Die approbierte Originalversion dieser Diplom-/Masterarbeit ist in der Hauptbibliothek der Technischen Universität Wien aufgestellt und zugänglich.
http://www.ub.tuwien.ac.at

The approved original version of this diploma or master thesis is available at the main library of the Vienna University of Technology.
http://www.ub.tuwien.ac.at/eng

FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

Text Detection and Recognition in Natural Scene Images

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Visual Computing

eingereicht von

Michael Opitz
Matrikelnummer 0828257

an der
Fakultät für Informatik der Technischen Universität Wien

Betreuung
Betreuer: Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Robert Sablatnig
Mitwirkung: Markus Diem, Stefan Fiel, Florian Kleber

Wien, 7. Oktober 2013

(Unterschrift Verfasser) (Unterschrift Betreuer)

FAKULTÄT
FÜR !NFORMATIK
Faculty of Informatics

Erklärung zur Verfassung der Arbeit

Michael Opitz
Hauptstraße 75
2463 Gallbrunn

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

Wien, 7. Oktober 2013

(Unterschrift Verfasser)

I dedicate this thesis to my family, which supported me in all these years during my study and my entire life.

Kurzfassung

Die Detektion und Erkennung von Text in Bildern hat zahlreiche Anwendungen im Gebiet der Bildverarbeitung. Dazu gehören zum Beispiel die Erkennung von Nummerntafeln, die automatische Erkennung von Straßenschildern, sowie Assistenzsysteme für Menschen mit Sehschwäche. Die Schwierigkeiten dieser Anwendungen besteht darin, dass oft Bilder mit einem komplexen Hintergrund, Texte mit verschiedenen Schriftgrößen und Arten, Textregionen die teilweise verdeckt sind, aber auch Bilder in denen Ausschnitte verschwommen sind, vorhanden sind.

Diese Arbeit präsentiert eine Methodik für eine wörterbuchbasierte End-to-End Textdetektion und Erkennung. Die Detektion des Textes wird mit einem AdaBoost Sliding Window Klassifikator gelöst, welcher Text skalierungsinvariant im Bild sucht. Der Klassifikator wurde mit unterschiedlichen Merkmalen evaluiert, die im Rahmen dieser Arbeit präsentiert werden. Modifizierte Local Ternary Pattern erweisen sich als das beste Merkmal für dieses Problem. Nach diesem Schritt werden Maximally Stable Extremal Regions aus dem Bild extrahiert und in die beiden Gruppen „Text“ und „kein Text“ klassifiziert. Textregionen werden zuerst zu Textzeilen gruppiert, und danach werden mit einer Word-Splitting Methode, welche auf k-Means und linearen Support Vector Maschinen basiert, die einzelnen Wörter detektiert.

Für die Texterkennung wird ein Deep Convolutional Neural Network (CNN) mit Backpropagation auf einem 1-dimensionalen Sliding Window trainiert. Um eine Überanpassung des Klassifikators zu vermeiden, wird das Netzwerk mit Dropout regularisiert. Die Methode um das wahrscheinlichste Wort in einem Wörterbuch zu finden basiert auf dem Viterbi Algorithmus. Die Einflüsse des Trainingssets und die unterschiedliche Anzahl der Layers im CNN werden evaluiert. Die präsentierte Methodik erzielt bessere Ergebnisse als aktuelle Methoden auf den ICDAR 2003 und 2011 Datensätzen für Textdetektion (F-Score 74.2% / 76.7%), wörterbuchbasierter Worterkennung (F-Score: 87.1% / 87.1%) und wörterbuchbasierte End-to-End Erkennung (F-score: 72.6% / 72.3%).

Abstract

Text detection and recognition in natural scene images has applications in computer vision systems such as license plate detection, automatic street sign translation, image retrieval and help for visually impaired people. Scene text, however, may have complex background, image blur, partially occluded text, variations in font-styles, image noise and varying illumination. Hence scene text recognition is a challenging computer vision problem. This work addresses the problem of dictionary driven end-to-end scene text recognition, which is divided into a text detection problem and a text recognition problem. For text detection an AdaBoost sliding window classifier is used to detect text in multiple scales. The effectiveness of several feature-sets for this classifier are compared and evaluated. A modified Local Ternary Pattern (LTP) feature-set is found as most effective for text detection. In a post-processing stage Maximally Stable Extremal Regions (MSER) are detected and labeled as text or non-text. Text regions are grouped to textlines. Textlines are split into words by a word-splitting method build upon k-means and linear Support Vector Machines (SVM). For text recognition a deep Convolutional Neural Network (CNN) trained with backpropagation is used as one-dimensional sliding window classifier. To avoid overfitting the network is regularized by Dropout. Recognition-responses are used in a Viterbi-style algorithm to find the most plausible word in a dictionary. The influence of training set size and size of convolutional layers is evaluated. The system presented outperforms state of the art methods on the ICDAR 2003 and 2011 dataset in the text-detection (F-score: 74.2% / 76.7%), dictionary-driven cropped-word recognition (F-score: 87.1% / 87.1%) and dictionary-driven end-to-end recognition (F-score: 72.6% / 72.3%) tasks.

Acknowledgments

I would like to thank the Computer Vision Lab team at TU Vienna for their helpful advice, time and patience.

Contents

CHAPTER 1

Introduction

In context of scanned machine written documents text-detection and -recognition is considered a solved problem. There are several commercial [1] and freely available systems [2], achieving a character recognition accuracy of up to 99% depending on the document quality [73].

Text detection and recognition in natural scene images is, however, a challenging computer vision problem [60]. Scene text has complex background, image blur, partially occluded text, variations in font-styles, image noise and varying illumination as illustrated in Figure 1.1 [2, 84]. Commercial systems do not work in these settings [24].

Hence, within the scope of the International Conference of Document Analysis and Recognition (ICDAR) several competitions were organised to assess the state of the art in text detection and recognition in natural scene images [86, 80]. To improve the state of the art, public datasets [86, 80, 94] with labeled ground-truth data are available, on which different methods are compared with each other.

The end-to-end scene text recognition problem is divided into a text-detection and text-recognition task. Text-detection is a preprocessing step for text-recognition. The text-detector has to locate words in natural scene images. Hence the detector has to distinguish text locations from non-text locations in an image. The text-recognizer predicts a word shown in a cropped image patch which is retrieved by the detector.

This work addresses the problem of end-to-end dictionary driven scene text recognition, in which the word-recognizer uses a dictionary to predict the resulting word.

1.1 Motivation

Text detection and recognition in natural scene images has applications in computer vision systems such as image retrieval, automatic license plate recognition, automatic street sign translation or help for visually impaired people.

[1] e.g. ABBYY FineReader, `http://www.abbyy.com/` - last checked on 22.09.2013

[2] Tesseract, `https://code.google.com/p/tesseract-ocr/` - last checked on 22.09.2013

Figure 1.1: Natural scene images containing text.

In image retrieval systems, end-to-end text recognition systems recognize text in images and index them into a document-index, so that they can later be retrieved. For automatic street sign translation systems, text recognition systems recognize text, which is processed by a machine translation tool to translate it into another language. By recognizing license plates automatic toll collection is possible. For visually impaired people text recognition systems can help to recognize and read text on street signs with text-to-speech systems.

Recent ICDAR competitions [86, 80] show, however, that state of the art methods achieve a detection performance, measured in F-score, of about 71.28% and the state of the art method in end-to-end recognition [96] achieves an F-score of 67% on end-to-end scene text recognition on ICDAR datasets. Hence, there is still room for improvement for bridging the gap to human performance.

1.1.1 Scope of Work

The scope of this work is text detection and dictionary driven text recognition in natural scene images. The scientific research question is: *Can the combination of local features, region based approaches and learned local features achieve competitive performance for text detection and dictionary driven recognition in natural scene images?*

To assess the performance the publicly available ICDAR 2003 [86] and ICDAR 2011 [80] datasets are used, on which several end-to-end systems proposed by other authors, which are listed in Chapter 2, are evaluated. Hence, the performance is compared to state of the art systems for end-to-end scene text recognition. Furthermore, the performance of each of the proposed subsystems is evaluated in isolation on the ICDAR datasets and compared to other methods.

Furthermore, in this work only the problem of dictionary driven scene text recognition is discussed.

1.1.2 Aim of this Thesis

The main aim of this thesis is to propose a dictionary driven end-to-end scene text recognition system, achieving competitive performance on the ICDAR datasets [86, 80].

Hence, a system is proposed consisting of a sliding window classifier which detects text regions. In these regions Connected Components (CCs) are labeled by machine learning models and grouped to textlines. Textlines are split into words, which are classified by the proposed text recognition system to predict a word for the cropped patch.

To answer the research question and achieve competitive performance, first several low-level feature-sets proposed in the literature are adapted and evaluated. Region based techniques which are proposed in the literature are combined for CC labeling. Finally, Dropout [40] is applied to deep Convolutional Neural Networks (CNNs) [49] to learn low-level vision features useful for cropped character recognition.

The main contributions of this work are as follows:

- The influence of different feature-sets for text-detection is evaluated.
- The Multilevel Color edge Local Binary Pattern (MACeLBP) proposed by Anthimopoulos et al. [2] is combined with Local Ternary Pattern (LTP) [87] to improve text-detection performance.
- Maximally Stable Extremal Regions (MSERs) adopted by Neumann et al. [63, 62, 61] for text detection and Conditional Random Fields (CRFs) proposed by Pan et al. [68, 69, 70] are integrated with linear Support Vector Machines (SVMs), k-means and a sliding window classifier to a competitive text labeling system for natural scene images.
- Supervised trained CNNs [49] with backpropagation [76], which are regularized by Dropout [40], are applied for cropped word recognition achieving state-of-the-art performance.
- Scene text detection and scene text recognition modules are integrated to achieve competitive end-to-end results.

1.2 Definition of Terms

In this section a definition of terms used in this work is presented.

`ANN` **Artificial Neural Network**: is a family of biologically inspired machine learning models, which use a model of a neuron for classification [23] (see Section 2.2).

`CC` **Connected Component**: is a coherent region of pixels which are grouped due to homogeneity properties.

`CNN` **Convolutional Neural Network**: is a ANN which is inspired by the visual cortex. It has layers with shared weights, which are convolution kernels. Predictions are computed by applying convolution operations layer by layer [49] (see Section 4.1.1).

`CRF` **Conditional Random Field**: is a graphical model, which is used to label nodes on a graph [44]. It combines unary features with pairwise features to minimize an energy function.

`HOG` **Histogram of Oriented Gradients**: is a gradient based feature set proposed by Dalal and Triggs [17] for pedestrian detection. Gradients are densely accumulated in cells which are aggregated into overlapping blocks (see Section 3.1.1).

ICDAR **International Conference of Document Analysis and Recognition**: is an international conference which is hold every two years. Within the scope of this conference datasets for text detection and recognition have been made publicly available.

LBP **Local Binary Pattern**: is a texture descriptor proposed by Ojala et al. [66]. Texture information for a pixel is summarized as 8 bit integer. From this information a histogram of texture information in a local window is accumulated (see Section 3.1.2).

LTP **Local Ternary Pattern**: is an extension of LBP to a ternary coding proposed by Tan et al. [87]. The texture information for a pixel is summarized as 2×8 bit integers and hence, the feature-set is more discriminative than LBPs.

MSER **Maximally Stable Extremal Region**: is a region detector proposed by Matas et al. [56] for robust wide-baseline stereo matching (see Section 3.2.1).

MACeLBP **Multilevel Color edge Local Binary Pattern**: is an LBP extension proposed by Anthimopoulos et al. [2] for text detection (see Section 3.1.2).

MLP **MLP**: Multilayer Perceptron is a ANN consisting of several layers of neurons followed by a non-linear activation function. By stacking several layers of neurons in this way, the classifier can do non-linear classification (see Section 2.2).

OCR **Object Character Recognition**: is an automatic conversion of printed text in images to machine readable text.

RBM **Restricted Boltzmann Machine**: is a generative ANN which encodes input data in a high-level representation by learning the distribution of the data. It is used by Hinton et al. [39] to train a deep belief network (see Section 2.2).

RF **Random Forest**: is a discriminative machine learning model proposed by Breiman et al. [7]. It is a bagged tree ensemble, where each tree is trained on a random subset of the feature vector.

SIFT **Scale Invariant Feature Transform**: is a keypoint detector proposed by David Lowe [55]. The descriptor accumulates gradients in a local window on detected interest points in cells.

SVM **Support Vector Machine**: is a discriminative machine learning model proposed by Vapnik et al. [8], which maximizes the minimum margin to a separating hyperplane of two classes. By applying a kernel trick the classifier can do non-linear classification.

SVT **Street View Text**: is an annotated dataset with dictionaries proposed by Wang et al. [94] for end-to-end text recognition.

SWT **Stroke Width Transform**: is an image operator proposed by Epshtein et al. [24] which assigns to each pixel the width of the stroke.

1.3 Results

The proposed text detector achieves a performance of 74% (F-score) on the ICDAR 2003 dataset [86] and 76% (F-score) on the ICDAR 2011 dataset [80] outperforming state-of-the-art methods. The proposed text recognizer achieves a performance of 87.1% accuracy on the ICDAR 2003 [86] cropped word recognition challenge, which is better than state-of-the-art methods. The end-to-end system proposed achieves a performance of 72.7% on ICDAR 2003 and a performance of 72.3 on ICDAR 2011 measured in F-score, outperforming other methods [94, 96].

Further, it is shown that the influence of the feature-set of the sliding window classifier used lies in a range of about 1.3% on detection performance measured in F-score. The detector achieves a performance of 71.4% on the ICDAR 2003 [86] dataset without any feature set. The maximum performance of the labeling stage given a perfect sliding window classifier is 77.0%.

Ceiling analysis shows that by improving the sliding window system, keeping the remaining pipeline fixed a maximum improvement of 2.1% on end-to-end performance (F-score) can be achieved. By replacing the CC labeling module a maximum improvement of 12.9% on end-to-end performance (F-score) can be achieved and by improving the word recognition system a maximum improvement of 12.3% on end-to-end performance (F-score) can be achieved.

1.4 Structure of this Work

The remainder of this work is structured as follows. In Chapter 2 an overview of state-of-the-art methods for text detection and recognition is given. Furthermore segmentation methods, object detection and recognition methods and ANN are reviewed.

In Chapter 3 the text detection system is presented. The chapter is divided into four sections. In Section 3.1 the choice of the sliding window classifier and the used feature-set is discussed. In Section 3.2, feature-sets and classifiers used for CC labeling are discussed. Section 3.3 and Section 3.4 discuss the textline grouping and word-splitting steps in the system proposed.

In Chapter 4 the proposed recognition system is presented. It is divided into Section 4.1 which presents the method used for recognizing cropped characters and in Section 4.2 which details about how the cropped character recognizer is used to predict words.

Results of this work are presented in Chapter 5. Finally, conclusions and directions for further work are drawn in Chapter 6.

CHAPTER 2

State of the Art

In this chapter an overview of state-of-the-art methods in text-detection and -recognition and an overview of object recognition methods which are related to this work are given. Since the method proposed relies on segmentation techniques, in Section 2.1 segmentation methods in context of document analysis and scene text-detection and -recognition are reviewed. The text-recognition system uses CNNs to recognize cropped characters. Hence, in Section 2.2 a historical overview of ANNs is given and recent work in this area is reviewed. In Section 2.3 an overview of object detection and recognition methods in computer vision, which are relevant to this work, is given.

In Section 2.4 an overview of text-detection and -recognition in natural scene images is presented. Finally, in Section 2.5 this chapter is summarized.

2.1 Segmentation Methods

Since the proposed method relies on segmentation methods, an overview of segmentation techniques used in document analysis and text detection and recognition in natural scenes is given in this section.

Traditional segmentation methods in OCR are divided into adaptive and non-adaptive approaches. Non-adaptive approaches estimate a global threshold for the whole document. Adaptive approaches use local thresholds which are estimated from a local window.

One of the most prominent global segmentation methods is Otsu [67]. Global segmentation methods, however, do not work on images having changes in illumination [77].

