Learning Representations for Word Images

Thesis submitted in partial fulfillment
of the requirements for the degree of

Doctor of Philosophy
in
Computer Science and Engineering

by

Praveen Krishnan
201150899
praveen.krishnan@research.iiit.ac.in

International Institute of Information Technology
(Deemed to be University)
Hyderabad - 500 032, INDIA
Nov 2020

Copyright © Praveen Krishnan, 2020
All Rights Reserved

International Institute of Information Technology

Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled "Learning Representations for Word Images" by Praveen Krishnan, has been carried out under my supervision and is not submitted elsewhere for a degree.

_____ _____

Date Adviser: C. V. Jawahar

To my sister,

Preeti

Acknowledgments

This is one of the last pieces of writing in my thesis and it's rightfully kept in the beginning because none of the following things would have been possible without these people and organizations, who supported me all through this journey.

First and foremost, I thank my adviser Prof. C.V. Jawahar for accepting me as his graduate student at CVIT, IIIT Hyderabad. CVIT lead me to a new world of learning and opportunities. As an adviser and a collaborator, he inculcated within me the right mindset in pursuing research by following its vigor, discipline, and patience. I am fortunate to learn under him. He was generous in providing his valuable time to discuss both the technical and the general aspects of academic life.

Secondly, I express my gratitude towards the research community which includes fellow lab-mates, researchers from other labs within IIIT, my thesis committee members, and the reviewers of my journal and conference proceedings. These people have given their sincere efforts to discuss, evaluate and provide critical feedback for my work. I am also fortunate to collaborate with Kartik, Minesh, Naveen, Ravi, Ajeet, Vijay, Anshuman, and Siddhant. Together as a team, we could do interesting works in our research domain.

I also express my deep gratitude to the funding agencies which supported me throughout my time at IIIT. I thank Tata Consultancy Services for their generous research fellowship, Amazon for Alexa Graduate Fellowship and IIIT for their institute fellowship program. I also acknowledge the travel grants received from Google and IIIT which made it possible for me to attend the conferences in our research domain.

My sincere thanks to the administrative staff at CVIT which includes Mr. Satyanarayana, (Late) Mr. Phaneendra, Siva, Silar, Ram, Sireesha, and Rohita. I also thank the administrative staff at IIIT which includes Ms. Prathima, Ms. Pushpalatha, Mr. Y Kishore, Mr. Appaji, Mr. Murthy, Mr. Satheesh, and Mr. Mahender. Thank you for supporting the needs of the research student and making the process smooth.

In my life journey, I have been fortunate to have great friends and my time at IIIT was no such exception. These people stood strong and supported me in all my ups and downs. Thank you Anand for being there all the time as a friend and mentor. I made many friends at IIIT. Friends like Aniket, Pritish, Aditya, Yasaswi, Nagendar, Deepayan, Natraj, Yashaswi, Saurabh Saini, Ajeet, Sourabh Daptardar, Rajvi, Swagatika, Aseem, Vikram, Udit, Avijit, Ajoy, Abhishek, Tejaswi, Parikshit, Rajan, and Nandini made my time at IIIT special. Together we juggled with our research problems while also bringing humor in daily activities, playing games, doing exercises, and making trips to the city. A special thanks to Thrupthi and Isha for our light daily conversations while sipping the green tea. A big thanks to the Malayali fraternity at IIIT incl. Devadath, Varun, Minesh, Jobin, Aquib, Mithun, Shajil, Jerin, Naveen, Vinitha, Sandhya, Litton, Ratheesh and others. We had great conversations on myriad topics such as politics, movies, food, and social issues. Thank you, guys. I would also never forget the wonderful faculty and friends for a life time that I got from my previous alma mater Amrita university.

Finally, I would like to express my sincere gratitude to my family who loved me unconditionally and allowed me to dream and pursue my chosen path. My thesis is also a small tribute to the teachers who taught me all this way.

Abstract

Reading and writing documents is one among the primary skills with which we gather and communicate information. With the emergence of artificial intelligence (AI), researchers are in constant pursuit to build intelligent algorithms that can bring our physical and digital worlds close to each other. One such important domain is document image analysis, where we delve into the problem of understanding content from scanned document image collections. Considering "words" as the basic unit in understanding a document, in this thesis, we address the problem of finding the best possible representation for word images.

Representation learning has been a key investigation for an AI problem. The primary goal of this thesis is to learn efficient representations for word images that encode its content. An ideal representation should be invariant to multiple fonts, handwritten styles and less sensitive to noise and distortions. In the past, representations have been handcrafted, specific to modalities (printed, handwritten), and sensitive to the complexities in handwriting in multi-writer scenarios. In this work, we choose the paradigm of learning from data using deep neural networks. We take our inspiration from the fact that given large amounts of annotated data, modern deep neural networks can inherently learn better representations. In this thesis, we also relax the need for large annotated datasets by heavily capitalizing on synthetically generated images. We also introduce a novel problem of learning semantic representation for word images which encodes the semantics of the word and reduces the vocabulary gap that exists between the query and the retrieved results.

The first contribution of this thesis is a simple technique to generate large amounts of synthetic data, useful for pre-training deep neural networks. This led to the creation of IIIT-HWS dataset which is now widely used in the document community. The other major contributions of this thesis are: (a) the design of a deep convolutional architecture (named as HWNet) for learning an efficient holistic representation for word images, (b) a joint embedding scheme to project words and textual strings onto a common subspace, and (c) a novel form of word image representation which respects the word form along with its

semantic meaning. The learned representations are evaluated under the tasks of word spotting and word recognition. We report state-of-the-art performance on popular datasets under both modern/historical and handwritten/printed document images while keeping the representation size compact in nature. Finally, in order to validate the proposed representations of this thesis, we present some interesting use cases such as (i) finding similarity between a pair of handwritten documents images, (ii) searching for keywords from online lecture videos, and (iii) building word retrieval system for Indic scripts.

Contents

Chapter Page

1 Introduction . . . 1
1.1 Motivation . . . 2
1.1.1 Scope . . . 3
1.2 Word Representation Learning . . . 5
1.2.1 Why Representation Learning? . . . 5
1.2.2 Problems of interest . . . 6
1.2.3 Challenges . . . 8
1.3 Contributions . . . 9
1.4 Publications . . . 11
1.5 Thesis Outline . . . 13

2 Background . . . 14
2.1 Classical Image Representations . . . 15
2.1.1 Bag of Visual Words (BoVW) Representation . . . 16
2.1.1.1 Detectors and Descriptors . . . 16
2.1.1.2 Visual Vocabulary . . . 17
2.1.1.3 Coding and Pooling . . . 18
2.1.2 Higher Order Representations . . . 18
2.1.2.1 Vector of Locally Aggregated Descriptors (VLAD) . . . 19
2.1.2.2 Fisher Vectors (FV) . . . 19
2.2 Neural Image Representations . . . 20
2.2.1 Convolutional Neural Networks . . . 21
2.2.1.1 Training . . . 24
2.2.1.2 Loss Functions . . . 25
2.2.1.3 Regularization . . . 26
2.2.2 ConvNet Architectures . . . 27
2.2.3 Transfer Learning . . . 30
2.3 Neural Word Embedding . . . 31
2.3.1 Language Modelling . . . 32
2.3.2 Neural Probabilistic Language Model . . . 33
2.3.3 C&W model . . . 34
2.3.4 Word2Vec . . . 35
2.3.5 Subword Models . . . 36
2.4 Image Retrieval . . . 38

2.4.1 Distance Measures . . . 38
2.4.2 Evaluation Measures . . . 40
2.5 Word Image Representation and Spotting . . . 40
2.5.1 Word Image Datasets . . . 42
2.5.2 Related Works . . . 43
2.5.2.1 Classical Representation . . . 43
2.5.2.2 Bag of Word Representation . . . 44
2.5.2.3 Learned Representations . . . 46
2.5.2.4 Segmentation-Free Approaches . . . 49
2.6 Summary . . . 50

3 HWNet: Word Image Representation . . . 51
3.1 Introduction . . . 52
3.2 Handwritten Synthetic Dataset . . . 54
3.2.1 Handwritten Font Rendering . . . 55
3.2.2 IIIT-HWS Dataset . . . 55
3.3 HWNet . . . 56
3.3.1 HWNet Baseline Architecture . . . 57
3.4 HWNet v2 . . . 58
3.4.1 Multi-Scale Training and ROI/TPP Pooling . . . 59
3.4.2 Data Augmentation and Elastic Distortion . . . 60
3.4.3 Curriculum Learning . . . 60
3.4.4 Transfer learning . . . 61
3.5 Visualizations . . . 61
3.6 Experiments . . . 64
3.6.1 Evaluation Protocol . . . 64
3.6.2 Ablation Studies . . . 64
3.6.3 Word Spotting Evaluation . . . 65
3.6.3.1 Architecture Evaluation . . . 65
3.6.3.2 State of the Art Comparison . . . 66
3.6.3.3 Segmentation-Free Word Spotting . . . 68
3.6.4 Transfer Learning . . . 70
3.6.5 t-SNE Embedding . . . 72
3.6.6 Compression of Representation . . . 73
3.6.7 Qualitative Results and Failure Scenarios . . . 74
3.6.8 Implementation Details . . . 75
3.7 Summary . . . 76

4 HWNet v3: A Joint Embedding Framework for Recognition and Retrieval of Handwritten Text 77
4.1 Introduction . . . 78
4.2 Related Works . . . 79
4.2.1 Deep Embedding . . . 79
4.2.2 Word Recognition . . . 80
4.3 HWNet Embedding . . . 81
4.3.1 Two-stage joint feature embedding . . . 82
4.3.2 Synthetic Attribute Embedding . . . 84

4.4 HWNet v3: End-2-End Embedding . . . 85
4.4.1 Variant I . . . 87
4.4.2 Variant II . . . 87
4.4.3 Training End-2-End Network . . . 88
4.4.4 Image and Text Representation . . . 88
4.5 Experiments . . . 89
4.5.1 Evaluation Measures . . . 89
4.5.2 Implementation Details . . . 90
4.5.3 Ablation Study . . . 90
4.5.4 Word Spotting Results . . . 92
4.5.5 Word Recognition Results . . . 95
4.5.6 Analysis on Embedding Size . . . 98
4.5.7 Effect of Pre-Training . . . 98
4.6 Visualization and Qualitative Results . . . 99
4.7 Summary . . . 101

5 Bringing Semantics into Word Image Representation . . . 103
5.1 Introduction . . . 104
5.1.1 Why Semantic Representation? . . . 106
5.1.2 Traditional Spotting vs. Semantic Spotting . . . 106
5.1.3 Contributions . . . 107
5.2 Related Works . . . 108
5.2.1 Textual Word Embedding . . . 108
5.2.2 Word Image Semantic Embedding . . . 109
5.3 Word Image Semantic Embedding . . . 109
5.3.1 Normalized Word Embedding . . . 110
5.3.2 Semantic Embedding . . . 111
5.3.2.1 Lexicon Based Recognition and Embedding . . . 112
5.3.2.2 Direct Embedding . . . 112
5.4 Experiments . . . 114
5.4.1 Evaluation Measures . . . 114
5.4.2 Implementation Details . . . 115
5.4.3 Normalized Spotting . . . 116
5.4.4 Semantic Spotting . . . 117
5.4.4.1 Ablation Study . . . 119
5.4.4.2 Qualitative Analysis . . . 120
5.5 Discussion . . . 121
5.6 Summary . . . 123

6 Use Cases of Word Image Representation . . . 124
6.1 Matching Handwritten Document Images . . . 125
6.1.1 Measure of Document Similarity (MODS) . . . 126
6.1.1.1 Document Segmentation . . . 127
6.1.1.2 Sum of Word Matches . . . 130
6.1.1.3 MODS matching . . . 130
6.1.2 Dataset and Evaluation Measures . . . 133

6.1.3 Results and Discussion . . . 135
6.1.4 Human Evaluations . . . 135
6.2 Word Spotting on Lecture Videos . . . 136
6.2.1 Datasets and Evaluation Protocol . . . 137
6.2.2 Results . . . 138
6.3 Word Retrieval on Indic Scripts . . . 139
6.3.1 Digital Library of India (DLI) . . . 140
6.3.2 Results . . . 140
6.3.2.1 Word Spotting using Lexical Representation . . . 141
6.3.2.2 Semantic Evaluation . . . 142
6.4 Summary . . . 143

7 Conclusion and Future Work . . . 144
7.1 Summary . . . 144
7.2 Future Directions . . . 145

Bibliography . . . 148

List of Figures

Figure Page

1.1 Sample document pages: (a) historical handwritten document written by George Washington [133], (b) a handwritten assignment written by a student in a course [90], (c) a video frame containing handwritten text on classroom board, and (d) a sample page taken from a printed book written in Hindi as part of DLI project [11]. 3

1.2 The top row shows two sample handwritten document images, while the bottom row shows few sample word images taken from the corpus containing these pages. 4

1.3 Illustration of the holistic word representation learning problem. (a) Word Image Representation in Lexical Space, (b) Joint Text and Image Embedding, and (c) Semantic Embedding. 7

1.4 Typical challenges which exists while learning of representation of word images: (a) multiple writer styles, (b) degradation and segmentation irregularities, and (c) extension to multiple scripts and languages. 8

2.1 Bag of Visual Words representation: The figure presents different stages in the computation of BOW for images. 16

2.2 Typical architecture of a convolutional neural network for image classification task along with filter visualizations. 21

2.3 Popular activation function used in a convolutional neural network. 22

2.4 (a) A typical convolutional layer, (b) max pooling layer. 23

2.5 (a) Basic inception module as introduced in [171], (b) Residual learning building block as presented in [69] 29

2.6 A typical convolutional neural network. The bottom axis shows the level of learned feature abstraction from a trained CNN network which enables the property of transfer learning. 30

2.7 Neural probabilistic language model. 33

2.8 Illustration of segmentation based word spotting task. Given a query word, we retrieve all relevant word images from a pre-segmented corpus of document images. Here each word image and the query is represented using an appropriate feature vector which is denoted by a histogram. 41

2.9 Sample word images taken from the popular datasets used in the document image community. (a) The IAM Handwriting Database, (b) George Washington (GW), and (c) Botany (top row) and Konzilsprotokolle (bottom row). 42

2.10 Evolution of word spotting methods from the perspective of different word image representation schemes. The evaluation is conducted on the IAM [109] dataset using mean Average Precision (mAP). 44
2.11 Bag of words based pipeline for learning word image representation. 46
2.12 Pyramid Histogram of Characters (PHOC) [9] 47
2.13 The word attribute embedding framework as proposed in [9]. Here, both the word images and its corresponding label are first embedded into an attribute space defined using PHOC and later projected onto a common subspace. 48

3.1 Sample nearest neighbor word images in the learned representation space. Here we show examples from historical datasets which contains degradation and irregular segmentation of words. One can also notice invariance of representation in terms of handwriting variations and noise in segmentation. 53
3.2 (a) Distribution of words in the IAM dataset. Here the 'x' axis corresponds to different words in the vocabulary which is sorted (in descending order) based on its frequency in the corpus. This is referred as 'Rank', while the 'y' axis shows the actual frequency in log units. (b) Sample word images from the IIIT-HWS dataset created as part of this work, to address the lack of training data for learning deep CNN networks. 54
3.3 Top two rows show the variations in handwritten images, the bottom two rows demonstrate the challenges of intra class variability in images across writers. 56
3.4 (a) HWNet v2 architecture which comprises of a deep CNN architecture using ResNet blocks along with a TPP pooling and fully connected layers, (b) Flowchart showing the transfer learning process where we first pre-train the network on synthetic data and later fine-tune it on real corpus. The features are extracted from the penultimate layer of the network. 57
3.5 (a-b) Multi-scale input, (c-d) region of interest pooling and, (e-f) temporal pyramid pooling shown at levels 2 and 3. 59
3.6 Data augmentation techniques: affine and elastic distortion. 61
3.7 Visualizations: (a) Layer 1 weights, (b) Four possible reconstructions [104] of sample word images shown in columns. These are reconstructed from the representation obtained from the penultimate layer, and (c) Visualization of the strongest activation [52] region of a particular neuron (each column refers to one neuron) of an intermediate convolutional layer. These regions are highlighted using a bounding box inside the word image. Here we notice that, in most of the cases, each neuron focus on detecting a semantically meaningful unit. 62
3.8 Bar plot showing the word spotting performance on IAM dataset with respect to different word lengths in the test dataset. Here, we have analysed word lengths ranging from (3,14). 69
3.9 Ablation study of word spotting in noisy word segmentation. Here we evaluate the word spotting results by perturbing the word segmentations of the test set. The perturbations are done randomly within an IoU range $(0.5, 1.0)$. 70
3.10 Graph analyzing the layer for efficient transfer learning. 72
3.11 t-SNE embedding of word image representation taken from the validation set from IAM dataset. 73
3.12 Evaluation of compression of learned representation using PCA on IAM and GW datasets. 74

3.13 Qualitative results of word spotting. In each row, the leftmost image is the query and remaining are the top retrieved word images based on features similarity. Here (a-d) refers to images taken from IAM, GW, Botany and Konzilsprotokolle datasets respectively. 75
3.14 Sample failure case images where the representation fails to match lexically correct neighbors. 76

4.1 The left part of the figure shows the motivation of HWNet features where we learn a style invariant representation. In this chapter, we extend our representation to textual space. Here we learn a common representation that embeds both word images and its corresponding text (as shown in the right figure) close to each other. 78
4.2 Overview of the two-stage joint feature embedding process. 82
4.3 Computation of PHOC representation using embedding functions ($\phi_{\mathcal{Y}}$, $\phi_{\mathcal{X}}$) for text and image respectively. 83
4.4 Synthetic attribute embedding. 85
4.5 The proposed variants of the HWNet v3 End-2-End embedding network for learning both image and textual embedding using multi-task loss function. (a) Variant I which was first proposed in [86], uses separate feature extraction channels for real and synthetic images, while (b) presents the Variant II of the End-2-End embedding network which allows better sharing of features among the real and synthetic image. 86
4.6 Effect on performance on varying HWNet v3 embedding size on the IAM dataset. The top figure shows the analysis of word spotting under both QBE and QBS setting. The bottom figure presents the word recognition CER and WER values. 97
4.7 tSNE image embedding 99
4.8 t-SNE common subspace embedding. The central plot shows a snapshot of limited number of word classes where each color denotes one word class. We have zoomed out two regions to show the projection of both word image and it's textual representation. Here the image is projected using colored round circles and the corresponding textual representation is shown using colored stars. 100
4.9 Qualitative results of word recognition on the IAM dataset using test lexicon. The top two rows show the successful cases, while the bottom row shows failure scenarios. Here the green box shows the text where the prediction matches the actual text. In failure cases, the red box shows the incorrect recognition, while the blue box shows the actual correct answer. 101
4.10 Qualitative results of query-by-string word spotting on (a) IAM, (b) GW, (c) Botany, (d) Konzilsprotokolle dataset respectively. The query is shown in the left most column and the word images shown on the right are the retrieved images in the ranked order. 102

5.1 Projection of word similarity scores onto visual and semantic axes as shown in the top and bottom locations respectively. Here the top axis uses word image representation [86] which only focuses on the visual space, while the bottom axis uses the proposed semantic representation which focuses on the meaning of the word. 105
5.2 The top row shows the nearest neighbors of a query "watched" in the traditional word spotting setting, while the middle row shows the word images that are related to query in terms of inflections or word morphology, while the bottom row shows the nearest neighbors which are either related in terms of inflections, or synonyms, or related in terms of semantics. 107

5.3 Intuition behind the semantic embedding process. Here each point refers to a word image embedding either shown in the lexical space (shown in left) or the semantic space (shown in right). We propose two schemes to embed word images from lexical to semantic space (a) Recognize+Embed, and (b) Direct Embedding. 111

5.4 Direct word image embedding architecture. Here, we use a semantic stream which provides the textual word embedding for the current word image being passed through the real stream. 113

5.5 (a-b) Rate of the performance change while increasing the lexicon size in lexicon based recognition and embedding for the IAM dataset. (a) shows semantic spotting performance in terms of mAP for exact evaluation, while the chart (b) reports the word and character error rates of the recognizer. (c) Qualitative results analogical tests. 120

5.6 Qualitative results of normalized and semantic word spotting. Note that, while showing the nearest neighbors, we have removed the consecutive similar word images to emphasize the distances among different lexical word images. 122

6.1 Given two document images $\mathcal{D}_i$ and $\mathcal{D}_j$, we are interested in computing a similarity score $\mathcal{S}(\mathcal{D}_i, \mathcal{D}_j)$ which is invariant to (i) writers, (ii) word flow across lines, (iii) spatial shifts, and (iv) paraphrasing. In this example, the highlighted lines from $\mathcal{D}_i$ and $\mathcal{D}_j$ have almost the same content but they widely differ in terms of the spatial arrangement of words. 126

6.2 MODS flowchart showing various modules used for the computation of similarity score between a pair of document images D_1 and D_2. 127

6.3 The segmentation proposals on a challenging document page from HW-1K dataset obtained from the segmentation algorithm. 129

6.4 A few major challenges of the matching process between a pair of documents $\mathcal{D}_1$ and $\mathcal{D}_2$. (i) Finding a unique match of each potential word, (ii) removal of stopwords, (iii) invariance to word overflow problems, and (iv) exploiting the loose ordering of words in matching. 131

6.5 Snapshot of document images taken from the HW-DocSim dataset, written by different writers. 134

6.6 Qualitative results of the MODS matching algorithm from the HW-DocSim dataset. Here we show two sample matching pairs in two columns. The top region is taken from the source and the bottom one is plagiarized. The highlighted words in rectangle have been correctly matched along with few words which remain undetected. 135

6.7 Sample frames from the LectureVideoDB datatset. Scenes cluttered with text and figures, black boards where text background contrast is poor, low resolution images and less legible handwriting make the text recognition harder in lecture videos. 137

6.8 Qualitative results of word spotting on LectureVideoDB dataset. Notice the robustness of the representation in-spite of wide diversity in the quality of images due to different writing mediums. 139

6.9 Qualitative results of query-by-string word spotting on Hindi, and Telugu datasets. The query is shown in the left most column and the word images shown on the right are the retrieved images in the ranked order. 141

6.10 Qualitative results of normalized and semantic word spotting on Hindi, and Telugu datasets. Note that, while showing the nearest neighbors, we have removed the consecutive similar word images to emphasize the distances among different lexical word images. 142

List of Tables

Table Page

2.1 The list of handwritten datasets used in this work. Here GW, Botany, and Konzilsprotokolle datasets are historical documents written primarily by a single author along with a few assistants(*). 42

2.2 Overview of major methods proposed in the literature, summarizing different types of word image representation schemes. 45

3.1 Summary of the HWNet v2 network configuration. The width, height, and number of channels of each convolution layer are shown in square brackets, with the number of layers that are stacked together. We present two variations in the network (as shown in the sixth column), using a single level ROI pooling or using temporal pyramid pooling (TPP) with three levels. 58

3.2 Ablation studies showing the effect of each of the enhancements to the baseline HWNet architecture on IAM dataset. 65

3.3 Comparative mAP evaluation of different deep networks with respect to the HWNet and HWNet v2 (TPP) network on IAM dataset. 66

3.4 Quantitative evaluation of word spotting on standard handwritten datasets in query-by-example setting. Here, results for DTW and FV are taken from [9], while all other related works are taken from their respective papers. 67

3.5 QBE mAP evaluation of HWNet v2 representation under segmentation-free scenario. Here we use the word proposals generated using the recent state of art method Ctrl-F-Mini [188]. As per the standard evaluation practice, we report the QBE mAP at overlap thresholds of 50% and 25%. 71

3.6 Evaluation of word spotting using mAP on IAM test dataset by training HWNet v2 with entire synthetic dataset while fine tuning on varying percentage of IAM training data. Here Train=0.0% refers only using synthetic data for training. 71

4.1 Ablation Study on the IAM dataset under two variants (column 1) of End-2-End embedding architectures, the choice of loss functions (column 2) and the choice of modality in each stream (column 3 and 4). 91

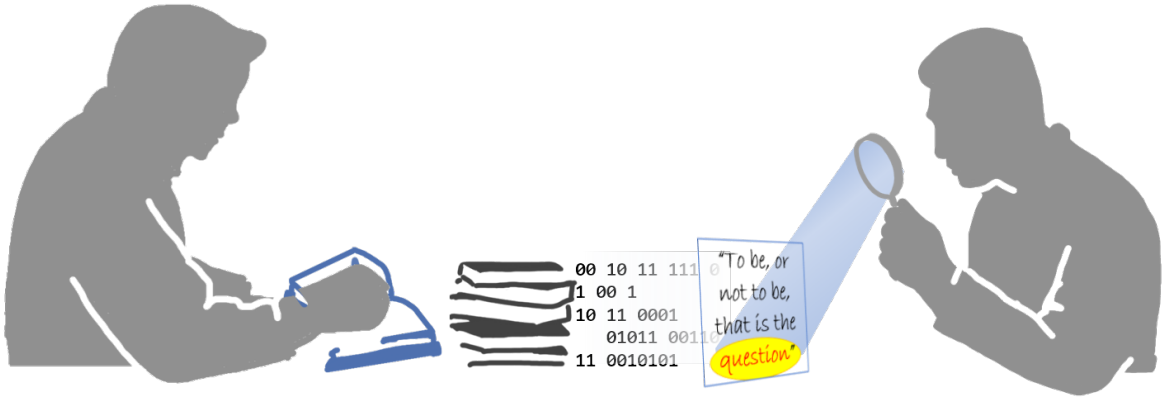
4.2 Quantitative evaluation of various word spotting methods on standard handwritten datasets. Here is the first block (rows 1-7) presents results from methods proposed in the literature. Note that in Triplet-CNN [186] we have taken the best results for IAM and GW across different word embedding used in the paper. The next three blocks of methods (rows 8-9, 10-13 and 14-17) present the different embedding schemes proposed in this chapter. 93
4.3 Word recognition results on IAM dataset under different settings of lexicon based evaluation for making the predictions. 96
4.4 Evaluation of word spotting and recognition on the IAM dataset by learning HWNet v3 representation with the entire synthetic dataset while fine-tuning on a varying percentage of IAM training data. Here Train=0.0% refers only using synthetic data for training. . . 98

5.1 Performance evaluation under normalized word spotting. Here we present embedding learned and evaluated in three different scenarios (Exact, Stem and Lemma). 116
5.2 Quantitative evaluation of word image semantic representation. Following shortened notation are used: word analogy (WA), query-by-example (QBE), query-by-string (QBS) and in-vocabulary QBS (In-QBS). Each of the performance measure vary between (0,1) where higher is the better. 117

6.1 Quantitative evaluation of various matching schemes on HW-DocSim dataset. We compare the performance of proposed MODS framework using CNN features over baseline methods such as NN, BOW, and embedded attributes proposed in [9]. 134
6.2 Details of the LectureVideoDB Dataset [42]. Here Type refers to the presentation/writing medium used by the instructor. 137
6.3 Word Spotting performance using end2end embedding architecture on the various splits of LectureVideoDB dataset. 138
6.4 The list of printed datasets used in this work. Here both Hindi and Telugu datasets are taken from Digital Library of India [11] corpus. 140
6.5 Quantitative evaluation of word spotting on printed datasets. 141
6.6 Semantic evaluation on printed datasets from English, Hindi and Telugu languages. . . 142

Chapter 1

Introduction



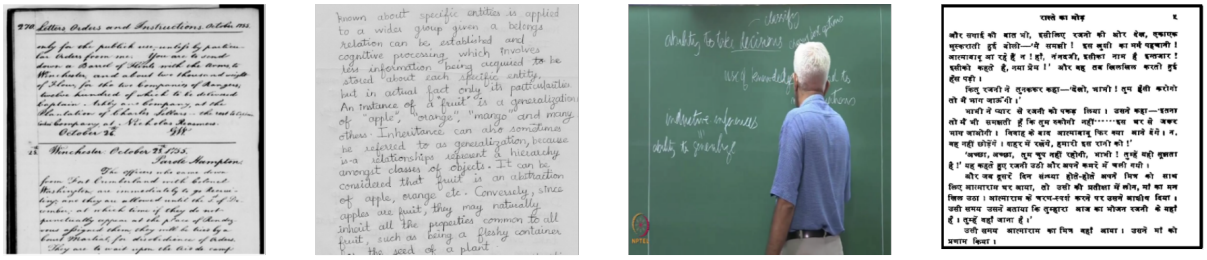
The human mind is highly inquisitive! It is always in pursuit of knowledge. In this process, it is also constantly documenting information in various forms. With a world of more than 7 billion people, there are lots of information to share, discuss and discover. In today's world, technology acts as a bridge that solves these problems and in recent times, artificial intelligence (AI) is becoming an answer to many of these challenges.

