 10, pp. 61–74.

[117] A. Alahi, R. Ortiz, and P. Vandergheynst, "Freak: Fast retina keypoint," in *Proceedings of the conference on Computer Vision and Pattern Recognition*, 2012, pp. 510–517. doi:10.1109/CVPR.2012.6247715

[118] G. Csurka and F. Perronnin, "An efficient approach to semantic segmentation," *International Journal of Computer Vision*, vol. 95, no. 2, pp. 198–212, November 2011. doi:10.1007/s11263-010-0344-8

[119] X. He, R. Zemel, and D. Ray, "Learning and incorporating top-down cues in image segmentation," in *Proceedings of the European Conference on Computer Vision*, ser. Lectures Notes in Computer Science, 2006, vol. 3951, pp. 338–351.

[120] B. Fulkerson, A. Vedaldi, and S. Soatto, "Class segmentation and object localization with superpixel neighborhoods," in *Proceedings of the International Conference on Computer Vision*, 2009, pp. 670–677. doi:10.1109/ICCV.2009.5459175

[121] J. Carreira, F. Li, and C. Sminchisescu, "Object recognition by sequential figure-ground ranking," *International Journal of Computer Vision*, vol. 98, no. 3, pp. 243–262, July 2012. doi:10.1007/s11263-011-0507-2

[122] J. Carreira, R. Caseiro, J. Batista, and C. Sminchisescu, "Semantic segmentation with second-order pooling," in *Proceedings of the European Conference on Computer Vision*, ser. Lecture Notes in Computer Science, vol. 7578, 2012, pp. 430–443. doi:10.1007/978-3-642-33786-4_32

[123] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, "Slic superpixels compared to state-of-the-art superpixel methods," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 11, pp. 2274–2282, November 2012. doi:10.1109/TPAMI.2012.120

[124] M. Van den Bergh, X. Boix, G. Roig, and L. Van Gool, "Seeds: Superpixels extracted via energy-driven sampling," *International Journal of Computer Vision*, vol. 111, no. 3, pp. 298–314, February 2015. doi:10.1007/s11263-014-0744-2

[125] T. Huang, G. Yang, and G. Tang, "A fast two-dimensional median filtering algorithm," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 27, no. 1, pp. 13–18, February 1979. doi:10.1109/TASSP.1979.1163188

[126] S. Perreault and P. Hebert, "Median filtering in constant time," *IEEE Transactions on Image Processing*, vol. 16, no. 9, pp. 2389–2394, September 2007. doi:10.1109/TIP.2007.902329

[127] M. Sizintsev, K. Derpanis, and A. Hogue, "Histogram-based search: A comparative study," in *Proceedings of the conference on Computer Vision and Pattern Recognition*, 2008, pp. 1–8. doi:10.1109/CVPR.2008.4587654

[128] F. Porikli, "Integral histogram: A fast way to extract histograms in cartesian spaces," in *Proceedings of the conference on Computer Vision and Pattern Recognition*, vol. 1, 2005, pp. 829–836. doi:10.1109/CVPR.2005.188

[129] Y. Wei and L. Tao, "Efficient histogram-based sliding window," in *Proceedings of the conference on Computer Vision and Pattern Recognition*, 2010, pp. 3003–3010. doi:10.1109/CVPR.2010.5540049

[130] D. Picard and P.-H. Gosselin, "Efficient image signatures and similarities using tensor products of local descriptors," *Computer Vision and Image Understanding*, vol. 117, no. 6, pp. 680–687, June 2013. doi:10.1016/j.cviu.2013.02.004

[131] X. Zhou, K. Yu, T. Zhang, and T. Huang, "Image classification using super-vector coding of local image descriptors," in *Proceedings of the European Conference on Computer Vision*, ser. Lecture Notes in Computer Science, vol. 6315, 2010, pp. 141–154. doi:10.1007/978-3-642-15555-0_11

[132] N. Slonim and N. Tishby, "Agglomerative information bottleneck," in *Proceedings of the Advances in Neural Information Processing Systems*, 1999, pp. 617–623.