Since natural scene images have varying illumination conditions, choosing a single threshold for segmenting the image results in segmentation errors. Figure 2.1 shows a specular highlight on a book cover, which results in several letters such as 'R' and 'E' segmented as a single component. In the context of document analysis several adaptive methods are proposed, which address the problem of varying illumination conditions in a document [72].

An example for an adaptive method is Niblack [65], which estimates a threshold T from the mean m and standard deviation s in a local window as follows:

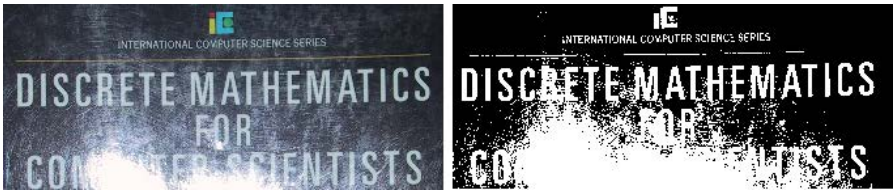


Figure 2.1: Segmentation errors caused by binarizing the image with Otsu [67].

$$T = m + k \cdot s \tag{2.1}$$

where k is a parameter of the method, m is the local mean and s the local standard deviation.

Extensions to this method are proposed by Sauvola et al. [77], which make binarization more robust against noise. Furthermore, a Niblack variant is proposed by Pan et al. [68, 69, 70] for binarization in natural scene images.

Traditional adaptive segmentation approaches for document analysis have limitations when applied to scene text segmentation. Characters which are too near to each other are segmented to a single CC, the shape of segmented blurred characters is distorted and characters which have the same contrast compared to background noise are segmented to a single CC with the background (see Figure 2.2). Hence, segmentation methods for text detection in natural scene images use approaches like MSER to alleviate these problems.

Figure 2.2: Segmentation errors due to background noise, blur and narrow text.

In the context of text-detection in natural scenes, segmentation of a dense stroke width image is proposed by Epshtein et al. [24]. The stroke width of pixels in the image is computed by the SWT which is an image operator operating on edges detected by the Canny edge detector [9]. Pixels with similar stroke width are grouped to letters, which are further grouped into words and textlines.

Recent methods, such as the winner of the ICDAR 2011 competition, use MSER [56] or Extremal Regions (ERs) [56] to detect candidate regions and use machine learning techniques or heuristics to label them as text or non-text [61, 62, 63, 10, 80]. MSER provide a multi-segmentation of an image by thresholding the image several times. Compared to adaptive thresholding approaches a higher number of candidate regions is generated. Due to the multi-segmentation, narrow text, blurred characters and text with weak contrast are less likely to be grouped together to a single CC. An overview of MSER is given in Section 3.2.1.

2.2 Artificial Neural Networks

ANNs are biologically inspired machine learning methods. The building block for these methods is a model of a neuron. The first neuron model is proposed by McCulloch and Pitts in 1943 [23]. The neuron model which is used in ANNs is shown in Figure 2.3 [23]. The inputs x_i for a neuron are weighted by weights w_i and summed together. If this sum exceeds a threshold the neuron is active:

$$g(\mathbf{x}) = \begin{cases} 1, & \sum_{i=0}^{n} x_i \cdot w_i > 0 \\ -1, & otherwise \end{cases} \tag{2.2}$$

where $g(\mathbf{x})$ is the activation function of the neuron.

In 1958 Frank Rosenblatt proposes a training algorithm for training a single neuron called Perceptron [23]. The training algorithm updates the weight vector w for misclassified training samples x of class $t \in \{1, -1\}$ as follows: $w = w + t \cdot x$. Rosenblatt et al. show that given a training set where object-classes lie on opposite sides of a hyperplane, the training algorithm terminates. Furthermore, the result of this algorithm is a hyperplane separating the two object classes.

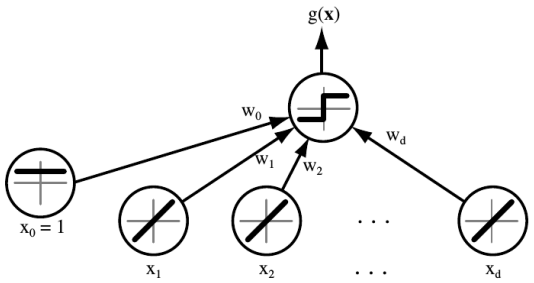


Figure 2.3: Single neuron "Perceptron" model which has several inputs x_i which are multiplied by weights w_i (Source: Duda et al. [23]).

Rumelhart, Hinton and Williams [76] propose backpropagation for training MLPs in 1958. MLPs consist of several layers of Perceptrons which are organized in a regular grid. A non-linear activation function is applied to responses from each layer. Hence, these networks overcome several limitations of Perceptrons and are capable of learning more complex functions such as the XOR function.

LeCun et al. [49, 50] propose CNNs for recognizing handwritten digits. CNNs have locally shared weights in form of convolution kernels. Responses are sub-sampled over spatial regions. Hence, neurons in upper layers have larger receptive fields than neurons in lower layers. Recent methods propose an additional response normalization step after the sub-sampling step [41]. An overview of CNNs in context with more recent research is given in Section 4.1.1.

Hinton et al. [39] propose in 2006 a greedy unsupervised pre-training method for training deep Neural Networks. RBMs are trained greedy, layer-by-layer to minimize the Kullback-Leibler divergence of the distribution of the data and the distribution defined by the model. RBMs encode the input in a high-level representation. After training a RBM, the output is used

to train a RBM for a higher layer. Finally, the whole model is finetuned by a supervised training method. This methodology makes it possible to train networks with a larger number of layers, which outperform networks with a smaller number of layers.

Since then, further research is done to improve methods for training deep neural networks.

Regularized Autoencoders [91, 75, 74, 37] are proposed as an alternative to RBMs for pre-training networks. Graphics Processing Unit (GPU) hardware is used to train deep networks faster [46, 13]. Deep CNNs are applied on labeled datasets consisting of over one million samples [46], outperforming other methods. Rectifier activation functions are proposed [59, 33], which are faster to train than sigmoid units [46].

Training methods for deep neural networks can be categorized into several groups [41]:

- Supervised trained networks which are trained on labeled datasets with backpropagation.
- Unsupervised pre-trained networks which are finetuned with a supervised learning algorithm.
- Unsupervised learned features which are used in a classifier like SVM. The feature layers do not receive any supervised finetuning.

Jarrett et al. [41] show that unsupervised pre-trained networks with a supervised fine-tuning stage perform best on the Caltech-101 [28] and NORB [52] datasets. The difference in performance compared to supervised trained networks is, however, in several experiments under 1%.

Later, several regularization methods are proposed which improve the generalization capability of deep networks. Hinton et al. [40] propose the regularization technique Dropout, which randomly sets inputs to zero in the training phase. A more complete overview of Dropout is given in Section 4.1.1. Zeiler and Fergus [104] propose Stochastic Pooling, a pooling operator for CNNs. In the training phase stochastic pooling samples responses from the local receptive fields according to a probability distribution defined by the activations of the neurons in the receptive field. Goodfellow et al. [36] propose a novel convex activation function Maxout, which is defined as the maximum over several linear units. Wan et al. [93] propose DropConnect, which is a generalization of Dropout, where weights instead of inputs are set to zero during training. Furthermore, Wang et al. [95] propose a fast method to train Dropout networks.

Recent deep learning methods set the state-of-the-art in several computer vision competitions. Krizhevsky et al. [46], winner of the ImageNet competition, uses deep CNNs with Rectified Linear Units (ReLUs), regularized by Dropout to classify 1.2 million images into 1000 different classes. Sermanet et al. [79] use unsupervised multi-stage features for a competitive pedestrian detection system. Goodfellow et al. [36] use convolutional Maxout networks for street view digit recognition. Cireşan et al. [12, 14] adopt CNNs for mitosis detection in breast cancer histology images and street sign classification. Tang et al. [88] classify facial expressions with deep CNNs, which are trained by minimizing the squared SVM hinge loss. Wang et al. [96] achieve state-of-the-art performance for end-to-end scene text recognition with CNNs which use unsupervised learned features. Farabet et al. [27] use CNNs for scene labeling. Le et al. [54] propose Independent Subspace Analysis for learning hierarchical features for action recognition in videos.

2.3 Object Detection and Recognition

Since a comprehensive overview of object-detection and -recognition in images is beyond the scope of this work, an overview of research relevant to this work is given.

For object detection in images sliding window classifiers are proposed [92, 11, 68, 17, 21, 98, 29, 54, 2, 79]. The approach of these methods is illustrated in Figure 2.4. First, to detect objects of multiple sizes, the image is sub-sampled several times to create an image pyramid. Each pyramid level is scanned by a local classification window which assigns a probability to a window indicating the certainty with which an object is shown in the window.

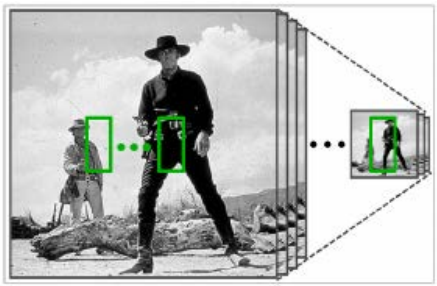


Figure 2.4: Sliding window detection for pedestrians proposed by Dollár et al. [20]

For classification discriminative machine learning models such as Support Vector Machines (SVMs) [8] or tree ensembles [31, 7] are used. The model is trained on cropped image patches containing samples of the object-class. Negative samples are found by mining false positive classifications on a set of training images containing no instance of the object-class being detected.

Since the performance of these models depends on the feature set, research has been devoted into developing and improving fast and discriminative low-level features.

Gabor filters are biologically inspired convolution filters [42]. They are proposed for recognition tasks such as iris recognition [18], face recognition [83] or text detection [54]. Furthermore, for texture classification Gabor responses are clustered to a high level "Texton" dictionary representation. [90]. For classification the image is convolved with a Gabor filter bank. Each pixel is assigned to the Texton-centroid having nearest distance to its responses. From this assignment a histogram is built which is used as feature vector.

Lowe et al. [55] propose SIFT as keypoint detector. The descriptor is computed on stable interest points and rotated according to the most dominant local gradient orientation. For object recognition the descriptor is used in bag-of-words models to learn high level features with dictionary learning methods [30]. In more recent methods the descriptors are computed on a regular grid, encoded as high-level representation and pooled over spatial regions [6, 47, 100]. These models are computationally expensive and due to the exhaustive search of sliding window detectors, over $100,000$ windows for images of size 1024×768 are classified. Hence, for object detection with these features a combined top-down and bottom-up approach is proposed [89]. A segmentation generates candidate locations, which are verified by the classifier.

Viola and Jones [92] propose Haar-like features for face detection which are shown in Figure

2.5. Grayscale values in the dark rectangular areas are summed and subtracted from grayscale values in the bright rectangular areas. A pool of over $180,000$ of such features is used for training an AdaBoost classifier consisting of decision stumps as weak learners. To compute these features efficiently, integral images are proposed with which the sum of grayscale values in a rectangular region can be computed in constant time.

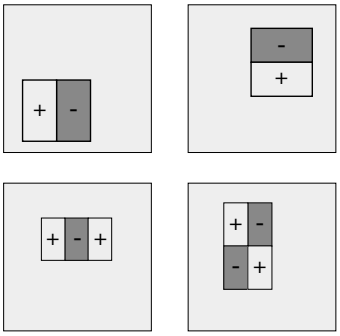


Figure 2.5: Haar-like features proposed by Viola and Jones for face detection [92].

Dalal and Triggs [17] propose HOG, which is a SIFT-like feature set for pedestrian detection. The feature-set is computed densely on the image. First, the gradient magnitude and orientation is computed. Then the gradient orientation is discretized in an orientation histogram. Each pixel votes with its gradient magnitude for an orientation histogram bin. These features are pooled locally to regular cells and blocks which are then concatenated to a feature vector. A more complete overview of HOG is given in Section 3.1.1.

Several extensions for HOG are proposed. Dollár et al. [21] integrate HOG with the integral image framework and use similar to Viola et al. [92] a pool of rectangular regions as features for training a tree ensemble. Watanabe et al. [98] propose Co-occurence HOG (CoHOG) which is a combination of co-occurrence histograms [38] with HOG. A color extension of HOG which incorporates color information is proposed by Kahn et al. [81].

LBPs [1, 66] is a feature set for encoding texture information, which is proposed for several object recognition and detection tasks like face recognition [1], pedestrian detection [97] or text detection [68, 2]. Several extensions are proposed to make this feature set more discriminative [87, 2]. A more complete overview of this feature set is given in Section 3.1.2.

More recent methods improve the sliding window approach by integrating low-level features with part-based models. Felzenszwalb et al. [29] propose a part-based model learned by a latent-SVM and uses HOG features as low-level features. The main idea of these models is that several object classes consist of parts like wheels on cars or motorbikes. The detector consists of a coarse root-template detector and several part-based detectors. The coarse detector detects candidate object locations. Within these locations part-based detectors detect object-parts such as wheels for motorbikes. If the scores obtained from the parts-detectors is sufficiently large the window is classified as containing the desired object-class. Hence, this detector is more robust against deformations of objects than the original linear HOG-classifier. Variations of this model have won the Pascal Visual Object Class (VOC) challenge, which is hold to assess the state-of-the-art in several computer vision tasks [26].

2.4 Related Work in Text-Detection and Text-Recognition

This section presents an overview of state-of-the-art scene text-detection and -recognition systems. Several competitions within the scope of the ICDAR have been organized to assess the state-of-the-art. Hence, there are publicly available datasets like the ICDAR dataset [86, 80] or the SVT dataset [94], on which objective comparisons of state-of-the-art methods can be done.

2.4.1 Text Detection Methods

Existing text-detection methods can be divided into region based and texture based methods [24, 2]. Region based methods rely on image segmentation. Pixels are grouped to CCs which are character candidates. These candidates are further grouped to candidate words and textlines based on geometric features. Texture based methods distinguish text from non-text based on local features and machine learning techniques. Text confidence maps are created, which are post-processed and converted into word level bounding boxes. Furthermore there exist hybrid approaches which group connected components into word candidates and use a texture based classifier to validate these candidates.

The disadvantage of region based approaches is that they are sensitive to clutter and occlusions which distort the shape of CCs [63]. Natural scene text is exposed to varying illumination conditions, has blur and is partially occluded by other objects [2]. With region based methods, however, pixel-accurate localization of characters and words is possible [70]. Furthermore the segmented characters can be directly used as input to an OCR engine [24].

Texture based methods use sliding windows to detect text regions in images. If the sliding window step size or the image scales are not estimated correctly regarding the text size, text detection errors can occur. In comparison to other object categories like faces, textlines and words have varying aspect ratio. Hence, detector-responses from a sliding window classifier must be further post-processed which is detailed in Section 3.2.

Text detection competitions are hold on a regular basis within scope of the ICDAR. Winner of the ICDAR 2011 competition is Chunghoon Kim [1] (F-score: 71.28%), who uses a hybrid detection method. Candidate textlines are generated by a MSER segmentation. Candidate letters are segmented and grouped to textline hypotheses, which are verified by a texture based method.

Second entry is the text detector proposed by Yi et al. [102] (F-score: 62.32%). This method generates candidate patches with a segmentation method presented in [102]. Based on Canny edges and k-means color clustering text components are segmented. Candidate patches are classified by an AdaBoost texture classifier based on gradient features in block-patterns proposed by Chen et al. [11].

Third entry is TH-TextLoc [80], proposed by Cheng et al. (F-score: 61.98%). CCs are extracted by an adaptive local segmentation method. Then, CCs are filtered by heuristics. In a fine-grained classification step a SVM is used on geometric features, shape related features and stroke width to classify text candidates into text or non-text.

Neumann et al. [62, 61] (F-score: 59.63%) comes on the fourth place of the ICDAR 2011 competition and the method is based on MSER and geometrical features only.

[1] Chunghoon Kim — Qualcomm Korea R&D Center, Seoul 443-749, South Korea — unpublished

The fifth entry is proposed by Shao et al. [82] (F-score: 58.09%), which is a region based approach. For CC classification gradient based features are used. Text-CCs are grouped to text regions.

In the following paragraphs region based approaches, texture based approaches and hybrid approaches are reviewed.