1.1 Motivation

The creation and dissemination of text are one of the primary ways we share and document information. Printed books, historical manuscripts remain at the forefront of organizing and archiving the knowledge of the world. With the advent of computers and the internet, the creation and sharing of text have become efficient, fast and scalable. With a growing world and its increasing dependence on technology, there is a dire necessity to bring offline documents into its digital format. These offline documents typically exist in the form of printed books, handwritten manuscripts, court records, bills, notes and letters etc. Bringing these documents to its digital counterparts would effectively open the vast amount of knowledge to the society. It will also seamlessly allow us to search and transfer data between our physical and digital worlds.

One of the first step towards bringing the physical documents to its digital form is digitization. There exist multiple digital library projects [4, 11] across the world where millions of documents are being scanned and stored in the form of images. These document images are in different languages and are either created from machine-printed books, handwritten manuscripts and belong to a modern or historical era. In the current world, the availability of images containing text is not limited to documents from digital libraries. The increasing usage of digital recording and capturing devices such as mobile cameras, allows a common user to quickly capture bills, medical reports, natural scenes with text and store it either locally or over a cloud network. One of the primary challenges which lie before the technology community is to provide an efficient content level access to these document images. An effective solution to this would enable applications such as: (i) search and retrieval from historical datasets, personal handwritten collections, instructional videos, and (ii) natural language applications such as finding near-duplicate documents, machine translation, question answering, etc. More importantly, this will also lead to an increase in digital content in non-Latin languages which would bring much more users closer to the digital revolution.

Understanding text from images and extracting information has been one of the primary interest areas of artificial intelligence (AI). These problems are pursued under the sub-domains of computer vision and natural language processing and are typically formulated in a machine learning framework. Advancements in AI are continuously bringing machines closer to human performances for tasks such as reading text from the visual domain, understanding speech from audio signals, interpreting and extracting knowledge from language. Traditionally, reading text from images is designed as an optical charac-



(a) (b) (c) (d)

Figure 1.1 Sample document pages: (a) historical handwritten document written by George Washington [133], (b) a handwritten assignment written by a student in a course [90], (c) a video frame containing handwritten text on classroom board, and (d) a sample page taken from a printed book written in Hindi as part of DLI project [11].

ter recognition (OCR) problem. An OCR problem has come a long way in AI history with recognition engines that can transcribe printed books in various languages. However, there still exist challenges in the recognition where the document images are from the handwritten domain, historical collections, etc. Fig. 1.1 shows sample images from different domains where the performance of traditional recognition engines is limited. Here: (a) shows a sample page from a historical handwritten document written by George Washington [133], (b) shows a handwritten assignment [90] created by a student, (c) shows a video frame containing handwritten text on classroom board, and (d) shows a sample page taken from a printed book written in Devanagari script and digitized as part of the "Digital Library of India" (DLI) project [11].

1.1.1 Scope

A document image is a structured entity with a particular layout that contains paragraphs, lines, words, and characters. Often such documents also contain figures, graphs, and other components thereby increasing the complexity of the layout. One could also observe that the layout of a handwritten document is more arbitrary as compared to a machine-printed document image. Given a document image, one has to first interpret the layout of the page and split it into its constituent parts (paragraphs, figures, graphics, etc.). This is referred to as layout analysis which is an open research problem [20] in the document community. Once the layout is inferred, the next problem to address is to segment out the constituent lines, words and character images. The existing methods in this space are robust for printed document images, however, it is challenging for handwritten collections. Given the seg-

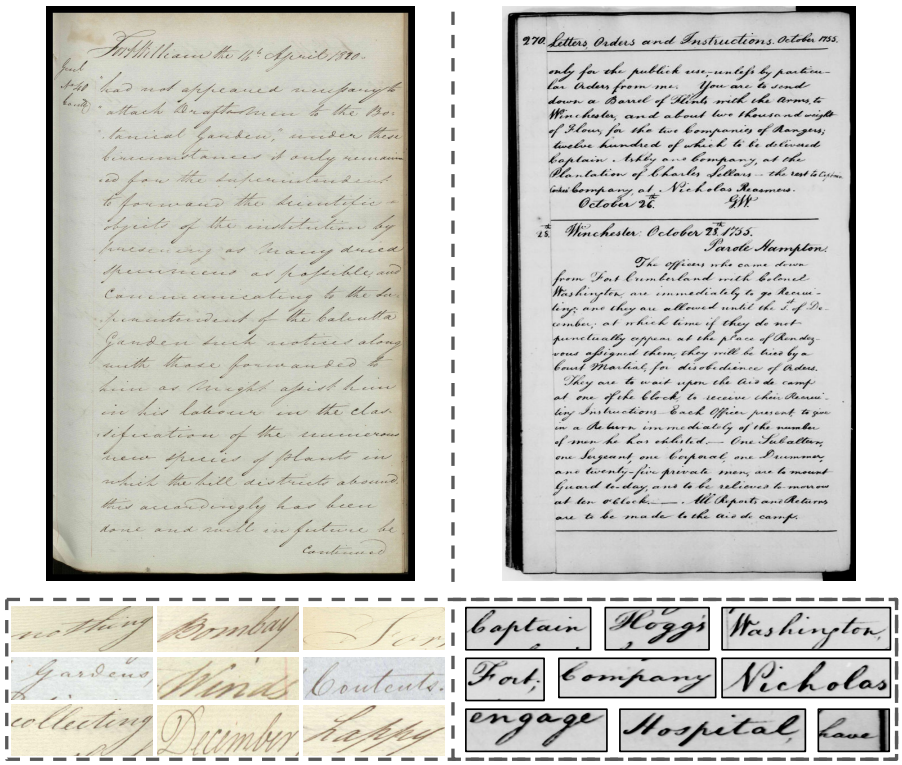


Figure 1.2 The top row shows two sample handwritten document images, while the bottom row shows few sample word images taken from the corpus containing these pages.

mentation (bounding box coordinates) of line and word images, there exist two broad categories of techniques in literature for achieving content level access in document images: "recognition-based" and "recognition-free".

Recognition based approaches are very popular and easily scalable. Here the text is usually generated from an (OCR). These systems have been highly successful in printed documents with standard fonts for Latin languages. However, the document images that we are interested in this thesis are handwritten, historical manuscripts, low resource languages (such as Indic scripts) and degraded printed books where traditional printed OCR based methods are limited. Moreover, the recognition based approaches also put a hard constraint that all the necessary symbols (textual units) which need to be recognized should be available in the training corpus of the underlying recognition model. This limits its applicability in historical documents which usually contains outdated characters and ligatures.

This leads us to the complementary method, where the idea is to re-formulate the problem in a "recognition-free" manner. This was popularized as "word spotting", originally proposed in speech

community [139] and later adapted to document images in [106]. Here the basic idea is, given a query (mostly a word), retrieve all its relevant instances in the entire corpus. One can perform such a task without the actual recognition of word, by comparing the feature representations in an appropriate space. These holistic word image representations can be learned either in an unsupervised or a supervised fashion and are found to be highly effective for complex handwritten and historical document collections.

Considering words as the basic unit of understanding a document, in this thesis, we set our granularity at the level of word images. We assume that the layout and the segmentation of a document image are given a priori which could be either provided manually or as an output of an existing method in which case it could be noisy. Fig. 1.2 shows two sample historical handwritten document images, where the left one is a manuscript taken from READ project[1] and the right image is one of the letters written by George Washington, archived [132] as part of George Washington Papers at the Library of Congress. The bottom row shows the sample word images extracted from the same corpus.

1.2 Word Representation Learning

Feature learning has been a key investigation for any artificial intelligence problem. In the domain of document images, the problem of defining an optimal feature at different granularities such as word [7,9, 106, 149], character/patch [137, 144, 175] has been an interesting quest in the community. In this thesis, we focus on the problem of learning efficient representations for "word" images. The term "efficient representation" within the scope of this thesis refers to the quality of representation in addressing the various challenges as presented in Section 1.2.3. Before we delve into our problems of interest, let us understand why we are interested in learning representation.

1.2.1 Why Representation Learning?

Broadly, there are two paradigms of approaches: (i) *recognition* (or transcription) of a word image into its constituent textual string and later transferring it into the representation space using existing textual IR algorithms, and (ii) a direct embedding of word images into the appropriate representation space through a learned model. A perfect recognition is all we want for building any type of IR or NLP application around document images. However, the document images that we are interested in this work

[1] https://read.transkribus.eu/

are handwritten, historical manuscripts and degraded printed books where traditional printed Optical Character Recognizer (OCR) based methods would result in noisy text and could lead to inferior results. This leads us to the complementary method, where the idea is to formulate the problem from a retrieval perspective. In this case, the focus is to learn an appropriate *representation* space which preserves the similarity among the content words irrespective of their variations due to style and degradation. One of the classical works in this space is the development of word spotting problem, originally proposed in the speech community [139] and later introduced in the document community [106]. Here, the basic idea is to represent the underlying word images into a representation space where matching between the query and the candidate is posed as a retrieval problem. In literature, there exist many successful representations which are learned either in an unsupervised setting [6, 7, 148, 149, 154, 189] or in a supervised fashion [9, 86, 91, 168, 186]. These representations are successful in capturing the lexical similarity at the level of word-forms while remaining invariant to a large extent to different (writer) styles and degradations. In this thesis, we also follow this later paradigm.

1.2.2 Problems of interest

Fig. 1.3 presents an illustration of the holistic word representation learning problem. The term "holistic" within our context refers to something which captures the whole property (lexical or semantic) by implicitly capturing the information from the parts. In the figure, the placeholders (a-c) signify the three major problems of interest of this thesis. As mentioned in the scope, we assume the document image is pre-segmented into word bounding boxes. Given the particular word images/textual-string "music" and "SONG", the goal is to compute an appropriate representation (shown as a histograms in the figure). In the bottom part of the figure, we show two such interesting representation spaces where the word images/textual-string are projected. In the following discussion, we enumerate each of the sub-problem:

a) **Word Image Representation:** One of the fundamental question that we pursue in our thesis is about finding an efficient holistic word image representation. The representation should be invariant to multiple styles, deformations, noise artifacts, while only respecting the lexical (verbatim) content of the word. In our illustration figure 1.3 (a), we present this problem of computing an appropriate representation for the word image "music" irrespective of its style. Learning such a representation enables to represent each word image from the corpus onto an appropriate lexical feature space where searching similar words is achieved by finding its nearest neighbors.

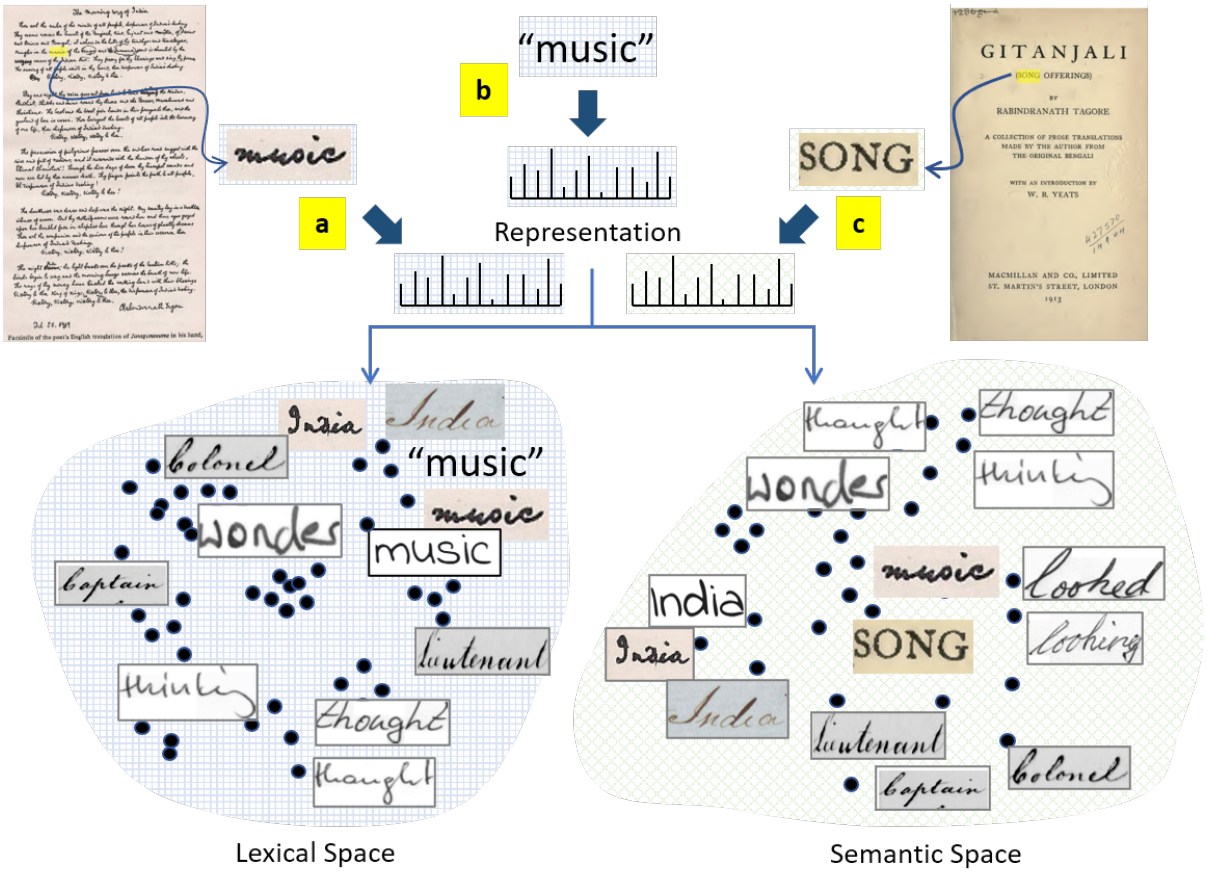


Figure 1.3 Illustration of the holistic word representation learning problem. (a) Word Image Representation in Lexical Space, (b) Joint Text and Image Embedding, and (c) Semantic Embedding.

b) **Joint Text and Image Embedding:** Our second problem of interest delves on the question of finding out a common representation space between textual strings and word images. Such a space where the text and images are mapped together is referred to as the embedding space. A common representation between the word images and its textual string enables searching using query-by-string/example or performing word recognition using a constrained lexicon. In our illustration figure (b), we show that the textual string "music" is also embedded into the same lexical space where its corresponding word images are lying close to each other.

c) **Semantic Embedding:** One of the interesting aspects in the domain of document images is its close relationship with the text-domain. A document image essentially contains text and therefore the "semantics" that are involved in the creation of the original text is directly applicable. Here, we take inspiration from the recent progress in the text domain where the community has adapted

Figure 1.4 Typical challenges which exists while learning of representation of word images: (a) multiple writer styles, (b) degradation and segmentation irregularities, and (c) extension to multiple scripts and languages.

to the distributed representation of words which encodes its semantics. In our thesis, we also explore this novel problem, where we want our representation of word images to be not limited by its lexical content but also include its semantic properties. In the illustration figure (c), we show the semantic space where word images are projected based on its semantic properties. For example the representation for word "SONG" and "music" lie close to each other. In this thesis, we restrict the scope of semantics at the level of individual word usages which involves "inflections" (the words that are related to each other by a common root word) and "synonymy" which refers to words having a similar meaning in a given context. Learning holistic representations in a semantically constrained manner helps to perform search in lexical and semantic space, and thereby reduce the vocabulary gap that exists between a query and its retrieved results.

1.2.3 Challenges

The foremost challenge in a representation learning problem is the intended type of in-variances. Some of the major challenges in our problem statement are listed below:

1. **Style and Font In-variances.** We prefer to learn a style and font invariant holistic representation which focuses solely on the content of the word. In the case of handwriting, the complexities occur due to various underlying factors such as writing speed, smoothness of pen/writing tool on the paper, etc. These factors make each writer have a unique style of handwriting. There are indeed many instances where even a human finds it difficult to interpret what is written without having enough contextual knowledge from nearby words. Figure 1.4 (a) presents a few sample word images written by different writers.

2. **Noise and Degradation.** In the case of the historical documents, one has to deal with noise and degradations caused on the manuscripts due to the damage that has occurred over time. Another source of noise arises from the scanning process of documents and is typically seen in large scale digitization projects.

3. **Segmentation Irregularities.** Finding accurate word-level segmentation can be demanding for complex documents with arbitrary layouts, touching lines and non-uniform kerning. Under these circumstances, one has to accommodate segmentation irregularities while finding the representation of these word images. Figure 1.4 (b) shows a few sample word images taken from historical documents with complex layouts.

4. **Extension to Multiple Scripts.** In our work, we would like to demonstrate a generic framework for learning word image representation which is not limited to a particular script or language. We are particularly interested to work with printed and handwritten Indic scripts that do not have robust text recognizers. These scripts posses additional challenges that arise from the intricacies of the script, and lack of enough training data to build and deploy machine learning algorithms. Figure 1.4 (c) shows a few sample word images from Indic script Devanagari and Bangla.

5. **Compactness.** In addition to the aforementioned challenges, we also prefer to learn compact representation which can facilitate faster search and lower index sizes.

1.3 Contributions

The thesis has the following set of significant contributions:

1. **Word Image Representation.** We propose a framework for learning efficient holistic representation for handwritten word images. The proposed method uses a deep convolutional neural network with a traditional classification loss. The major strengths of our work lie in: (i) the efficient usage of synthetic data to pre-train a deep network, (ii) an adapted version of ResNet-34 architecture with spatial pyramid pooling, which learns discriminative features for variable-sized word images, and (iii) realistic augmentation of training data which mimics the handwritten data. We further investigate the process of transfer learning at various layers to reduce the domain gap between synthetic and real domain and also analyze the in-variances learned at different layers using various visualization techniques. Our representation leads to a state of the art word spotting

performance on standard handwritten datasets and historical manuscripts in different languages with minimal representation size.

IIIT-HWS Synthetic Dataset. As part of this work, we also release a 9M synthetic handwritten word image corpus, useful for training deep network architectures and advancing the performance in handwritten word spotting and recognition tasks. We present this contribution along with the dataset in Chapter 3.

2. **Joint Image and Text Embedding.** We propose a deep convolutional feature representation that achieves state of the art performance for word spotting and recognition for handwritten images. The major contributions are: (i) we propose different schemes of embedding label information in the process of learning word image representation, (ii) we enable query-by-string word spotting and constrained word recognition, (iii) we present a novel end-to-end embedding framework for learning the common subspace using a multi-task loss function, and (iv) we also validate the role of synthetic data as a complementary modality. We present our results on both handwritten and printed document datasets and show considerable improvement under word spotting and recognition tasks. The details on the joint embedding process is presented in Chapter 4.

3. **Semantic Embedding.** In the domain of document images, for decades we have appreciated the need for learning a holistic word-level representation, popularly used for the task of word spotting. However, in the past, we have restricted the representation which only respects the word form, while completely avoiding its meaning. In this work, we attempt to bridge this gap by encoding the notion of semantics by introducing two novel forms of word image representation. The first form learns an inflection invariant representation, thereby focusing on the root of the word, while the second form is built along the lines of recent textual word embedding techniques such as Word2Vec [113] and has much broader scope for semantic relationships. We observe that such representations are useful for both traditional word spotting and also enriches the search results by accounting for the semantic nature of the task. We report interesting results for the standard semantic evaluation tasks such as word analogy and word similarity. In Chapter 5, we present our semantic representation along with its evaluation studies.

4. **Use Cases of Word Representation.** In Chapter 6, we present the following interesting use cases from learned representations which extend our work to newer tasks and validate its robustness to different modalities.

(a) We demonstrate an interesting problem of predicting similarity between a pair of document images written by potentially different individuals. This has applications related to matching and mining in image collections containing handwritten content.

(b) We build a keyword based retrieval system from online lecture videos. These videos contain text written on blackboards, whiteboards, paper, slides etc. Our retrieval system provide a reasonable performance, in-spite of having such a diverse set of writing content with distortions and poor resolution of images.

(c) Finally, we also demonstrate the generic nature of the proposed representation schemes (Lexical and Semantic) to newer scripts and languages. Here we conduct word spotting experiments for major Indic scripts such as Devanagari and Telugu.

1.4 Publications

Part of the work described in this thesis has previously been presented as following publications. The total number of citations for these publications are 188 [Source: Google scholar, 28 December 2019 [2]]

Journal:

1. **Praveen Krishnan**, and C. V. Jawahar, *HWNet v2: an efficient word image representation for handwritten documents*, IJDAR 2019.

2. **Praveen Krishnan**, and C. V. Jawahar, *Bringing semantics into word image representation*, Pattern Recognition, 2020.

Conferences:

3. **Praveen Krishnan**, Kartik Dutta and C. V. Jawahar, *Word Spotting and Recognition using Deep Embedding* (Oral, IAPR Nakano Best Paper Award), DAS 2018.

4. Kartik Dutta, **Praveen Krishnan**, Minesh Mathew and C. V. Jawahar, *Towards Spotting and Recognition of Handwritten Words in Indic Scripts* (Oral), ICFHR 2018

5. Kartik Dutta, Minesh Mathew, **Praveen Krishnan** and C. V. Jawahar, *Localizing and Recognizing Text in Lecture Videos*, ICFHR 2018.

[2] https://goo.gl/mPDeSZ

6. **Praveen Krishnan** and C. V. Jawahar, *Matching Handwritten Document Images*, ECCV 2016.

7. **Praveen Krishnan**, Kartik Dutta and C. V. Jawahar, *Deep Feature Embedding for Accurate Recognition and Retrieval of Handwritten Text* (Oral), ICFHR 2016.

8. **Praveen Krishnan** and C. V. Jawahar, *Bringing Semantics in Word Image Retrieval*, ICDAR 2013.

9. **Praveen Krishnan**, Ravi Sekhar and C. V. Jawahar, *Content Level Access to Digital Library of India Pages* (Oral), ICVGIP 2012.

Secondary publications which are related to the work done in this thesis:

10. Siddhant Bansal, **Praveen Krishnan** , and C.V. Jawahar, *Improving Word Recognition using Multiple Hypotheses and Deep Embeddings*, ICPR 2020.

11. Siddhant Bansal, **Praveen Krishnan** , and C.V. Jawahar, *Fused Text Recogniser and Deep Embeddings Improve Word Recognition and Retrieval* (Oral), DAS 2020.

12. Vijay Rowtula, **Praveen Krishnan**, and C. V. Jawahar, POS Tagging and Named Entity Recognition on Handwritten Documents, ICON 2018.

13. Kartik Dutta, **Praveen Krishnan**, Minesh Mathew and C. V. Jawahar, *Improving CNN-RNN Hybrid Networks for Handwriting Recognition* (Oral), ICFHR 2018.

14. Kartik Dutta, **Praveen Krishnan**, Minesh Mathew and C. V. Jawahar, *Offline Handwriting Recognition on Devanagari using a new Benchmark Dataset* (Oral), DAS 2018.

15. Kartik Dutta, **Praveen Krishnan**, Minesh Mathew and C. V. Jawahar, *Towards Accurate Handwritten Word Recognition for Hindi and Bangla* (Oral), NCVPRIPG 2017.

16. Anshuman Majumdar, **Praveen Krishnan** and C. V. Jawahar, *Visual Aesthetic Analysis for Handwritten Document Images*, ICFHR 2016.

17. **Praveen Krishnan**, Ravi Sekhar and C. V. Jawahar, *Towards a Robust OCR System for Indic Scripts*, DAS 2014.

1.5 Thesis Outline

- In **Chapter 2**, we present the necessary background from the domain of computer vision (CV), natural language processing (NLP), information retrieval (IR) and machine learning (ML) for understanding the concepts and methods presented in this thesis. We also present an overview of related methods of word image representations proposed in the document community.

- In **Chapter 3**, we present our first major contribution by proposing a deep convolutional neural architecture for learning efficient word image representation. We present a detailed analysis of the training process, augmentation schemes, and evaluate the quality of our representation on standard handwritten and printed datasets.

- In **Chapter 4**, we present our second major contribution which proposes different variants of the joint embedding scheme for word images and its corresponding text. The proposed representation reports state-of-the-art results for the task of word spotting and recognition on major handwritten and printed datasets.

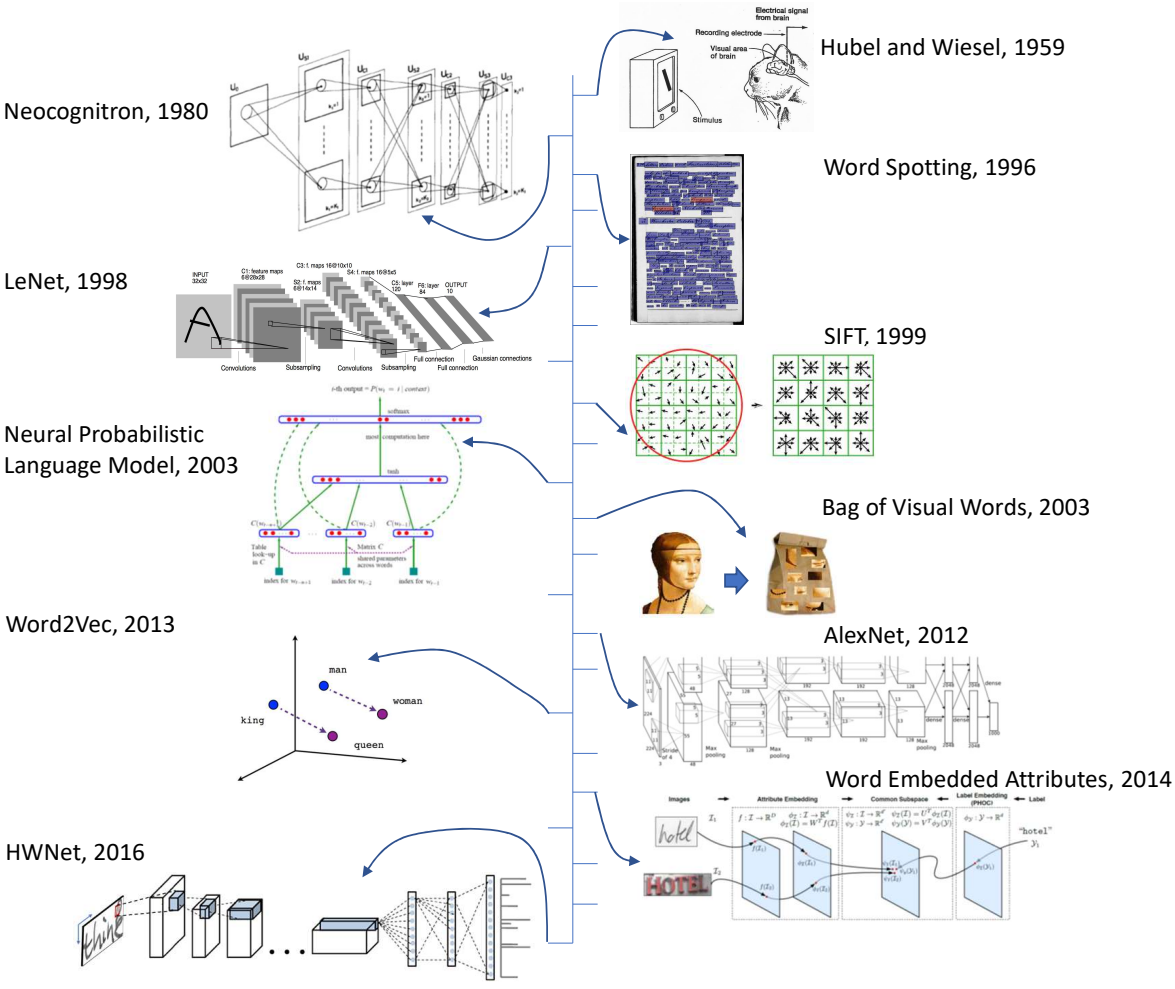
- In **Chapter 5**, we introduce the novel form of semantic representation for word images. The proposed semantic representation presents interesting results for semantic spotting task and is also evaluated under standard performance metrics used in IR literature such as word analogy and word similarity.