[133] P. Bennett, "Using asymmetric distributions to improve text classifier probability estimates," in *Proceedings of the Special Interest Group on Information Retrieval (SIGIR)*, 2003, pp. 111–118. doi:10.1145/860435.860457

[134] A. Niculescu-Mizil and R. Caruana, "Predicting good probabilities with supervised learning," in *Proceedings of the International Conference on Machine Learning*, 2005, pp. 625–632. doi:10.1145/1102351.1102430

[135] B. Zadrozny and C. Elkan, "Obtaining calibrated probability estimates from decision trees and naive bayesian classifiers," in *Proceedings of the International Conference on Machine Learning*, 2001, pp. 609–616.

[136] S. Shalev-Shwartz, Y. Singer, and N. Srebro, "Pegasos: Primal estimated sub-gradient solver for svm," in *Proceedings of the Advances in Neural Information Processing Systems*, 2007, pp. 807–814.

[137] A. Bordes, L. Bottou, and P. Gallinari, "Sgd-qn: Careful quasi-newton stochastic gradient descent," *Journal of Machine Learning Research*, vol. 10, pp. 1737–1754, December 2009.

[138] C.-J. Lin, R. Weng, and S. Keerthi, "Trust region newton method for large-scale logistic regression," *Journal of Machine Learning Research*, vol. 9, pp. 627–650, June 2008.

[139] J. Zhang, M. Marszałek, S. Lazebnik, and C. Schmid, "Local features and kernels for classification of texture and object categories: A comprehensive study," *International Journal of Computer Vision*, vol. 73, no. 2, pp. 213–238, June 2006. doi:10.1007/s11263-006-9794-4

[140] S. Maji, A. Berg, and J. Malik, "Efficient classification for additive kernel svms," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 1, pp. 66–77, January 2013. doi:10.1109/TPAMI.2012.62

[141] D. Aldavert, A. Ramisa, R. Toledo, and R. López de Mántaras, "Efficient object pixel-level categorization using bag of features," in *Advances in Visual Computing*, ser. Lectures Notes in Computer Science, 2009, vol. 5875, pp. 44–54.

[142] ——, "Fast and robust object segmentation with the integral linear classifier," in *Proceedings of the conference on Computer Vision and Pattern Recognition*, 2010, pp. 1046–1053. doi:10.1109/CVPR.2010.5540098

[143] F. Crow, "Summed-area tables for texture mapping," in *Proceedings of the International Conference and Exhibition on Computer Graphics and Interactive Techniques (SIGGRAPH)*, 1984, pp. 207–212. doi:10.1145/800031.808600

[144] P. Viola and M. Jones, "Robust real-time face detection," *International Journal of Computer Vision*, vol. 57, no. 2, pp. 137–154, May 2004. doi:10.1023/B:VISI.0000013087.49260.fb

[145] J. Gemert, J.-M. Geusebroek, C. Veenman, and A. Smeulders, "Kernel codebooks for scene categorization," in *Proceedings of the European Conference on Computer Vision*, ser. ECCV' 08, 2008, pp. 696–709. doi:10.1007/978-3-540-88690-7_52

[146] J. Sanchez, F. Perronnin, and T. de Campos, "Modeling the spatial layout of images beyond spatial pyramids," *Pattern Recognition Letters*, vol. 33, no. 16, pp. 2216–2223, December 2012. doi:10.1016/j.patrec.2012.07.019

[147] N. Elfiky, J. Gonzalez, and F. Roca, "Compact and adaptive spatial pyramids for scene recognition," *Image and Vision Computing*, vol. 30, no. 8, pp. 492–500, August 2012. doi:10.1016/j.imavis.2012.04.002

[148] Y. Jia, C. Huang, and T. Darrell, "Beyond spatial pyramids: Receptive field learning for pooled image features," in *Proceedings of the conference on Computer Vision and Pattern Recognition*, 2012, pp. 3370–3377. doi:10.1109/CVPR.2012.6248076

[149] G.-X. Yuan, K.-W. Chang, C.-J. Hsieh, and C.-J. Lin, "A comparison of optimization methods and software for large-scale l1-regularized linear classification," *Journal of Machine Learning Research*, vol. 11, pp. 3183–3234, December 2010.