Region Based Methods

Epshtein et al. [24] propose the SWT for text detection. The SWT is an image operator which assigns a stroke width to each pixel of an image (see Figure 2.6). The stroke width is determined by shooting rays on edges in the direction of the gradient. If the ray hits an edge with the same gradient direction modulo 180 degrees, the length of the ray determines the stroke width of the underlying pixels. Pixels which are adjacent to each other belong to the same character if their stroke width is similar. Hence, CCs are formed by segmenting the stroke width map. Adjacent CCs are grouped to words if the median stroke width ratio of two CCs does not exceed 2.0, the height ratio of the two components is smaller than 2.0 and distance and average color difference are within thresholds learned from the training set.

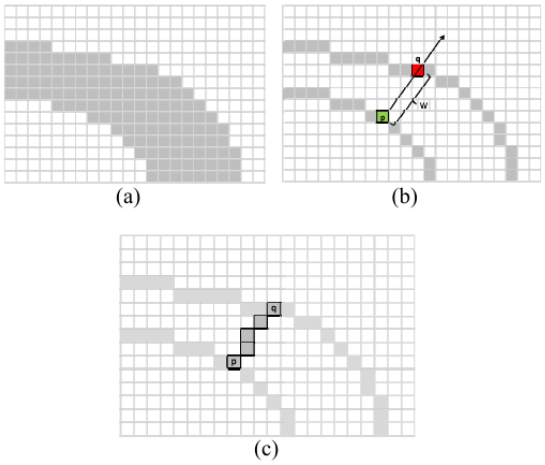


Figure 2.6: The SWT for a stroke shown in (a) is computed by shooting a ray from the edges in the gradient direction (b). If another edge with the same gradient direction is found along the ray, the width between the start- and end-point is assigned as stroke width to the underlying pixels (c). (Source: Epshtein et al. [24])

Chen et al. [10] propose a text detection method using MSER. The outlines of MSER are enhanced by Canny edges. This makes MSER less sensitive to blur. Based on geometric cues these candidate character regions are then grouped to words and textlines.

Neumann et al. [61, 62, 63] propose ERs for segmenting regions. ERs are extracted on the RGB, HSI and gradient images to retrieve character candidate regions. Instead of using heuristics as Epshtein et al. [24] for labeling text, an AdaBoost classifier based on geometric features is used. Text-CCs are then grouped to words.

Yao et al. [101] generalizes the SWT proposed by Epshtein et al. [24] for arbitrary oriented text. Camshift is used to find character orientation, length of major and minor axis and barycenter. CCs are filtered and grouped to chains. False positive chains are eliminated by a RF classifier.

Texture Based Methods

Chen et al. [11] propose an AdaBoost classifier based on x- and y-derivative features, intensity features and edge-linking features computed in block-patterns of the detector-window (see Figure 2.7). Text-patches have low-entropy in this block-patterns.

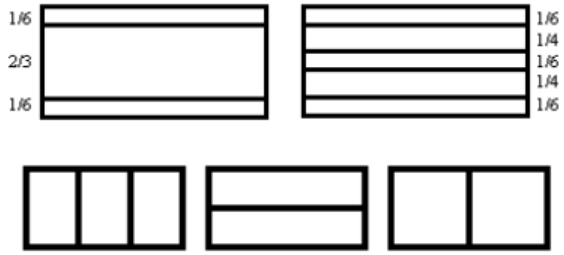


Figure 2.7: Block-patterns proposed from Chen et al. [11].

Pan et al. [68, 69, 70] propose an AdaBoost detector based on HOG and multiscale LBP (msLBP) features computed in blocks as proposed by Chen et al. [11]. The detected regions are post-processed with a CC-labeling stage. The image is segmented with Niblack. CCs are then grouped to a graph and labeled by a CRF. Textlines are formed by minimizing energy functions of a learned distance-metric.

Lee et al. [54] adopt an AdaBoost classifier and use gradient, histogram, CC, color gradient, Gabor and wavelet coefficient features for text detection. Confidence maps are processed by heuristics to find word-level bounding boxes.

Anthimopoulos et al. [2] compute 8 LBP feature maps with 8 different thresholds. The featureset is used in a RF ensemble to create text confidence maps. These confidence maps are then post-processed by smearing techniques on the gradient image and a segmentation to find word-level bounding boxes.

Coates et al. [15] detects text with unsupervised learned features. A k-means variant is used to cluster whitened 8×8 patches to convolution filters. Responses are average-pooled over a local 5×5 field and used as features for a SVM classifier.

Hybrid approaches

Minetto et al. [57] propose a hybrid approach where CCs are labeled with shape descriptors like zernike moments as text or non-text regions. CCs are then grouped to textline candidates. This candidates are then verified by discriminative classifiers using a novel Fuzzy HOG (F-HOG) feature set.

The winner of the recent ICDAR 2011 competition [80] Chunghoon Kim et al. [2] uses MSER to extract candidate letters, which are grouped to candidate textlines and then verified by a texture-based system.

Gonzalez et al. [34] propose a hybrid localization method based on a combination of MSER and local adaptive thresholding for segmentation. CCs are filtered based on geometric features such as aspect ratio or occupy ratio. Hypotheses are created which are verified by a texture based classifier.

2.4.2 Text Recognition

Similar to text-detection systems, recognition systems can be divided into region-based and texture-based approaches. Texture-based approaches use low-level vision features to assign candidate labels to text-windows. These candidate labels are then post-processed by models which incorporate language information to predict the final word. On the contrary, region-based methods assign a candidate label to segmented CCs. Similar to texture-based methods, a language model is then used to predict the final word.

Region Based Methods

Neumann et al. [61, 62, 63] propose a recognition system which classifies MSER regions which are detected by a text-detection system as text-components. Boundaries of normalized MSER regions are inserted into separate images based on their orientation. In total 8 orientations are used. Each orientation image is filtered with a Gaussian and sub-sampled in a 5×5 image. Hence, a $5 \times 5 \times 8 = 200$ dimensional feature vector is used for classifying MSER regions. To improve the prediction, a language model based on bigrams is used.

Texture Based Methods

Wang et al. [94] propose HOG features with a Random Ferns classifier to detect and classify text in an end-to-end setting. The multiclass-detector is trained on cropped synthetic and real-world letters. Non-maxima of the detector responses are suppressed. The remaining letters are then combined in a Pictorial Structure framework, where letters are parts of words. For each word in a dictionary, the most plausible character responses are found in the image. Detected words are then re-scored based on geometric information and non-maxima suppression is done to remove overlapping word-responses.

Mishra et al. [58] propose, similar to Wang et al. [94], HOG features with SVMs to detect and classify text. Instead of using Pictorial Structures, CRFs are used to predict the final word. The model penalizes overlapping candidate-windows, incorporates language information, geometric information and classifier information to predict a candidate word. The final word-prediction is done by retrieving the word in a dictionary with the shortest edit distance.

Yildirim et al. [103] propose Hough Forest (HF) for text recognition. Cross-scale binary features, which are thresholded differences of regions across different scales are proposed for

[2] Chunghoon Kim — Qualcomm Korea R&D Center, Seoul 443-749, South Korea

word recognition. For predicting the final word from candidate-responses a CRF energy function is minimized, which incorporates geometrical information and lexicon priors.

Wang et al. [96] propose deep CNNs with a fixed unsupervised feature layer, which is trained by the method Coates et al. [15] propose. 8×8 whitened patches, which are randomly cropped from text-samples, are clustered with a k-means variant to learn a dictionary of convolution-filters. These features are used for training a two layer CNN with backpropagation, where the first layer is fixed. For post-processing a simple dynamic programming algorithm is proposed, which computes a matching score given the responses of the classifier and a dictionary word. The final word prediction is then done by retrieving the word in a dictionary with the highest matching score.

2.5 Summary

In this chapter an overview of segmentation methods, ANNs and object detection and recognition methods related to this work was given.

Furthermore, an overview of text-detection and -recognition methods was presented. Text detection and recognition methods were divided into region based methods and texture based methods. Recent hybrid text-detection methods were presented, which combine region based methods and texture based methods.

CHAPTER 3

Text Detection

An overview of the proposed end-to-end system is given in Figure 3.1. It is divided into a text-detection module (top) and a text-recognition (bottom), each consisting of several steps. In this chapter the text-detection system is presented.

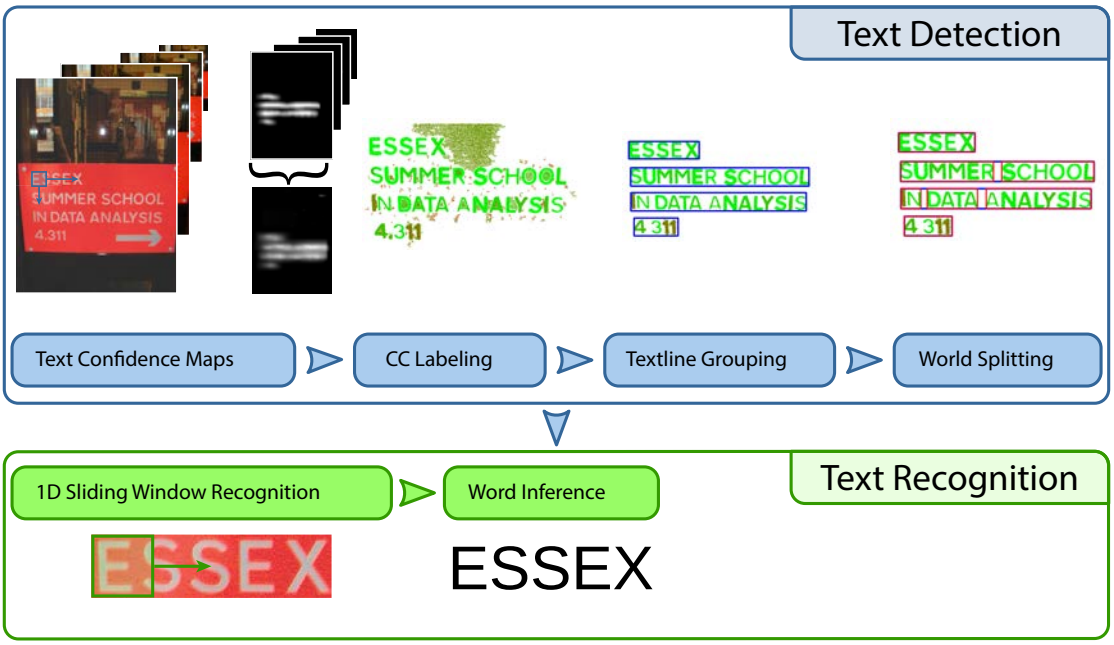


Figure 3.1: The proposed system is divided into a text detection (top) and text recognition module (bottom).

The text-detection module is responsible for creating word-level bounding boxes around words in natural scene images. This task is divided into several steps. First, an AdaBoost classifier detects text in a local window. The classifier is shifted over an image in multiple

scales. Detection responses are accumulated into a text confidence map. Next, within detected regions CCs are extracted and labeled as either text or non-text. From text CCs textlines are formed, which are then split into word-level bounding boxes.

Due to varying illumination, blur, partial occlusions, varying font-sizes and complex background in images [2], the text detection problem is posed as object detection problem. Hence, a sliding window classifier is used to detect textlines in multiple scales in an image. Compared to object detection tasks such as pedestrian detection [17] or face detection [92], text detection has several unique challenges. First, the aspect ratio of a textline varies, which is different from other object classes such as pedestrians or faces, where a fixed aspect ratio or a finite number of different aspect ratios is assumed [29]. Second, textlines have different shapes since textlines consist of different characters. Next, the ICDAR evaluation criteria requires bounding boxes which are pixel-accurate. Hence, approaches such as non-maxima suppression are not sufficient to retrieve bounding boxes. Furthermore, for evaluation word-level bounding boxes and no textline-level bounding boxes are required.

Hence, the text detection system is divided into the following parts:

- **Text Confidence Maps** are created by a sliding window detector.
- **Connected Component Labeling** is done on MSER which are pruned by the text confidence maps.
- **Textline Grouping** is done by linear SVMs to group CCs to textlines. Low-confidence textlines are then, removed.
- **Word Splitting** is done to split textlines into words, obtaining the final bounding boxes.

The remainder of this chapter is divided as follows: in Section 3.1 the AdaBoost sliding window classifier is discussed. In Section 3.2 CC labeling is discussed. In Section 3.3 the textline grouping stage is explained. In Section 3.4 the word-splitting method is detailed. Finally, in Section 3.5 a summary of the detection system is given.

3.1 Text Confidence Maps

Text confidence maps are created by a sliding window classifier. Since for detection tasks about $100,000$ classifications in a 1024×768 image are done, a fast classifier and a fast to compute feature set is needed. Furthermore, since the dimensionality of the feature-sets considered is high ($5,000$-$28,000$), it is desirable to have a classifier which performs feature selection. Features used by the classifier are then computed as needed for the decision making. Further, AdaBoost is used in cascades [92] and soft-cascades [21] with which fewer features have to be computed for non-text regions. Hence, the AdaBoost variant GentleBoost [32] is used for simplicity reasons, and trained in a softcascade as proposed by Dollár et al. [21]. A sigmoid function is used to bound raw-responses of the classifier between $[-1, 1]$. Each window classified as text is accumulated with its response on a response map as illustrated in Figure 3.2. Response maps having a maximum response lower than 1.0 contain mostly non-text and are discarded. This threshold is chosen as the 0.95 quartile of responses on non-text images.

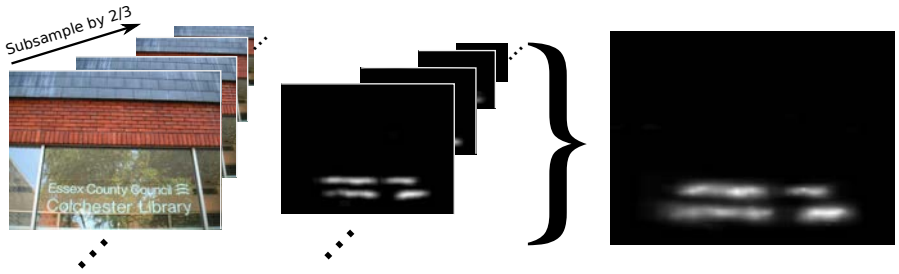


Figure 3.2: For each scale a response map is generated, which are then accumulated into a single map for further processing.

Since several individual characters such as 'I' or 'O' have similar shape to natural scene objects such as vertical edges, fences or street-signs, the detector is trained on random horizontally translated and scaled textline patches as shown in Figure 3.3. The window size is set to 24×12, as proposed by Marios Anthimopoulos et al. [2], such that it contains several characters and patches from the smallest textline in the training set fit into the window. The image is resized at most 10 times by $\frac{2}{3}$ for each scale as proposed by Marios Anthimopoulos et al. [2] and a step-size of 4 pixel is used due to computational reasons. Since the following CC-labeling step of this pipeline is not using an image pyramid, the responses are accumulated in a single response map.

Figure 3.3: 100 sampled training patches.

Since there is more background than text in images, object detection tasks are unbalanced classification tasks. Therefore, bootstrap training [92, 17, 21, 54] is performed. An initial working set of positives is created and an equally amount of negatives is sampled from non-text image regions. An AdaBoost classifier is trained which mines false positive patches over a set of non-text images. These "hard" negatives are added to the pool of negative samples and the classifier is retrained. This process is repeated until the performance on validation images decreases. The influence of the inital working set size, the AdaBoost hyperparameters, the feature-set and the number of bootstrapping iterations are discussed in the following sections.

The requirements of the feature-set for a detector are low computational time and discriminativeness. Hence the following feature sets are evaluated:

- HOG [17] as proposed by Pan et al. [68], since it has a low computational cost, it is

discriminative and several variations are used in state-of-the-art models in object detection [25]. The gradients in this HOG-variant are accumulated in block-patterns proposed by Chen et al. [11].

- HOG as proposed by Dollár et al. [21] in the framework of Integral Channel Features, which is one of the state-of-the-art feature sets used for pedestrian detection [22]. The gradients in this HOG-variant are accumulated in $30,000$ random block-patterns. An AdaBoost learning algorithm is used to select blocks which are most discriminative for classification.
- MACeLBP proposed by Anthimopoulos et al. [2], which achieves state-of-the-art performance in text detection.
- A combination of MACeLBP [2] and LTP [87].