- In **Chapter 6**, we present three interesting use cases from the proposed representations of this thesis. These are: (i) matching handwritten document images, (ii) word spotting on lecture videos, and (iii) word retrieval on Indic scripts.

- Finally, in **Chapter 7**, we present the conclusion of the thesis and mention some of the future research works in our domain which are interesting to pursue.

Chapter 2

Background



Feature engineering has been a key investigation for any pattern recognition problem. In the visual domain, feature extraction is a process of mapping an image represented as $I \in \mathcal{I}$ to a feature space $X \in \mathcal{X}$ or equivalently learning a mapping function $\mathcal{F} : \mathbb{R}^p \rightarrow \mathbb{R}^d$. Here p denotes the number of pixels in the given image and d refers to feature dimension and in most cases $d << p$. Depending on the task at hand, typically a feature mapping function is expected to have the following properties:- (i) in-variance to translation, scale & illumination, (ii) robustness to noise and degradation, (iii) compactness, and (iv) the computational efficiency in calculation.

In this chapter, we present the necessary background and the related works from the literature which are essential to the topic of feature learning and evaluation. We divide our discussion into five major sections where we begin with Ssection 2.1 on classical image representation schemes which were dominant to the computer vision community until the last five years. With the resurgence of neural networks and its success to address a diverse set of problems, in Section 2.2 we will present the basics of convolutional neural networks (CNN), which also has been the underlying machine learning architecture used in this thesis. It will be followed by a discussion on how CNNs are useful for learning generic features for images. In Section 2.3, we bring our discussions in the text or natural language processing domain and present the popular methods from the literature which enables learning textual word-level representations. Understanding textual word representation (popularly referred to as word embedding) is important since one of the contributions of this thesis is in semantic embedding where we are interested to find semantic representation for word images. In Section 2.4, we will discuss popular distance measures, evaluation measures and few other techniques commonly used in image retrieval tasks. Finally, in Section 2.5, we present an overview of the related works in the domain of word image representation and retrieval.

2.1 Classical Image Representations

Among numerous types of features proposed in the community, in the early 2000's, gradient-based local features such as Scale Invariant Feature Transform (SIFT [102]) & Histogram of Oriented Gradients (HOG [34]) became popular for a wide range of vision problems. These descriptors summarize the information within a local patch by computing the histogram of edge directions calculated from gradients. Around the same time, a representation scheme motivated by Bag of Words (BOW) [107] in text categorization was introduced for visual domain problems. In vision, it is commonly referred to

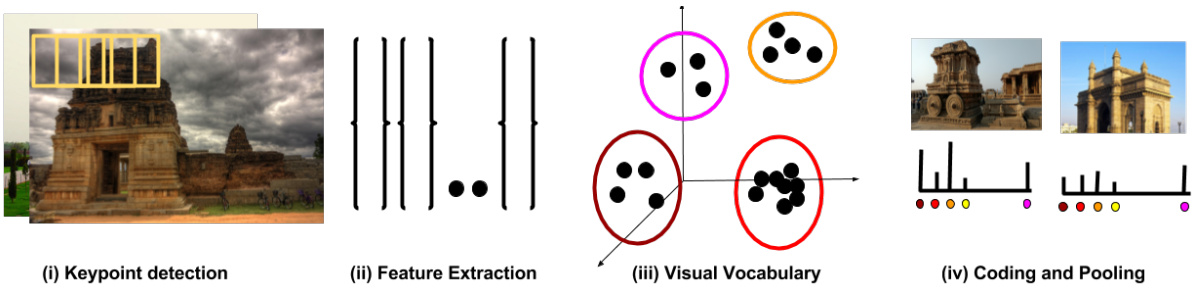


Figure 2.1 Bag of Visual Words representation: The figure presents different stages in the computation of BOW for images.

as Bag of Visual Words (BOVW) [33, 161]. In this section, we discuss in detail the BOVW based image representation which is widely used for image retrieval tasks.

2.1.1 Bag of Visual Words (BoVW) Representation

The fundamental assumption in BOVW based model is that an image is considered as a collection of visual words, similar to the relationship between a textual document and its words. These visual words referred to as $b \in B$, are part of a visual vocabulary B which is learned from a subset of the data (collection of images) in an unsupervised manner. Given an image, a naive BOVW representation is computed by quantizing each local descriptor to its nearest visual word and pooling them into a histogram of visual words. A typical pipeline for BOW is shown in Fig. 2.1 which contain following stages such as:- (i) interest point detection, (ii) computation of local descriptors, (iii) creation of visual vocabulary (also referred as quantization), (iv) coding and (v) pooling. The following sections present different stages of BOW pipeline as shown in Fig. 2.1 in detail.

2.1.1.1 Detectors and Descriptors

A visual word describes a local patch, which is denoted as a region of neighboring pixels. The size of a region can span from small regions capturing only the edges to large regions referred to as parts [79, 160] which capture much more semantic information. Given an image, the foremost step is to extract the interest points (keypoints) where the descriptors are computed. Interest points could be sparse or dense in nature, and 'repeatability' is one of the fundamental desired properties of a detector which makes it robust to noise and other degradations. Feature detection is a low-level image processing technique that utilizes edges, corners or blob level information. Some of the popular keypoint detectors

are Harris affine [66], Difference of Gaussians (DoG) [102] and maximally stable extremal regions (MSER) [110]. Harris affine detector computes corner points and preserves the higher response points. The DoG detector was proposed as part of SIFT descriptor which computes local maxima points in a scale-space which is given as the difference of Gaussian pyramids. While different detectors are typically suitable for different domain tasks, it was later found that dense or regular keypoints [26, 98] at different scales also results in competitive performances. Given a set of local interest points computed from images, local features such as SIFT [102], HOG [34], Local Binary Pattern [119] (LBP) are commonly used to compute the descriptors. Here both SIFT and HOG are gradient-based descriptors, which divides an image into multiple cells (either at keypoints or uniformly), accumulate the gradient directions into bins of fixed sizes, and summarize it into a histogram. HOG performs an additional local contrast normalization on overlapping cells. These descriptors are quite robust to degradation along with achieving scale and translation invariance properties.

2.1.1.2 Visual Vocabulary

Let $x_1, x_2, \ldots, x_K \in \mathbb{R}^d$ be the K descriptors computed from a random subset of the entire corpus of images. As a working example, we take SIFT descriptor ($d = 128$) which is considered as the state of the art descriptor for BOVW based representation for image retrieval problems. Learning a visual vocabulary is essentially partitioning the local descriptor space into informative regions. Typically we use k-means based clustering algorithm, which minimizes the sum of squared Euclidean distances between sample points with their respective cluster centers given as:-

$$B^* = \underset{B}{\operatorname{argmin}} \sum_{i=1}^{M} \sum_{x \in S_i} \parallel x - \mu_i \parallel_2^2 \tag{2.1}$$

where $S = \{S_1, S_2, \ldots, S_M\}$ are the set of M clusters, where each cluster S_i contains the nearest set of descriptors from the cluster mean given as μ_i. We refer $B = [\mu_1, \mu_2, \ldots, \mu_M]$ as the learned visual vocabulary also called the codebook, individual μ_i's are the codewords and the entire process is referred to as quantization. The size of the codebook M should not be small enough to under-represent the patches from the corpus and neither too large which can introduce quantization errors. Quantization errors refer to ambiguity in assigning codewords to descriptors that lie near the boundary of two or more clusters. Here, M is fixed empirically by cross-validation. In general, learning visual vocabulary using k-means on a real-world dataset with millions of descriptors is computationally demanding. There are

efficient clustering schemes such as hierarchical clustering proposed by Nister et al. [118] and approximate k-means using randomized kd-trees proposed by Philbin et al. [127] which bring the computational complexity in assigning a sample to cluster $O(d \log M)$.

2.1.1.3 Coding and Pooling

The next stage in the pipeline is "coding" which performs the mapping from input feature space ($\mathbb{R}^d$) to codewords ($\mathbb{R}^M$). The most popular and basic coding method is vector quantization (VQ) which assigns each descriptor to its nearest codeword from the codebook. It represents a hard quantization scheme where, exactly one codeword is assigned to each descriptor. The VQ for an image having N descriptors is given mathematically as:-

$$\begin{aligned} C^* &= \underset{C}{\operatorname{argmin}} \sum_{i=1}^{N} \parallel x - Bc_i \parallel^2 \\ \text{s.t. } &\parallel c_i \parallel_{l_0} = 1, \parallel c_i \parallel_{l_1} = 1, c_i \geq 0, \forall i \end{aligned} \tag{2.2}$$

where $c_i \in \mathbb{R}^M$ is the respective code for the i^{th} descriptor. To derive the BOW representation from $C = \{c_1, c_2, \ldots, c_N\}$ codes obtained from a given image, we perform "pooling". There are two major types of pooling:- (i) *sum* (or *average* pooling) given as $h = c_1 + c_2 + \ldots + c_N$, and (ii) *max* pooling given as $h = \max(c_1, c_2, \ldots, c_N)$. Here h refers to the pooled representation, which has the following advantages:- (i) derives a compact representation, (ii) brings invariance against position & illumination conditions, and (ii) robustness to clutter, etc. However, doing a naive pooling also results in loss of geometrical information (a fundamental limitation in the bag of words model), which is quite important to build robust image retrieval or recognition systems. In order to partially recover geometry, a spatial pooling scheme is proposed by Lazebnik et al. [95], which divides an images into multiple spatial pyramids and concatenates the weighted histograms from individual regions. The weights are set high at the fine level as compared to coarse level.

2.1.2 Higher Order Representations

The encoding schemes discussed so far are based only on the count statistics of visual words. In terms of moments in statistics, the BOVW representation can be thought to capture the zeroth moment from a set of samples. However, one could think of deriving much higher level representation if first and second-order moments are utilized. In this space, the two popular representation:- (i) Vector of Linearly

Aggregated Descriptors (VLAD) [78] proposed by Jégou et al. utilizes the first moment information, and (ii) Fisher Vectors (FV) [123] by Perronnin and Dance, uses first, second and third moment statistics.

2.1.2.1 Vector of Locally Aggregated Descriptors (VLAD)

Given a codebook $B = \{b_i, i = 1, \dots M\}$ learned with k-means and a set of local descriptors $X = \{x_t, t = 1, \dots, T\}$, the VLAD computation is done in three stages:-

Stage1: Assign $NN(x_t) = \text{argmin}_{b_i} ||x_t - b_i||$

Stage2: Compute residual vectors $v_i = \sum_{x_t : NN(x_t) = b_i} (x_t - b_i)$

Stage3: Concatenate and L_2 Normalize

As shown in the above stages, the final representation is an L_2 normalized concatenated representation from each visual word. Here, each visual word is represented by aggregating the residual vectors (first order moment) of all the descriptors in X which are assigned to that particular visual word. The earlier coding schemes resulted in a dimension of M where M is the number of visual words, whereas VLAD has an increased dimensionality of $d \times M$, where d is the dimensionality of the descriptor.

2.1.2.2 Fisher Vectors (FV)

FVs are an instantiation of Fisher Kernels (FK) [72] on visual vocabulary which gives a dense description of an image using smaller vocabularies. FK measures the similarity between a sample and a statistical model, and was introduced to combine the benefits of generative and discriminative approaches. Given a likelihood function P_λ with parameters λ, the scoring function of a given sample X is given as:

$$\phi_\lambda(X) = \bigtriangledown_\lambda \log p(X|\lambda) \tag{2.3}$$

The above scoring function is given by the direction in which the parameters λ of the model should be modified to better fit the data. This results in a fixed length vectorial representation whose size is fixed by the number of parameters. The Fisher Kernel is defined as:-

$$K(X, Y) = \phi(X|\lambda)' F_\lambda^{-1} \phi(Y|\lambda) \tag{2.4}$$

where F_λ is the Fisher information matrix (FIM) which normalizes the input vector and is given as:-

$$F_\lambda = E_X[\bigtriangledown_\lambda \log p(X|\lambda) \bigtriangledown_\lambda \log p(X|\lambda)'] \tag{2.5}$$

Perronnin et al. [123] introduced FK for visual vocabulary where the vocabularies are represented by means of a Gaussian Mixture Model (GMM). Let $X = \{x_t, t = 1, \dots T\}$ denote the set of T i.i.d d-dimensional local descriptors from an image, and λ be the set of parameters of the GMM. Then $\lambda = \{w_i, \mu_i, \Sigma_i, i = 1, \dots, M\}$ where w_i,μ_i and Σ_i denote the weights, mean vector and covariance matrix of Gaussian i respectively and M denotes the number of Gaussians. The scoring function for X is given as below, which resembles average pooling as a direct consequence of the i.i.d assumption:-

$$\phi_\lambda(X) = \frac{1}{T} \sum_{i=1}^{T} \bigtriangledown_\lambda \log p(x_t|\lambda) \tag{2.6}$$

$$\text{where, } p(x_t|\lambda) = \sum_{i=1}^{M} w_i p_i(x_t|\lambda) \tag{2.7}$$

$$\sum_{i=1}^{M} w_i = 1 \tag{2.8}$$

$$p_i(x|\lambda) = \frac{exp\{-\frac{1}{2}(x-\mu_i)'\Sigma_i^{-1}(x-\mu_i)\}}{(2\pi)^{D/2}|\Sigma_i|^{1/2}} \tag{2.9}$$

The scoring function $\phi_\lambda(X)$ can be computed by taking derivatives with respect to each w_i, μ_i and Σ_i and normalized by the Fisher Information Matrix and concatenated to get the final representation of an image. This leads to a representation of size $(2 \times d + 1) \times M$.

2.2 Neural Image Representations

In Section 2.1, we discussed some of the prominent image representation schemes built successfully on top of highly engineered features such as SIFT, HOG etc. Until last five years, these features were used to build state of art image retrieval and classification methods. More recently there has been a paradigm shift from 'feature engineering' to 'feature learning' due to the resurgence of neural networks and is mostly credited to the revival of Convolutional Neural Networks (CNN). LeNet-5 [96] is one of the earliest proposed work, presents an end to end trainable pattern recognition framework which takes raw pixels as input and learns a feature representation and classifies handwritten digits. Although the recent architectures are inspired by LeNet-5, they use much deeper networks, better optimization algorithms, more data and increased computing power using graphical processing units. In the next subsections, we will present the basics of CNN architecture along with the recent advances and understand how

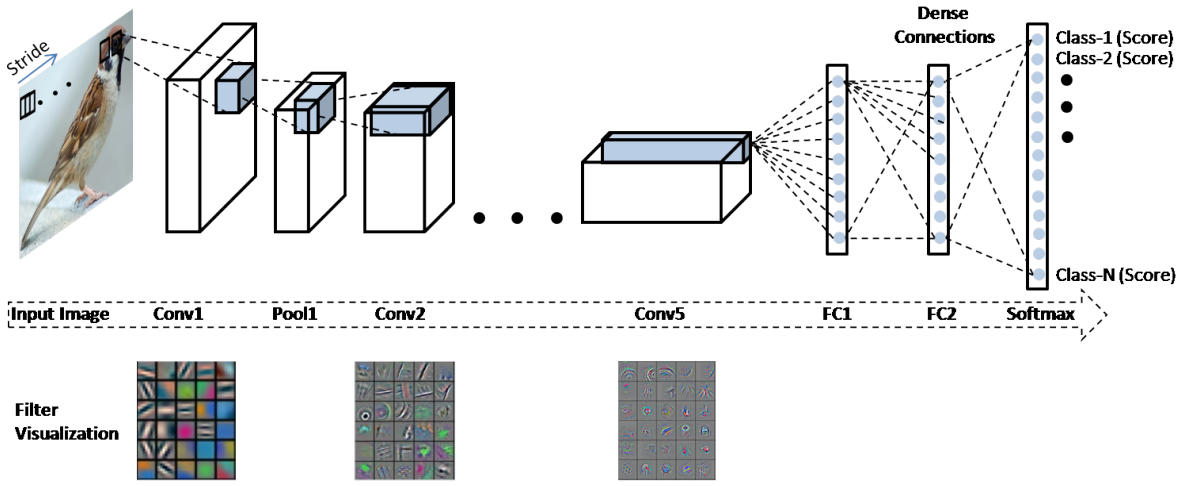


Figure 2.2 Typical architecture of a convolutional neural network for image classification task along with filter visualizations.

these architectures lead to state of the art of image representations in the domain of natural images to document images.

2.2.1 Convolutional Neural Networks

Convolutional neural network (CNN) is a particular type of feed forward network g which maps an input $x \in \mathbb{R}^p$ to output $y \in \mathbb{R}^d$. More specifically [179] $g = f_L \circ \cdots \circ f_1$ is a sequence of simpler functions f_l, called as **layers**. Let $x_0, x_1, \ldots, x_{L-1}$ be the outputs of each function in the network and x_0 is the input image. Output at a specific layer l is given as $f(x_{l-1}, w_l)$, where w_l are the layer parameters (weights and biases). Given an image $I \in \mathbb{R}^{H \times W \times F}$, where H, W, F are the height, width and number of channels (for RGB image, $F = 3$), the output at l^{th} layer of CNN is given as a spatial map $x_l \in \mathbb{R}^{H_l \times W_l \times D_l}$ where H_l, W_l are the spatial resolution and D_l is the number of feature channels present in layer l. There are different types of functional layers, each one performing a fixed or a learned function to transform the inputs from one space to another. Some of important functional layers are:-

A **convolutional** layer computes the convolution between an input map $x \in \mathbb{R}^{H \times W \times F}$ with a bank of K multidimensional filters $f \in \mathbb{R}^{H' \times W' \times F \times K}$ to produce an output $y \in \mathbb{R}^{H'' \times W'' \times K}$. The convolu-

tional operation is shown in Fig. 2.4(a), which is denoted as:-

$$y_j^n = f(\sum_{k=1}^{F} x_k^{n-1} * w_{kj}^n), j = \{1, \ldots, K\} \tag{2.10}$$

where 'n' denotes the layer index, k is the input channel index, $w_{kj}^n \in \mathbb{R}^{h \times w}$ is the j^{th} set of filter weights, $*$ denotes the convolutional operator, $f(.)$ is an activation function and y_j^n is the j^{th} output feature map. Convolution is a linear and translation invariant operator that acts as a local detector for images. Here, K filters are used and each filter is applied as a sliding window to a local region of x^{n-1}. The parameters are shared across different local region and thereby limits the number of parameters to be learned. It also exploits the stationarity principle of an image signal and preserves the locality of pixel dependencies. To learn complex transformation in input space, non-linear activation functions are used after each convolutional operation. The popular activation functions are presented in Fig. 2.3. Here both the sigmoid and tanh functions suffer from vanishing gradient phenomena due to its smoothness towards the lower and higher domain values which leads to slower training. Rectified Linear Units (ReLUs) [116] solves this problem by having non-saturating gradients, however, it can suffer from having dead units by learning high negative bias. To overcome this, leaky ReLU units are proposed which can overcome this property over the course of training. Maxout [57] activation units are a generalization for ReLUs which learn a non-linear activation function by approximating it with multiple piece-wise linear units.

The hyper-parameters of a convolutional layer are the spatial resolution (h, w) of the filter, the number of filters (K), the stride (s) and the padding width p. Here the stride refers to step size in the x and y dimensions which are set as per the required output resolution. Padding is usually set to preserve the spatial dimensionality.

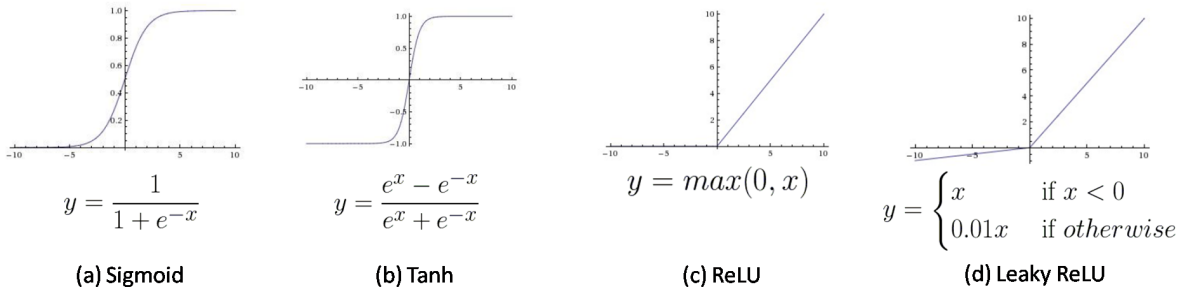


Figure 2.3 Popular activation function used in a convolutional neural network.

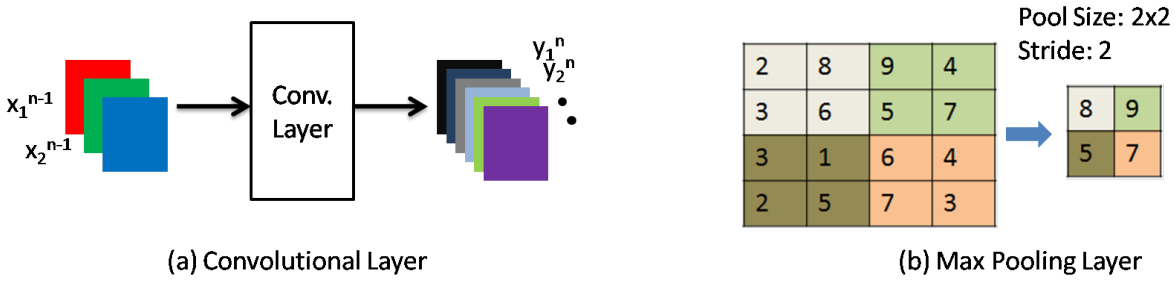


Figure 2.4 (a) A typical convolutional layer, (b) max pooling layer.

In a typical CNN architecture, a **pooling layer** (Fig. 2.4(b)) follows a convolutional layer and typically uses a stride more than one, thereby reducing the spatial resolution of the feature maps. Pooling does the role of an aggregator as we move from fine-scale to coarse-scale of image and is applied over the local spatial map which brings in-variance to image transformation and noise. The popular pooling schemes are sum (average) and max pooling. The hyper-parameters of the pooling layers are similar to that convolutional except the number of output feature channels remains the same to that of input. As we tend to go further down the layers starting from the input layer at the highest scale, we progressively reduce the resolution of the features using the pooling layers and thereby increasing the feature capacity by capturing higher-level semantics. Therefore the network capacity is increased by increasing the number of filters (parameters) in the later layers. In Fig. 2.2, we show 5 conv. layers and 3 pooling layers along with the filter dimensions and the resolution of the feature maps.

Given the abstract feature representation from a set of conv. and pooling layers, a set of **fully connected (FC) layers** are used which is treated as a classifier that maps the conv. features to output space with dimension equivalent to the number of classes. FC layer is a multi-layer perceptron with dense connections where all its units are connected to the previous layer units. Here, in our figure, we show three FC layers and the output from the last FC layer is given to a softmax function with maps the score of each neuron between (0,1) using the softmax function as given below:-

$$\hat{y}_i = \frac{e^{x_i^n}}{\sum_k e^{x_k^n}} \tag{2.11}$$

The output from a softmax layer can be treated as a normalized sample from a probability distribution.

2.2.1.1 Training

Training a CNN network requires to calculate the optimal set of weights for each learnable parameter in the network. These are filter weights (W) and biases (b) which corresponds to the convolutional and fully connected layers. For finding the optimal weights, we formulate the learning process in terms of an optimization problem with the following objective function to minimize:-

$$\theta^* = \underset{\theta}{\operatorname{argmin}} \sum_{n=1}^{N} \mathcal{L}(x^n, y^n) + \lambda ||W||_2^2 \tag{2.12}$$

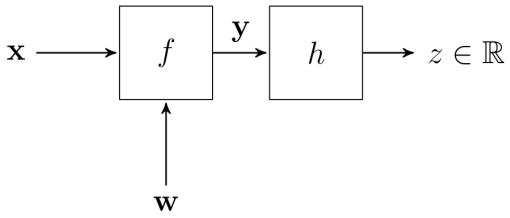
where $\mathcal{L}(.)$ denotes the loss function which is parameterized by θ which contains both W, b. Here x^n, y^n denotes the input sample and the ground truth label vector for the n^{th} sample respectively. The second part of the equation minimizes the L_2 norm of the weight coefficients also referred as the weight decay which prevents over-fitting by reducing the complexity of the network. This is also known as the regularization process.

In general the loss function for a deep convolutional network with non-linear activation functions is highly complex and non-convex. The most popular optimization algorithm used is the mini-batch stochastic gradient descent (SGD) with momentum. SGD is a first order iterative optimization method which computes the gradient of the loss and iteratively update the weights in the direction where the loss decreases as given below:-

$$\begin{aligned} \theta_{n+1} &= \theta_n - \eta \triangle \\ \triangle &= \mu \triangle + \frac{\partial L}{\partial \theta_n} \end{aligned} \tag{2.13}$$

where η is the learning rate which decides the step size of the update and μ is the momentum factor which decides how much gradient direction from the previous step should be taken into consideration in the current update. Note that there exists many weight initialization schemes [53, 68] for better learning the optimal weights.

In order to update the parameters of each layer in CNN network, the backpropagation [147] algorithm is used which is an application of chain rule for derivatives. Consider the below intermediate computational block [179] from CNN network with layer f with parameters w with input x and output y. For simplicity of explanation, we avoided the use of activation functional block.



Here, function h computes the loss which is scalar output z. The derivative $h \circ f$ with respect to x and w is given as:-

$$\frac{\partial z}{\partial w} = \frac{\partial z}{\partial y}\frac{\partial y}{\partial w} \tag{2.14}$$

$$\frac{\partial z}{\partial x} = \frac{\partial z}{\partial y}\frac{\partial y}{\partial x} \tag{2.15}$$

The above derivatives are back-propagated from each intermediate layer to the input layer using the chain rule and are applicable for all directed acyclic graphs (DAG) with differentiable layer functions. To summarize the training process, we require two computational paths which are:-

Forward pass: which takes input x and parameters w and computes the output y.

Backward pass: which takes input x, parameters w and $\frac{\partial z}{\partial y}$ and computes $\frac{\partial z}{\partial w}$ and $\frac{\partial z}{\partial x}$. Once the gradients are computed we can use SGD algorithm as shown in Equation 2.13 to update the parameters.