[150] H.-F. Yu, F.-L. Huang, and C.-J. Lin, "Dual coordinate descent methods for logistic regression and maximum entropy models," *Machine Learning*, vol. 85, no. 1–2, pp. 41–75, October 2011. doi:10.1007/s10994-010-5221-8

[151] C.-J. Lin, R. Weng, and S. Keerthi, "Trust region newton methods for large-scale logistic regression," in *Proceedings of the International Conference on Machine Learning*, 2007, pp. 561–568. doi:10.1145/1273496.1273567

[152] A. Bordes, L. Bottou, P. Gallinari, J. Chang, and S. Smith, "Erratum: Sgdqn is less careful than expected," *Journal of Machine Learning Research*, vol. 11, pp. 2229–2240, August 2010.

[153] S. Shalev-Shwartz, Y. Singer, N. Srebro, and A. Cotter, "Pegasos: Primal estimated sub-gradient solver for svm," *Mathematical Programming*, vol. 127, no. 1, pp. 3–30, March 2011. doi:10.1007/s10107-010-0420-4

[154] I. Pratikakis, K. Zagoris, B. Gatos, J. Puigcerver, A. Toselli, and E. Vidal, "Icfhr2016 handwritten keyword spotting competition (h-kws 2016)," in *Proceedings of the International Conference on Frontiers in Handwriting Recognition*, 2016, pp. 613–618. doi:10.1109/ICFHR.2016.0117

[155] C. Park and H. Park, "A comparison of generalized linear discriminant analysis algorithms," *Pattern Recognition*, vol. 41, no. 3, pp. 1083–1097, 2008. doi:10.1016/j.patcog.2007.07.022

[156] T. Wilkinson and A. Brun, "Semantic and verbatim word spotting using deep neural networks," in *Proceedings of the International Conference on Frontiers in Handwriting Recognition*, 2016, pp. 307–312. doi:10.1109/ICFHR.2016.0065

[157] S. Sudholt and G. Fink, "Attribute cnns for word spotting in handwritten documents," *International Journal on Document Analysis and Recognition*, vol. 21, pp. 199–218, September 2018. doi:10.1007/s10032-018-0295-0

[158] P. Krishnan and C. Jawahar, "Hwnet v2: an efficient word image representation for handwritten documents," *International Journal on Document Analysis and Recognition*, vol. 22, pp. 387–405, December 2019. doi:10.1007/s10032-019-00336-x

[159] A. Silberpfennig, L. Wolf, N. Dershowitz, S. Bhagesh, and B. Chaudhuri, "Improving ocr for an under-resourced script using unsupervised word-spotting," in *Proceedings of the International Conference on Documents Analysis and Recognition*, 2015, pp. 706–710. doi:10.1109/ICDAR.2015.7333853

[160] G. Retsinas, G. Louloudis, N. Stamatopoulos, and B. Gatos, "Keyword spotting in handwritten documents using projections of oriented gradients," in *IAPR Workshop on Document Analysis Systems (DAS)*, 2016, pp. 411–416. doi:10.1109/DAS.2016.61

[161] G. Sfikas, G. Retsinas, and B. Gatos, "Zoning aggregated hypercolumns for keyword spotting," in *Proceedings of the International Conference on Frontiers in Handwriting Recognition*, 2016, pp. 283–288. doi:10.1109/ICFHR.2016.0061

[162] G. Retsinas, G. Louloudis, N. Stamatopoulos, and B. Gatos, "Efficient learning-free keyword spotting," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 7, pp. 1587–1600, 2019. doi:10.1109/TPAMI.2018.2845880

[163] M. Stauffer, P. Maergner, A. Fischer, and K. Riesen, "Cross-evaluation of graph-based keyword spotting in handwritten historical documents," in *Graph-Based Representations in Pattern Recognition*, 2019, pp. 45–55. doi:10.1007/978-3-030-20081-7_5

[164] M. Ameri, M. Stauffer, K. Riesen, T. Bui, and A. Fischer, "Graph-based keyword spotting in historical manuscripts using hausdorff edit distance," *Pattern Recognition Letters*, vol. 121, pp. 61–67, 2019. doi:10.1016/j.patrec.2018.05.003

[165] M. Stauffer, A. Fischer, and K. Riesen, "Keyword spotting in historical handwritten documents based on graph matching," *Pattern Recognition*, vol. 81, pp. 240–253, 2018. doi:10.1016/j.patcog.2018.04.001

[166] P. Riba, A. Fischer, J. Lladós, and A. Fornés, "Learning graph edit distance by graph neural networks," arXiv, Tech. Rep. 2008.07641, 2020.