3.1.1 Histogram of Oriented Gradients

HOG [17] is a SIFT-like [55] gradient based feature set which is computed densely on the image. The descriptor as proposed by Dalal and Triggs [17] is illustrated in Figure 3.4. First, the input image is normalized. Then the x-derivatives and y-derivatives of the image are computed. From x- and y-derivatives the gradient magnitude and orientation for a pixel (x, y) is computed as follows:

$$\begin{aligned} \theta(x,y) &= atan2(G_y(x,y), G_x(x,y)) \\ m(x,y) &= \sqrt{G_x(x,y)^2 + G_y(x,y)^2} \end{aligned} \tag{3.1}$$

where $m(x, y)$ is the gradient magnitude, $\theta(x, y)$ is the gradient orientation and $G_x(x, y)$ and $G_y(x, y)$ are the x- and y-derivatives at pixel-position (x, y). The gradient responses are accumulated over local cells into local gradient orientation histograms. The orientations are quantized into discrete bins modulo 180 degrees. Each pixel votes with its gradient magnitude for the entry in the histogram corresponding to its gradient orientation. Cells are then further grouped into overlapping blocks. To achieve robustness against illumination changes each block is normalized. The final feature vector is built by concatenating all normalized blocks in the detection window.

For text detection Pan et al. [68] integrates HOG with a block structure proposed by Chen et al. [11] (see Figure 3.5). For each block the gradients are binned in a 4-orientation gradient histogram. Each histogram is normalized by the standard-deviation of the gradient magnitude of the whole detector window. For the experiments in this work, the number of orientations of this feature set is extended to 8 and for normalization the L1-sqrt norm proposed by Dalal and Triggs is used [17]. This norm is faster to compute than the L2-hys [17] norm and with this configuration the feature set achieves better results on the validation set than the L1-norm [21] and normalization by standard deviation [68].

Dollár et al. [21] extend HOG by computing the gradient histogram in a pool of about 30,000 non-regular random blocks as opposed to a regular grid. An AdaBoost classifier selects blocks which minimize the misclassification error. To compute the feature responses in blocks efficiently, Integral Images [92] on the oriented gradient images are adopted. For the experiments

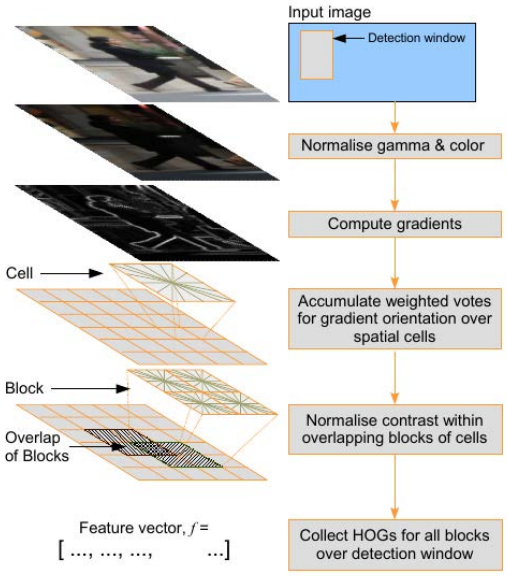


Figure 3.4: An overview of HOG [17]. (Source: [16])

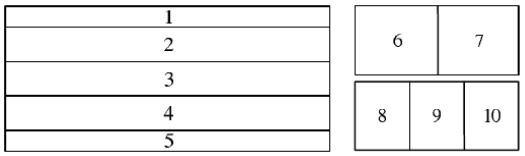


Figure 3.5: Block patterns adopted by Pan et al. [68] for text detection.

in this work, this feature set is called *ChnFtrs*. Furthermore, 8 directional histograms are used and the blocks are normalized by the L1-sqrt norm proposed by Dalal and Triggs [17], which performs better than the L1-norm the authors propose. The pool size is reduced to $5,000$ features, which perform not worse than the proposed $30,000$ features since the window size for the proposed text-detection system (24×12) is several times smaller than the window size used on pedestrian datasets (64×128).

3.1.2 Local Binary Patterns

LBPs [66] are texture descriptors which are computed densely on the image. Features in a window are accumulated and classified subsequently. Formally, the LBP-operator centered on pixel (x_c, y_c) is defined by:

$$LBP(x_c, y_c) = \sum_{n=0}^{7} 2^n s(i_n - i_c) \tag{3.2}$$

where i_n is the grayscale intensity of a pixel in the 8-neighborhood of the pixel positioned on (x_c, y_c), i_c is the grayscale intensity of the pixel (x_c, y_c) and $s(x)$ is defined as:

$$s(x) = \begin{cases} 1, & x > 0 \\ 0, & \text{otherwise} \end{cases} \tag{3.3}$$

This operator is visualized in Figure 3.6. Each 3×3 window is thresholded by the central pixel. From the thresholded boundary pixels of the 3×3 window a binary code is computed. Binary codes in the detector-window are then accumulated into histograms and used as a 256 dimensional feature vector for classification.

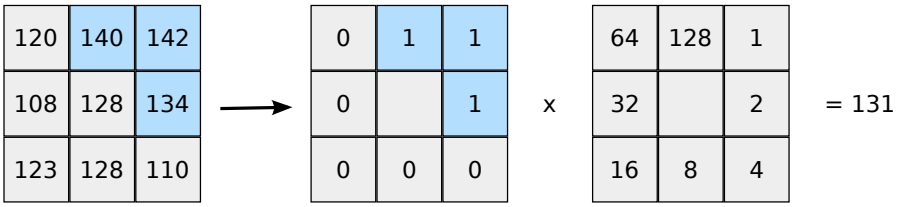


Figure 3.6: LBPs threshold a 3×3 window by the central pixel intensity. The binary pattern of the boundary pixels are then encoded as number between 0 and 255 [87].

MACeLBP proposed by Anthimopoulos et al. [2] is an extension for LBPs designed for detection of text in natural scene images. The proposed feature set encodes texture information in multiple contrast levels. Formally, the coding between central pixel and adjacent pixels in a 3×3 window is defined as follows:

$$s_e(x) = \begin{cases} 1, & |x| > e \\ 0, & otherwise \end{cases} \tag{3.4}$$

where e is a threshold which is estimated locally from the oriented absolute value gradient image, which is shown in Figure 3.7. The gradient of a pixel is defined as maximum absolute difference of the RGB channels. The local mean of the horizontal, vertical, diagonal and anti-diagonal gradient images is computed in a 9×9 window. From these local means thresholds are computed which are used for the LBP encoding.

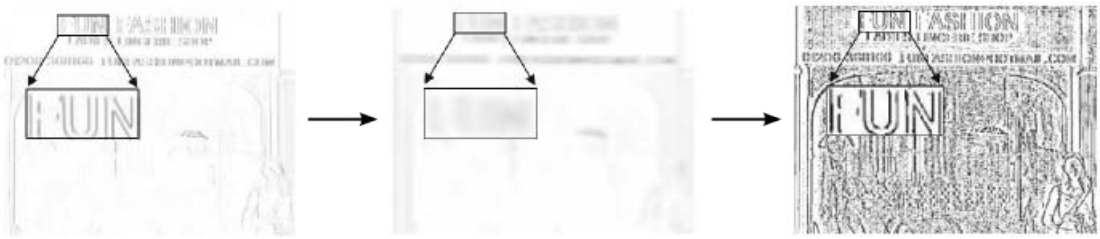


Figure 3.7: Local means of the horizontal absolute gradient images are computed and used to determine the threshold e for the horizontally adjacent neighbors of a center pixel. (Source: [2])

Since the feature set encodes texture information in multiple contrast levels, several thresholds for e are used and the resulting LBP histograms are concatenated to a single feature vector.

Absolute gradient responses are assumed to have a Laplacian distribution. Hence, thresholds for e cluster the Probability Density Function (PDF) . It is desirable to cluster the PDF into clusters of equal probability. Hence, the parameters of the PDF are estimated in a local 9×9 window by computing the mean absolute gradient responses. For the Laplacian distribution a quantile function is defined as follows:

$$F_{exp}(l) = -\lambda^{-1} \cdot ln(1 - l) \tag{3.5}$$

where λ^{-1} is the local mean and $0 \leq l < 1$ is a probability. The function returns a threshold e for which gradient responses smaller than e fall with probability l.

Anthimopoulos et al. [2] use 8 thresholds which cluster the PDF into clusters of equal probability and compute 8 LBP histograms which are used as feature for classification.

3.1.3 Proposed

The proposed feature set is a combination of LTPs [87] and MACeLBPs. LTPs extend the binary coding of the patterns to a ternary pattern. The coding is defined as follows:

$$s_e(x) = \begin{cases} 1, & x > e \\ -1, & x < -e \\ 0, & \text{otherwise} \end{cases} \tag{3.6}$$

where e is a user-defined threshold. This coding is illustrated in Figure 3.8 for $e = 10$. Neighbors which are within radius e of the central pixel are encoded as 0, neighbors bigger than the central pixel are encoded as 1 and neighbors smaller than the central pixel as -1. Furthermore, for encoding the patterns as feature vector, the LTPs are split up into two LBP histograms. Each LBP histogram is then accumulated into a 256 dimensional histogram, which results in a 512 dimensional feature vector.

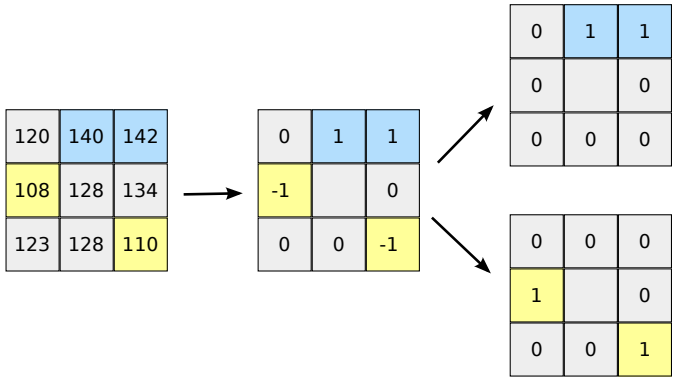


Figure 3.8: LTP encoding of a 3×3 window with threshold $e = 10$ [87].

The LTP pattern is extended by choosing 8 thresholds for e which are estimated as proposed by Marios Anthimopoulos et al. [2] on the oriented gradient image. Instead of accumulating feature responses over the whole detector window, the responses are accumulated in 5 additional

blocks inspired by blocks proposed by Chen et al. [11] (see Figure 3.9). Since a single LTP histogram has a dimensionality of 2×256, the feature vector proposed has a dimensionality of $2 \times 256 \times 8 \times 6 = 24,576$.

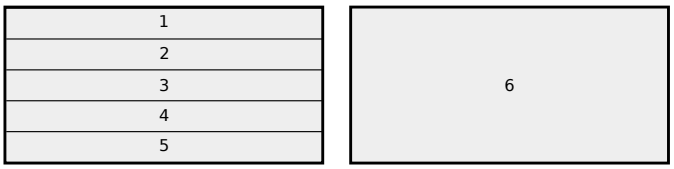


Figure 3.9: Proposed sub-blocks in which the features are extracted.

3.1.4 Comparison of Feature Sets

In this section an overview of the classification performance of feature sets on validation images is given. Since Pan et al. [68] use 5,000 positive samples for their experiments and a combinatorial evaluation of training set size, hyperparameters and feature-set is computationally expensive, for each experiment in this section, the number of positive and negative samples is fixed to 5,000 cases which are randomly sampled from the ICDAR 2003 training images. Furthermore 5,000 positive and negative holdout samples are sampled for calibrating the AdaBoost soft-cascade. In the literature the number of weak learners for the AdaBoost algorithm are set to 92 [11], 100 [54], 121 [68] and 1,000 [21] and the depth of the tree is set to 1 [92, 11], 2 [68, 21], 5 [54]. Hence, a compromise of 200 weak learners and depth 5 decision trees are used for this experiments. As it is shown later, performance for the feature set with the highest dimensionality saturates at about 5,000-10,000 positive samples and an AdaBoost ensemble with these hyperparameters is for the given task optimal.

For each feature set false positives are mined over a set consisting of about 1,000 non-text images. In each bootstrapping iteration 10,000 sampled hard negatives are added to the pool of training samples and the AdaBoost classifier for the feature set is trained from scratch with this bigger pool of training samples.

For evaluating the classifier in isolation from the remaining system, an evaluation method proposed by Coates et al. [15] is used. Each pixel inside a text bounding box is assigned a positive label, and each pixel outside a text bounding box is assigned a negative label. From the response maps a precision-recall curve is then swept out by thresholding the responses with several thresholds. For images having no responses 100% precision is assumed and 0% recall.

For each feature set several bootstrapping iterations are done. After each iteration the feature set is evaluated on the validation set and precision, recall, F-score and Area Under Curve (AUC) are recorded for the best threshold.

The performance of each feature set in question and the influence of the number of bootstrapping iteration is shown in Table 3.1. The precision recall curves are shown in Figure 3.10. The proposed feature set outperforms the MACeLBP and HOG feature sets. Furthermore 1-2 bootstrapping iterations are sufficient for reaching optimal performance. Sampling more hard negatives reduces recall and improves precision. Mining no hard negatives in images results in

insufficient performance. It can be seen that texture based features outperform gradient based features.

Feature Set	# FP	AUC	Precision	Recall	F-score
Proposed	5000	0.489	0.429	0.630	0.511
Proposed	15000	**0.701**	0.671	0.655	**0.663**
Proposed	25000	0.651	0.739	0.592	0.657
MACeLBP	5000	0.494	0.463	0.602	0.524
MACeLBP	15000	0.651	0.628	0.600	0.613
MACeLBP	25000	0.650	0.623	0.601	0.612
MACeLBP	35000	0.638	0.632	0.588	0.609
ChnFtrs	5000	0.382	0.356	0.550	0.432
ChnFtrs	15000	0.596	0.661	0.545	0.597
ChnFtrs	25000	0.594	0.644	0.577	0.609
ChnFtrs	35000	0.576	0.665	0.547	0.600
ChnFtrs	45000	0.561	0.632	0.530	0.577
HOG	5000	0.368	0.380	0.489	0.428
HOG	15000	0.588	0.573	0.596	0.584
HOG	25000	0.587	0.605	0.565	0.584
HOG	35000	0.567	0.595	0.539	0.565

Table 3.1: Validation image results on the best two feature sets.

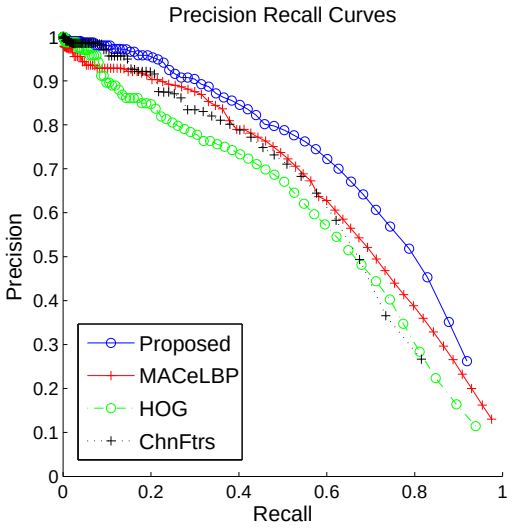


Figure 3.10: Precision and recall curves of several feature sets.

3.1.5 Experiments on the Number of Initial Positive Samples

For the best performing feature set experiments for choosing the number of initial positive samples are conducted. Similar to the experiments in the previous section, the classifier is trained from an initial working set of an equal number of positive and negative samples. The initial size of the positive samples in this experiments are 2,500, 5,000, 10,000 and 20,000. The upper limit of samples is due to computational constraints.

Several bootstrapping iterations over a set of images containing no text patches are done until the F-score on the holdout validation images decreases. Results are shown in Table 3.2 and Figure 3.11. The performance increases with a bigger set of positive samples until it saturates between 5,000 and 10,000 positive samples. Furthermore, for bigger initial training set sizes more hard negatives are mined in training images. Since there are only about 700 different textlines from which the classifier learns to detect different shaped text, the performance saturates. Due to this limitation, at about 20,000 positive samples the performance slightly decreases, since there are more hard negatives in the training set, compared to the experiment with 10,000 samples.