2.2.1.2 Loss Functions

The loss function measures the compatibility between the predictions and the ground truth. Some of the popular loss functions for classification and retrieval problems are:-

1. Cross entropy loss: $L = -\sum_i y_i \log(\hat{y}_i)$

   Here y_i is the one-hot representation for the i^{th} class and $\hat{y}_i$ is the probability of prediction of the i^{th} class as obtained from the network after the softmax layer.

2. Mean square loss: $L = \frac{1}{2}||y_i - \hat{y}_i||_2^2$

3. Hinge loss: $L = max(0, 1 - y_i\hat{y}_i)$

In addition to these there exists different loss functions such as negative log likelihood, KL divergence, cosine embedding, margin based ranking loss etc. which are suited for different class of problems. Note that almost all loss functions are continuous and differentiable except hinge loss where the sub-gradient exists.

2.2.1.3 Regularization

To generalize well on the unseen examples of the test set and to prevent overfitting on the training data, there are a couple of regularization methods popularly used in machine learning which are also applicable in the context of deep learning. In this section, we discuss three such methods from the literature.

Parameter Norm Penalty

Deep neural networks with numerous parameters typically learn complex non-linear functions that can easily overfit on the training set. To provide restriction over the learned function and prefer simpler models, a regularizer in the form of a penalty is added over the norm of parameters. The most popular one is a L_2 norm added in the form of $\lambda||w||^2$ which was also shown in our earlier equation 2.12. Here λ controls the amount of regularization needed. This form of regularization is also called as weight decay since it multiplicatively shrinks the weight vector by a constant factor. The other common form of weight penalty is using a L_1 norm.

Dropouts

Dropouts [163] is a way of stochastic regularization where at each iteration of training, few neurons are dropped randomly with a probability of p and thereby all connections to that neuron are also made zero. It breaks the internal symmetry between the layers which causes over-fitting and demands each neuron to learn an individual activity. While testing, all neuron units are preserved but scaled with a factor of p. Adding dropouts is similar to learning ensemble of networks which is a form of generalization.

Data Augmentation

One could generalize better if there exists more data. In practical scenarios, this is usually not that simple because annotations are expensive or even the access to the data is sometimes limited (e.g. in the medical domain). In such a scenario, data augmentation schemes help in generating data by transforming the training data in ways specific to a domain. For example, in the visual domain, one can perform image translation, rotation, scaling which is consistent with the task being considered. One has to be careful and rely only on label preserving transformations. Injecting noise also helps in both generating newer data and for bringing robustness in the learned features. Most of the recent state of

the art architecture trained for different tasks deploys such data augmentation techniques to improve the performance.

2.2.2 ConvNet Architectures

We now discuss some of the popular convolutional architectures proposed in the literature. We will begin with the classical work by LeCun [96] and continue our discussion with some of the prominent architectures introduced in the recent past.

LeNet-5

As mentioned in Section 2.2, the LeNet-5 architecture [96] is one of the first incarnation of convolutional networks in the form we see them today. The architecture was trained for building the handwritten digit recognition task and comprised of 3 convolutional layers (C1,C3,C5) which includes 2 alternate sub-sampling layers (S1, S2) in between and the last conv. layer essentially covers the entire feature map. Following the last convolutional layer is a fully-connected layer and its output contains 10 hidden units, one unit for each digit symbol. In contrast to the modern conv. architectures, the sub-sampling layers (comparable to pooling layers) in LeNet-5 contains trainable parameters and the intermediate conv. layer (C3) filters did not connect with all feature maps produced from the S2 layer. This was done with the motivation of breaking symmetry in the network learning, thereby forcing different feature maps to learn complementary features. The other difference from recent networks is on the output layer of LeNet-5. It is composed of Euclidean Radial Basis Function (RBF), one of each class. Each of the RBF units computes a Euclidean distance from the input vector and the parameter vector is hard-coded. Also, $tanh$ was preferred as the activation function for the conv. and the fully connected layers except for the output layer.

AlexNet

In 2012, with the seminal work [93] of Alex Krizhevsky, Ilya Sutskever and Geoffrey E. Hinton, modern-day CNNs marked its entry by winning the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) competition. The competition dataset contains roughly 1.2 million training images and 150K test images from 1000 classes. The proposed CNN architecture which was later referred to as AlexNet successfully demonstrated training of the deep network, having around 60 million parameters

on two parallel GPUs. AlexNet contains 5 conv. and 3 fully connected layers, along with max-pooling layers after 1st, 2nd, and 5th conv. layer. In terms of activation function, it used rectified linear units (RELU) which do not suffer from the problems of saturation. Saturating units such as $tanh$ and sigmoid functions typically slows down the network convergence. The paper also introduces local response normalization layers after the 1st and 2nd conv. layers that do the task of brightness normalization by normalizing the response of feature maps in a local neighborhood. To reduce overfitting, during training various data augmentation techniques are performed. These included image translation, horizontal reflection, random crops and altering the intensity of RGB channels. For better generalization, dropout [163] layers are used successively after the first two fully connected layers. The output from the last fully connected layer is given to the 1000-way softmax layer.

VGGNet

The success of AlexNet made it clear that going deep is one of the ways to improve performance and the architecture VGGNet [158] proposed by Simonyan and Zisserman was first among to lead the race of increasing depth. The proposed work introduced multiple variants of deep architecture starting with 11 layers to 19 layers. One of the major contributions of VGGNet is the idea of using multiple smaller filter sizes (3×3) which could cover larger receptive fields with less computational complexity. They also use the 1×1 convolutional layers originally proposed in the work of [99] as a way of adding more non-linearity to the network with the limited expense of parameters. The VGG16 architecture with 16 weight layers (13 convolutional and 3 fully-connected layers) contains around 138 million trainable parameters. They also performed multi-scale training by rescaling the image randomly within a certain range. VGGNet secured the first and second position of localization and classification tasks of the ILSVRC competition of 2014.

GoogLeNet

In [171], Szegedy et al. introduced a 22 layer deep convolutional architecture named as GoogLeNet. The proposed architecture consists of multiple inception modules that try to approximate the local sparse structure using dense components. Fig. 2.5(a) shows the naive inception module which contains multiple conv. layers in parallel with sizes $1 \times 1, 3 \times 3, 5 \times 5$ for multi-scale processing along with max pooling. It also uses 1×1 conv. layers for adding the extra non-linearity and dimensionality reduction. The GoogLeNet architecture avoids using fully connected layers which typically contains the maximum

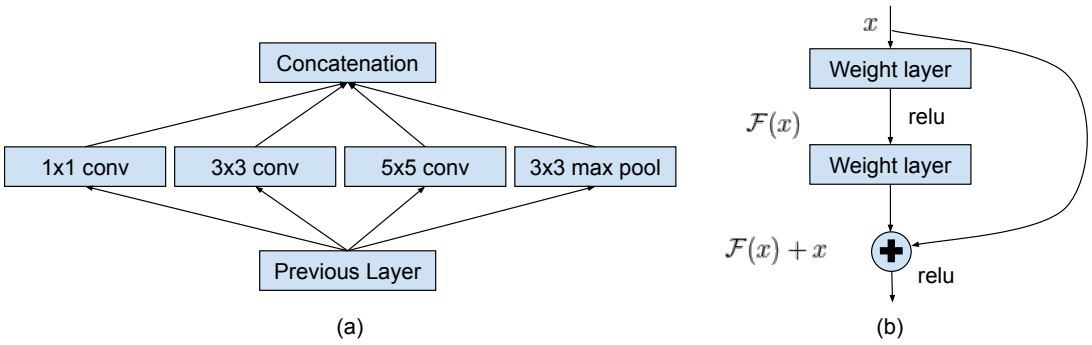


Figure 2.5 (a) Basic inception module as introduced in [171], (b) Residual learning building block as presented in [69]

number of trainable parameters and instead uses global average pooling. This limits the number of parameters in 22 layer GoogLeNet to 5 million which is significantly less than both AlexNet and VGGNet. The proposed architecture also uses auxiliary classifiers in intermediate layers for better propagation of gradients to initial layers. In the later set of works [172], the authors improved their architectures with newer inception modules (v2-v3) with factorized convolutions.

ResNet

In pursuit of going further deep, in [69] the authors report a strange problem referred to as degradation which makes deeper architectures to have a larger test and training error as compared to the shallower architectures. This was not due to overfitting. To avoid this problem and also enable training of much deeper architectures, in this work the authors introduce the concept of residual training. Consider the desired mapping function to be learned as $\mathcal{H}$, the residual learning framework fits another mapping $\mathcal{F} := \mathcal{H} - x$. Therefore the original mapping function is recast into $\mathcal{F} + x$. Fig. 2.5(b) presents the basic building block of residual learning where it is realized by a feedforward network with shortcut connections. Note that the shortcut connection does not incur additional parameters. The proposed module allows learning identity mapping with ease and if the optimal mapping is closer to identity, it is easier to find by small perturbations. The work also introduces multiple ResNet architectures with varying depth sizes of 34, 50, 101, 152, etc. ResNet-152 layer is also the winner of 2015 ImageNet Classification challenge.

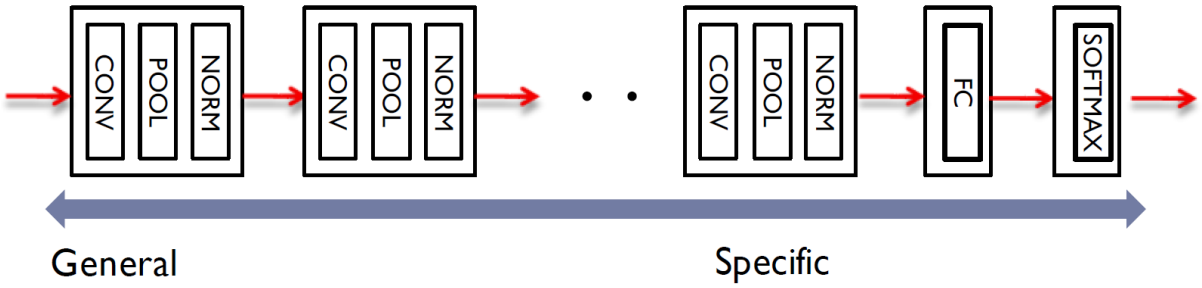


Figure 2.6 A typical convolutional neural network. The bottom axis shows the level of learned feature abstraction from a trained CNN network which enables the property of transfer learning.

2.2.3 Transfer Learning

One of the key observations from a trained deep CNN network using a large dataset (e.g. AlexNet [93], VGGNet [158] trained on imagenet corpus) is that the feature maps obtained at the intermediate layers are generic in nature and can be utilized for other domain tasks. More specifically there is a progression from generic to specific features [14, 38, 192] from initial layers to final layers in the network. Fig. 2.6 presents this transition. This key property of a trained convolutional network led to the popular and pervasive technique called as transfer learning or fine-tuning. In this setting, a pre-trained network trained on a related task is adapted [15, 134] to obtain reasonable and even state-of-the-art performance (features) for newer tasks. The typical rule of thumb during adaptation is as follows. If the target task has a relatively small dataset, then fix all weights of the pre-trained network and only retrain the classifier (final layer). In case of having larger dataset for the target task, then we fine-tune by keeping the old weights of the pre-trained network as initialization and re-train only the last few layers of the network. During re-training, the learning rates are either kept same or lower as compared to the ones used while training the original network. Keeping a lower learning rate avoids network to forget the features which were learned in the original network with a much larger dataset. These fine-tuned features [134] have shown comparable and even better results for problems such as scene classification, human attribute detection, instance retrieval, etc. One of the major advantages of transfer learning is that one may not require large training data in the target domain since the pre-trained network acts as a good initializer of weights.

In [192], Yosinski et al. performed an interesting experiment to validate how transferable are the features in the deep neural network. Their observation are: (i) higher layer features are more specialized to the original task for which the network was trained, (ii) there existed a co-adaptation among neurons

of successive layers which should be taken care while freezing initial layers while transfer learning, (iii) fine-tuning recovers both co-adapted interactions, and improves generalization to newer tasks.

2.3 Neural Word Embedding

In the domain of text and natural languages, representation learning has majorly focussed on finding the right vectorial representation for words. Although in this domain, the representations may exist at different granularities such as phrases, sentences, and documents, in our discussion we limit ourselves to "words". Unsupervised word embedding has been quite popular and successful in NLP domain and became a de facto representation useful for multiple downstream NLP tasks such as machine translation, sentiment analysis, document clustering, recommendation systems, etc. In this section we will focus on neural word embedding, embeddings learned using neural networks. However, there exists classical methods called as topic models including latent semantic analysis (LSA) [35], probabilistic latent semantic indexing (PLSI) [70] and latent Dirichlet allocation (LDA) [21] which were quite popular and extremely useful. In literature one could also find related keywords such as word vectors, distributed representations, semantic representation, however, the most popular usage in recent times is "word embedding" and we will also try to adhere to that in our discussions.

A word in a language in its most basic form is represented as a one-hot vector $w \in \{0,1\}^n$: $\sum_{i=1}^{n} w_i = 1$. Here n is the size of the vocabulary. Such a sparse word vector is a valid feature for a word and quite useful for building a vector space model of a document using a bag of words representation. However, the one-hot feature in itself does not convey anything useful other than its presence. One would ideally expect to extract useful properties, motivated by the linguistic traditions which dictate the usage of words and its similarity with other words. For example, the word "find" and "finding" are related to each other by their common root. The words "college" and "university" are related to other since both are related to educational institutions. According to the distributional hypothesis [67] theory, the words that occur in the same context tend to have similar meanings. In other words, "a word is characterized by the company it keeps". Here, the scope of the context changes in different approaches wherein traditional topic models uses context as the entire document whereas in recent word embedding methods the context is restricted to few ngrams or small windows of words. Before we delve into different neural word embedding techniques, we first bring our discussion on

the fundamentals of language modeling formulation which forms the underlying task in many of these techniques.

Notations: Let's assume our corpus contains T words $w_1, w_2, \ldots, w_T$ in a sequence and the number of unique words in the vocabulary is $|V|$. Also, let v_w, v'_w represent the input and output word embedding for a word w. The role of input and output embedding will be discussed in word embedding techniques.

2.3.1 Language Modelling

A language model defines a probability distribution of words in a language. Given a set of words in a sequence, a language model assigns a probability which measures the validity of that sentence in a language. Using chain rule, one can formulate the problem as below:

$$p(w_1, \ldots, w_T) = \prod_i p(w_i | w_1, \ldots, w_{i-1}) \tag{2.16}$$

Under Markov assumptions, we can approximate the above probability by taking into context only the last n words as below:

$$p(w_1, \ldots, w_T) = \prod_i p(w_i | w_{i-1}, \ldots, w_{i-n+1}) \tag{2.17}$$

Under the classical technique of ngram based language modelling, we represent the probability of words based on the frequencies of the respective ngrams:

$$p(w_i | w_{i-1}, \ldots, w_{i-n+1}) = \frac{\text{count}(w_i, w_{i-1}, \ldots, w_{i-n+1})}{\text{count}(w_{i-1}, \ldots, w_{i-n+1})} \tag{2.18}$$

Traditionally, ngram based language modelling also uses smoothing techniques such as Laplace smoothing or Kneser-Ney [117] to deal with unknown sequence of words during testing.

Language modeling finds applications in spell correction, machine translation, speech recognition and building optical character recognizers. There exist two forms of evaluations: implicit and explicit for a language model. In explicit evaluation, we measure the performance of the downstream application which utilizes it. For example, in an optical character recognizer, one would measure in terms of word accuracy before and after using the language model. An implicit evaluation technique uses the concept of perplexity, borrowed from the domain of information theory. Perplexity is the inverse probability of the test set, normalized by the number of words.

$$PP(W) = p(w_1, \ldots, w_T)^{-\frac{1}{T}} \tag{2.19}$$

In general, lower the perplexity, better is the model.

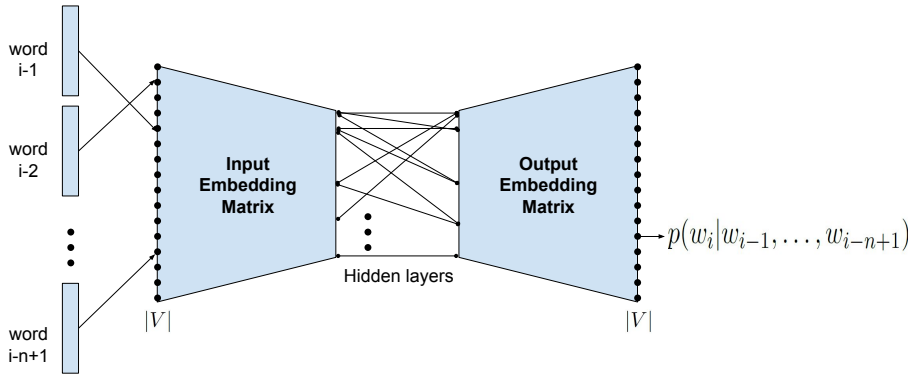


Figure 2.7 Neural probabilistic language model.

2.3.2 Neural Probabilistic Language Model

In [18], Bengio et al. introduced the notion of distributed representation of words which allows to learn a probability function for a sequence of words in a language. Here, both the word representation and probability function is learned jointly using a neural network. Fig. 2.7 presents the neural architecture comprising of a shared projection layer, a hidden layer with $tanh$ non-linearity and an output softmax layer. Considering the neural network as a function approximator, the objective is to learn a function $f(w_i, w_{i-1}, \ldots, w_{i-n+1}) = p(w_i|w_{i-1}, \ldots, w_{i-n+1})$. Here the function is decomposed into two parts as follows:

$$f(i, w_{i-1}, \ldots, w_{i-n+1}) = g(i, C(w_{i-1}), C(w_{i-2}), \ldots, C(w_{i-n+1})) \tag{2.20}$$

where $C(w)$ is the mapping of a word $w \in V$ to a vector $v_w \in \mathbb{R}^d$, also referred as input word embedding. This is implemented as shared layer which is stored in the form of the parameter matrix of size $|V| \times d$. The function g provides a conditional probability distribution of a word w_i given its context vectors $C(w_{i-1}), C(w_{i-2}), \ldots, C(w_{i-n+1})$. The parameters of this function corresponds to the weights present in the input to hidden layer which could be represented by a matrix H with dimensions $h \times (n-1)d$ and hidden to output layer which corresponds to weight matrix U with dimension $h \times |V|$. Here, h is the number of hidden layer neurons. Let the overall parameters in the network correspond to $\theta = (C, H, U)$ excluding biases, then the training objective will be to find the best θ which maximizes the log likelihood L:

$$L = \frac{1}{T} \sum_{i=1}^{T} \log f(i, w_{i-1}, \ldots, w_{i-n+1}; \theta) + R(\theta) \tag{2.21}$$

where $R(\theta)$ is the regularization term, typically used as a weight decay parameter. In order to ensure function $f(.)$ to represent as probability, a softmax output layer is used which guarantees the scores to sum to one. Therefore,

$$p(w_i|w_{i-1}, \ldots, w_{i-n+1}) = \frac{\exp(h^T v'_{w_i})}{\sum_{w_k \in V} \exp(h^T v'_{w_k})} \tag{2.22}$$

Here, h represents the hidden representation and v'_{w_i} represents the output embedding which is stored in the output weight matrix of U for each word w_k of the vocabulary. Note that in this work, authors identifies that the intermediate hidden layers can either use a feed forward network or a recurrent network. Also the major computational complexity in training these network is the final softmax layer which is proportional to number of words in the vocabulary and could easily span to millions of words.

The above architecture and its basic components such as the shared input embedding layer, and the intermediate hidden layers remain the basic underlying framework for many word embedding models. Almost all neural architectures for NLP tasks which takes input as words would require a form of input embedding layer which converts discrete textual data into a continuous form and therefore requires input word embedding. The major distinction among these methods comes from whether they learn the input embedding explicitly or implicitly. The input embedding learned for a particular task may not be general enough to be useful for other tasks. Among the most general task is language modeling which provides semantically rich word embedding. We now present other architectures in this space which tries to either avoid softmax or approximate it for training on the large corpus and leads to much effective word feature vectors.

2.3.3 C&W model

Collobert and Weston in their work [32] successfully demonstrated the role of shared input embedding for different NLP tasks such as part-of-speech tagging, chunking, named entity recognition, semantic role labeling, etc. In their formulation for language modeling, instead of using the softmax layer they used a pairwise ranking criterion as given below:

$$L = \sum_{s \in S} \sum_{w \in V} max\{0, 1 - f_\theta(s) + f_\theta(s^w)\} \tag{2.23}$$

Here S is the set of valid sentence windows from the training corpus, and s^w is a corrupted sentence where the middle word is replaced by w taken from the vocabulary. By minimizing this objective, the network learns to output higher scores for valid sentences as compared to invalid ones. The formulation

also uses a margin of 1. The word features obtained in the shared layer was able to cluster semantically similar words and proved quite useful for solving other NLP tasks.

2.3.4 Word2Vec

One of the most popular recent methods in word embedding is word2vec proposed by Mikolov et al. in their seminal works [112, 113]. Word2Vec simplifies previous embedding models by explicitly addressing the problem of learning efficient representation for words. It removes the expensive hidden layer (hence not a deep architecture) and enables the language model to consider the additional context. The work also proposes newer training strategies and approximates the softmax based loss function by using a technique called negative sampling which is found to be extremely fast and useful for training with a corpus of size in billions of words. Word2Vec introduced two different architectural variants that utilize different contextual information.

Continuous Bag-of-Words Model (CBOW)

One of the first relaxation in the language modeling framework that word2vec introduced is that both past and future words could be taken for learning word embeddings. In CBOW model, n words are taken from both before and after the current word w_i. All these context words are projected into the same position and averaged to predict the current word w_i. Under the bag assumption, the order of the words is also ignored. The objective function is given below:

$$L = \frac{1}{T}\sum_{i=1}^{T} \log p(w_i | w_{i-n}, \ldots, w_{i-1}, w_{i+1}, \ldots, w_{i+n}) \tag{2.24}$$

Continuous Skip-gram Model

In the skip-gram model, the roles are flipped and instead of predicting the center word, the surrounding words are predicted using the center word. The corresponding objective function is given as:

$$L = \frac{1}{T}\sum_{i=1}^{T} \sum_{-n \leq j \leq +n, \neq 0} \log p(w_{i+j} | w_i) \tag{2.25}$$

In the basic formulation of the above variants, a softmax function is used to compute the probability of the current or context word. For example, in skip-gram formulation this would be given as:

$$p(w_{i+j} | w_i) = \frac{\exp(v_{w_i} v'_{w_{i+j}})}{\sum_{w_k \in V} \exp(v_{w_i} v'_{w_k})} \tag{2.26}$$

Note that, in the above equation instead of using h we use v_{w_i} which refers to the input embedding vector for word w_i since word2vec has removed the notion of hidden layer. The architecture only contains the input and output word embedding matrices.

Training

In [113], Mikolov et al. proposed efficient schemes for training word2vec models using computationally efficient approximation as compared to softmax. The first technique is a hierarchical softmax which uses a binary tree representation for the output layer by splitting it into intermediate nodes and leaves. The leaves contain the words from the vocabulary and intermediate nodes represent the relative probabilities of its child nodes. The major computational advantage here is that instead of computing |V| output nodes to obtain the probability distribution, one has to evaluate only $\log_2(|V|)$ nodes.

Another technique proposed in the above work is using a sampling-based approach which avoids the need for computing expensive softmax. Here the authors introduce a simplification of noise contrastive estimation [64] approach referred to as negative sampling. The idea is to learn a model that differentiates data from noise using logistic regression. The idea is similar to the one proposed in Collobert and Weston as discussed in section 2.3.3. The negative sampling objective to maximize is given as:

$$L = \log \sigma({v'_{w_O}}^T v_{w_I}) + \sum_{i=1}^{k} \mathbb{E}_{w_i \sim P_n(w)} \left[\log \sigma(-{v'_{w_i}}^T v_{w_I}) \right] \tag{2.27}$$

Here, v_{w_I}, v'_{w_O} refers to input and output embedding for words w_I, w_O respectively, and $\sigma(.)$ refers to the logistic function. The formulation is designed to distinguish the target word w_O from k noise words which are sampled from the distribution $P_n(w)$ using logistic regression. The paper uses a unigram distribution $U(w)$ raised $3/4th$ power to sample noise words. Also, k is chosen between $5-20$ samples.

The word embeddings produced by word2vec encodes general semantic relationships and is now being used as the de facto input representation for most of the NLP tasks. Moreover, the learned representation expresses linear relationships such as the famous example $king - man + women = queen$ which is quite popular in the community.

2.3.5 Subword Models

Most of the previous models represent each word using a distinct vector. This ignores the internal structure of words, thereby missing the details on morphological features which is quite important in

many languages. Moreover, a word which is not present in the training corpus, one cannot associate any word vector. Such rare words in a language make it difficult to learn good quality word representation. In order to solve these problems, recent methods in this domain use the notion of sub-word level embedding which helps to compose the embedding of entire words by exploiting the internal structure present in them. We discuss one such popular method which extends word2vec to subword level as presented in [24].

In [24], Bojanowski et al. abstract the notion of similarity between word embedding using a scoring function s which maps pairs of (word, context) to real-valued scores. Let us rewrite Mikolov's skip-gram objective using negative sampling in terms of s as below:

$$L = \log \sigma(s(w_O, w_I)) + \sum_{i=1}^{k} \mathbb{E}_{w_i \sim P_n(w)} \left[\log \sigma(-s(w_i, w_I))\right] \tag{2.28}$$

In their work, each word w is represented as a bag of character n-grams. It also contains special symbols $<, >$ to mark the beginning of a word and its ending. This allows differentiating between prefix and suffix of the character sequences. For example: for the word $school$, its 3grams would be $\{< sc, sch, cho, hoo, ool, ol >\}$ and the set will also include the entire word $< school >$. In their implementation, the authors use n from 3-6. Let G_S be the dictionary of an entire set of n-grams including the unique words from the corpus and let $G_w \subset G_S$ are the n-grams present for word w. Let z_g be the vector representation to each n-gram g, a word is represented by the sum of the vector representations of its constituent n-grams from this dictionary.

$$s(w_I, w_O) = \sum_{g \in G_w} z_g^T v'_{w_O} \tag{2.29}$$

The above model allows sharing the representations across words. Thereby providing better embedding for rare words that are common in morphologically rich languages. Also, for out-of-vocabulary words whose constituent n-gram representation is available, we are able to compose its embeddings.

The above discussion on word embedding presents a few major techniques from the literature which has been used in our work for learning semantic representation for word images. Interested readers are recommended to refer to a recent survey presented in [10] for knowing other methods in this space.

2.4 Image Retrieval

Given the representation of an image, one of the fundamental tasks is image retrieval. This would validate the effectiveness of the representation and also enables content level access from the corpus of images. In an image retrieval task, given a query in the form of an exemplar image or a textual string, we need to retrieve all the relevant images in a ranked fashion based on its similarity or distance from the query. In this section, we discuss popular distance measures computed in the representation space. Further, we also present the important evaluation measures used in the information retrieval IR community for measuring the performance.