[167] V. Balntas, E. Riba, D. Ponsa, and K. Mikolajczyk, "Learning local feature descriptors with triplets and shallow convolutional neural networks," in *Proceedings of the British Machine Vision Conference (BMVC)*, 2016, pp. 119.1–119.11. doi:10.5244/C.30.119

[168] G. Retsinas, B. Gatos, N. Stamatopoulos, and G. Louloudis, "Isolated character recognition using projections of oriented gradients," in *Proceedings of the International Conference on Documents Analysis and Recognition*, 2015, pp. 336–340. doi:10.1109/ICDAR.2015.7333779

[169] D. Kobak, W. Brendel, C. Constantinidis, C. Feierstein, A. Kepecs, Z. Mainen, X.-L. Qi, R. Romo, N. Uchida, and C. Machens, "Demixed principal component analysis of neural population data," *eLife*, vol. 5, p. e10989, April 2016. doi:10.7554/eLife.10989

[170] L. Lei, W. Xiao-dan, L. Xi, and S. Ya-fei, "Hierarchical error-correcting output codes based on svdd," *Pattern Analysis and Applications*, vol. 19, no. 1, pp. 163–171, February 2016. doi:10.1007/s10044-015-0455-5

[171] M. Brown and W. Seales, "Image restoration of arbitrarily warped documents," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 10, pp. 1295–1306, October 2004. doi:10.1109/TPAMI.2004.87

[172] A. Graves, M. Liwicki, S. Fernández, R. Bertolami, H. Bunke, and J. Schmidhuber, "A novel connectionist system for unconstrained handwriting recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 5, pp. 855–868, May 2009. doi:10.1109/TPAMI.2008.137

[173] V. Shapiro, G. Gluhchev, and V. Sgurev, "Handwritten document image segmentation and analysis," *Pattern Recognition Letters*, vol. 14, no. 1, pp. 71–78, January 1993. doi:10.1016/0167-8655(93)90134-Y

[174] A. Antonacopoulos and D. Karatzas, "Document image analysis for world war ii personal records," in *Proceedings of IEEE International Workshop on Document Image Analysis for Libraries*, 2004, pp. 336–341. doi:10.1109/DIAL.2004.1263263

[175] L. Likforman-Sulem, A. Hanimyan, and C. Faure, "A hough based algorithm for extracting text lines in handwritten documents," in *Proceedings of the International Conference on Documents Analysis and Recognition*, vol. 2, 1995, pp. 774–777. doi:10.1109/ICDAR.1995.602017

[176] F. Cruz and O. Terrades, "Handwritten line detection via an em algorithm," in *Proceedings of the International Conference on Documents Analysis and Recognition*, 2013, pp. 718–722. doi:10.1109/ICDAR.2013.147

[177] R. Saabni, A. Asi, and J. El-Sana, "Text line extraction for historical document images," *Pattern Recognition Letters*, vol. 35, pp. 23–33, January 2014. doi:10.1016/j.patrec.2013.07.007

[178] D. Fernández-Mota, J. Lladós, and A. Fornés, "A graph-based approach for segmenting touching lines in historical handwritten documents," *International Journal on Document Analysis and Recognition*, vol. 17, no. 3, pp. 293–312, September 2014. doi:10.1007/s10032-014-0220-0

[179] J. Pastor-Pellicer, M. Afzal, M. Liwicki, and M. Castro-Bleda, "Complete system for text line extraction using convolutional neural networks and watershed transform," in *IAPR Workshop on Document Analysis Systems (DAS)*, 2016, pp. 30–35. doi:10.1109/DAS.2016.58

[180] B. Moysset, C. Kermorvant, and C. Wolf, "Full-page text recognition: Learning where to start and when to stop," in *Proceedings of the International Conference on Documents Analysis and Recognition*, 2017, pp. 871–876. doi:10.1109/ICDAR.2017.147

[181] L. Likforman-Sulem, A. Zahour, and B. Taconet, "Text line segmentation of historical documents: a survey," *International Journal on Document Analysis and Recognition*, vol. 9, no. 2, pp. 123–138, April 2007. doi:10.1007/s10032-006-0023-z