Positive Samples	# False Positives	F-score
2500	12500	0.647
5000	15000	0.663
10000	30000	0.676
20000	40000	0.669

Table 3.2: Influence of the number of positive training samples on the validation set performance.

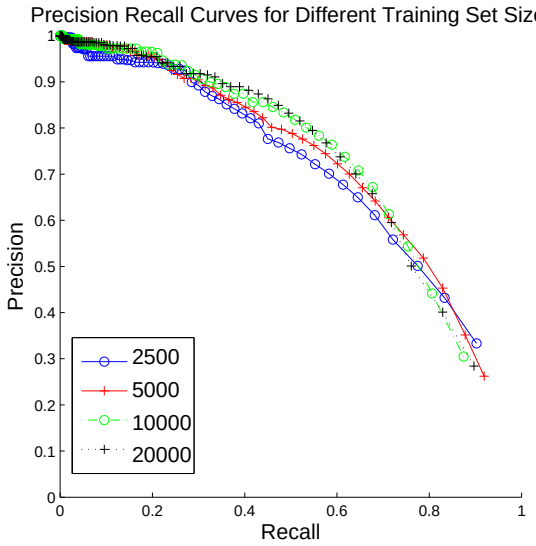


Figure 3.11: Precision and recall curves for training set sizes of 2,500, 5,000, 10,000 and 20,000 positive samples.

3.1.6 Experiments on the Classifier Hyperparameters

The proposed AdaBoost algorithm has two hyperparameters to tune: the number of weak learners and the maximum depth of the decision trees. In previous work decision trees are proposed as weak learners. Depth of trees is set to 1 (decision stumps) [92], 2 [68] or 5 [54]. Therefore experiments with this 2 hyperparameter configurations are conducted. Since in the literature weak learners of size 92 [11], 100 [54], 121 [68] and 1,000 [21] are used, for the number of weak learners, a logarithmical grid of 100, 200, 400, 800 and 1,600 learners is chosen. As training set a bootstrapped training set mined with an AdaBoost classifier consisting of 200 weak learners is chosen. The set consists of 5,000 positive samples and 15,000 negative samples.

The results are shown in Table 3.3 and Figure 3.12. The performance of the feature set is stable across a variety of different hyperparameters. For deeper trees a lower number of weak learners is required to get comparable performance to shallower trees. Furthermore, the AdaBoost learning algorithm does not overfit on the parameter grid analyzed. Since more weak learners require higher computational effort at classification and training time, a compromise of 200 weak learners and depth 5 decision trees is chosen as hyperparameters for the final system.

Depth	# Trees	AUC	Precision	Recall	F-score
2	1600	0.70	0.67	0.66	0.66
2	800	0.71	0.65	0.67	0.66
5	800	0.69	0.69	0.63	0.66
5	200	0.70	0.69	0.63	0.66
5	1600	0.68	0.69	0.63	0.66
2	400	0.70	0.66	0.65	0.66
5	400	0.70	0.67	0.64	0.66
5	100	0.69	0.67	0.64	0.65
1	800	0.68	0.68	0.63	0.65
1	1600	0.68	0.64	0.67	0.65
2	200	0.69	0.66	0.63	0.65
1	400	0.68	0.64	0.65	0.65
1	200	0.66	0.64	0.62	0.63
2	100	0.67	0.60	0.65	0.63
1	100	0.63	0.63	0.59	0.61

Table 3.3: Performance of different Hyperparameters on the validation set.

3.2 Connected Component Labeling

Text confidence maps provide an approximate location of the textline which is shown in Figure 3.13. The responses overlap with background noise in the top and bottom part of the image. When the accumulated responses are processed by smearing techniques on the gradient magnitude image as proposed by Anthimopoulos et al. [2], a box for a horizontal edge above “PostPak” and a enlarged box for the text “Office” is retrieved.

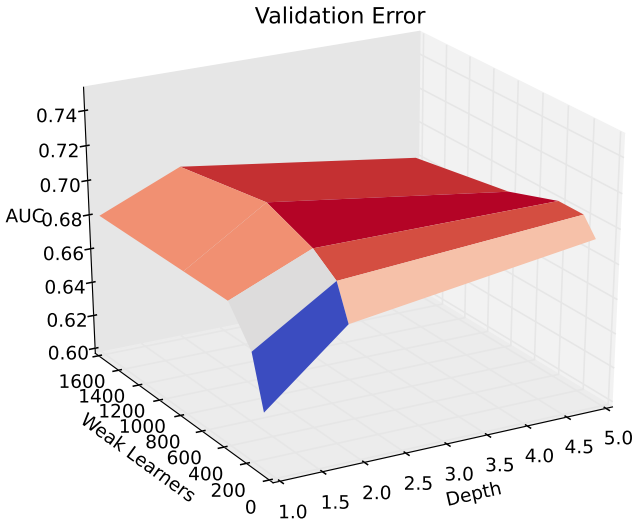
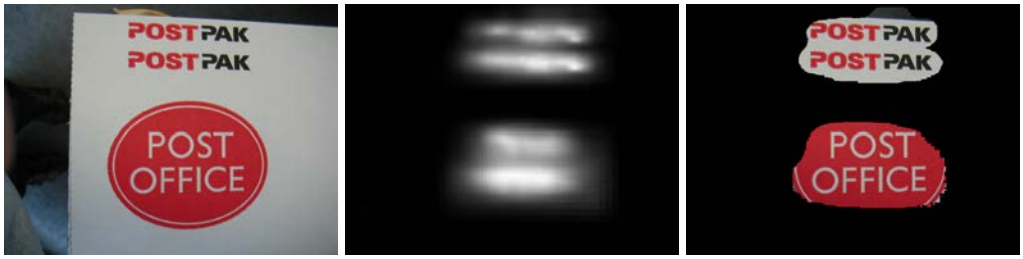


Figure 3.12: Performance of different Hyperparameters on the validation set.



(a) Input image. (b) Accumulated responses. (c) Thresholded responses.

(d) Boxes retrieved by a smearing technique on gradient magnitude image.

(e) Boxes retrieved by the method proposed.

Figure 3.13: Exact localization issues response-maps.

Since the evaluation criteria requires pixel-accurate word-level bounding boxes, a segmentation is performed However, regions overlapping with confidence maps contain background noise. Hence, machine learning techniques are used to label CCs as either text or non-text.

For segmentation MSER [56] are used, which are proposed in several text detection systems [63, 80, 10]. As opposed to segmentation approaches in document analysis such as Otsu [67] or Niblack [65], MSER provide a multi-segmentation of the image. Hence, more candidate regions are extracted which are classified by a machine learning pipeline as either text or non-text. Furthermore, Neumann et al. [63] shows that extracting ERs on color-channels (RGB and HSI) and *intensity gradient-image* increases the character detection rate of ERs to 94.8%. The *intensity gradient-image* is defined as maximum difference of a pixel and its neighbors on the I-channel of the HSI color representation. Since the proposed labeling-pipeline is computationally expensive, MSER on the grayscale image and the U, V channels of the LUV colorspace representation of the image are extracted. The reason for this is shown in Figure 3.14. The image has a luminance gradient in the grayscale image and thus cannot be segmented with MSER on the grayscale image. On the L and U channel of the LUV representation, however, the text does not have a gradient and hence can be segmented.

(a) Color image. (b) Grayscale Image. (c) U channel of the LUV representation. (d) V channel of the LUV representation.

Figure 3.14: Different color-channels of an image with a luminance gradient.

MSER are first pruned against thresholded text confidence maps by $0.1 \times MV$ where MV is the maximum value of the response map. Next, several features are extracted for each region, which are used by a RF [7] and SVM [8] ensemble to predict a confidence with which the region is a text-region. Overlapping MSER are removed by using the component-tree provided by the algorithm. From the remaining regions a graph is constructed by linking components close to each other. Two components are close to each other if the following condition is met [68]:

$$dist(i,j) < 2 \cdot min(max(w_i, h_i), max(w_j, h_j)) \tag{3.7}$$

where w_i, h_i are width and height of component i and $dist(i,j)$ is the euclidean distance between centroids of components i and j.

A CRF is used to label the nodes of the graph by combining text-confidences with geometric information about the neighboring CCs. Pan et al. [68, 69, 70] shows the effectiveness of CRFs for CCs labeling in scene text. Furthermore, CRFs combine unary and, in this method, pairwise features to make a decision. Since adjacent text has similar properties like color, bottom alignment, ..., these neighboring information help to distinguish text from non-text.

3.2.1 Maximally Stable Extremal Regions

Matas et al. [56] propose MSER for stereo matching. MSER is a region detection method, which detects regions invariant to monotonic illumination changes and affine transformations. The MSER algorithm works as follows: The grayscale image is thresholded several times with increasing thresholds t. Each thresholded image consists of several CCs which are called ER. ERs in images of different thresholds form a parent-child relationship where child-regions are nested in parent regions. Hence, a component-tree is build. Let $\mathcal{Q}_1, \mathcal{Q}_2, \ldots \mathcal{Q}_n$ be a sequence of nested ERs. For each ER $\mathcal{Q}_i$ a stability measure $q(i)$ is defined as follows:

$$q(i) = \frac{|\mathcal{Q}_{i+\Delta} \backslash \mathcal{Q}_{i-\Delta}|}{|\mathcal{Q}_i|}$$

where $|\cdot|$ denotes the area of a region, $A \backslash B$ denotes the set of pixels in A which are not in B, and Δ is a parameter of the method. A larger Δ results in less regions retrieved, since a region i has to be stable over a larger grayscale range. Hence, Δ is set to 1 for the proposed system. $q(i)$ measures the area of change of the regions $\mathcal{Q}_{i-\Delta}$ and $\mathcal{Q}_{i+\Delta}$, normalized by the region $\mathcal{Q}_i$. Hence, ERs having a local minima of $q(i)$ are defined as MSER.

The algorithm as explained above detects MSER which are local minima in the grayscale image. To detect local maxima, the grayscale image is inverted and the algorithm is applied again.

In the system proposed MSER are extracted from the grayscale, U and V image. To efficiently prune overlapping MSER the component-tree is used. Leaf nodes are propagated up the hierarchy, replacing their parents if the maximum of their confidence is bigger than the confidence of the parent. Since the MSER algorithm detects one component tree for each pass, the components of $3 \times 2 = 6$ component trees are pruned against each other. Hence, components C_i and C_j are replaced by the one with the one with the bigger confidence if the following condition is holds:

$$\frac{|CC_i \cap CC_j|}{|CC_i \cup CC_j|} > 0.4$$

where $\cap$ is the intersection of the components and $\cup$ the union of the components. The threshold of 0.4 is found experimentally by optimizing the F-score on the ICDAR sample set.

3.2.2 Feature-Set

Since the system proposed uses discriminative models and graphical models for CC labeling, the feature-set consists of unary features computed for single CCs and pairwise features for pairs of CCs. The feature-pool is a combination of features proposed by Neumann et al. [63], Pan et al. [68, 69, 70] and Epshtein et al. [24] and extended with 3 features derived from the minimum area rectangle: area over minimum area rectangle area, minimum area rectangle compactness and minimum area aspect ratio. The list of the 14 features used is shown in Table 3.4, where C denotes a set of pixels corresponding to a CC and $|\cdot|$ denotes the area of the CC. w and h are the width and height of the component. Furthermore, $P(C)$ denotes the set of pixels on the contour

of C, $H(C)$ denotes the convex hull of C and $SW(C)$ the mean stroke width of C. For features fitting a minimum area rectangle, w_{mr} and h_{mr} denotes the width and height of this rectangle.

aspect ratio	$a = \frac{w}{h}$
contour gradient	$\frac{1}{\lvert P(C)\rvert} \cdot \sum_{(x,y)\in P(C)} G(x,y)$ where $G(x,y)$ denotes the gradient magnitude at position (x,y)
area over bounding box area	$\frac{\lvert C\rvert}{w \cdot h}$
area over convex hull area	$\frac{\lvert C\rvert}{\lvert H(C)\rvert}$
compactness	$\frac{\lvert C\rvert}{\lvert P(C)\rvert^2}$
bounding-box compactness	$\frac{w \cdot h}{\lvert P(C)\rvert^2}$
euler number	
three horizontal crossings	as proposed by Neumann et al. [63]
stroke width standard deviation	
mean stroke width	$\frac{SW(C)}{max(w,h)}$
area over minimum area rectangle area	$\frac{\lvert C\rvert}{w_{mr} \cdot h_{mr}}$
minimum area rectangle compactness	$\frac{w_{mr} \cdot h_{mr}}{\lvert P(C)\rvert^2}$
minimum area rectangle aspect ratio	$\frac{w_{mr}}{h_{mr}}$

Table 3.4: Proposed unary feature-set.

The stroke width of a MSER is estimated with the SWT operator proposed by Epshtein et al. [24]. The SWT is performed on Canny-edges and the gradient image. Hence, SWT-computation is fast and does not need to be repeated for every MSER, which is desirable for images consisting of several thousands of regions. Since the SWT is performed on Canny-edges and the gradient image, regions detected by Canny-edges differ from MSER. This is illustrated in Figure 3.15. The smaller 'A' is detected by the MSER-detector and edges of the bigger 'A' are detected by the Canny edge detector. Non-negative stroke-entries of the SWT image are used to compute the proposed stroke width features. Since the SWT is computed in two passes, the stroke width image with the most non-negative overlapping entries is used for feature computation.

For the pairwise features a combination of features proposed by Pan et al. [68, 69, 70] and Epshtein et al. [24] is used. Let w_i, h_i denote the width and height, x_i, y_i the centroid

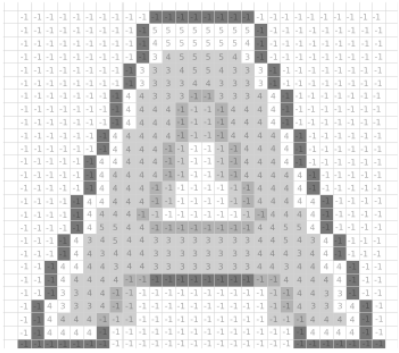


Figure 3.15: MSER (gray) and edges detected by Canny are not overlapping. The stroke widths are densely computed by the SWT [24].

coordinates, t_i, b_i the y-coordinate of the upper boundary and the lower boundary and $\mathbf{c_i}$ the RGB-color vector of a CC C_i. The proposed feature-set is shown in Table 3.5.

horizontal distance	$\frac{\lvert x_i - x_j \rvert}{max(w_i, w_j)}$
vertical distance	$\frac{\lvert y_i - y_j \rvert}{max(h_i, h_j)}$
difference to upper-boundary	$\frac{\lvert t_i - t_j \rvert}{max(h_i, h_j)}$
difference to lower-boundary	$\frac{\lvert b_i - b_j \rvert}{max(h_i, h_j)}$
color difference	$\lVert \mathbf{c_i} - \mathbf{c_j} \rVert_2$
vertical ratio	$\frac{max(h_i, h_j)}{min(h_i, h_j)}$
horizontal ratio	$\frac{max(w_i, w_j)}{min(w_i, w_j)}$
stroke width ratio	$\frac{min(SW(C_i), SW(C_j))}{max(SW(C_i), SW(C_j))}$
vertical overlap	$\frac{min(b_i, b_j) - max(t_i, t_j)}{max(h_i, h_j)}$

Table 3.5: Proposed pairwise feature-set.

MSER regions on the ICDAR 2003 training set are labeled as either text or non-text. Unary and pairwise features are extracted. On the unary features a discriminative model is trained to classify a CC as either text or non-text. On pairwise features three one-vs-all RF models are trained, since two adjacent CCs are either labeled as both text, text and non-text, or both as non-text. The predictions of these models are combined with a CRF which combines unary and pairwise information to label a component as either text or non-text. The CRF is trained with dlib [43].

For MSER labeling in context of text detection a high recall of about 95% [63] is desired. Hence, models are compared by precision at a fixed recall of 95%.