2.4.1 Distance Measures

In the case of fixed size representations such as BOW for images, neural codes, etc. there exist different distance measures to compute the similarity between a query and the images present in the corpus. Let use denote h_1, h_2 as representations for two different images, some of the popular distance measures are:

- Cosine distance: $D(h_1, h_2) = 1 - \frac{<h_1, h_2>}{||h_1||||h_2||}$
- L2 distance: $D(h_1, h_2) = \sqrt{\sum_{i=1}^{M}(h_1(i) - h_2(i))^2}$
- Chi-Square distance: $D(h_1, h_2) = \sum_{i=1}^{M} \frac{(h_1(i) - h_2(i))^2}{(h_1(i) + h_2(i))}$
- Hellinger distance: $D(h_1, h_2) = \sum_{i=1}^{M}(\sqrt{h_1} - \sqrt{h_2})^2$

Here, the first two distance measures (cosine and L2) are generic in nature while the chi-square measure is more appropriate for finding distances between histograms. The L_1 normalized BOW based representation can be assumed as probability distribution and therefore the class of distance measures which compare two probability distributions can be used instead of the former ones. Hellinger distance measure is one such example.

Another popular distance metric is the Earth Mover Distance (EMD) [146], which is computes the dissimilarity between two probability distribution by solving the below optimization problem. Let $S = \{w_{s_1}, w_{s_2}, \ldots, w_{s_m}\}$ be the source distribution with 'm' elements and let $S = \{w_{q_1}, w_{q_2}, \ldots, w_{q_n}\}$ be the target distribution with 'n' elements. EMD is formulated as optimizing the flow between the two

distribution by minimizing the cost given as:

$$\begin{aligned} \min_{f_{i,j}} \sum_{i=1}^{m}\sum_{j=1}^{n} f_{i,j}d_{i,j} \\ f_{i,j} \geq 0 \text{ for } 1 \leq i \leq m, 1 \leq j \leq n \\ \sum_{j=1}^{n} f_{i,j} \leq w_{s_i} \text{ for } 1 \leq i \leq m \\ \sum_{i=1}^{m} f_{i,j} \leq w_{q_j} \text{ for } 1 \leq j \leq n \\ \sum_{i=1}^{m}\sum_{j=1}^{n} f_{i,j} = \min(\sum_{i=1}^{m} w_{s_i}, \sum_{j=1}^{n} w_{q_j}) \end{aligned} \tag{2.30}$$

The above formulation is posed as a transportation problem from source to destination between a supplier and a consumer. Here, the weights of the suppliers define the respective amount of supplies present with each supplier, while the weights of the consumer are the requirement of supplies for each consumer. The distance between each supplier and consumer is defined using the $d_{i,j}$ variables, and the flow variables ($f_{i,j}$) need to be optimized, which defines how much supply each supplier should send to each consumer. The first constraint defines the non-negativity of supply from a supplier to a consumer. The second constraint defines the total supply sent from each supplier to all consumers should not exceed beyond the capacity of each supplier. Similarly, the third constraint defines the amount of supply received by each consumer should not exceed its requirements. The last constraint restricts the overall flow to be the minimum to the overall capacity of all suppliers and the overall requirement of all the consumers together. In the case of normalized BOVW representation of two images, $d_{i,j}$ is defined as the distance between the i^{th} and j^{th} visual word in the $\mathbb{R}^d$ space and the weights are the representation of codes for each visual word. Although EMD is a robust similarity measure, it is computationally demanding to solve the linear optimization problem for finding the distances between each pair of images.

Another popular similarity measure between a pair of temporal and variable-length sequences is defined using Dynamic Time Warping (DTW) [150]. It was originally developed as a time series alignment algorithm to solve the speech recognition problem, and uses the dynamic programming algorithm. In the domain of document images, a word can be assumed as a temporal sequence of features of variable length and DTW based distance measures were quite popular in classical methods such as [133]. However the computational complexity of $O(N^2)$ is a major challenge while searching in large databases.

2.4.2 Evaluation Measures

The standard evaluation measures [107] used in the IR domain are based on precision and recall of the ranked list. Given a ranked list for a particular query along with the list of ground truth occurrences in the corpus, precision is defined as the ratio between the number of relevant instances retrieved out of all the retrieved instances. The recall is defined as the number of relevant instances retrieved out of all the relevant instances in the ground truth. In formal terms, these are given as:

$$\text{Precision (P)} = \frac{TP}{TP + FP} \tag{2.31}$$

$$\text{Recall (R)} = \frac{TP}{TP + FN} \tag{2.32}$$

Here, TP, FP, FN represent the true positives, false positives and false negatives rate respectively. There are two popular ways to combine precision and recall into a single unit which are:

- F-measure: It is the harmonic mean between precision and recall and given as F-Measure = $2.\frac{P.R}{P+R}$
- Average precision (AP): AP is calculated as the area under the precision and recall curve.

In some cases, we also use a measure called precision at N ($prec@N$) which measures the number of relevant results obtained in the top-N instances in the ranked list.

Another methodology used in the IR domain for evaluating graded similarity is normalized discounted cumulative gain ($nDCG$) [77]. The discounted cumulative gain (DCG) at position p is given as:

$$DCG_p = \frac{\sum_{i=1}^{p}(2^{rel_i} - 1)}{(log_2(i+1))} \tag{2.33}$$

where rel_i is the ground truth relevance for the image at rank i. The normalized measure $nDCG$ is defined as $DCG_p/IDCG_p$, where $IDCG$ is the DCG measure for the ideal ranking. $nDCG$ values scale between $0.0 - 1.0$, with 1.0 for the ideal ranking.

2.5 Word Image Representation and Spotting

Learning representation for word images has been traditionally associated to accomplish the word spotting task. Word spotting [106] refers to locate a keyword (e.g word image) given in the form of a query which is either an exemplar word image from the underlying corpus or could be the corresponding

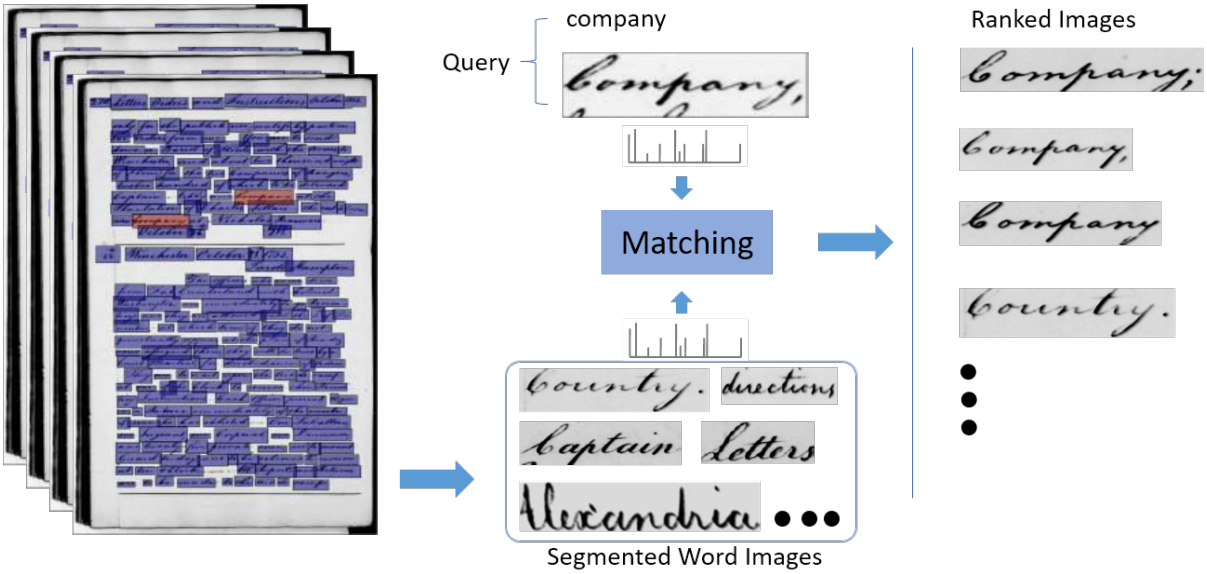


Figure 2.8 Illustration of segmentation based word spotting task. Given a query word, we retrieve all relevant word images from a pre-segmented corpus of document images. Here each word image and the query is represented using an appropriate feature vector which is denoted by a histogram.

text itself. The former setting is referred to as "query-by-example" and the latter is called "query-by-text". The method is inspired by the traditional IR techniques and is widely used in both audio and vision communities. The basic idea is to represent word images using appropriate features. Fig. 2.8 presents an illustration of the segmentation-based word spotting task. Here the corpus of document images is pre-segmented into words and each word image is represented using an appropriate feature representation which encodes its lexical content. Given a query word image or its textual string, it is also projected onto the same feature space where the corpus word images are embedded. The ranked images are given by a matching function which finds similarity using an appropriate distance metric.

There are numerous successful attempts done in the past which focus on different aspects of the problem such as the modality of data (printed, handwritten and scene text), nature of representation either fixed or variable length, and type of embedding schemes. We start our discussion on related works with an introduction of popular datasets used in the community to evaluate word spotting and recognition methods. In our later chapters, we would also be using these datasets to evaluate and compare the proposed representation of thesis with related works. Our discussion on related works is broadly split into four major parts: (i) the classical methods built using variable length representation schemes, (ii)

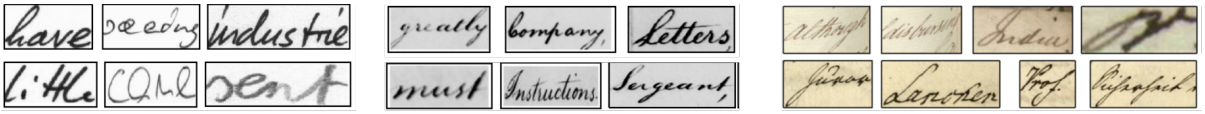


Figure 2.9 Sample word images taken from the popular datasets used in the document image community. (a) The IAM Handwriting Database, (b) George Washington (GW), and (c) Botany (top row) and Konzilsprotokolle (bottom row).

fixed length representation which are achieved using bag of words framework, (iii) learned representation using, different classifier models built on top of handcrafted features and using the deep learning networks, and (iv) segmentation-free approaches.

2.5.1 Word Image Datasets

In this section, we present the popular datasets used in document image analysis community for evaluating word image representation. Table 2.1 shows different datasets and their statistics in terms of the number of words and the number of writers in case of handwritten documents. Fig 2.9 presents sample word images taken from these datasets.

The IAM Handwriting Database [109]: It includes contributions from 657 writers making a total of 1,539 handwritten pages comprising of 115,320 words and is categorized as part of modern collection. The database is labeled at the sentence, line, and word levels. We use the official partition for writer independent text line recognition that splits the pages into training, validation, and test sets which are writer independent.

Dataset	**Historical**	**#Words**	**#Writers**
IAM	No	1,15,320	657
GW	Yes	4,894	1*
Botany	Yes	20,004	1*
Konzilsprotokolle (Konz.)	Yes	12,993	1*

Table 2.1 The list of handwritten datasets used in this work. Here GW, Botany, and Konzilsprotokolle datasets are historical documents written primarily by a single author along with a few assistants(*).

George Washington (GW) [133]: It contains 20 pages of letters written by George Washington and his associates in 1755 and thereby categorized into historical collection. The images are annotated at word level and contain approximately 5,000 words. Since there is no official partition, we use a random set (similar to [9]) of 75% for training and validation and the remaining 25% for testing.

Botany and Konzilsprotokolle [131]: These two datasets are parts of ICFHR 2016 Handwritten Keyword Spotting Competition [131]. The original competition contains data both segmentation based and free scenario. We took only the segmentation based data which contained cropped word images split into training and test sets. There were also three partitions of training sets small, medium, and large. Here we took the largest partition for conducting experiments.

In this thesis, we will be using these datasets along with in-house synthetic datasets to evaluate and compare the proposed representation.

2.5.2 Related Works

Fig 2.10 presents an evolution of some of the key methods proposed in the space of word spotting, which uses some of the recent representations proposed for handwritten word images. It also includes the work proposed as part of this thesis, referred to as HWNet. As one can clearly notice, with the introduction of deep learning, there has been a significant boost in the performance. The proposed representation scheme of this thesis using HWNet v2 architecture further boosts the performance in this space. In Table 2.2, we present a quick overview of the related works that we discuss in this section based on its representation type. One can also refer to the detailed survey presented in [50] which reviews major representation schemes proposed for the task of word spotting.

2.5.2.1 Classical Representation

Learning holistic representation for word images was popularized as word spotting which was originally proposed in [139] for speech processing. Within the document community, initial attempts in this space mostly focused on variable length representations of word images by considering it as a temporal sequence. Most of these methods used profile features [106, 108, 132] which are computed at each column of the word image and are summarized using various pixel level statistics. Dynamic Time Warping (DTW) based algorithms were found to be useful for matching variable length representations and is quite popular in speech [115, 151] and other sequence matching problems. In [132], Rath et al. used profile features namely vertical profile, upper & lower word profile, and background to ink transi-

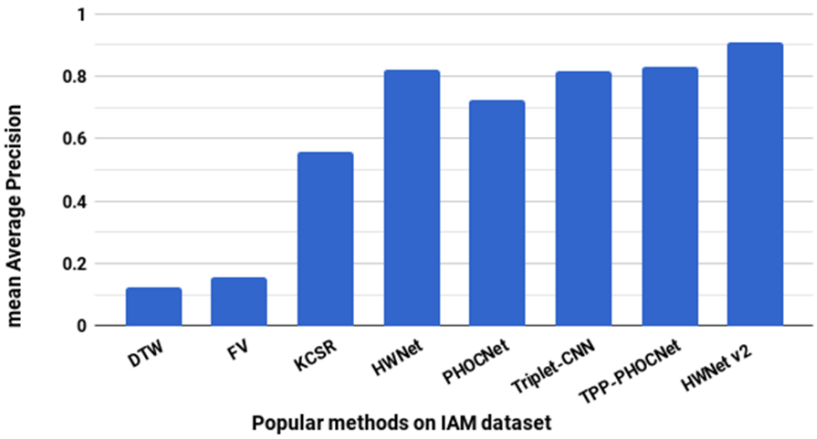


Figure 2.10 Evolution of word spotting methods from the perspective of different word image representation schemes. The evaluation is conducted on the IAM [109] dataset using mean Average Precision (mAP).

tions. In [16, 111], profile features were combined with the shape based structural features for a partial matching scheme using DTW. Although these features are fast to compute, it is susceptible to noise and common degradation present in documents.

With the popularity of the local gradient features such as SIFT [102], HOG [34] which describes a patch using histograms of edge orientations computed from a gradient image, the features are less susceptible to stray pixels and variations in brightness and contrast. Methods such as [137, 175] adapted local gradient features for word spotting where [175] used a continuous DTW algorithm for partial word matching from the line images and [137] used Hidden Markov Model (HMM) based classification method. Most of the features discussed above are not robust to different fonts, writing styles and required careful image pre-processing techniques such as binarization, slant and skew correction which remain hard for handwritten and historical documents. Moreover, the methods such as DTW and HMM based scheme of matching variable length representations do not scale to large datasets due to the higher time complexity. Hence, the later methods appreciated more on fixed length representations built on top of highly engineered features proposed in computer vision.

2.5.2.2 Bag of Word Representation

Fig. 2.11 presents the typical BOW pipeline for learning representation on word images. As explained earlier in this chapter on our discussion of classical image representation using BOW in Section 2.1.1,

Method	Representation
Profile [132] Profile+Moments [108] Profile+DFT [94] Slit HoG [175]	Variable Length
Local Gradient Histogram [137] SIFT [142]	HMM
SIFT [6, 7, 148, 154, 189]	BoWs
SIFT [8]	Fisher
SIFT [9]	PHOC Attributes
Deep Learning [90]	Neural Codes
Deep Learning [85, 166, 167, 186]	PHOC Attributes
Deep Learning [54]	Levenshtein Embedding

Table 2.2 Overview of major methods proposed in the literature, summarizing different types of word image representation schemes.

there exists multiple stages starting from local feature detection and extraction, codebook generation, encoding and spatial pyramid pooling. The top part of the figure visually demonstrated each of this stage, while the bottom part visualizes the corresponding feature computation. The popularity of bag of words (BOW) [33, 161] framework using local gradient features such as SIFT and HOG, led to its proliferation to document images [6, 7, 148, 149, 154, 189]. Rusinol et al. [148, 149], presented a patch based framework using BOW histograms computed from the underlying SIFT descriptors. The histogram based representation was further projected onto a topic space using latent semantic indexing (LSI) [35], where the latent topic space is assumed to preserve the lexical content of word images. In [154, 189], BOW based representation was adapted for word image retrieval for machine printed documents using standard keypoint detectors such as Harris [66], FAST [141] corner detectors, while the local features at the keypoints were computed using SIFT. Due to the fixed length and sparse nature of the representation, the matching was done using cosine distance and an inverted index was used for faster retrieval. Aldavert [7] et al., presents a detailed survey for BOW based representation for handwritten word spotting with its analysis on the effect of codebook size, choice of encoding, and type of normalization. In a

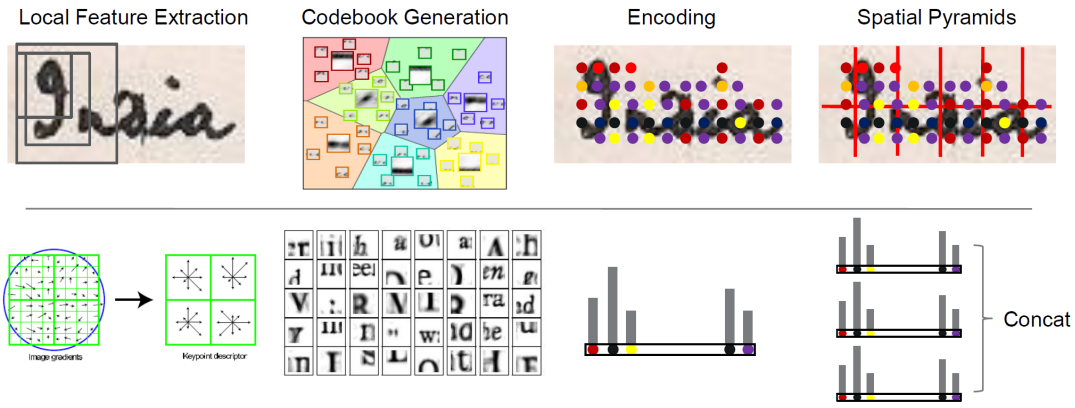


Figure 2.11 Bag of words based pipeline for learning word image representation.

similar line of work using local features and codebook, Almazán et al. [8] uses an exemplar-SVM [105] for representing a query and performs the initial scoring of candidate words. Given the initial matches, the list is re-ranked using a Fisher vector based representation [124] which is a generalization of BOW using higher order statistics. In general, the unsupervised nature of learning of BOW based methods make them directly applicable to historical databases where the annotation is hard and costly. However, the limitation of local features (SIFT,HOG) to capture the larger part level information from word images restricted these methods to work only for limited writer datasets where the variations are less.

Fusion of OCR and BoWs

In one of our earlier works [92] in this space, we propose a search scheme which fuses search results obtained from noisy OCR text and word spotting using BOW representation. In this work, we exploit the fact that in terms of search performance, OCR has a high precision while BOW has a high recall. We use a modified edit distance which takes into account the confusion of different character classes of the OCR based recognizer to re-score the results obtained using BOWs. Our fusion scheme demonstrated an improvement of search performance on the large corpus of the Digital Library of India pages in Hindi and Telugu languages.

2.5.2.3 Learned Representations

Most of the previous set of works in word image representation, either uses hand-engineered features or learns it in an unsupervised fashion. In recent methods, supervised features have gained greater

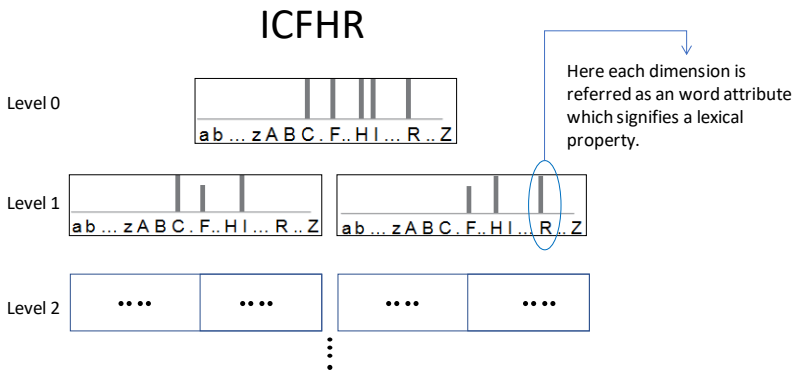


Figure 2.12 Pyramid Histogram of Characters (PHOC) [9]

attention due to their superiority and robustness. We now present one of the major works in this space on word attributes and later discuss the most recent methods using deep learning.

Word Attributes

In the domain of word spotting, the concept of word attributes [9,125] using the pyramidal histogram of characters (PHOC) has been a key contribution to the community. An attribute representation such as PHOC becomes the common link that connects a word image to its text. PHOC attributes are calculated by dividing the word into multiple pyramid levels and at each level, the histogram of characters and bi-grams are computed. The final representation is the concatenation of each level of representation. The intuition behind the attribute is that it denotes the presence/absence of a character at a particular spatial position. Figure 2.12 presents an illustration of PHOC feature computation.

Similar to the textual representation, PHOC representation for word images could be derived from the scores of attribute level binary classifiers. In [9], Almazán et al. uses Fisher features of word images for learning word image attributes while PHOC for text labels are extracted by the spatial position of each character. Fig. 2.13 presents the attribute embedding pipeline as proposed in [9]. Here $f(I), \phi_I(I)$ represents the Fisher feature for the word image I and its corresponding PHOC based attribute representation. Given the attribute level representation for both word images ($\phi_I(I)$) and text ($\phi_y(Y)$), the work also proposes a common subspace regression formulation with a closed-form solution to project these representations onto a common subspace. Here, $\psi_I(I), \psi_y(Y)$ represents the common subspace representation for image I and its label Y respectively. The projection effectively allows to capture the correlation among the attributes and leads to better representation than the naive comparison. Although

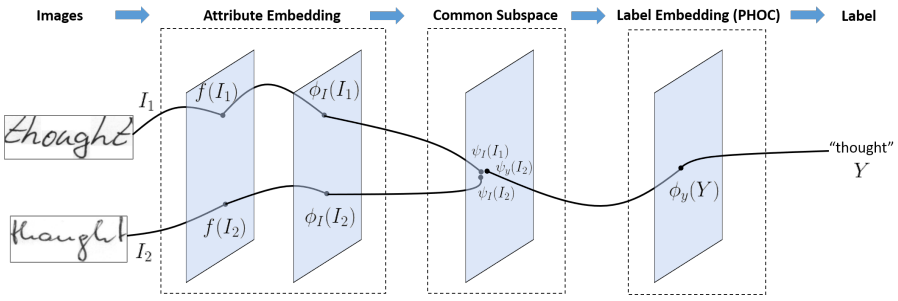


Figure 2.13 The word attribute embedding framework as proposed in [9]. Here, both the word images and its corresponding label are first embedded into an attribute space defined using PHOC and later projected onto a common subspace.

the word attribute framework is generic, the underlying handcrafted features limit the robustness of the learned holistic features. In recent methods, this is addressed using deep neural networks to learn features in an end to end hierarchical manner which results in better generalization.

Deep Learning

With the advancements in deep learning, there is a paradigm shift in feature engineering where features are now learned during the training process which customizes itself to the domain of training data. Among different types of neural networks, deep convolutional neural networks (CNN) [93, 158, 171] have revolutionized the way features are learned for specific tasks. In the domain of word images, Jaderberg et al. [73, 74, 76], proposed three different architecture models (char, nGram and dictionary words) for scene text recognition. Taking inspirations for word attributes, for handwritten images, Poznanski et al. [130] adapted VGGNet [158] for recognizing PHOC attributes by having multiple parallel fully connected layers, each one predicting PHOC attributes at a particular level. In similar spirits, different architectures [166, 168, 186] were proposed using CNN networks which embed features into different textual embedding spaces defined by PHOC. In [166], Sudholt et al. proposes an architecture to directly embed image features to PHOC attributes by having sigmoid activation in the final layer and thereby avoiding multiple fully connected layers as presented in [130]. It is referred to as PHOCNet, which uses the final layer activation to derive a holistic representation for word spotting. In the later set of works [167, 168] from the same group, PHOCNet was adapted with temporal pooling layer (TPP-PHOCNet) and evaluated under different loss functions and optimization algorithms which further improved the word spotting performance. In [186], the authors propose a two stage architecture where

a triplet CNN network is trained to reduce the distance between the anchor word image and a similar labeled (positive) word image, while simultaneously increasing the distance between the anchor and negative labeled word image. In the second stage, the learned image representation is embedded into a word embedding space (PHOC, DCTOW, ngram etc) using a fully connected neural network. In the above methods where the target embedding is an attribute space, one can query the representation space either using query-by-string or query-by-example setting.

2.5.2.4 Segmentation-Free Approaches

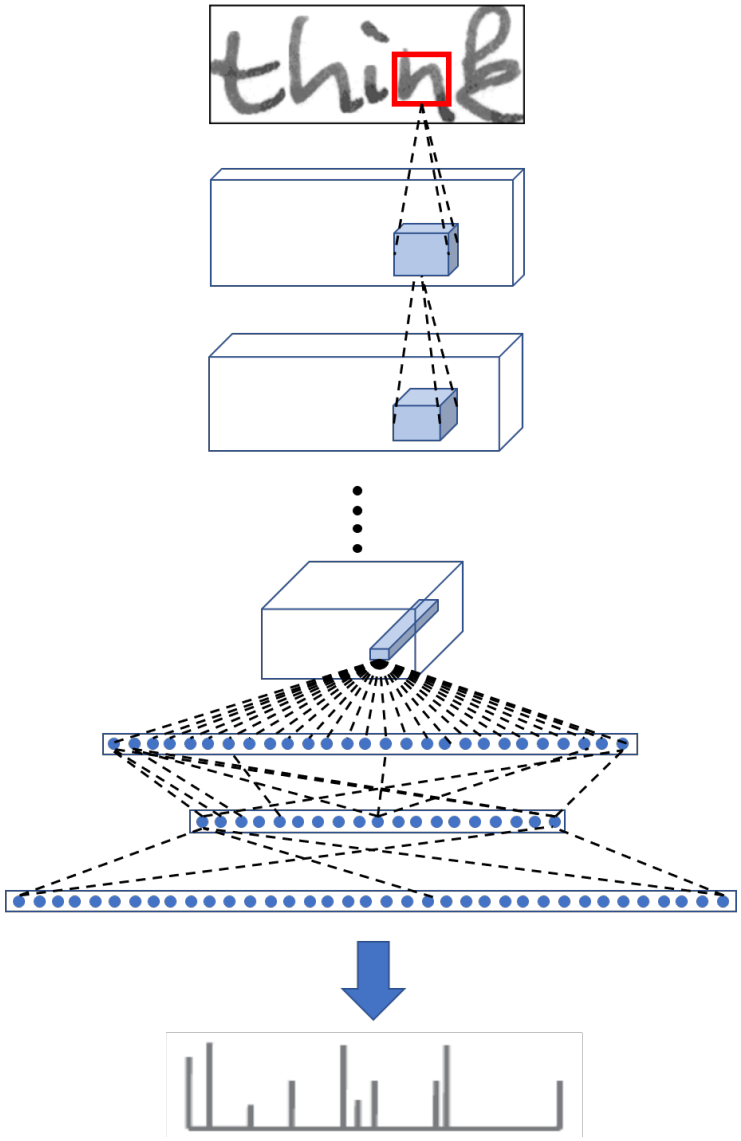
In addition to the different representation schemes, one can also classify the methods in terms of segmentation-based and segmentation-free word spotting approach. Most of the methods presented so far belong to the setting where the segmentation of words is available in the form of ground truth. In segmentation-free setting, the input is a page image and the underlying method first proposes potential word hypothesis before computing its representation. In literature, one can place the segmentation-free approaches into three broad categories. The first category of methods [8, 48, 142, 148] use a sliding window technique where the regions are proposed along a regular grid. This typically results in a dense extraction of bounding boxes and are computationally expensive to process. The second category of methods utilize connected components [49, 84] along with mathematical morphological operations to extract characters/words from the page image. Most of these methods work in a bottom-up fashion and utilize certain rules to extract the final bounding box of words. The number of proposals obtained using this approach is far less than sliding windows, however, they are sensitive to page quality and degradation as seen in historical documents. In [143], authors propose a hybrid approach where the document image is first subjected to dense text detection using sliding windows and later the word hypothesises are computed using the set of extremal regions. The third category of methods [13, 187, 188] in the segmentation-free setting is inspired by the recent success of region proposal based object detection techniques such as Faster R-CNN [136]. The Ctrl-F-Net [187] model proposes an end to end trainable detection and embedding network. It utilizes a localization layer to predict potential word proposals along with its wordness score. The initial predictions are further filtered and presented to the embedding network. The method also utilizes a complementary external region proposal method called as Dilated Text Proposals (DTP) to improve the overall recall of the system. The authors extended their method in [188] by simplifying the architecture (Ctrl-F-Mini) by only utilizing the external proposals computed using DTP. This performs faster and better in certain situations than the original architecture.