[182] W. Freeman and E. Adelson, "The design and use of steerable filters," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, no. 9, pp. 891–906, September 1991. doi:10.1109/34.93808

[183] L. van Vliet, I. Young, and P. Verbeek, "Recursive gaussian derivative filters," in *Proceedings of the International Conference on Pattern Recognition*, 1998, pp. 509–514. doi:10.1109/ICPR.1998.711192

[184] A. Bieniek and A. Moga, "An efficient watershed algorithm based on connected components," *Pattern Recognition*, vol. 33, no. 6, pp. 907–916, June 2000. doi:10.1016/S0031-3203(99)00154-5

[185] Q. Vo and G. Lee, "Dense prediction for text line segmentation in handwritten document images," in *Proceedings of the International Conference on Image Processing*, 2016, pp. 3264–3268. doi:10.1016/S0031-3203(99)00154-5

[186] S. Kahan, T. Pavlidis, and H. Baird, "On the recognition of printed characters of any font and size," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 9, no. 2, pp. 274–288, March 1987. doi:10.1109/TPAMI.1987.4767901

[187] A. Nicolaou and B. Gatos, "Handwritten text line segmentation by shredding text into its lines," in *Proceedings of the International Conference on Documents Analysis and Recognition*, 2009, pp. 626–630. doi:10.1109/ICDAR.2009.243

[188] B. Gatos, N. Stamatopoulos, G. Louloudis, G. Sfikas, G. Retsinas, V. Papavassiliou, F. Sunistira, and V. Katsouros, "Grpoly-db: An old greek polytonic document image database," in *Proceedings of the International Conference on Documents Analysis and Recognition*, 2015, pp. 646–650. doi:10.1109/ICDAR.2015.7333841

[189] G. Louloudis, B. Gatos, I. Pratikakis, and C. Halatsis, "Text line detection in handwritten documents," *Pattern Recognition*, vol. 41, no. 12, pp. 3758–3772, December 2008. doi:10.1016/j.patcog.2008.05.011

[190] B. Moysset and C. Kermorvant, "On the evaluation of handwritten text line detection algorithms," in *Proceedings of the International Conference on Documents Analysis and Recognition*, 2013, pp. 185–189. doi:10.1109/ICDAR.2013.44

[191] Z. Shi, S. Setlur, and V. Govindaraju, "A steerable directional local profile technique for extraction of handwritten arabic text lines," in *Proceedings of the International Conference on Documents Analysis and Recognition*, 2009, pp. 176–180. doi:10.1109/ICDAR.2009.79

[192] M. Diem, F. Kleber, S. Fiel, G. Tobias, and B. Gatos, "cbad: Icdar2017 competition on baseline detection," in *Proceedings of the International Conference on Documents Analysis and Recognition*, 2017, pp. 1355–1360. doi:10.1109/ICDAR.2017.222

[193] U.-V. Marti and H. Bunke, "The iam-database: an english sentence database for offline handwriting recognition," *International Journal on Document Analysis and Recognition*, vol. 5, no. 1, pp. 39–46, November 2002. doi:10.1007/s100320200071

[194] A. Fischer, E. Indermühle, H. Bunke, G. Viehhauser, and M. Stolz, "Ground truth creation for handwriting recognition in historical documents," in *IAPR Workshop on Document Analysis Systems (DAS)*, 2010, pp. 3–10. doi:10.1145/1815330.1815331

[195] B. Gatos, N. Stamatopoulos, and G. Louloudis, "Icdar2009 handwriting segmentation contest," *International Journal on Document Analysis and Recognition*, vol. 14, no. 1, pp. 25–33, March 2011. doi:ICDAR2009 Handwriting Segmentation Contest

[196] P. Héroux, S. Diana, A. Ribert, and E. Trupin, "Classification method study for automatic form class identification," in *Proceedings of the International Conference on Pattern Recognition*, 1998, pp. 926–928. doi:10.1109/ICPR.1998.711385

[197] N. Chen and D. Blostein, "A survey of document image classification: problem statement, classifier architecture and performance evaluation," *International Journal on Document Analysis and Recognition*, vol. 10, no. 1, pp. 1–16, June 2006. doi:10.1007/s10032-006-0020-2

[198] M. Rusiñol, V. Frinken, D. Karatzas, A. Bagdanov, and J. Lladós, "Multimodal page classification in administrative document image streams," *International Journal on Document Analysis and Recognition*, vol. 17, no. 4, pp. 331–341, December 2014. doi:10.1007/s10032-014-0225-8

[199] M. Rusiñol, D. Karatzas, A. Bagdanov, and J. Lladós, "Multipage document retrieval by textual and visual representations," in *Proceedings of the International Conference on Pattern Recognition*, 2012, pp. 521–524.