Radial Basis Function (RBF)-SVMs and RFs are considered for discriminative models. RFs are tree ensembles and hence can cope with inputs having diversity of ranges. Furthermore RFs, predict a probability which is bounded in $[0, 1]$. By bounding the predictions in a range no further normalization is required when the predictions are used for training a CRF. RBF-SVMs are applied to several real-world computer vision problems [17, 71, 100, 98, 29, 58]. The RBF-SVM is calibrated to output probability predictions between $[0, 1]$, such that it can be combined with the RF by simple averaging. Inputs for the SVM are normalized to have zero mean and unit standard deviation.

Cross-validation results for recall-values of about 95% for each model are shown in Table 3.6.

Method	Precision	Recall
CRF	0.825	0.955
RF-SVM	0.703	0.955
RF	0.686	0.955
SVM	0.546	0.955

Table 3.6: Cross-validation results for each model at recall of 0.955.

3.3 Textline Grouping

The next step in the text-detection system is grouping text-components to textlines. Since there are false positive text-components from the previous step (see Figure 3.16), a robust method for grouping components to textlines is proposed. In the literature for region based methods, thresholds [24, 11, 62] optimized on the training set, as well as SVMs [61] and distance metric learning with energy functions [68, 69, 70] are proposed. Since parameter search on a grid for 9 features is computationally expensive on a single machine, for simplicity reasons a combination of a trained SVM on these 9 features and thresholds for 2 features found by gridsearch on the ICDAR sample set is used to group components.

First, two components are considered for grouping if they are close to each other as defined in Equation 3.7. Next, to avoid grouping text components with non-text components a linear SVM is trained to classify a pair of CCs as belonging to the same text line or not.

The SVM is trained on the pairwise features presented in Section 3.2.2. The positive training set consists of adjacent CCs in the same textline. The negative training set consists of positive CC pairs which are close to each other but in different textlines, or CC pairs which are close to each other and having different class labels. These three cases are illustrated in Figure 3.17. Green lines correspond to positive samples and red lines to negative samples. The edges connecting the first letters of “computer” to ’M’ in the top part of the figure are negatives, since the textline differs. Edges from ’H’, ’S’, ’B’, C’ to non-text regions are examples for the second subcategory of negatives.

Figure 3.16: The grouping stage has to be robust against false positives from the labeling stage.

Figure 3.17: Adjacent text-CCs in the same textline are positive cases (top and bottom), adjacent text-CCs in different textlines (top) and adjacent text and non-text CCs (bottom) are negative cases. Negative cases in the top figure are shown for 'M'.

In addition to the SVM prediction, the minimum vertical overlap and horizontal ratio is constrained by 0.4 and 2.0 respectively. These parameters are found by optimizing F-score on the ICDAR sample dataset.

Textline candidates which are grouped according to these rules are pruned with several rules to eliminate false positives. First, as proposed by Epshtein et al. [24], groups consisting of two or fewer CCs are removed.

Next, groups with a lower confidence score than a threshold g_c are discarded. Due to misclassification errors, textline candidates contain non-text components which are shown in Figure 3.18. These non-text components are assigned a low-confidence score by the discriminative model and reduce the average confidence score in candidate textlines. Since the successive steps in this algorithm are designed to be robust against such noise, low-confidence CCs below g_c are ignored if by removing them the area of the textline does not shrink below $0.9 \times a_l$ where a_l is

the area of the textline. This allows to set higher thresholds for g_c and increase the robustness of the performance on the ICDAR sample set by 3% on F-score.

Finally, overlapping groups are removed. Let G_i denote group i, G_j group j and $|\cdot|$ the bounding box area of a component. G_i is overlapping with G_j if the following condition holds:

$$\frac{A_{G_i \cap G_j}}{A_{G_i}} > 0.6$$

where $A_{G_i \cap G_j}$ is the area of intersection of the CCs of each group as defined by:

$$A_{G_i \cap G_j} = \sum_{\substack{CC_i \in G_i, \\ CC_j \in G_j}} |CC_i \cap CC_j|$$

and A_{G_i} is defined as:

$$A_{G_i} = \sum_{CC_i \in G_i} |CC_i \cap CC_j|$$

G_i is discarded if it is overlapping with G_j and its confidence is lower than G_j and vice versa.

Figure 3.18: False positive CCs in a textline have low-confidence scores.

3.4 Word Splitting

The last step of the text detector is splitting grouped textlines into words. Two models are considered for splitting textlines into words. The first model classifies a gap between letters as either word-gap or non-word-gap.

The second model is motivated by the fact that natural scene text has variation in letter-spacing (see Figure 3.19). Distances between words are clustered by k-means into two clusters. Distances belonging to the smaller cluster are non-word-gaps and distances belonging to the bigger cluster are word-gaps.

This works if the textline consists of several words. Since natural scene text has short textlines consisting of single words, a method is needed which distinguishes single-word textlines from multi-word textlines. In textlines with several words the distance between the two cluster-centroids is large compared to the height of the letters and the median distance between components. Hence, a linear SVM is trained to distinguish these two cases with the following two features:

- centroid difference over height: $\frac{|c_i - c_j|}{H}$ where H denotes the height of the textline.

- centroid difference over median distance: $\frac{|c_i - c_j|}{MD}$ where MD denotes the median distance between the components.

The training set is extracted from the ICDAR 2003 training images.

If the SVM classifies the textline as multi-word textline, the k-means textline splitting algorithm is applied. Otherwise a bounding box for the whole textline is returned. Furthermore, textlines consisting of only a single gap are considered as single-word textline.

Since in several candidate texlines false positives are persistent, a robust method for calculating the distance between components is used.

Figure 3.19: Text in natural scene images has varying letter spacing.

To compute distances, projection profiles on the segmented components are used to estimate distances between components. Since components are connected with background noise or have a serif typeface, distances on the vertical projection profiles are non-uniform in words. Hence, projection profiles are thresholded by the 0.25 quartile of the histogram. To avoid splitting characters with small stroke like 'U', the bounding boxes of the CCs are narrowed on the border of the box as shown in Figure 3.20. The vertical projection profiles and the 0.25 quartile for the word is calculated. Each bounding box is narrowed up to $0.1 \times w$ where w is the width of the box. The distances between the vertical projections of the boxes are used for further processing.

For the k-means approach a further heuristic is introduced which prohibits splitting textlines into words if the number of word-gaps is larger than the number of non-word-gaps. This reduces the impact of misclassification errors of the SVM. If a single-word textline is misclassified as multi-word the textline is split by k-means as illustrated in Figure 3.21. By assuming that a textline consists of more non-word gaps than word-gaps, these errors are removed.

The model predicting word gaps as component distance over height achieves a cross-validation accuracy of 88.6%, and the combined model achieves a crossvalidation-accuracy of 97.5% for predicting word-gaps.

3.5 Summary

In this chapter the text detection algorithm was presented. The detection task was split up into a confidence map generation step, a connected component labeling step and a textline grouping and word-splitting step.

First confidence maps were generated by a sliding window classifier which used a combination of LTP [87] and MACeLBP [2]. It was shown that on holdout validation images the feature set outperforms MACeLBP [2], HOG [68] and a random block HOG variant proposed

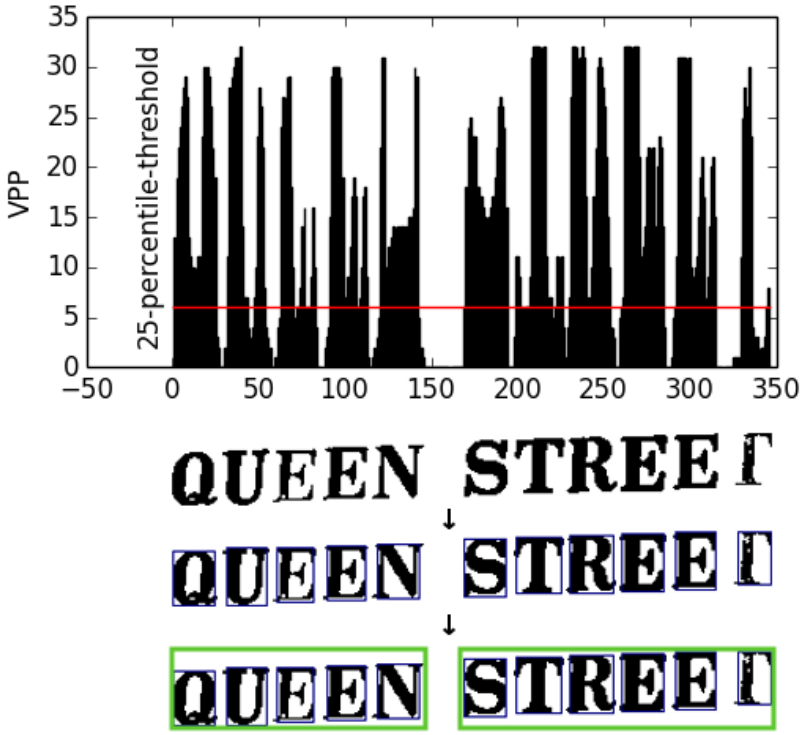


Figure 3.20: Vertical projection profiles for word splitting.

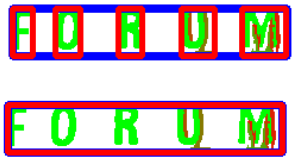


Figure 3.21: Splitting due to SVM misclassifications.

by Dollár et al. [21]. Furthermore, size of the training-set and influence of hyperparameters were discussed.

The extracted confidence maps were thresholded and used for connected component labeling. MSER were extracted on the grayscale, U and V image, pruned with the thresholded maps and several features were computed from these regions. A graph from the components was built which was labeled with a CRF. The CRF utilizes unary predictions from a SVM-RF ensemble and pairwise predictions from three RF classifiers.

Labeled CCs were grouped to textlines with a linear SVM. Low-confidence textlines and textlines consisting of fewer than 2 components are pruned.

Finally, textlines were classified as multi-word and single-word. Multi-word textlines were split by k-means into several words.

The main advantage of the proposed system is its robustness. The sliding window classifier and the graph-labeling step filter out non-text components. Furthermore, due to the linear SVM and the thresholds for maximum height ratio and minimum vertical overlap in the grouping stage, non-text components are not grouped into a textline. To alleviate segmentation errors components are additionally extracted from color channels. Finally, the word splitting stage is designed to split textlines containing false positive components.

One disadvantage of the proposed system is its runtime performance. A separate sliding window classifier is used to create text confidence maps. Further, the LTP feature-set is compared to HOG computationally more expensive and requires more memory. The feature-set in the CCs labeling stage scales linear with the number of extracted components. Furthermore, labeling requires classifications of two unary classifiers, three pairwise classifiers and a CRF inference. Next, the system is designed to detect horizontal text, since features such as difference to lower-boundary, difference to upper-boundary or vertical overlap are used for grouping and the ICDAR training set consists of horizontally aligned text. Finally, the system uses a separate segmentation step, which is sensitive to clutter and occlusions which distort the components [63].

CHAPTER 4

Text Recognition

In this chapter an overview of the text recognition system is presented. The text recognition part is responsible for recognizing cropped words given a list of candidate words from a dictionary. This problem is divided into two tasks. First a 1D sliding window classifier predicts a candidate character sequence for a textline. This candidate predictions are integrated with a language model which incorporates a dictionary to recognize the final word.

The remainder of this chapter is structured as follows. In Section 4.1 the sliding window classifier are discussed. In Section 4.2 an overview of the language model used to infer the word is given. Finally, in Section 4.3 a summary of the recognition module is given.

4.1 Sliding Window Classifier

The task of the sliding window classifier is to predict the character shown in a cropped window. For text classification deep CNNs are used, since they outperform handcrafted features on large datasets in several classification tasks including scene character and scene digit recognition [46, 14, 12, 88, 36, 96]. Furthermore, the network proposed is trained with backpropagation without unsupervised pretraining. Recent work shows that such networks have competitive performance compared to unsupervised pretrained and finetuned models and outperform only unsupervised pretrained models [41]. Since unsupervised pretraining with supervised finetuning is computationally expensive, and have a larger hyperparameter-space to explore, in this work a CNN trained with backpropagation is proposed.

The CNN is used as 1D sliding window classifier on cropped word images to create candidate character predictions, which are used in the prediction step to infer the most plausible word in a dictionary.

The remainder of this section is structured as follows. In Section 4.1.1 an overview of CNNs, backpropagation and Dropout is given. In Section 4.1.2 the training set is discussed. In Section 4.1.3 network architecture and hyperparameter choice is discussed.

4.1.1 Convolutional Neural Network Overview

This section provides an overview of Feed Forward ANNs, backpropagation, CNNs and Dropout. The notation used in figures and equations follows the conventions of Bishop et al. [5].

Feed Forward ANNs are classifiers which consist of several layers of neurons which is shown in Figure 4.1. Neuron activations are computed layer by layer. The activations of the last layer are the predictions of the network.

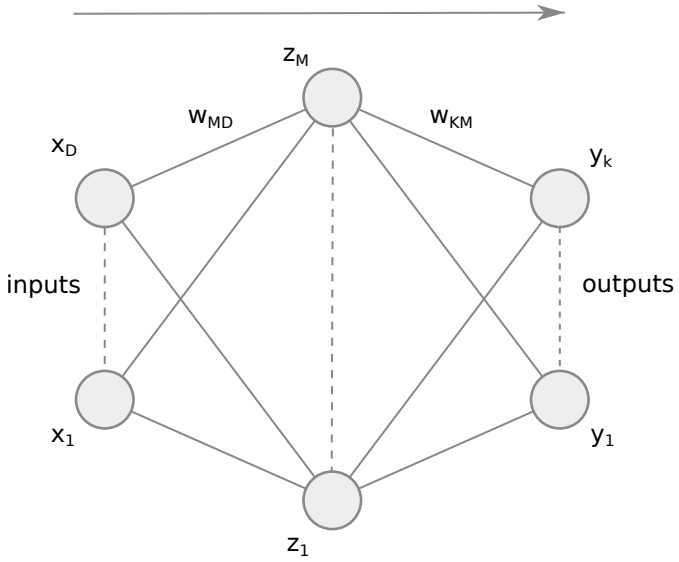


Figure 4.1: Feed Forward ANNs. (Source: [5])

More formally let x_i be an input, the activation of the hidden layer z_j is computed as:

$$z_j = h\left(\sum_{i=1}^{D} w_{ji} \cdot x_i\right)$$

where $h(x)$ is an activation function and w_{ji} are the input-to-hidden weights. For clarity reasons the bias term is left out.

The output y_k for the network in Figure 4.1 is computed by:

$$y_k = g\left(\sum_{j=1}^{M} w_{kj} \cdot z_j\right)$$

where $g(x)$ is an activation function, which is not necessarily the same as $h(x)$, and w_{kj} are the hidden-to-output weights. The activations of neurons are defined as:

$$a_k = \sum_{j=1}^{M} w_{kj} \cdot z_j$$

$$a_j = \sum_{i=1}^{D} w_{ji} \cdot x_i$$

where a_k are activations of neurons y_k and a_j are activations of neurons z_j.

The weights w_{ij} and w_{kj} are learned by training on labeled data with backpropagation. Backpropagation minimizes an error criterion $E(\mathbf{w})$. To minimize this criterion the gradient of the function $E(\mathbf{w})$ with respect to $\mathbf{w}$ is computed, which is then used in a numeric optimization algorithm such as Stochastic Gradient Descent (SGD) to update the weights. Gradients are computed layer by layer starting from the output layer (see Figure 4.2). First, the gradients from the output layer are backpropagated to the neurons of the hidden layer. Hidden Layer neurons accumulate all gradients from outgoing connections and propagate the error back to the inputs of these neurons.

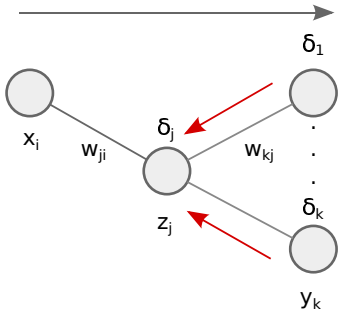


Figure 4.2: Backpropagation in Feed Forward ANNs. (Source: [5]).