2.6 Summary

In this chapter, we presented the necessary background from computer vision, information retrieval and natural language processing domains. We present our contributions in subsequent chapters by taking insights from the existing literature. In the last section, we also presented related works from word image representation and spotting domain. Many of these are directly related to our problems of interest and will be compared in our experimental analysis. Our thesis contributions build on top of modern CNN architectures for learning word image representation. In the following chapters, we present in detail each of our contributions.

Chapter 3

HWNet: Word Image Representation



In this chapter, we present a framework for learning an efficient holistic representation for handwritten and printed word images. As defined in the Section 1.2.2, the term "holistic" in this thesis refers to something which captures the whole property (restricted to lexical property in this chapter) by implicitly extracting the information from the parts. In case of word images, one could consider the parts to be characters and n-grams. The proposed method uses a deep convolutional neural network with traditional classification loss. The major strengths of our work lie in: (i) the efficient usage of synthetic data to pre-train a deep network, (ii) an adapted version of the ResNet-34 architecture with the region of interest pooling which learns discriminative features for variable sized word images, and (iii) realistic augmentation of training data with multiple scales and distortions which mimics the natural process of handwriting. We further investigate the process of transfer learning to reduce the domain gap between synthetic and real domain, and also analyze the in-variances learned at different layers of the network using visualization techniques proposed in the literature. Our representation leads to a state-of-the-art word spotting performance on standard handwritten datasets and historical manuscripts in different languages with minimal representation size. We also present results on printed document datasets in English and Indic scripts which validates the generic nature of the proposed framework for learning word image representation. This work has been previously presented as part of our publications [90, 91].

3.1 Introduction

Feature engineering has been a key investigation for any pattern recognition problem. In the domain of document images, the problem of defining an optimal feature which describes a word [7, 9, 106, 149], character/patch [137, 144, 175] has been an interesting quest in the community over the last two decades. With the improved representations over time, there has been a significant impact on the larger goals of the community such as recognition and retrieval of documents, script recognition, layout analysis, etc. In the domain of word images, initial features proposed were based on pixel level statistics [108, 132] which worked only on limited settings of the writers and font variations. Later, features such as scale-invariant feature transform (SIFT) [102] and histogram of gradients (HOG) [34], were adapted in document image community to build word level representations. These features are invariant to scale, translation, and common degradation. The bag of visual words (BOW) [33, 161], were built using these local features along with advanced encoding schemes such as Fisher [123], locality constrained linear coding (LLC) [184], and sparse codes [191]. These features along with the learned models such

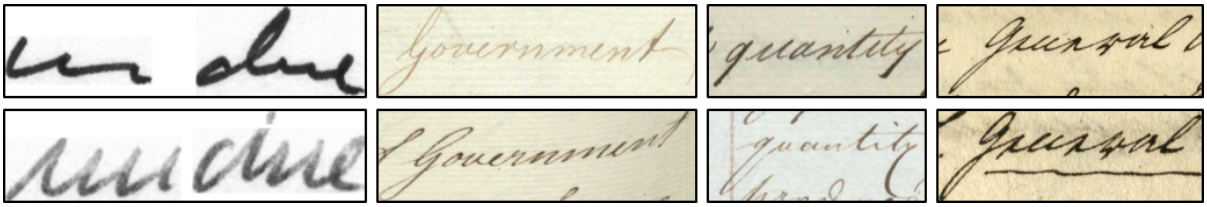
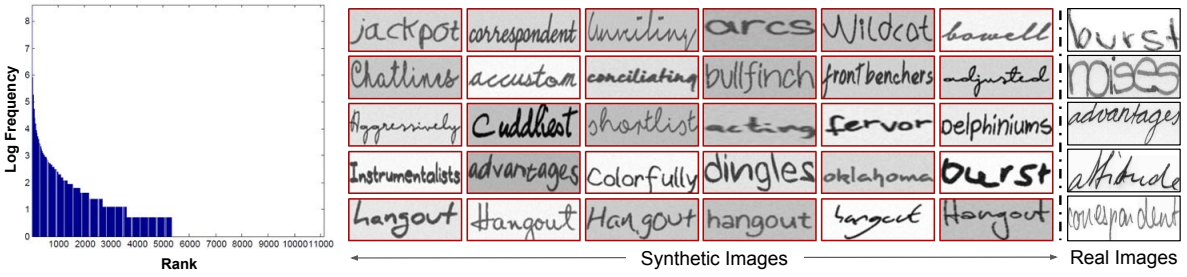


Figure 3.1 Sample nearest neighbor word images in the learned representation space. Here we show examples from historical datasets which contains degradation and irregular segmentation of words. One can also notice invariance of representation in terms of handwriting variations and noise in segmentation.

as [9], obtained state-of-the-art word spotting and recognition for historical manuscripts and multi-writer handwritten documents.

More recently, there has been a paradigm shift from 'feature engineering' to 'feature learning' due to the resurgence of neural networks. It is mostly credited to the revival of convolutional neural networks (CNN) [93], availability of large scale of annotated data [36], and increased computing power using graphical processing units (GPU). The features are learned on the fly during training and gets adapted to the task of interest. Networks trained on large data sets also learn generic feature representation that can be used for related tasks [135], and in many cases, have reported state-of-the-art results as compared to the handcrafted features. When the data is limited, fine-tuning a pre-trained network has also been demonstrated to be very effective. In the domain of document images these features have shown better performance for word spotting [90, 166, 168, 186], recognition [130], document classification [65], layout analysis [28], etc. In this work, we propose a deep CNN architecture named as HWNet v2, for the task of learning an efficient word level representation for handwritten documents which can handle multiple writers and, is robust to common forms of degradation and noise. We also show the generic nature of the proposed representation and architecture which allows it to be used as an off-the-shelf feature for printed documents and in building state-of-the-art word spotting systems for various languages. In order to derive a compact representation for an efficient storage and retrieval, we evaluate the performance of compressed feature codes, where we push the compression to an order of 16-to-32 dimensions with minimal drop in performance. Fig. 3.1 shows sample word images which are considered as nearest neighbors in the proposed representation space. The shown images are quite challenging in terms of handwriting variations, distortion created due to scanning, and irregular segmentation which are common in historical manuscripts. These images are taken from the test sets of the datasets used in this work, which are explained more in detail in Section 3.6.



(a) (b)

Figure 3.2 (a) Distribution of words in the IAM dataset. Here the 'x' axis corresponds to different words in the vocabulary which is sorted (in descending order) based on its frequency in the corpus. This is referred as 'Rank', while the 'y' axis shows the actual frequency in log units. (b) Sample word images from the IIIT-HWS dataset created as part of this work, to address the lack of training data for learning deep CNN networks.

3.2 Handwritten Synthetic Dataset

Quality data [36,43] has always played a pivotal role in the advancement of pattern recognition problems. Some of the key properties for any dataset are: (i) a good sample distribution which mimics the real world unseen examples, (ii) quality of annotation, and (iii) scale. With the success of deep learning based methods [73, 93, 158, 171], there has been a surge in newer supervised learning architectures which are ever more data hungry. These architectures have millions of parameters to learn, thereby need a large amount of training data to avoid over-fitting and to generalize well. In general, data creation is a time consuming and expensive process which requires huge human efforts. More recently, an alternative form of data generation process with minimal supervision is getting popular [73, 140, 145], which uses synthetic mechanisms to render and annotate images in an appropriate form. The simple idea of generating data synthetically allows overcoming the challenges in obtaining the data. In this work, we address the need for large scale annotated datasets for handwritten images by generating synthetic words with natural variations. Fig. 3.2 (b) shows sample handwritten word images generated using the proposed framework which looks quite natural and comparable with its counterparts from the real world which are shown in the last column of the figure.

Some of the popular datasets in handwritten domain are IAM handwriting dataset [109], George Washington [45, 106], Bentham manuscripts [27], Parzival database [45] etc. Except for IAM, the remaining datasets are part of the historical collections which were created by one or very few writers.

IAM is a relatively modern dataset, which consists of unconstrained text written in forms by around 657 writers. The vocabulary of IAM is limited to nearly 11K words whereas any normal dictionary in the English language would contain more than 100K words. Fig. 3.2 (a) shows the distribution of entire words in IAM vocabulary which follows the typical Zipf's law [196]. As one can notice that, out of 11K words, nearly 10.5K word classes contain fewer than 20 samples or instances. Also, the majority of remaining words are stop words which are shorter in length and are less informative. The actual samples in training data are much smaller than this, which limits building efficient deep learning networks such as [93].

3.2.1 Handwritten Font Rendering

We use publicly available handwritten fonts for our task. The vocabulary of words is chosen from a dictionary. For each word in the vocabulary, we randomly sample a font and render[1] its corresponding image. During this process, we vary the following parameters: (i) kerning level (inter character space), (ii) stroke width, from a defined distribution. In order to make the pixel distribution of both foreground (F_g) and background (B_g) pixels more natural, we sample the corresponding pixels, for both regions from a Gaussian distribution where the parameters such as mean and standard deviation are learned from the F_g and B_g region of IAM dataset. Finally, Gaussian filtering is done to smooth the rendered image.

3.2.2 IIIT-HWS Dataset

To address the lack of data for training handwritten word images for English, we build a synthetic handwritten dataset of 1 million word images. We named this dataset as IIIT-HWS [89]. Some of the sample images from this dataset are shown in Fig. 3.2 (b). Note that these images are very similar to natural handwriting. The IIIT-HWS dataset is formed out of 750 publicly available handwritten fonts. We use the popular Hunspell dictionary and pick a unique set of 90K words for this purpose. For each word, we randomly sample 100 fonts and render its corresponding image. Moreover, we prefer to learn a case-insensitive model for each word category, hence we perform three types of rendering, namely, all letters capitalized, all letters lower, and only the first letter in caps.

[1]We use ImageMagick for rendering the word images. URL: `http://www.imagemagick.org/script/index.php`

Figure 3.3 Top two rows show the variations in handwritten images, the bottom two rows demonstrate the challenges of intra class variability in images across writers.

3.3 HWNet

In the quest for learning better holistic features for word images, we leveraged recent CNN architectures to learn discriminative representations. Fig. 3.3 demonstrates the challenges across writers. The top two rows show the variations across images in which some are even hard for humans to read without enough context of nearby characters. The bottom two rows show different instances of the same word written by the different writers, e.g., "inheritance" and "Fourier", where one can clearly notice the variability in shape for each character in the word image. The learned representation needs to be invariant to (i) both inter and intra class variability across the writers, (ii) presence of skew, (iii) quality of ink, and (iv) quality and resolution of the scanned image. One can also notice that there can be instances where few characters are completely distorted or degraded due to the cursive nature of word formation. However, the knowledge of vocabulary and its overall appearance, humans can still make out the word. One of the key differentiation of word images with respect to natural scene images is that a word image is inherently a variable length representation and making it fixed size would distort the individual characters non-uniformly.

We propose a deep CNN architecture named as HWNet, first presented in [90] for learning representation for handwritten word images, and further enhance the network capacity to address issues specific to handwriting. The improved network architecture is one of the major contributions of this work which is named as HWNet v2. We formulate the problem as word classification on a given vocabulary, however, given a trained network, we are interested to derive holistic features $f \in \mathbb{R}^d$ which obeys lexical similarity to all possible words in that language (irrespective of the trained vocabulary).

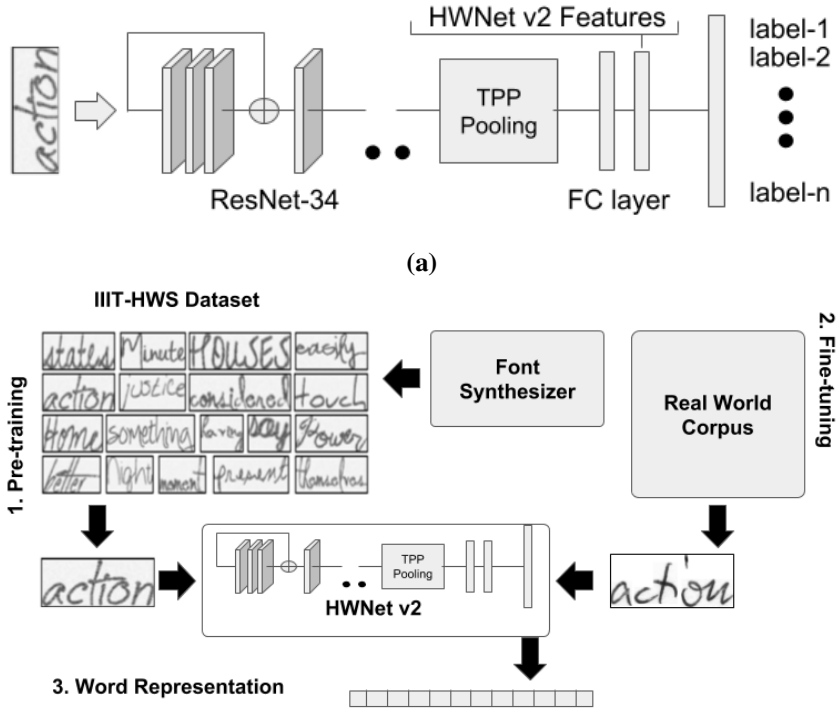


Figure 3.4 (a) HWNet v2 architecture which comprises of a deep CNN architecture using ResNet blocks along with a TPP pooling and fully connected layers, (b) Flowchart showing the transfer learning process where we first pre-train the network on synthetic data and later fine-tune it on real corpus. The features are extracted from the penultimate layer of the network.

3.3.1 HWNet Baseline Architecture

In the baseline model, we use a fixed sized gray scale word image of dimension 48×128. As noted earlier, this would result in distortion of aspect ratio, however, we show in experiments, the learned features are robust, and quite better than previous variable size representations. The underlying architecture of our CNN model is inspired from [93]. We use a CNN with five convolutional layers with 64, 128, 256, 512, and 512 square filters with dimensions: 5, 5, 3, 3 and 3 respectively. The next two layers are fully connected ones with 2048 neurons each. The last layer uses a fully connected (FC) layer with dimension equal to the number of classes (vocabulary of training and validation set), and is further connected to the softmax layer to compute the class specific probabilities. Rectified linear units (ReLU) are used as the non-linear activation units after each weight layer except the last one, and 2×2 max pooling is applied after first, second, and fourth convolutional layers. We use a stride of one

Conv1	Block1	Block2	Block3	Block4	ROI/TPP	FC
3x3	[3x3,64] x 3	[3x3,128] x 4	[3x3,256] x 6	[3x3,512] x 3	ROI{6x12} TPP{1,2,3}	[2048] x 2

Table 3.1 Summary of the HWNet v2 network configuration. The width, height, and number of channels of each convolution layer are shown in square brackets, with the number of layers that are stacked together. We present two variations in the network (as shown in the sixth column), using a single level ROI pooling or using temporal pyramid pooling (TPP) with three levels.

and padding is done to preserve the spatial dimensionality. We empirically observed that using batch normalization [71] after each convolutional and fully connected layer, resulted in lower generalization error as compared to dropouts. We use cross entropy loss function to predict the word class labels, and the weights are updated using the mini batch gradient descent algorithm with momentum.

3.4 HWNet v2

In our original HWNet architecture, we limited the number of convolutional layers to five, which was equivalent to layers proposed in AlexNet. More recently, newer architectures such as VGGNet [158], GoogLeNet [171], and ResNets [69] have shown deeper CNN networks for better performance and the resulting features to be more discriminative. Some of the key architectural changes brought in these networks, which lead to efficient training are: (i) use of lower dimensional filters (3×3) thereby having less parameters from larger sized filters, which also acts as a forced regularizer, (ii) use of (1×1) filters which acts as dimensionality reduction unit to keep the no. of parameters in control, (iii) use of inception layer [171] which introduces multi-scale processing by having multiple parallel layers operating at different scales, and (iv) use of residual blocks [69] to learn residual function $\mathcal{F}(x) := \mathcal{H}(x) - x$. Here $\mathcal{H}(x)$ is the desired underlying mapping. A residual layer is typically implemented using a shortcut connection without any parameters. In our improved HWNet architecture, named as HWNet v2, we use the ResNet34 [69] network with four blocks where each block contains multiple ResNet modules. Instead of using global average pooling (as proposed along with ResNet architecture [69]), we found fully connected layers in the end for learning better features from the penultimate layer. Table 3.1 shows the summary of HWNet v2 network configuration. Here each ResNet module consists of two convolutional layers and a shortcut connection to enable residual learning. There is no max pooling in

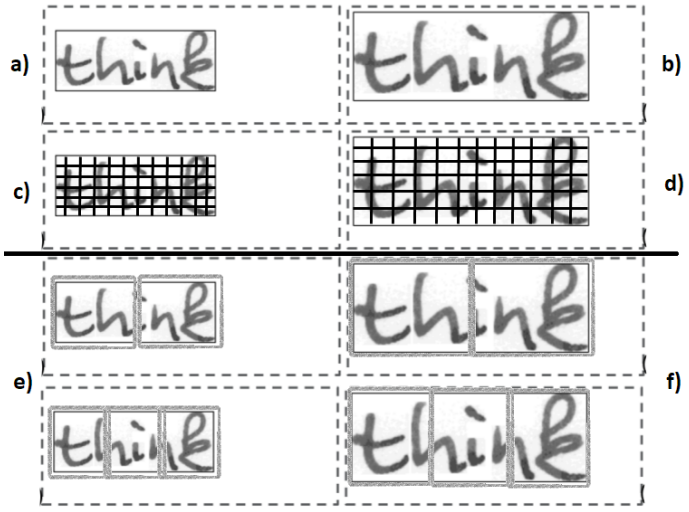


Figure 3.5 (a-b) Multi-scale input, (c-d) region of interest pooling and, (e-f) temporal pyramid pooling shown at levels 2 and 3.

the network and the spatial resolution is down sampled using a stride of 2 at the first convolutional layer of block 2, 3 and 4. As stated earlier, we also use batch normalization after each convolutional and fully connected layer except the last one.

3.4.1 Multi-Scale Training and ROI/TPP Pooling

One of the major limitations in the previous architecture was the requirement to use fixed dimensional inputs so that it remains compatible with fully connected layers of the network. However as mentioned earlier, this leads to distortion of aspect ratio which manipulates the appearance of characters present in the word images arbitrarily. Another aspect which we tend to ignore so far due to the restriction of fixed size is, the ability to train word images in multiple scales so that the network remains invariant to multiple character scales. It has been observed that different writers typically write at different scales which leads to the presence of a variable sized sequence of characters. To overcome these issues, we use a fixed size padded image (128×384) to accommodate variable sized input word image. As shown in Fig. 3.5(a-b), we render different scale input image to learn scale invariant representation. Given the output feature maps from the last convolutional layer, we only keep the activations coming from the input region belonging to word image using ROI pooling. ROI pooling [51], layer gives a differentiable pooling (max/average) mechanism from variable sized input feature maps into fixed sized output maps, by constructing a grid with variable sized cells. Fig. 3.5(c-d), shows the ROI pooling where the number

of grids in both images remains same while the size of each grid cell varies as per the scale of the image. In another variant of the HWNet v2 network, we use temporal pyramid pooling (TPP) similar to [167]. Both ROI and TPP takes variable length representation and produces a fixed length output depending on the number of grids. In our usage of TPP along HWNet v2, we follow a similar paradigm of creating pyramid levels only vertically [154, 167] with levels set at 1, 2 and 3. This would essentially capture the temporal properties present in the word image. Fig. 3.5(e-f), shows the temporal pyramid pooling at level 2 and 3. We also set max pooling as the preferred pooling operation within each grid for both ROI and TPP variants. In general, both ROI, TPP pooling methods don't require a fixed sized padded image. We made such a decision from an implementation point of view so that we can train in batches of images with the fixed size which wouldn't be possible otherwise.

3.4.2 Data Augmentation and Elastic Distortion

While training a CNN network, data augmentation [93] is a common practice to introduce artificial variations in data to make network robust to intra-class variations and prevent over-fitting. Popular data augmentation techniques are random crops, horizontal reflection, random flipping of pixels, and affine transformations such as scaling and translation. In this work, while training HWNet v2, we perform two major augmentation schemes which are: (i) affine transformation and (ii) elastic distortion. In affine transformation, we generalize to translation, scaling, rotation, and shearing. Here rotation and shearing are restricted to certain angles which mimic the skew and cursiveness present in natural handwriting. Elastic distortion [157] has been used in the past successfully for recognizing handwritten digits. It mimics variations created from the oscillation of hand and inertia exerted on the writing medium. The basic idea is to generate a random displacement field which dictates the computation of new location to each pixel through interpolation. The displacement field is smoothed using a Gaussian filter of standard deviation σ and scaled using constant factor α. Both σ, α are set empirically by visualizing the quality of distorted images. Fig. 3.6 shows different possible variations created for each word image.

3.4.3 Curriculum Learning

Training with a large number of classes (10K for both HWNet and HWNet v2) typically results in slow convergence. To avoid such scenarios, we use the strategy from curriculum learning [19], where we start the training process from synthetic images showing easy examples (based on the number of characters) first and harder later. We also perform an incremental learning scheme where at initial

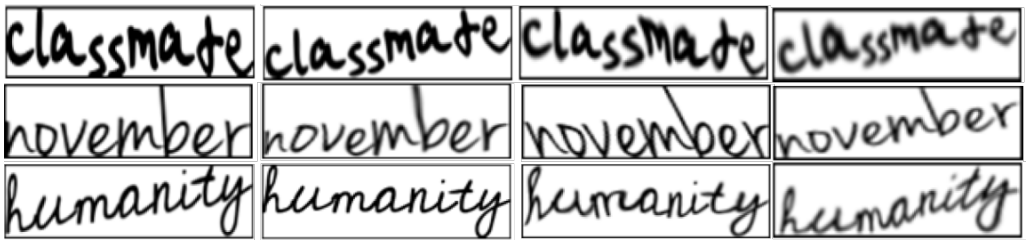


Figure 3.6 Data augmentation techniques: affine and elastic distortion.

epochs, we limit the number of classes to 500 and gradually increase the classes after achieving partial convergence. While increasing the number of classes, we copy the weights from last trained network and randomly initialize the newer weights. This improves the training process in terms of faster convergence in the presence of huge data (#classes).

3.4.4 Transfer learning

It is well-known that off-the-shelf CNNs [39, 135] trained for a related task could be adapted or fine-tuned to obtain reasonable and even state-of-the-art performance for new tasks. In our case, we prefer to perform transfer learning from the synthetic domain (IIIT-HWS) to real world setting. In general, real world handwritten labeled corpora are not large enough to train such deep networks which contain millions of parameters and can easily over-fit on smaller datasets. Moreover, the use of synthetic corpora would give us better vocabulary coverage and also capture some frequent patterns in ngrams which commonly occur in a particular language. In our case, transfer learning achieves to reduce the domain gap between the synthetic and real world data. We employ a similar approach as presented in [193], to do a careful study in transferring feature at a particular layer. The details of the study are presented in the experimental Section 3.6.4.

3.5 Visualizations

One of the most intriguing question while using a deep network is "what is the designated behavior of a neuron trained at a particular layer?". The question is more relevant since we are dealing with a large scale training machine with millions of parameters. Some of the recent works [52, 104, 173, 194, 195] partly answer this question with meaningful insights on what happens behind the scenes.

Figure 3.7 Visualizations: (a) Layer 1 weights, (b) Four possible reconstructions [104] of sample word images shown in columns. These are reconstructed from the representation obtained from the penultimate layer, and (c) Visualization of the strongest activation [52] region of a particular neuron (each column refers to one neuron) of an intermediate convolutional layer. These regions are highlighted using a bounding box inside the word image. Here we notice that, in most of the cases, each neuron focus on detecting a semantically meaningful unit.

Fig. 3.7 shows the visualization of the trained HWNet v2 architecture. It is easy to visualize weights for layer 1 since the filter dimension is $64 \times 1 \times 3 \times 3$ where we have 64 output channels (# filters), 1 is the channel size since the input image is gray scale, and the spatial dimension of the 3×3 filters. Fig. 3.7 (a) visualizes these weights which bear a resemblance to the Gabor filters and detects edges in different orientations. Visualizing neurons after layer 1 is non-trivial since the receptive field of these neurons keeps exploding and the filters are present in higher dimensions. Here we interpret these neurons using the non-parametric technique proposed in [52], which probes from the maximum neuron activations and visualizes the receptive fields in the original image. Fig. 3.7 (c) shows such interesting patches which

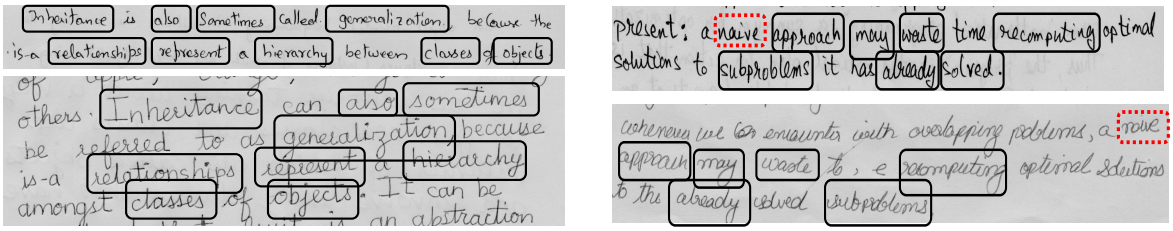


Figure 6.6 Qualitative results of the MODS matching algorithm from the HW-DocSim dataset. Here we show two sample matching pairs in two columns. The top region is taken from the source and the bottom one is plagiarized. The highlighted words in rectangle have been correctly matched along with few words which remain undetected.

6.1.3 Results and Discussion

We now establish two baselines for comparison. Our first approach uses a classical visual bag of words (BOW) approach computed at the interest points. The BOW representation has been successfully used in many image retrieval tasks including the document images [155, 189]. We use SIFT descriptors, quantized using LLC and represented using a spatial pyramid of size 1×3. Our second baseline (NN) uses the classical word spotting scheme based on profile features similar to [133]. While the first one is scalable for large datasets, the second one is not appropriate due to the time complexity of classical DTW. In both these methods, the best match is identified as the document which has the most number of word/patch matches. Table 6.1 reports the quantitative evaluation for various matching schemes along with the baselines. The proposed MODS framework along with HWNet features [90] performs better in both evaluation measures consistently. Using SWM word matching scheme over the proposed HWNet features, we achieve a $nDCG$ score of 0.8569 and AUC of 0.9465. This is further improved in the MODS framework, which incorporates loose ordering and is invariant to word overflow problems. Note that, in both cases (SWM and MODS), the stopwords are removed as preprocessing. We also evaluate our framework with the word attribute framework proposed in [9] and observe a similar trend that validates the effectiveness of deep features and the matching scheme provided by MODS. Fig. 6.6 shows some qualitative results of matching pairs from the HW-DocSim dataset.