[200] P. Viola, J. Rinker, and M. Law, "Automatic fax routing," in *Proceedings of the International Worksop on Document Analysis Systems: Document Analysis Systems VI*, ser. Lectures Notes in Computer Science, 2004, vol. 3163, pp. 484–495.

[201] C. Peanho, H. Stagni, and F. da Silva, "Semantic information extraction from images of complex documents," *Applied Intelligence*, vol. 37, no. 5, pp. 543–557, December 2012. doi:10.1007/s10489-012-0348-x

[202] Y. Ishitani, "Model-based information extraction method tolerant of OCR errors for document images," in *Proceedings of the International Conference on Documents Analysis and Recognition*, 2001, pp. 908–915. doi:10.1109/ICDAR.2001.953918

[203] ——, "Model-based information extraction and its applications for document images," in *Proceedings of the Workshop on Document Layout Interpretation and its Applications*, 2001.

[204] M. Rusiñol, T. Benkhelfallah, and V. d'Andecy, "Field extraction from administrative documents by incremental structural templates," in *Proceedings of the International Conference on Documents Analysis and Recognition*, 2013, pp. 1100–1104. doi:10.1109/ICDAR.2013.223

[205] D. Schuster, K. Muthmann, D. Esser, A. Schill, M. Berger, C. Weidling, K. Aliyev, and A. Hofmeier, "Intellix – end-user trained information extraction for document archiving," in *Proceedings of the International Conference on Documents Analysis and Recognition*, 2013, pp. 101–105. doi:10.1109/ICDAR.2013.28

[206] K. Santosh and A. Belaïd, "Pattern-based approach to table extraction," in *Proceedings of the Iberian Conference on Pattern Recognition and Image Analysis*, 2013, pp. 766–773. doi:10.1007/978-3-642-38628-2_91

[207] B. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Proceedings of the International Joint Conference on Artificial Intelligence*, vol. 2, 1981, pp. 674–679.

[208] S. Baker and I. Matthews, "Lucas-kanade 20 years on: An unifying framework," *International Journal of Computer Vision*, vol. 56, no. 3, pp. 221–255, February 2004. doi:10.1023/B:VISI.0000011205.11775.fd

[209] D. Dimmick, M. Garris, and C. Wilson, "Structured forms database," National Institutte of Standards and Technology, Tech. Rep., 1991.

[210] I. Matthews and S. Baker, "Active appearance models revisited," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 135–164, November 2004. doi:10.1023/B:VISI.0000029666.37597.d3

[211] G. Evangelidis and E. Psarakis, "Parametric image alignment using enhanced correlation coefficient maximization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 10, pp. 1858–1865, October 2008. doi:10.1109/TPAMI.2008.113

[212] S. Lucey, R. Navarathna, A. Ashraf, and S. Sridharan, "Fourier lucas-kanade algorithm," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 6, pp. 1383–1396, June 2013. doi:10.1109/TPAMI.2012.220

[213] G. Tzimiropoulos, S. Zafeiriou, and M. Pantic, "Robust and efficient parametric face alignment," in *Proceedings of the International Conference on Computer Vision*, 2011, pp. 1847–1854. doi:10.1109/ICCV.2011.6126452

[214] E. Antonakos, J. Alabort-i Medina, G. Tzimiropoulos, and S. Zafeiriou, "Feature-based lucas-kanade and active appearance models," *IEEE Transactions on Image Processing*, vol. 24, no. 9, pp. 2617–2632, September 2015. doi:10.1109/TIP.2015.2431445

[215] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 9, no. 1, pp. 62–66, January 1979. doi:10.1109/TSMC.1979.4310076