More formally, let $E(\mathbf{w}) = \frac{1}{2}C\sum_{k=1}^{K} max(0, 1 - t_k \cdot y_k)^2$ be the squared hinge loss, $t_k \in \{-1, 1\}$ the targets for dimension k and y_k the activation for the last layer. Then by applying the chain rule:

$$\frac{\partial E(\mathbf{w})}{\partial w_{kj}} = \frac{\partial E(\mathbf{w})}{\partial a_k} \cdot \frac{\partial a_k}{\partial w_{kj}} = \delta_k \cdot z_j$$

where δ_k is defined as:

$$\delta_k \equiv \frac{\partial E(\mathbf{w})}{\partial a_k} = C \cdot max(0, 1 - t_k \cdot y_k) \cdot (-t_k) \cdot g'(a_k)$$

The partial derivatives for the input-to-hidden weights are computed as follows:

$$\frac{\partial E(\mathbf{w})}{\partial w_{ji}} = \frac{\partial E(\mathbf{w})}{\partial a_k} \cdot \frac{\partial a_k}{\partial a_j} \cdot \frac{\partial a_j}{\partial w_{ji}} = \delta_j \cdot x_i$$

where

$$\delta_j \equiv \frac{\partial E(\mathbf{w})}{\partial a_j} = \sum_{k=0}^{K} \delta_k \cdot \frac{\partial a_k}{\partial a_j} = h'(a_j) \cdot \sum_{k=0}^{K} \delta_k \cdot w_{jk}$$

CNNs are ANNs which have layers with shared weights as illustrated in Figure 4.3 for a one dimensional CNN. Each neuron in layer l_{i+1} is connected with three neurons in layer l_i. The weights between these layers are shared, which is indicated by the colors red, green and blue.

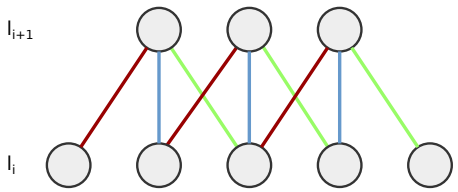


Figure 4.3: Illustration of an one dimensional CNN. (Source: [85])

To predict the activations for the next layer a convolution operation on the inputs of the layer is applied.

Convolutional Layers are followed by a subsampling operation, where responses are pooled in a local receptive field to a single value by computing mean, max, Lp-pooling [78], or sampling [104].

Dropout is a regularization method proposed by Hinton et al. [40] for regularizing ANNs by randomly setting inputs during training to 0. This can be seen as efficiently training an ensemble of 2^N different networks, where N is the number of neurons in the network. During prediction the geometric mean of these 2^N networks is computed efficiently by scaling the weights by 0.5.

4.1.2 Training Set

Since labeled datasets such as MNIST [51], CIFAR-10 [45], CIFAR-100 [45], SVHN [60] and ImageNet [19] on which CNNs are applied range from 50,000 to over 1 million samples and the ICDAR training set consists of about 6,000 samples for 62 classes, a synthetic text-dataset for training is created. Synthetic training sets for text recognition are proposed by Wang et al. [94], Yildirim et al. [103] and Wang et al. [96] for training character classifiers for natural scene text.

Similar to Wang et al. [96] and Yildirim et al. [103] from a database of about 500 fonts a synthetic dataset consisting of 50,000 samples is generated (see Figure 4.4). Each training patch shows a centered character with 0-2 adjacent characters. The text has random text-color, is blurred by a Gaussian filter with probability 0.05, perspectively distorted and blended with a natural scene image. The font-database consists of no small-caps fonts. Lowercase letters which do not have ascenders are padded with 3-7 pixels on the top border of the patch. The distribution of characters in the synthetic dataset matches the distribution of characters in the ICDAR training set. Optimal distortion parameters are found by optimizing performance by training a RF classifier on HOG features (4×4 pixel cell size, 8×8 pixel overlapping blocks with a stride of 4 pixels) on the synthetic training set and maximizing accuracy on the ICDAR

training set. The reason for the choice of HOG features and RF for optimizing performance is computational expense of CNNs. HOG parameters are found by optimizing accuracy on the ICDAR training set.

Figure 4.4: 100 sampled synthetic training samples.

The patch-size is set to 30×30 pixels, which is of a similar size proposed by Wang et al. [96] (32×32), Yildirim et al. [103] (24×24) and Wang et al. [94] (48×48).

For training the ICDAR 2003 training set is split in two subsets. One half is used as holdout validation set for hyperparameter estimation. The other half is perspectively distorted, blurred and added to the training set. The final training set consists of about 60,000 samples.

4.1.3 Network Architecture

An overview of the proposed network architecture is shown in Figure 4.5 and is similar to other architectures proposed for computer vision tasks [50, 46, 40, 36]. The network consists of three convolutional layers with a kernel size of 5×5 pixels. The number of different feature maps learned per convolutional layer is 32-128-128 which is constrained due to computational reasons. Each convolution layer is followed by a max-pooling operation which operates on 3×3 receptive fields with a stride of 2×2. Hence, receptive fields overlap on the border. Each input for a convolution layer is padded by a border of 2 such that the result of the convolution has the same dimension as the input. The convolutional layer is followed by a fully connected layer consisting of 2048 neurons. The last layer is a classification layer. For activation ReLUs ($g(x) = max(0, x)$, where $g(x)$ is the activation) are used, since they are faster to train and perform better for some problems compared to sigmoids [46, 41]. As proposed by Wang et al. [96] and Tang et al. [88], for the classification layer the squared SVM-Hinge loss ($\sum_{i=1}^{N} max(0, 1 - y_i \cdot t_i)^2$, where $t_i \in \{-1, 1\}$ are the ground-truth labels and y_i are the predictions) is backpropagated.

For preprocessing local contrast normalization as proposed by Jarrett et al. [41] is performed. Patches are normalized within a 7×7 window to have zero mean and unit standard deviation. The network is trained by a modified version of the Pylearn2 library [35] and optimized CUDA kernels for fast convolutions from Krizhevsky et al. [46].

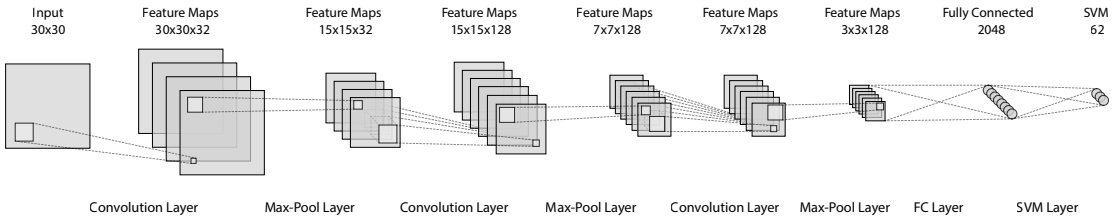


Figure 4.5: Proposed CNN architecture, adapted from LeCunn et al. [50]

For regularization the L^2 norm of the kernel-weights in convolutional layers and the column norms of the remaining layers are constrained. Furthermore, layers are regularized by Dropout. The optimal hyperparameters are found via human guided search on the validation set described in Section 4.1.2 in about 100 experiments each of which takes 4-8 hours to on a NVIDIA 770 GTX complete. For training the networks the advice of Bengio et al. [3] is followed. First the highest learning rate is found with which the network does not diverge. Too high learning rates result in too high updates of weights which prevent the network from learning anything. Too low learning rates result in too slow updates and the optimization is subject to get stuck in local optimums. Then a CNN is trained to achieve 0% training error, with which the capacity needed to learn the training set is known. Since the network is regularized with Dropout, which changes the effective size of the network to 50% during training, the size of the second and third convolution layer as well as the last hidden layer are doubled. On the bigger network the learning rate is adjusted to the highest value with which the network does not diverge. The remaining parameters are found by a combination of random gridsearch [4] and coordinate descent optimization of individual parameters. Finally, the termination criterion of the network is relaxed, so that the network trains for a larger number of epochs. Several experiments are conducted to find out if by locally changing any regularization parameters a better validation error is achieved.

Hence, the final model is trained by constraining the L^2 norm for the convolutional layers to 1.5, 1.9 and 1.9 and the L^2 norm for the remaining layers to 1.9. Dropout is applied with probability of 0.1 in convolutional layers and probability of 0.5 in fully connected layers.

The filters learned are shown in Figure 4.6. The first layer consists of oriented edge detectors. In the second layer filters for corners and rounded contours are learned. The second-layer filters are visualized numerical maximization of the input by the method used by Le et al. [48].

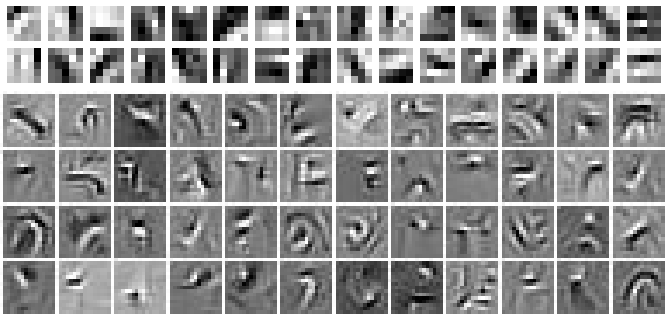


Figure 4.6: Random sample of layer 1 and layer 2 filters learned by the network.

4.2 Cropped Word Recognition

To predict a word from the confidence scores of the CNN the method proposed by Wang et al. [96] is used, since it is simpler compared to other methods proposed by Mishra et al. [58] and Wang et al. [94].

The result of the sliding window classifier is a $62 \times N$ dimensional response matrix $\tilde{M}$, where N is the number of sliding windows in the cropped word patch. Since word recognition is evaluated case-insensitive, an augmented $36 \times N$ dimensional response matrix M is computed where predictions for case-insensitive letters are defined as maximum over the lowercase and uppercase letters.

Next, non-maxima-suppression as proposed by Wang et al. [96] is done. For non-maxima-suppression a confidence score c_j for each column of the matrix M is computed. The confidence c_j for column j is defined as difference of the highest and second highest response over the 36 classes in column j. To obtain candidate characters, non-maxima on the confidence scores c_j in 1×4 window are suppressed. Furthermore entries of c_j below a confidence threshold t_c are suppressed. For suppressed entries of c_j, the columns j of matrix M are set to $-\infty$.

To predict the final word, Wang et al. [96] propose a Viterbi-style algorithm, which minimizes the following score for a given word w as follows:

$$S_M^w = max_{l^w \in L^w} \left(\sum_k^{|w|} M(w_k, l_k^w) \right)$$

where S_M^w is the score of a dictionary word w given a matrix M, l_w is an alignment vector between characters of the word w and positions on the response matrix. $l_k^w = n$ means that the k^{th} character in the word is assigned to the n^{th} column of M.

Since choosing the word with best score according to S_M^w from a dictionary does not work, Wang et al. [96] introduces additional penalty terms.

Word-assignments which do not start or end at the border of the response map are penalized by $\frac{d_l+d_r}{l} \cdot p_b$, where d_l is the distance of the first assignment to the left border of the response-matrix, d_r is the distance from the last assignment to the right border of the response-matrix, l is the number of columns of the response-matrix and p_b is a parameter.

Furthermore, assignments are penalized by the standard-deviation of the character distances. Let x_1, x_2, ...,x_n denote the columns of matrix M which are assigned to word w. Then $d_i = |x_{i+1} - x_i|, i \in [1, n-1]$ denotes the distance between assignments. For the set of distances $d_i, i \in [1, n-1]$ the standard deviation σ_d is computed. The score is hence penalized by $\sigma_d \cdot p_s$ where p_s is a parameter.

If a word does not contain an 'i' or 'l' it is penalized for an average gap-size smaller than 8 pixels by $max(0, 8.0 - \frac{l}{|w|}) \cdot p_n$, where l is the number of columns of the response-matrix, and p_n a parameter.

Finally, adjacent assignments which deviate from 8 pixels are penalized by $-e^{-\frac{(|x_i - x_{i+1}| - 8)^2}{4}} \cdot p_a$ where x_i denotes a column of matrix M which is assigned to the letter i of word w and p_a is a parameter.

The parameters t_c, p_b, p_s, p_n, p_a are found by random parameter search [4] on the cropped word training set on a logarithmic grid of $\{0\} \cup [10^{-2}, 10]$ for the parameters t_c, p_a, p_s, a grid of $[10^{-2}, 10^2]$ for p_n, and a grid of $\{0\} \cup [10^{-3}, 10^2]$ for p_b.

4.3 Summary

In this chapter the text recognition system was presented. It consisted of a one-dimensional sliding window classifier which recognized characters in a 30×30 window. The detector was slided horizontally over a cropped word. Non-maxima of the recognizer-responses were suppressed. A Viterbi-style algorithm proposed by Wang et al. [96] was used to compute a matching score for a candidate word from a dictionary. The best-matching candidate word was used as prediction for the cropped word image patch.

The main advantages of the proposed text recognition system are that it does not need a separate segmentation step and that the word-inference method is simpler compared to other methods [58, 103, 84]. Furthermore, no prior knowledge in feature-engineering is needed, since features are learned by the CNN.

The disadvantages of the system are that deep CNN are computationally expensive at classification time and training time and that the word-inference step does not scale to dictionary sizes of thousands of words. Further, since the ICDAR ground-truth boxes are axis oriented rectangles, words are assumed to be horizontally aligned

CHAPTER 5

Results

In this chapter the performance of the system proposed is assessed experimentally. To compare this work with other methods, it is evaluated on several publicly available datasets where ground-truth data is available. The remainder of this chapter is structured as follows. Section 5.1 gives an overview of the text detection results on the ICDAR 2003 [86] and ICDAR 2011 [80] datasets. In Section 5.2 the character recognizer and cropped word recognizer are evaluated on the ICDAR 2003 [86] and the ICDAR 2011 [80] dataset and the SVT [94] dataset. Section 5.3 shows end-to-end scene text recognition results. In Section 5.4 errors of the end-to-end system are analyzed. Finally, Section 5.5 summarizes this chapter.

5.1 Text Detection Results

In this section quantitative comparisons of the text detector on the ICDAR 2003 and ICDAR 2011 dataset with other methods are done. Furthermore, to compare the effectiveness of different proposed models, experiments with different labeling classifiers are done on the ICDAR 2003 dataset in Section 5.1.3. For each configuration the group threshold g_c, maximum height ratio, minimum vertical overlap are fixed to values optimized on the ICDAR sample set. For discriminative models a lower confidence threshold l_c is introduced and optimized on the ICDAR sample set. CCs in the CC-labeling stage, which are below this threshold are removed.

In Section 5.1.4 the influence of the grouping parameters and the linear SVM in the grouping step of the algorithm is discussed.

Next, in Section 5.1.5 the influence of the UV-color channels of the LUV image representation are discussed on the ICDAR 2003 dataset on the best classifier combination. The parameters g_c, maximum height ratio, minimum vertical overlap are fixed to values found on the ICDAR sample set. The system is evaluated in a setting where CCs are only extracted from the grayscale image, and in a setting where CCs are extracted from grayscale L and U image.

In Section 5.1.6 the influence of the word splitting method is compared against a splitting method which classifies gaps between CCs based on distances-over-height as feature as described in Section 3.4.

Next, to compare the influence of response maps on the text-detector, the proposed RF-SVM-CRF labeling system is evaluated on response maps created by an AdaBoost classifier trained on feature sets presented in Section 5.1.7. Furthermore the system is evaluated on ground-truth response-maps and blank response maps, simulating a setting with a perfect sliding window classifier and a setting with no sliding window classifier at all. In these experiments g_c and the threshold of the response map are swept out to create a precision-recall curve.

Finally, in Section 5.1.8 several errors of the detection module are listed and a quantitative error analysis on 30 sampled images is conducted.

5.1.1 ICDAR 2003 Results

In this section comparative results with other methods on the ICDAR 2003 dataset are presented. The system is evaluated according to the ICDAR 2003 evaluation protocol [86] with the publicly available Deteval tool [99]. The results are shown in Table 5.1. The proposed LTP-RF-SVM-CRF detector achieves a precision of 0.825, a recall of 0.675 and a F-score of 0.742, achieving competitive performance on this dataset.

Anthimopoulos et al. [2] uses a RF with MACeLBP to create text confidence maps and smearing techniques on the gradient image to detect word level bounding boxes. The improvement over their method is due to a more discriminative featureset and a more robust post-processing stage.