6.1.4 Human Evaluations

To validate the performance of the system on an unrestricted collection, we introduce the HW-1K dataset which is collected from the real assignments of a class as part of an active course in the university.

The dataset contains nearly 1K handwritten pages from more than 100 students. The content in these documents varied text, figures, plots, and mathematical symbols. Most of the documents follow a complex layout with misalignment in paragraphs, huge variations in line and word spacing and a high degree of skewness over the content. The scanned images also possessed degradation in quality due to loose handling by the students which created folds and noise over the paper. We perform a human evaluation where we picked a set of 50 assignment images written by different students, and gathered the top-1 similar document image present in the corpus using MODS. We asked five humans evaluators to give a score to the top-1 retrieval on a Likert scale of $0-3$ where 0 is "*very dis-similar*", 1 is "*similar only for few word matches*", 2 is "*partially similar*" and 3 is "*totally similar*". Here, the scale-1 refers to the case where the document pair refers to the same topic. Thus there could be individual word matches but the text is not plagiarized. The average agreement to the human judgments as evaluated for the top-1 similar document is reported at 2.356 with 3 as the best score.

6.2 Word Spotting on Lecture Videos

Lecture videos are getting abundant with the increasing interest in e-learning in the form of Open-CourseWare (OCW) lectures and Massive Open Online Courses (MOOCs). Text is present almost everywhere in a lecture video; particularly in lectures on Science, Mathematics and Engineering. Text alone could be used for a variety of of tasks like keyword generation, video indexing and enabling search and extracting class notes [97, 177, 183, 190]. Text in lecture videos comprises of handwritten text written on a blackboard or a paper, text written using a stylus on a tablet and displayed on a screen or font rendered text appearing in presentation slides (digital text). Lectures are recorded using one or more cameras, and the camera(s) are typically positioned to directly face the blackboard or the presentation slides. Usually text recognition from presentation slides is less challenging as the text is more legible, and with little variations in style. However, the text on blackboards, whiteboards, and on paper is handwritten and not very legible due to poor lighting, smaller size or poor contrast. In this section, we present an application of our representation in searching text inside such lecture videos. This work is part of the larger work around benchmarking the problem of localization and recognition of text inside lecture videos and is first published in [42]. In this application, we limit our discussion on word spotting experiment using word embeddings as proposed in Chapter 4, Section 4.4.

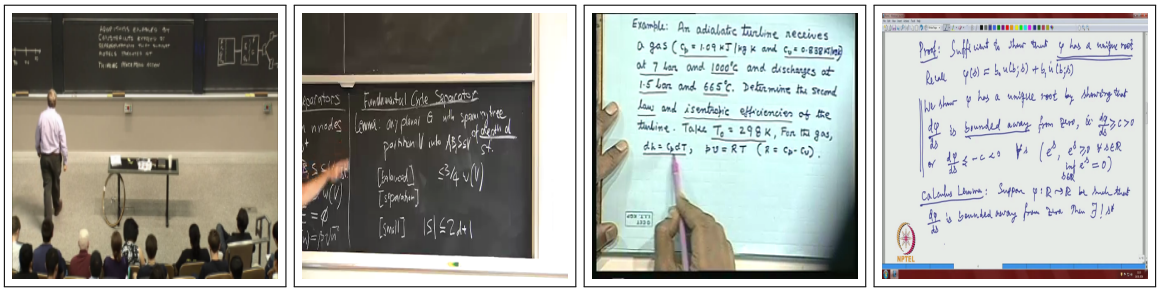


Figure 6.7 Sample frames from the LectureVideoDB datatset. Scenes cluttered with text and figures, black boards where text background contrast is poor, low resolution images and less legible handwriting make the text recognition harder in lecture videos.

6.2.1 Datasets and Evaluation Protocol

We use the Lecture Video (LectureVideoDB) Dataset created as part of the work presented in [42]. The dataset is created from course videos of 24 different courses across science, management and engineering. These courses are offered by e-learning initiatives such as MIT OCW [2], Khan Academy [1] and NPTEL [3]. Fig. 6.7 shows sample frame images as part of this dataset. Table 6.2 describes the breakdown of the LectureVideoDB across four modalities: slides, whiteboard, paper, and blackboard. The ground truth information contains both the bounding box coordinates for each word image and its text for each sampled frame taken from the video. Since we are interested only on the word spotting task, we use the ground truth bounding box coordinates as the segmentation proposal. In addition to LectureVideoDB, we also use IAM handwriting database [109] for the purpose of pre-training. As described in our earlier chapters, the dataset consists of contributions from over 657 writers and comprises of 115,320 words in English.

We follow the evaluation protocol for word spotting as presented in [9] using the train/val/test splits created for the LectureVideoDB dataset. We conduct both query-by-string (QBS) and query-by-example

Type	#Frames	#Words	#Writers
Slides	1145	52225	5
Whiteboard	945	21160	7
Paper	1281	27900	9
Blackboard	2103	36460	14

Table 6.2 Details of the LectureVideoDB Dataset [42]. Here Type refers to the presentation/writing medium used by the instructor.

Train Dataset	mAP		Data-Split	mAP	
	QBE	QBS		QBE	QBS
IAM	0.4311	0.4531	Slides	0.7272	0.7157
			Whiteboard	0.3628	0.3234
			Paper	0.2165	0.3882
			Blackboard	0.0721	0.2156
LectureVideoDB	**0.7909**	**0.7404**	Slides	0.8726	0.7977
			Whiteboard	0.6205	0.4799
			Paper	0.8035	0.8037
			Blackboard	0.7151	0.7028

Table 6.3 Word Spotting performance using end2end embedding architecture on the various splits of LectureVideoDB dataset.

(QBE) on the test corpus. For QBE setting, the queries are the subset of words taken from the test corpus having a frequency more than 1. However all the words are kept in the retrieval set. For QBS scenario, we take the unique set of strings in the test set as queries. In both cases, we report the mean average precision value (mAP) which is standard measure for a word spotting task. We also removed stopwords from the query set and the performance is evaluated in a case-insensitive manner.

6.2.2 Results

As mentioned earlier, we use the embeddings computed from our End-2-End embedding network for representing both word images and the textual strings. Table 6.3, presents the quantitative results of word spotting on LectureVideoDB dataset under both QBE and QBS setting. Here we first evaluate the baseline performance of the pre-trained End2End embedding network on IAM train set. As one can observe the performance is quite inferior where we report the QBE and QBS mAP of 0.4311 and 0.4531. We believe this is due to the huge biasness in the training set of IAM where the images are only written on white paper with ink pens, whereas the LectureVideoDB dataset contain various other modalities such white text on blackboards, text on slides with different colors etc. In our next experiment we fine-tuned the network using the training samples from the LectureVideoDB to remove this biasness. Here we observe a clear improvement of performance with QBE being reported at 0.7909 and QBS of 0.7404 which is encouraging. However, the performance still does not come close with the level of performance of word spotting on the IAM test set. We attribute this to the complexity of the underlying problem for

Figure 6.8 Qualitative results of word spotting on LectureVideoDB dataset. Notice the robustness of the representation in-spite of wide diversity in the quality of images due to different writing mediums.

spotting text in lecture videos due to the presence of multiple modalities along with the challenges posed by image capture and low resolution videos of some courses.

Fig. 6.8 presents sample qualitative results from of the word spotting using QBS. One can notice the diversity of results where the word images are from different modalities, resolutions and degradations. Also note the robustness in retrieving certain word images such as “orthogonal” where the word image is highly skewed in nature.

6.3 Word Retrieval on Indic Scripts

In our final application, we extend and validate the representations proposed as part of this thesis to Indic scripts. In this work, we present the word retrieval system for two popular Indic scripts – Devanagari and Telugu. A lot of documents in these scripts have been made available by Digital Library of India (DLI) [11] which contains historical documents, ancient manuscripts and literary resources with cultural significance. Devanagari and Telugu are among the two most popular Indian scripts. Unlike Latin script, Indic scripts contains higher number of character set, the concept of modifiers (consonant is modified in-case it is followed by a vowel), and compound characters (a consonant follows one or more consonant(s), a new character gets formed). These challenges creates complexities in understanding text from Indic documents. For more details about the Devanagari and Telugu scripts, interested reader can refer to [120].

6.3.1 Digital Library of India (DLI)

Digital Library of India (DLI) has emerged as one of the largest collections of document images in Indian scripts [11]. DLI, as a part of Million Book Project (MBP), has contributed to the free access of knowledge to Billions of people. In addition, it also helped in digitally archiving the rare and precious books in many of the Indian languages. DLI archives printed content (mostly books and journals) as scanned document images. A major challenge presently faced by the DLI is the lack of content level access to the individual pages. As it stands, content (in Indian languages) can be primarily accessed only by the meta data of the books, which is manually created. These books are available as binary images, and the binarization has already resulted in significant cuts and merges in the word images. We argue that, even if a reliable high performance OCR is not readily available, effective search is still possible. In our experiments, we use a subset of DLI corpus which we annotated as part of our work in [92]. The details are given below:

DLI Hindi and Telugu [92]: These two datasets, belonging to Hindi (Devanagari script) and Telugu languages from Indic scripts and are part of the DLI project. We take one such subset [92] which was annotated at the level of lines and words and referred to as HS1 and TS1 datasets.

English-1601 [189]: In addition to printed Indic scripts, we also pick a English book for our experimentation. This is a book in English titled "Adventures of Sherlock Holmes" written by Arthur Conan Doyle. This was first used in [189] for comparing OCR based results with image search. Note that this book is not from the DLI corpus. Table 6.4 lists outs the details of the above three dataset.

6.3.2 Results

We validate our representations (Lexical and Semantic) which are discussed in Chapters 3,4, and 5 on above datasets. The evaluation protocol remains the same. For lexical representation, we follow the rules of word spotting as mentioned in Chapter 4, Section 4.5.1. While for semantic evaluation we conduct

Dataset	**#Pages**	**#Words**
English-1601	310	1,13,008
DLI Hindi (HS1)	1,533	4,20,100
DLI Telugu (TS1)	1,005	1,61,265

Table 6.4 The list of printed datasets used in this work. Here both Hindi and Telugu datasets are taken from Digital Library of India [11] corpus.

Method	Supervision	English		Hindi		Telugu	
		mAP-QBE	mAP-QBS	mAP-QBE	mAP-QBS	mAP-QBE	mAP-QBS
Yalniz et.al [189]	No	0.9300	-	-	-	-	-
Krishnan et.al [92]	No	-	-	0.6055	-	0.7438	-
HWNet v2	Yes	0.9570	-	0.9509	-	**0.9582**	-
HWNet v3	Yes	**0.9939**	**0.9921**	**0.9537**	**0.9659**	0.9499	**0.9474**

Table 6.5 Quantitative evaluation of word spotting on printed datasets.

Figure 6.9 Qualitative results of query-by-string word spotting on Hindi, and Telugu datasets. The query is shown in the left most column and the word images shown on the right are the retrieved images in the ranked order.

only the word analogy task and exact evaluation which are mentioned in Chapter 5, Section 5.4.1. For better understanding of the notations used, readers are requested to refer these respective sections.

6.3.2.1 Word Spotting using Lexical Representation

Table 6.5 presents the quantitative results of word spotting on the printed datasets. Here the methods proposed in Yalniz et al. [189] and Krishnan et al. [92] uses an unsupervised bag of word framework for deriving a holistic word-level descriptor. These methods are not directly comparable with HWNet based methods which are learned in a supervised fashion. However such comparison highlights the superiority of supervised methods. Note that the previous methods have not reported query-by-string results since those methods didn't embed textual strings into the representation space. From the results, we notice

Method	Supervision	English			Hindi			Telugu	
		WA	mAP-Exact		WA	mAP-Exact		mAP-Exact	
			QBE	QBS		QBE	QBS	QBE	QBS
fastText [59]	-	0.8140	-	-	0.5714	-	-	-	-
Yalniz et.al [189]	No	-	0.9300	-	-	-	-	-	-
Krishnan et.al [92]	No	-	-	-	-	0.6055	-	0.7438	-
HWNet v2	Yes	-	0.9570	-	-	0.9509	-	**0.9582**	-
HWNet v3	Yes	-	**0.9939**	**0.9921**	-	**0.9537**	**0.9659**	0.9499	**0.9474**
Sem-Rank	Yes	0.7901 $\pm$0.003	0.9905	0.9566	0.5321 $\pm$0.018	0.9428	0.9359	0.9538	0.9461

Table 6.6 Semantic evaluation on printed datasets from English, Hindi and Telugu languages.

Figure 6.10 Qualitative results of normalized and semantic word spotting on Hindi, and Telugu datasets. Note that, while showing the nearest neighbors, we have removed the consecutive similar word images to emphasize the distances among different lexical word images.

that the performance of word spotting has improved significantly on all these datasets and for printed English, one could directly use HWNet v2/v3 features as off-the-shelf for various document tasks. The improvement of results in Hindi and Telugu also suggests that such an architecture can be used for various other languages with wide variations in scripts and language constructs. Fig. 6.9 presents the qualitative results of word spotting in Indic scripts. The first four rows shows query-by-example (QBE) results and the last four rows shows query-by-string (QBS) results.

6.3.2.2 Semantic Evaluation

Table 6.6 presents the quantitative analysis of the proposed semantic representation from Chapter 5. As mentioned in that chapter, we use word analogy measure from textual word embedding literature to evaluate our representation. For English, we use the word analogy dataset introduced in [112], while

for Hindi we use the one provided in [59]. To our knowledge there does not exist a similar dataset for Telugu language. We also report the mAP-Exact measure to observe the drop in performance for lexical (verbatim) word spotting. To embed word images into semantic space, we used fastText pre-trained models [59] for English, Hindi and Telugu languages.

The first method reported in the table (fastText) evaluates our ground truth representation under word analogy task. Here, for the analogy task (WA), we only take those words (analogy question pairs) which are present in our test corpus of each dataset. The numbers reported under fastText could be taken as the upper bound for our proposed semantic representation for word images. The next three rows reports the word spotting performance under lexical representations. The last row (Sem-Rank) reports the evaluation of the proposed semantic representation from Chapter 5 under both word analogy and mAP-Exact measures. We observe that in terms of analogy performance (WA), the gap between the fastText and Sem-Rank is quite closer than the one reported for the English in handwritten domain. Also, one can see that the drop in mAP-Exact performance for Sem-Rank quite minimal as compared to HWNet v3. Fig. 6.10 presents qualitative results for Hindi and Telugu language word images under semantic word spotting. Note that the existing dataset is binary in nature and contains degradations due to the binarization process. Here the top results are meaningful and semantically coherent in nature.

6.4 Summary

In this chapter, we bring out three different applications using the representations proposed in this thesis. In our first application, we propose the MODS method which estimates a measure of similarity for two handwritten documents. In the next two applications, we validate the performance of our representation in building retrieval system under newer modalities such as lecture videos and printed documents from Indic scripts. Under both scenarios, we are able to successfully demonstrate a promising system which is highly useful to the community.

Chapter 7

Conclusion and Future Work

In this thesis, we presented different representation schemes for word images that respect the lexical and semantic properties of a word. The word image representation, denoted by a vector of real-valued numbers could be thought in a similar manner to what the ASCII or UNICODE representation does for the text. We validated our primary argument that one could learn such representations in a data-driven manner where the reliance on the annotated dataset can be minimized using the synthetic data. In this chapter, we present the summary of the thesis, its contributions, impact in the document community and also list out few important ideas to pursue in the future.

7.1 Summary

The thesis focuses on the problem of learning efficient representations for word images from scanned handwritten and machine-printed documents. The proposed representations are compact in size and encode the content information while remaining invariant to different styles, fonts, degradation, and noise. Our learning framework is supervised in nature and generic to different scripts and languages. The thesis presents two broad categories of representation: (i) lexical, and (ii) semantic. In our primary contribution, we introduced the learning architecture HWNet in Chapter 3 which provides an efficient word image representation in lexical space. Our architecture is one of the initial deep convolutional neural networks proposed in the community for word representation learning. In this work, we also release a large scale synthetic word image dataset (IIIT-HWS) useful for pre-training the deep network and generalize well on real-world handwritten collections. In our next contribution, in Chapter 4 we presented different ways of jointly embedding word images and its corresponding textual strings onto a common subspace. The common word representation across image and text modality allows both

query-by-text/image and also enables word recognition under constrained lexicon. In Chapter 5, we introduced the novel problem of learning semantic representation from word images. The representation encodes the semantics of the word and helps to identify synonyms, words related to common root and essentially reduces the vocabulary gap that exists between the query and the retrieved results. We presented detailed evaluation and visualization studies to understand different representations. Under lexical representation, we evaluated for the tasks of word spotting and word recognition, whereas for semantic representation we used word similarity and analogical tests. The HWNet v2 and HWNet v3 representation report state of the art results for most of the standard handwritten and printed datasets. Finally, in Chapter 6, we presented different use cases built using these representations. Here at first, we demonstrated an interesting case study on the problem of matching pairs of handwritten document images. The proposed system provides a similarity score on the basis of patterns of text re-usages between document images irrespective of writing style and content re-ordering. We tested our system on real-world handwritten assignments given as part of the course conducted in our university. In our second use case, we demonstrated a keyword retrieval system from online lecture videos. The retrieval system provides a reasonable performance, in-spite of having such a diverse set of writing content on blackboards, whiteboards, paper, slides etc. Finally, we also evaluated our representations for popular Indic scripts (Devanagari and Telugu) to validate the generic nature of our learning framework.

We consider the thesis work to be significant in the document image community. The IIIT-HWS dataset has gained popularity for pre-training deep neural networks in our domain. The learned representations have significantly improved the state of art performance for word spotting and word recognition in handwritten images. With the demonstration of newer applications, we have essentially widened the scope of utilizing such representations for enabling natural language applications. We also consider our works in Indic scripts, under both printed and handwritten domains [41] to be a significant benchmark for the document community in India.

7.2 Future Directions

The thesis opens up many promising directions as listed below:

- **Spotting Words in Complex and Unstructured Documents:** It is quite amazing to see how humans can quickly spot a word within a complex layout document where segmentation is extremely hard. These scenarios are quite common for both modern and historical handwritten documents.

More recently, there have been interesting line of works [187,188] in literature which builds word spotting in a segmentation-free manner. These networks are inspired by the success of region proposal based object detection techniques such as Faster R-CNN [136] and adapted to document images. One could also think of using attention mechanism [60] to process an entire document page and focus onto the region where a particular keyword is likely to be present. Such an integration of detection/attention mechanism within the representation learning framework would allow us to work with complex layout documents.

- **Domain Adaptation:** An important question that intrigues us: Can we build a real-world handwriting recognition or spotting system entirely using synthetic data? Here also we would like to raise our expectation to human levels where we see that humans can generalize well to unseen styles by learning through typewritten fonts. Given the availability of a large amount of annotated synthetic data almost free and a huge corpus of unsupervised images in the target language, there exists a scope to formulate a domain adaptation task which transfers knowledge to the target domain. The solution to such a problem will provide a boost in solving problems in low resource languages, historical documents which typically contain very little or no annotated data.

- **Sub-word Representations:** It would be interesting to explore the granularity of representation at the level of character n-grams/patches by augmenting the representation with the intermediate layer activations. Such representations have recently gathered lots of interest in the natural language community by providing better representations for out of vocabulary words and rare words of a language. We also believe that such an enriched representation would capture local information which would be useful to distinguish between classes with minimum edit distance.

- **Synthetic Data Generation:** Our thesis highlighted the role of synthetic data in training deep networks. It's no doubt that with much more realistic synthetic data, the dependence on real-world annotated data would reduce. The recent success in generative models such as GAN [56], VAE [83], and PixelRNN [178] has provided interesting ways to generate realistic data in a particular domain. We consider this track of research of generating synthetic word/line images in multiple fonts and styles to be a direction worth pursuing.

- **Indic Scripts:** Although, this thesis demonstrates an application of Indic script word spotting, the scope of works that one could pursue within Indian language documents is huge. This includes

working with multiple languages/scripts, various historical documents that are typewritten, handwritten, on palm leaves, etc. There also exists a dire necessity to build better handwriting recognition systems for both offline and online domains. One of the primary challenges to address is the lack of annotated datasets in these languages. Since many of these languages have some sort of shared properties and origins, one could also think of solutions that can exploit these factors. The other challenges exist due to the script complexities and the agglutinative properties of the language. Given its huge community base, simple contributions would also make a huge impact on society.

- **Applications in Educational Domain:** There are many exciting applications in the educational domain where our work is directly applicable. One such important problem is providing content level access from instructional videos which are uploaded online as part of Massive Open Online Courses (MOOC). Extracting content from videos needs one to interpret speech, text written on blackboards, slides, etc. which could be posed as a multi-modal information retrieval problem. In one of our recent works [42], we worked upon the problem of spotting text inside these videos to facilitate text level searching using representations proposed in this thesis. The other set of problems that are interesting in this domain are: how to automatically grade handwritten answer sheets, finding plagiarism in assignments, collect and interpret feedback. We believe that the automation of these tasks could help modern educational institutions cater to a large number of students.

Bibliography

[1] Khan Academy. https://www.khanacademy.org/, Last Retrieved 2018-03-28.

[2] MIT opencourseware. https://ocw.mit.edu/index.htm, Last Retrieved 2018-03-28.

[3] NPTEL. http://nptel.ac.in/, Last Retrieved 2018-03-28.

[4] Project Gutenberg. (n.d.). `www.gutenberg.org`, Last Retrieved 2019-11-24.

[5] C. Adak, B. B. Chaudhuri, and M. Blumenstein. Named entity recognition from unstructured handwritten document images. In *12th IAPR Workshop on Document Analysis Systems, DAS*, pages 375–380, 2016.

[6] D. Aldavert, M. Rusiñol, R. Toledo, and J. Lladós. Integrating visual and textual cues for query-by-string word spotting. In *International Conference on Document Analysis and Recognition, ICDAR*, pages 511–515, 2013.

[7] D. Aldavert, M. Rusiñol, R. Toledo, and J. Lladós. A study of bag-of-visual-words representations for handwritten keyword spotting. *Int. J. Document Anal. Recognit.*, 18(3):223–234, 2015.

[8] J. Almazán, A. Gordo, A. Fornés, and E. Valveny. Segmentation-free word spotting with exemplar svms. *Pattern Recognit.*, 47(12):3967–3978, 2014.

[9] J. Almazán, A. Gordo, A. Fornés, and E. Valveny. Word spotting and recognition with embedded attributes. *IEEE Trans. Pattern Anal. Mach. Intell.*, 36(12):2552–2566, 2014.

[10] F. Almeida and G. Xexéo. Word embeddings: A survey. *CoRR*, abs/1901.09069, 2019.

[11] V. Ambati, N. Balakrishnan, R. Reddy, L. Pratha, and C. Jawahar. The digital library of india project: Process, policies and architecture. In *International Conference on Digital Libraries, ICDL*, 2006.

[12] C. Anderson. Essentials of linguistics. `https://creativecommons.org/licenses/by-sa/4.0/`, Last Retrieved 2019-01-19.

[13] G. Axler and L. Wolf. Toward a dataset-agnostic word segmentation method. In *International Conference on Image Processing, ICIP*, pages 2635–2639, 2018.

[14] H. Azizpour, A. S. Razavian, J. Sullivan, A. Maki, and S. Carlsson. From generic to specific deep representations for visual recognition. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops, CVPRW*, pages 36–45, 2015.

[15] A. Babenko, A. Slesarev, A. Chigorin, and V. Lempitsky. Neural codes for image retrieval. In *European Conference on Computer Vision, ECCV*, pages 584–599, 2014.

[16] A. Balasubramanian, M. Meshesha, and C. V. Jawahar. Retrieval from document image collections. In H. Bunke and A. L. Spitz, editors, *International Workshop on Document Analysis Systems, DAS*, volume 3872, pages 1–12, 2006.

[17] V. Balntas, E. Johns, L. Tang, and K. Mikolajczyk. PN-Net: Conjoined triple deep network for learning local image descriptors. *CoRR*, abs/1601.05030, 2016.

[18] Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin. A neural probabilistic language model. *J. Mach. Learn. Res.*, 3:1137–1155, 2003.

[19] Y. Bengio, J. Louradour, R. Collobert, and J. Weston. Curriculum learning. In *International Conference on Machine Learning, ICML*, volume 382, pages 41–48, 2009.

[20] G. M. BinMakhashen and S. A. Mahmoud. Document layout analysis: A comprehensive survey. *ACM Comput. Surv.*, 52(6):109:1–109:36, 2020.

[21] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, 2003.

[22] T. Bluche, J. Louradour, and R. O. Messina. Scan, attend and read: End-to-end handwritten paragraph recognition with MDLSTM attention. In *International Conference on Document Analysis and Recognition, ICDAR*, pages 1050–1055, 2017.

[23] T. Bluche, H. Ney, and C. Kermorvant. A comparison of sequence-trained deep neural networks and recurrent neural networks optical modeling for handwriting recognition. In *Statistical Language and Speech Processing - Second International Conference, SLSP*, volume 8791, pages 199–210, 2014.

[24] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov. Enriching word vectors with subword information. *Trans. Assoc. Comput. Linguistics, TACL*, 5:135–146, 2017.

[25] S. E. Boquera, M. J. C. Bleda, J. Gorbe-Moya, and F. Zamora-Martínez. Improving offline handwritten text recognition with hybrid HMM/ANN models. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(4):767–779, 2011.

[26] A. Bosch, A. Zisserman, and X. Muñoz. Scene classification via plsa. In *9th European Conference on Computer Vision, ECCV*, volume 3954, pages 517–530, 2006.

[27] T. Causer and V. Wallace. Building A volunteer community: Results and findings from *Transcribe Bentham*. *Digit. Humanit. Q.*, 6(2), 2012.

[28] K. Chen, M. Seuret, M. Liwicki, J. Hennebert, and R. Ingold. Page segmentation of historical document images with convolutional autoencoders. In *International Conference on Document Analysis and Recognition, ICDAR*, pages 1011–1015, 2015.

[29] S. Chopra, R. Hadsell, and Y. LeCun. Learning a similarity metric discriminatively, with application to face verification. In *Computer Vision and Pattern Recognition, CVPR*, pages 539–546, 2005.

[30] O. Chum, J. Philbin, and A. Zisserman. Near duplicate image detection: min-hash and tf-idf weighting. In *British Machine Vision Conference, BMVC*, pages 1–10, 2008.

[31] P. D. Clough and M. Stevenson. Developing a corpus of plagiarised short answers. *Language Resources and Evaluation, LREC*, 45(1):5–24, 2011.

[32] R. Collobert and J. Weston. A unified architecture for natural language processing: deep neural networks with multitask learning. In *International Conference Machine Learning, ICML*, volume 307, pages 160–167, 2008.

[33] G. Csurka, C. Dance, L. Fan, J. Willamowski, and C. Bray. Visual categorization with bags of keypoints. In *Workshop on statistical learning in computer vision, ECCV*, volume 1, pages 1–2, 2004.

[34] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, CVPR*, pages 886–893, 2005.

[35] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. Indexing by latent semantic analysis. *J. Am. Soc. Inf. Sci.*, 41(6):391–407, 1990.

[36] J. Deng, W. Dong, R. Socher, L. Li, K. Li, and F. Li. Imagenet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 248–255, 2009.

[37] P. Doetsch, M. Kozielski, and H. Ney. Fast and robust training of recurrent neural networks for offline handwriting recognition. In *International Conference on Frontiers in Handwriting Recognition, ICFHR*, pages 279–284, 2014.

[38] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *International Conference on Machine Learning, ICML*, volume 32, pages 647–655, 2014.

[39] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *International Conference on Machine Learning, ICML*, volume 32, pages 647–655, 2014.

[40] K. Dutta, P. Krishnan, M. Mathew, and C. V. Jawahar. Improving CNN-RNN hybrid networks for handwriting recognition. In *International Conference on Frontiers in Handwriting Recognition, ICFHR*, pages 80–85, 2018.

[41] K. Dutta, P. Krishnan, M. Mathew, and C. V. Jawahar. Towards spotting and recognition of handwritten words in Indic scripts. In *International Conference on Frontiers in Handwriting Recognition, ICFHR*, pages 32–37, 2018.

[42] K. Dutta, M. Mathew, P. Krishnan, and C. V. Jawahar. Localizing and recognizing text in lecture videos. In *International Conference on Frontiers in Handwriting Recognition, ICFHR*, pages 235–240, 2018.

[43] M. Everingham, L. V. Gool, C. K. I. Williams, J. M. Winn, and A. Zisserman. The pascal visual object classes (VOC) challenge. *Int. J. Comput. Vis.*, 88(2):303–338, 2010.

[44] L. Finkelstein, E. Gabrilovich, Y. Matias, E. Rivlin, Z. Solan, G. Wolfman, and E. Ruppin. Placing search in context: The concept revisited. *ACM Transactions on information systems*, 2002.

[45] A. Fischer, A. Keller, V. Frinken, and H. Bunke. Lexicon-free handwritten word spotting using character HMMs. *Pattern Recognit. Lett.*, 33(7):934–942, 2012.

[46] A. Gandhi and C. V. Jawahar. Detection of cut-and-paste in document images. In *International Conference on Document Analysis and Recognition, ICDAR*, pages 653–657, 2013.

[47] B. Gatos, N. Stamatopoulos, and G. Louloudis. ICDAR2009 handwriting segmentation contest. *Int. J. Document Anal. Recognit.*, 14(1):25–33, 2011.

[48] S. K. Ghosh and E. Valveny. A sliding window framework for word spotting based on word attributes. In *Pattern Recognition and Image Analysis, IbPRIA*, volume 9117, pages 652–661, 2015.

[49] S. K. Ghosh and E. Valveny. Text box proposals for handwritten word spotting from documents. *Int. J. Document Anal. Recognit.*, 21(1-2):91–108, 2018.

[50] A. P. Giotis, G. Sfikas, B. Gatos, and C. Nikou. A survey of document image word spotting techniques. *Pattern Recognit.*, 68:310–332, 2017.

[51] R. Girshick. Fast R-CNN. In *IEEE International Conference on Computer Vision, ICCV*, pages 1440–1448, 2015.

[52] R. B. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Computer Vision and Pattern Recognition, CVPR*, pages 580–587, 2014.

[53] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *International Conference on Artificial Intelligence and Statistics, AISTATS*, volume 9, pages 249–256. JMLR.org, 2010.

[54] L. Gomez-Bigorda, M. Rusiñol, and D. Karatzas. LSDE: levenshtein space deep embedding for query-by-string word spotting. In *International Conference on Document Analysis and Recognition, ICDAR*, pages 499–504, 2017.

[55] L. Gomez-Bigorda, M. Rusiñol, and D. Karatzas. LSDE: levenshtein space deep embedding for query-by-string word spotting. In *International Conference on Document Analysis and Recognition, ICDAR*, pages 499–504, 2017.

[56] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in neural information processing systems, NIPS*, pages 2672–2680, 2014.

[57] I. J. Goodfellow, D. Warde-Farley, M. Mirza, A. C. Courville, and Y. Bengio. Maxout networks. In *International Conference on Machine Learning, ICML*, volume 28, pages 1319–1327, 2013.

[58] A. Gordo, J. Almazán, N. Murray, and F. Perronnin. LEWIS: latent embeddings for word images and their semantics. In *International Conference on Computer Vision, ICCV*, pages 1242–1250, 2015.

[59] E. Grave, P. Bojanowski, P. Gupta, A. Joulin, and T. Mikolov. Learning word vectors for 157 languages. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation, LREC*, 2018.

[60] A. Graves. Generating sequences with recurrent neural networks. *CoRR*, abs/1308.0850, 2013.

[61] A. Graves, S. Fernández, F. J. Gomez, and J. Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the Twenty-Third International Conference, ICML*, volume 148, pages 369–376, 2006.

[62] A. Graves, M. Liwicki, S. Fernández, R. Bertolami, H. Bunke, and J. Schmidhuber. A novel connectionist system for unconstrained handwriting recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, 31(5):855–868, 2009.

[63] A. Graves and J. Schmidhuber. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks*, 18(5-6):602–610, 2005.

[64] M. Gutmann and A. Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, AISTATS*, volume 9, pages 297–304, 2010.

[65] A. W. Harley, A. Ufkes, and K. G. Derpanis. Evaluation of deep convolutional nets for document image classification and retrieval. In *International Conference on Document Analysis and Recognition, ICDAR*, pages 991–995, 2015.

[66] C. G. Harris and M. Stephens. A combined corner and edge detector. In C. J. Taylor, editor, *Alvey Vision Conference, AVC*, pages 1–6, 1988.

[67] Z. S. Harris. Distributional structure. *Word*, 10(2-3):146–162, 1954.

[68] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *IEEE International Conference on Computer Vision, ICCV*, pages 1026–1034, 2015.

[69] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 770–778, 2016.

[70] T. Hofmann. Probabilistic latent semantic analysis. In K. B. Laskey and H. Prade, editors, *Uncertainty in Artificial Intelligence, UAI*, pages 289–296, 1999.

[71] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In F. R. Bach and D. M. Blei, editors, *International Conference on Machine Learning,ICML*, volume 37, pages 448–456, 2015.

[72] T. S. Jaakkola and D. Haussler. Exploiting generative models in discriminative classifiers. In *Advances in Neural Information Processing Systems, NIPS*, pages 487–493, 1998.

[73] M. Jaderberg, K. Simonyan, A. Vedaldi, and A. Zisserman. Synthetic data and artificial neural networks for natural scene text recognition. *CoRR*, abs/1406.2227, 2014.

[74] M. Jaderberg, K. Simonyan, A. Vedaldi, and A. Zisserman. Reading text in the wild with convolutional neural networks. *Int. J. Comput. Vis.*, 116(1):1–20, 2016.

[75] M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu. Spatial transformer networks. In *Advances in Neural Information Processing Systems, NIPS*, pages 2017–2025, 2015.

[76] M. Jaderberg, A. Vedaldi, and A. Zisserman. Deep features for text spotting. In *European Conference on Computer Vision, ECCV*, volume 8692, pages 512–528, 2014.

[77] K. Järvelin and J. Kekäläinen. Cumulated gain-based evaluation of ir techniques. *ACM Transactions on Information Systems, TOIS*, 20(4):422–446, 2002.

[78] H. Jégou, M. Douze, C. Schmid, and P. Pérez. Aggregating local descriptors into a compact image representation. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 3304–3311, 2010.

[79] M. Juneja, A. Vedaldi, C. V. Jawahar, and A. Zisserman. Blocks that shout: Distinctive parts for scene classification. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 923–930, 2013.

[80] Y. Ke, R. Sukthankar, and L. Huston. An efficient parts-based near-duplicate and sub-image retrieval system. In *ACM International Conference on Multimedia, ACM-MM*, pages 869–876, 2004.

[81] Y. Kim. Convolutional neural networks for sentence classification. In A. Moschitti, B. Pang, and W. Daelemans, editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP*, pages 1746–1751. ACL, 2014.

[82] Y. Kim, Y. Jernite, D. A. Sontag, and A. M. Rush. Character-aware neural language models. In D. Schuurmans and M. P. Wellman, editors, *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, pages 2741–2749, 2016.

[83] D. P. Kingma and M. Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations, ICLR*, 2014.

[84] A. Kovalchuk, L. Wolf, and N. Dershowitz. A simple and fast word spotting method. In *International Conference on Frontiers in Handwriting Recognition, ICFHR*, pages 3–8, 2014.

[85] P. Krishnan, K. Dutta, and C. V. Jawahar. Deep feature embedding for accurate recognition and retrieval of handwritten text. In *International Conference on Frontiers in Handwriting Recognition, ICFHR*, pages 289–294, 2016.

[86] P. Krishnan, K. Dutta, and C. V. Jawahar. Word spotting and recognition using deep embedding. In *IAPR International Workshop on Document Analysis Systems, DAS*, pages 1–6, 2018.

[87] P. Krishnan and C. Jawahar. Bringing semantics into word image representation. *Pattern Recognition*, 108:107542, 2020.

[88] P. Krishnan and C. V. Jawahar. Bringing semantics in word image retrieval. In *12th International Conference on Document Analysis and Recognition, ICDAR*, pages 733–737, 2013.

[89] P. Krishnan and C. V. Jawahar. Generating synthetic data for text recognition. *CoRR*, abs/1608.04224, 2016.

[90] P. Krishnan and C. V. Jawahar. Matching handwritten document images. In *European Conference on Computer Vision, ECCV*, volume 9905, pages 766–782, 2016.

[91] P. Krishnan and C. V. Jawahar. Hwnet v2: an efficient word image representation for handwritten documents. *Int. J. Document Anal. Recognit., IJDAR*, 22(4):387–405, 2019.

[92] P. Krishnan, R. Shekhar, and C. V. Jawahar. Content level access to digital library of india pages. In B. Triggs, K. Bala, and S. Chandran, editors, *Indian Conference on Vision, Graphics and Image Processing, ICVGIP*, page 5, 2012.

[93] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems, NIPS*, pages 1106–1114, 2012.

[94] A. Kumar, C. V. Jawahar, and R. Manmatha. Efficient search in document image collections. In *Asian Conference on Computer Vision, ACCV*, volume 4843, pages 586–595, 2007.

[95] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, volume 2, pages 2169–2178, 2006.

[96] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[97] G. C. Lee, F. Yeh, Y. Chen, and T. Chang. Robust handwriting extraction and lecture video summarization. *Multim. Tools Appl.*, 76(5):7067–7085, 2017.

[98] F. Li and P. Perona. A bayesian hierarchical model for learning natural scene categories. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 524–531, 2005.

[99] M. Lin, Q. Chen, and S. Yan. Network in network. In *2nd International Conference on Learning Representations, ICLR*, 2014.

[100] E. Loper and S. Bird. NLTK: the natural language toolkit. *CoRR*, cs.CL/0205028, 2002.

[101] G. Louloudis, B. Gatos, I. Pratikakis, and C. Halatsis. Text line and word segmentation of handwritten documents. *Pattern Recognit.*, 42(12):3169–3183, 2009.

[102] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vis., IJCV*, 60(2):91–110, 2004.

[103] L. v. d. Maaten and G. Hinton. Visualizing data using t-sne. *Journal of machine learning research, JMLR*, 9:2579–2605, 2008.

[104] A. Mahendran and A. Vedaldi. Understanding deep image representations by inverting them. In *Computer Vision and Pattern Recognition, CVPR*, pages 5188–5196, 2015.

[105] T. Malisiewicz, A. Gupta, and A. A. Efros. Ensemble of exemplar-SVMs for object detection and beyond. In *IEEE International Conference on Computer Vision, ICCV*, pages 89–96, 2011.

[106] R. Manmatha, C. Han, and E. M. Riseman. Word spotting: A new approach to indexing handwriting. In *Computer Vision and Pattern Recognition, CVPR*, pages 631–637, 1996.

[107] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to information retrieval*. Cambridge University Press, 2008.

[108] U. Marti and H. Bunke. Using a statistical language model to improve the performance of an hmm-based cursive handwriting recognition system. *Int. J. Pattern Recognit. Artif. Intell.*, 15(1):65–90, 2001.

[109] U. Marti and H. Bunke. The IAM-database: an english sentence database for offline handwriting recognition. *Int. J. Document Anal. Recognit.*, 5(1):39–46, 2002.

[110] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide-baseline stereo from maximally stable extremal regions. *Image Vis. Comput.*, 22(10):761–767, 2004.

[111] M. Meshesha and C. V. Jawahar. Matching word images for content-based retrieval from printed document images. *Int. J. Document Anal. Recognit.*, 11(1):29–38, 2008.

[112] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. In *1st International Conference on Learning Representations, ICLR*, 2013.

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

[114] G. A. Miller. Wordnet: A lexical database for english. *Commun. ACM*, 38(11):39–41, 1995.

[115] C. Myers, L. Rabiner, and A. Rosenberg. Performance tradeoffs in dynamic time warping algorithms for isolated word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 28(6):623–635, 1980.

[116] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *International Conference on Machine Learning, ICML*, pages 807–814, 2010.

[117] H. Ney, U. Essen, and R. Kneser. On the estimation of 'small' probabilities by leaving-one-out. *IEEE Trans. Pattern Anal. Mach. Intell., PAMI*, 17(12):1202–1212, 1995.

[118] D. Nistér and H. Stewénius. Scalable recognition with a vocabulary tree. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR*, pages 2161–2168, 2006.

[119] T. Ojala, M. Pietikäinen, and T. Mäenpää. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Trans. Pattern Anal. Mach. Intell., PAMI*, 24(7):971–987, 2002.

[120] U. Pal and B. B. Chaudhuri. Indian script character recognition: a survey. *Pattern Recognition*, 37(9):1887–1899, 2004.

[121] A. Passos, V. Kumar, and A. McCallum. Lexicon infused phrase embeddings for named entity resolution. In R. Morante and W. Yih, editors, *Proceedings of the Eighteenth Conference on Computational Natural Language Learning, CoNLL*, pages 78–86, 2014.

[122] J. Pennington, R. Socher, and C. D. Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP*, pages 1532–1543. ACL, 2014.

[123] F. Perronnin and C. Dance. Fisher kernels on visual vocabularies for image categorization. In *Computer Vision and Pattern Recognition, CVPR*, pages 1–8, 2007.

[124] F. Perronnin and J. A. Rodríguez-Serrano. Fisher kernels for handwritten word-spotting. In *International Conference on Document Analysis and Recognition, ICDAR*, pages 106–110, 2009.

[125] F. Perronnin and J. A. Rodríguez-Serrano. Fisher kernels for handwritten word-spotting. In *International Conference on Document Analysis and Recognition, ICDAR*, pages 106–110, 2009.

[126] V. Pham, T. Bluche, C. Kermorvant, and J. Louradour. Dropout improves recurrent neural networks for handwriting recognition. In *International Conference on Frontiers in Handwriting Recognition, ICFHR*, pages 285–290, 2014.

[127] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *IEEE conference on computer vision and pattern recognition, CVPR*, pages 1–8, 2007.

[128] M. F. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

[129] M. Potthast, M. Hagen, A. Beyer, M. Busse, M. Tippmann, P. Rosso, and B. Stein. Overview of the 6th international competition on plagiarism detection. In *Working Notes for CLEF Conference*, volume 1180 of *CEUR Workshop Proceedings*, pages 845–876, 2014.

[130] A. Poznanski and L. Wolf. Cnn-n-gram for handwriting word recognition. In *Computer vision and pattern recognition, CVPR*, pages 2305–2314, 2016.

[131] I. Pratikakis, K. Zagoris, B. Gatos, J. Puigcerver, A. H. Toselli, and E. Vidal. ICFHR2016 handwritten keyword spotting competition (H-KWS 2016). In *International Conference on Frontiers in Handwriting Recognition, ICFHR*, pages 613–618, 2016.

[132] T. M. Rath and R. Manmatha. Word image matching using dynamic time warping. In *Computer Vision and Pattern Recognition, CVPR*, pages 521–527, 2003.

[133] T. M. Rath and R. Manmatha. Word spotting for historical documents. *Int. J. Document Anal. Recognit.*, 9(2-4):139–152, 2007.

[134] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson. CNN features off-the-shelf: An astounding baseline for recognition. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops, CVPRW*, pages 512–519, 2014.

[135] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson. CNN features off-the-shelf: An astounding baseline for recognition. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops, CVPRW*, pages 512–519, 2014.

[136] S. Ren, K. He, R. B. Girshick, and J. Sun. Faster R-CNN: towards real-time object detection with region proposal networks. In *Neural Information Processing Systems, NIPS*, pages 91–99, 2015.

[137] J. A. Rodriguez and F. Perronnin. Local gradient histogram features for word spotting in unconstrained handwritten documents. pages 7–12, 2008.

[138] J. A. Rodríguez-Serrano and F. Perronnin. A model-based sequence similarity with application to handwritten word spotting. *IEEE Trans. Pattern Anal. Mach. Intell.*, 34(11):2108–2120, 2012.

[139] J. R. Rohlicek, W. Russell, S. Roukos, and H. Gish. Continuous hidden markov modeling for speaker-independent word spotting. In *International Conference on Acoustics, Speech, and Signal Processing,*, pages 627–630, 1989.

[140] G. Ros, L. Sellart, J. Materzynska, D. Vázquez, and A. M. López. The SYNTHIA dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 3234–3243, 2016.

[141] E. Rosten and T. Drummond. Machine learning for high-speed corner detection. In *European Conference on Computer, ECCV*, volume 3951, pages 430–443, 2006.

[142] L. Rothacker, M. Rusiñol, and G. A. Fink. Bag-of-features hmms for segmentation-free word spotting in handwritten documents. In *International Conference on Document Analysis and Recognition, ICDAR*, pages 1305–1309, 2013.

[143] L. Rothacker, S. Sudholt, E. Rusakov, M. Kasperidus, and G. A. Fink. Word hypotheses for segmentation-free word spotting in historic document images. In *IAPR International Conference on Document Analysis and Recognition, ICDAR*, pages 1174–1179, 2017.

[144] P. P. Roy, F. Rayar, and J. Ramel. Word spotting in historical documents using primitive codebook and dynamic programming. *Image Vis. Comput.*, 44:15–28, 2015.

[145] A. Rozantsev, V. Lepetit, and P. Fua. On rendering synthetic images for training an object detector. *Comput. Vis. Image Underst.*, 137:24–37, 2015.

[146] Y. Rubner, C. Tomasi, and L. J. Guibas. The earth mover's distance as a metric for image retrieval. *International journal of computer vision, IJCV*, 40(2):99–121, 2000.

[147] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986.

[148] M. Rusiñol, D. Aldavert, R. Toledo, and J. Lladós. Browsing heterogeneous document collections by a segmentation-free word spotting method. In *International Conference on Document Analysis and Recognition, ICDAR*, pages 63–67, 2011.

[149] M. Rusiñol, D. Aldavert, R. Toledo, and J. Lladós. Efficient segmentation-free keyword spotting in historical document collections. *Pattern Recognit.*, 48(2):545–555, 2015.

[150] H. Sakoe and S. Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE transactions on acoustics, speech, and signal processing*, 26(1):43–49, 1978.

[151] H. Sakoe and S. Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE transactions on acoustics, speech, and signal processing*, 26(1):43–49, 1978.

[152] S. Schleimer, D. S. Wilkerson, and A. Aiken. Winnowing: Local algorithms for document fingerprinting. In *International Conference on Management of Data, SIGMOD*, pages 76–85, 2003.

[153] A. Sharma and D. B. Jayagopi. Automated grading of handwritten essays. In *16th International Conference on Frontiers in Handwriting Recognition, ICFHR*, pages 279–284. IEEE Computer Society, 2018.

[154] R. Shekhar and C. V. Jawahar. Word image retrieval using bag of visual words. In *IAPR International Workshop on Document Analysis Systems, DAS*, pages 297–301, 2012.

[155] R. Shekhar and C. V. Jawahar. Document specific sparse coding for word retrieval. In *International Conference on Document Analysis and Recognition, ICDAR*, pages 643–647, 2013.

[156] B. Shi, X. Bai, and C. Yao. An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. *IEEE transactions on pattern analysis and machine intelligence*, 39(11):2298–2304, 2016.

[157] P. Y. Simard, D. Steinkraus, and J. C. Platt. Best practices for convolutional neural networks applied to visual document analysis. In *International Conference on Document Analysis and Recognition, ICDAR*, pages 958–962, 2003.

[158] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations, ICLR*, 2015.

[159] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations, ICLR*, 2015.

[160] S. Singh, A. Gupta, and A. A. Efros. Unsupervised discovery of mid-level discriminative patches. In *12th European Conference on Computer Vision, ECCV*, volume 7573, pages 73–86, 2012.

[161] J. Sivic and A. Zisserman. Video google: A text retrieval approach to object matching in videos. In *IEEE International Conference on Computer Vision, ICCV*, pages 1470–1477, 2003.

[162] C. Spearman. The proof and measurement of association between two things. *The American Journal of Psychology*, 100(3/4):441–471, 1987.

[163] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958, 2014.

[164] N. Stamatopoulos, B. Gatos, G. Louloudis, U. Pal, and A. Alaei. ICDAR 2013 handwriting segmentation contest. In *International Conference on Document Analysis and Recognition, ICDAR*, pages 1402–1406, 2013.

[165] B. Stuner, C. Chatelain, and T. Paquet. Cohort of LSTM and lexicon verification for handwriting recognition with gigantic lexicon. *CoRR*, abs/1612.07528, 2016.

[166] S. Sudholt and G. A. Fink. PHOCNet: A deep convolutional neural network for word spotting in handwritten documents. In *International Conference on Frontiers in Handwriting Recognition, ICFHR*, pages 277–282, 2016.

[167] S. Sudholt and G. A. Fink. Evaluating word string embeddings and loss functions for CNN-based word spotting. In *International Conference on Document Analysis and Recognition, ICDAR*, pages 493–498, 2017.

[168] S. Sudholt and G. A. Fink. Attribute cnns for word spotting in handwritten documents. *Int. J. Document Anal. Recognit.*, 21(3):199–218, 2018.

[169] J. Sueiras, V. Ruíz, Á. Sánchez, and J. F. Vélez. Offline continuous handwriting recognition using sequence to sequence neural networks. *Neurocomputing*, 289:119–128, 2018.

[170] Z. Sun, L. Jin, Z. Xie, Z. Feng, and S. Zhang. Convolutional multi-directional recurrent network for offline handwritten text recognition. In *International Conference on Frontiers in Handwriting Recognition, ICFHR*, pages 240–245, 2016.

[171] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. E. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Computer Vision and Pattern Recognition, CVPR*, pages 1–9, 2015.

[172] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 2818–2826, 2016.

[173] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. J. Goodfellow, and R. Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations, ICLR*, 2014.

[174] K. Takeda, K. Kise, and M. Iwamura. Real-time document image retrieval for a 10 million pages database with a memory efficient and stability improved LLAH. In *International Conference on Document Analysis and Recognition, ICDAR*, pages 1054–1058, 2011.

[175] K. Terasawa and Y. Tanaka. Slit style HOG feature for document image word spotting. In *International Conference on Document Analysis and Recognition, ICDAR*, pages 116–120, 2009.

[176] J. I. Toledo, S. Sudholt, A. Fornés, J. Cucurull, G. A. Fink, and J. Lladós. Handwritten word image categorization with convolutional neural networks and spatial pyramid pooling. In *Structural, Syntactic, and Statistical Pattern Recognition - Joint IAPR International Workshop, S+SSPR*, volume 10029 of *Lecture Notes in Computer Science*, pages 543–552, 2016.

[177] T. Tuna, J. Subhlok, and S. Shah. Indexing and keyword search to ease navigation in lecture videos. In *Applied Imagery Pattern Recognition Workshop: Imaging for Decision Making, AIPR*, pages 1–8, 2011.

[178] A. van den Oord, N. Kalchbrenner, and K. Kavukcuoglu. Pixel recurrent neural networks. In M. Balcan and K. Q. Weinberger, editors, *International Conference on Machine Learning, ICML*, volume 48, pages 1747–1756, 2016.

[179] A. Vedaldi and K. Lenc. Matconvnet: Convolutional neural networks for MATLAB. In *ACM Conference on Multimedia Conference, MM*, pages 689–692, 2015.

[180] A. Vinciarelli and S. Bengio. Offline cursive word recognition using continuous density hidden markov models trained with PCA or ICA features. In *International Conference on Pattern Recognition, ICPR*, pages 81–84, 2002.

[181] S. N. P. Vitaladevuni, F. Choi, R. Prasad, and P. Natarajan. Detecting near-duplicate document images using interest point matching. In *International Conference on Pattern Recognition, ICPR*, pages 347–350, 2012.

[182] P. Voigtlaender, P. Doetsch, and H. Ney. Handwriting recognition with large multidimensional long short-term memory recurrent neural networks. In *International Conference on Frontiers in Handwriting Recognition, ICFHR*, pages 228–233, 2016.

[183] F. Wang, C. Ngo, and T. Pong. Synchronization of lecture videos and electronic slides by video text analysis. In *ACM International Conference on Multimedia, ACMMM*, pages 315–318, 2003.

[184] J. Wang, J. Yang, K. Yu, F. Lv, T. S. Huang, and Y. Gong. Locality-constrained linear coding for image classification. In *Computer Vision and Pattern Recognition, CVPR*, pages 3360–3367, 2010.

[185] C. Wigington, S. Stewart, B. L. Davis, B. Barrett, B. L. Price, and S. Cohen. Data augmentation for recognition of handwritten words and lines using a CNN-LSTM network. In *IAPR International Conference on Document Analysis and Recognition, ICDAR*, pages 639–645, 2017.

[186] T. Wilkinson and A. Brun. Semantic and verbatim word spotting using deep neural networks. In *International Conference on Frontiers in Handwriting Recognition, ICFHR*, pages 307–312, 2016.

[187] T. Wilkinson, J. Lindström, and A. Brun. Neural ctrl-f: Segmentation-free query-by-string word spotting in handwritten manuscript collections. In *International Conference on Computer Vision, ICCV*, pages 4443–4452, 2017.

[188] T. Wilkinson, J. Lindström, and A. Brun. Neural word search in historical manuscript collections. *CoRR*, abs/1812.02771, 2018.

[189] I. Z. Yalniz and R. Manmatha. An efficient framework for searching text in noisy document images. In *IAPR International Workshop on Document Analysis Systems, DAS*, pages 48–52, 2012.

[190] H. Yang, M. Siebert, P. Lühne, H. Sack, and C. Meinel. Lecture video indexing and analysis using video OCR technology. In *International Conference on Signal-Image Technology and Internet-Based Systems, SITIS*, pages 54–61, 2011.

[191] J. Yang, K. Yu, Y. Gong, and T. S. Huang. Linear spatial pyramid matching using sparse coding for image classification. In *Computer Vision and Pattern Recognition, CVPR*, pages 1794–1801, 2009.

[192] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. How transferable are features in deep neural networks? In *Advances in Neural Information Processing Systems, NIPS*, pages 3320–3328, 2014.

[193] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. How transferable are features in deep neural networks? In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems, NIPS*, pages 3320–3328, 2014.

[194] J. Yosinski, J. Clune, A. M. Nguyen, T. J. Fuchs, and H. Lipson. Understanding neural networks through deep visualization. *CoRR*, abs/1506.06579, 2015.

[195] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *European Conference on Computer Vision, ECCV*, volume 8689, pages 818–833, 2014.

[196] G. K. Zipf. *The psycho-biology of language: An introduction to dynamic philology*, volume 21. Psychology Press, 1999.