Pan et al. [68, 69, 70] uses as segmentation a Niblack variant to label CCs. Furthermore, energy models are used for textline grouping. The improvement is due to a better segmentation stage.

Method	P	R	F
LTP-RF-SVM-CRF (Proposed)	**0.825**	0.675	**0.742**
Anthimopoulos et al. [2]	0.82	0.61	0.70
Lee et al. [54]	0.66	**0.75**	0.70
Pan et al. [69]	0.67	0.68	0.69
Gonzalez et al. [34]	0.81	0.57	0.67
Epshtein et al. [24]	0.73	0.60	0.66
Yao et al. [101]	0.68	0.60	0.66
Chen et al. [10]	0.73	0.60	0.66
Robust Reading Competition 2005			
Hinnerk Becker	0.62	0.67	0.62
Alex Chen	0.60	0.60	0.58
Qiang Zhu	0.33	0.40	0.33
Jisoo Kim	0.22	0.28	0.22
Nobuo Ezaki	0.18	0.36	0.22
Robust Reading Competition 2003			
Ashida	0.55	0.46	0.50
HWDavid	0.44	0.46	0.45
Wolf	0.30	0.44	0.35
Todoran	0.19	0.18	0.18

Table 5.1: Results on ICDAR 2003.

5.1.2 ICDAR 2011 Results

On the ICDAR 2011 dataset the proposed method is evaluated according to the ICDAR 2011 evaluation protocol [80] with the publicly available Deteval tool [99]. The method proposed achieves a precision of 0.845, a recall of 0.702 and a F-score of 0.767 (see Table 5.2).

The improvement over Neumann et al. [63] is achieved by combining text confidence maps with better labeling model.

Method	P	R	F
LTP-RF-SVM-CRF (Proposed)	**0.845**	**0.702**	**0.767**
Neumann et al. [63]	0.731	0.647	0.687
Yao et al. [101]	0.776	0.555	0.647
Gonzalez et al. [34]	0.727	0.56	0.632
Robust Reading Competition 2011 [80]			
Kim et al.	0.83	0.625	0.713
Yi et al.	0.672	0.5810	0.623
TH-TextLoc System	0.670	0.577	0.62
Neumann's Method	0.689	0.525	0.60
TDM_IACS	0.635	0.535	0.581
LIP6-Retin	0.63	0.507	0.558
KAIST AIPR System	0.446	0.597	0.510
ECNU-CCG Method	0.35	0.383	0.366
Text Hunter	0.501	0.26	0.342

Table 5.2: Results on ICDAR 2011.

5.1.3 Comparison of Labeling Classifier

To assess the performance differences of the labeling classifier presented in Section 3.2, the proposed classifiers are evaluated on the LTP feature set, since it has the highest validation set performance as shown in Section 3.1.4. Performance is measured with the ICDAR 2003 evaluation protocol and ICDAR 2011 evaluation protocol. The parameters g_c, l_c are set according to experiments on the ICDAR 2003 sample set. The results are shown in Table 5.3. Difference between the best-performing and worst-performing system is 1.3% (F-score). The SVM has an impact on performance on the metric proposed by Wolf et al. [99] of 3.2%. To illustrate the reason for these differences, qualitative comparisons for each model against the best performing LTP-RF-SVM-CRF are done. For each image the difference in F-score (Wolf) to the result of LTP-RF-SVM-CRF is computed. The top-3 images with highest difference are retrieved and compared against each other.

Method	P	R	F	P (Wolf)	R (Wolf)	F (Wolf)
LTP-RF-SVM-CRF	0.825	0.675	0.742	0.811	0.661	0.728
LTP-RF-CRF	0.800	0.675	0.732	0.752	0.635	0.689
LTP-RF-SVM	0.790	0.677	0.729	0.777	0.666	0.718
LTP-RF	0.802	0.669	0.729	0.75	0.630	0.685

Table 5.3: Precision Recall and F-scores on the ICDAR 2003 dataset evaluated with the ICDAR 2003 evaluation protocol and the protocol proposed by Wolf et al. [99] which is used for ICDAR 2011 evaluation.

The top-3 images where the LTP-RF-SVM-CRF detector outperforms the LTP-RF detector are shown in Figure 5.1. Due to the low threshold $l_c = 0.1$, which is found by optimizing

performance on the ICDAR sample set, RF fails to retrieve low-confidence text-components which align with text components. Furthermore, since the minimum vertical overlap is higher for this detector-combination, letter-parts on the boundary are not grouped to textlines (see Figure 5.1e). Half of the letter 'G' lies outside the image. Hence, the letter is segmented into two CCs. The lower letter part is not grouped to the textline by the LTP-RF-SVM classifier, since the vertical overlap of the CC is too small.

(a) Result of LTP-RF-SVM-CRF (+0.50).

(b) Result of LTP-RF.

(c) Input.

(d) Result of LTP-RF-SVM-CRF (+0.50).

(e) Result of LTP-RF.

(f) Input.

(g) Result of LTP-RF-SVM-CRF (+0.50).

(h) Result of LTP-RF.

(i) Input.

Figure 5.1: Qualitative comparison of results for LTP-RF-SVM-CRF and LTP-RF.

Comparisons of the LTP-RF-SVM-CRF detector and the LTP-RF-SVM detector are shown in Figure 5.2. Errors of LTP-RF-SVM are due to grouping errors as illustrated in Figure 5.2a, 5.2b and 5.2c. In the first figure the letter '6' is split into two components. The LTP-RF-SVM classifier fails to group the component to the textline resulting in a smaller bounding box. According to the evaluation protocol proposed by Wolf et al. [99] the retrieved box is not detected. In the second letter a part of letter 'G' is not grouped to the textline due to too small vertical overlap, which results in a lower F-score for the LTP-RF-SVM method. In the last picture the same problem happens with part of the letter 'N'. On the retrieved textline the word splitting algorithm fails, since the gap between 'E' and 'S' is assigned to the intra-word cluster by k-means.

Comparisons of LTP-RF-SVM-CRF and LTP-RF-CRF are shown in Figure 5.3. The re-

(a) Result of LTP-RF-SVM-CRF (+1.00).

(b) Result of LTP-RF-SVM.

(c) Input.

(d) Result of LTP-RF-SVM-CRF (+0.50).

(e) Result of LTP-RF-SVM.

(f) Input.

(g) Result of LTP-RF-SVM-CRF (+0.40).

(h) Result of LTP-RF-SVM.

(i) Input.

Figure 5.2: Qualitative comparison of results for LTP-RF-SVM-CRF and LTP-RF-SVM.

trieved boxes are the same as for the LTP-RF detector. Errors are due to grouping errors and misclassifications, which result in errors in the word splitting step (see Figure 5.3h). The letters of 'STAR WARS' are not detected by the RF, the part of the letter 'G' is not grouped to the textline due to a too small vertical overlap, and in the last image too much background noise is grouped into the textline, due to which the word-splitting algorithm fails.

(a) Result of LTP-RF-SVM-CRF (+0.50).

(b) Result of LTP-RF-CRF.

(c) Input.

(d) Result of LTP-RF-SVM-CRF (+0.50).

(e) Result of LTP-RF-CRF.

(f) Input.

(g) Result of LTP-RF-SVM-CRF (+0.50).

(h) Result of LTP-RF-CRF.

(i) Input.

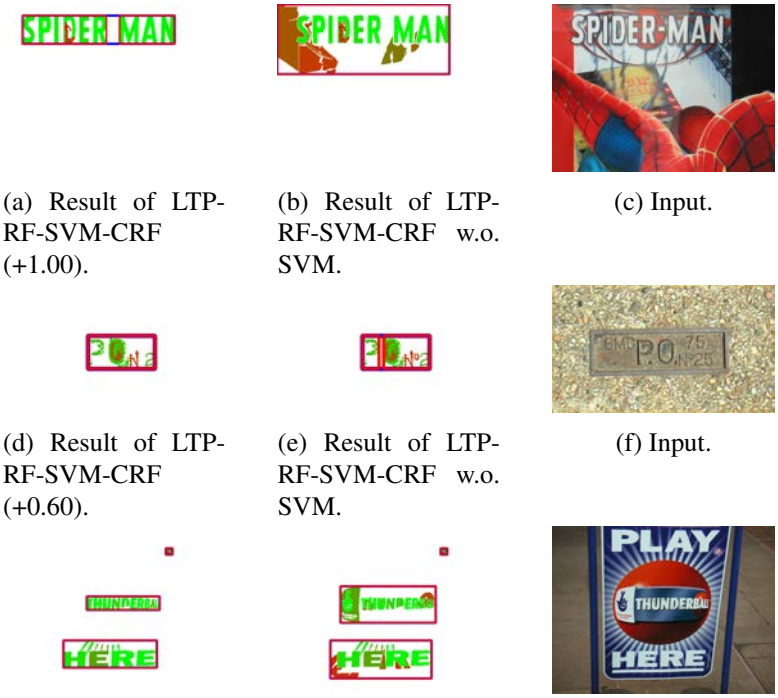
Figure 5.3: Qualitative comparison of results for LTP-RF-SVM-CRF and LTP-RF-SVM.

5.1.4 Influence of the Grouping Parameters and Grouping SVM

To assess the influence of the linear SVM in the grouping stage, the detector is run on the ICDAR 2003 dataset without the linear SVM. Hence, all components for which the grouping thresholds are valid and which are close to each other, as defined in Equation 3.7, are grouped to a textline. Results of this experiment is shown in Table 5.4. The performance difference is 1.4% (0.9% [99]). The grouping thresholds in the textline-grouping stage of the system work reliably without the SVM. The SVM improves performance by filtering out not aligned text components. The top-3 images with highest difference are shown in Figure 5.4. In the first and last images background components are grouped into the textline. In the second example the degree-sign is grouped into the textline due to which the word-splitting algorithm fails.

Method	P	R	F	P (Wolf)	R (Wolf)	F (Wolf)
LTP-RF-SVM-CRF	0.825	0.675	0.742	0.811	0.661	0.728
LTP-RF-SVM-CRF w.o. SVM	0.798	0.669	0.728	0.789	0.661	0.719

Table 5.4: Comparison of LTP-RF-SVM-CRF with and without grouping SVM.



(a) Result of LTP-RF-SVM-CRF (+1.00).

(b) Result of LTP-RF-SVM-CRF w.o. SVM.

(c) Input.

(d) Result of LTP-RF-SVM-CRF (+0.60).

(e) Result of LTP-RF-SVM-CRF w.o. SVM.

(f) Input.

(g) Result of LTP-RF-SVM-CRF (+0.33).

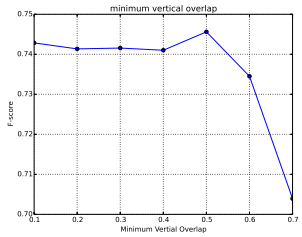
(h) Result of LTP-RF-SVM w.o. SVM.

(i) Input.

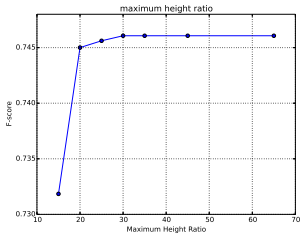
Figure 5.4: Qualitative comparison of LTP-RF-SVM-CRF with and without grouping SVM.

Forthermore, to show the influence of the minimal vertical overlap and maximal horizontal ratio parameter, gridsearch on these two parameters is performed. In this experiments the linear SVM classifier for grouping is not disabled. In Figure 5.6a the influence of the minimal vertical overlap is shown, keeping the minimal height ratio fixed at the best found value 2.5, and in Figure 5.6b the influence of the minimal height ratio is shown, keeping the minimal vertical overlap parameter fixed to the best found value of 0.5. A plot of both parameters is shown in Figure 5.6c. It is shown that minimal height ratio is stable starting from a threshold of 2.0. Minimum vertical overlap stays constant at 0.4. Disabling the grouping heuristics by setting minimal vertical overlap to -10,000 and minimal height ratio to 100, results in an F-score of 73.8% (71.9% [99]). If the same experiment is repeated for the LTP-RF-SVM combination similar results are achieved (see Figure 5.6). The influence of maximal height ratio is stable if it is big enough (2.0) and performance decreases if minimal vertical overlap decreases. The performance difference is, however, larger compared to LTP-RF-SVM-CRF, since the discriminative models filter fewer false positives at this stage of the algorithm. Hence, more false positive CCs are grouped to textlines in the grouping stage. Disabling the parameters by setting minimal vertical overlap to -10,000 and minimal height ratio to 100, results in an F-score of 71.9% (70.8% [99]).

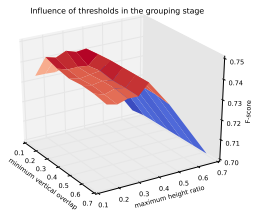
Further, the influence of the threshold for the minimum number of CCs is assessed. The LTP-RF-SVM-CRF detector is run on the ICDAR 2003 dataset with a minimum number of



(a) Minimal vertical overlap with fixed maximal height ratio at 2.5.

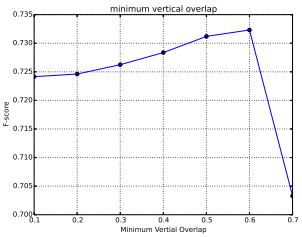


(b) Maximal height ratio with fixed minimal vertical overlap at 0.5

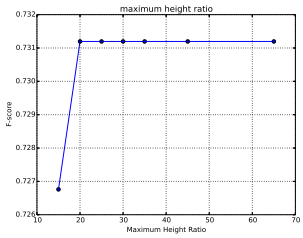


(c) Performance by varying maximal height ratio and minimal vertical overlap.

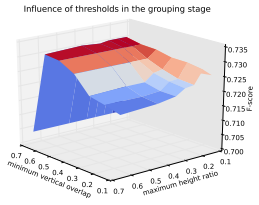
Figure 5.5: Influence of the grouping parameter.



(a) Minimal vertical overlap with fixed maximal height ratio at 2.5.



(b) Maximal height ratio with fixed minimal vertical overlap at 0.5



(c) Performance by varying maximal height ratio and minimal vertical overlap.

Figure 5.6: Influence of the grouping parameter of RF-SVM.

components in a textline of 2 and 1. Results are shown in Table 5.5. With a grouping threshold of 1, the performance drops significantly.

Method	P	R	F	P (Wolf)	R (Wolf)	F (Wolf)
LTP-RF-SVM-CRF	0.825	0.675	0.742	0.811	0.661	0.728
LTP-RF-SVM-CRF-2	0.797	0.693	0.741	0.772	0.669	0.717
LTP-RF-SVM-CRF-1	0.630	0.714	0.667	0.634	0.673	0.654

Table 5.5: Influence of the minimum number of CCs in a textline.

5.1.5 Influence of UV Color Channels

The influence of detecting MSER regions in the UV color channels is presented in this section. The best detector LTP-RF-SVM-CRF is applied on just the grayscale image. Performance of the two systems on the ICDAR 2003 dataset is shown in Table 5.6.

As Neumann et al. [63, 62, 61] demonstrates, retrieving MSER from color channels improves performance of the detector. The LTP-RF-SVM-CRF detector achieves an 1% higher

Method	P	R	F	P (Wolf)	R (Wolf)	F (Wolf)
LTP-RF-SVM-CRF-Gray	0.83	0.65	0.732	0.62	0.792	0.693
LTP-RF-SVM-CRF	0.825	0.675	0.742	0.81	0.661	0.728

Table 5.6: Influence on color channels on the ICDAR 2003 dataset.

F-score the ICDAR 2003 metric and a 3.5% higher F-score on the metric proposed by Wolf et al. [99], which is used for evaluation on the ICDAR 2011 dataset. The top 3 images where the difference in F-score on the metric proposed by Wolf et al. [99] is largest are listed in Figure 5.7. By extracting MSER from U and V channels, more candidate characters are segmented. The grayscale segmentation for the first picture is shown in Figure 5.7c. The character-parts are not detected by the detector and hence the textline is not retrieved. Characters detected in Figure 5.7g have segmentation errors on the grayscale image and hence are not detected by LTP-RF-SVM-CRF-Gray. The yellow letters in Figure 5.7l are not a MSER on the grayscale image and hence not retrieved by LTP-RF-SVM-CRF-Gray.

(a) Result of LTP-RF-SVM-CRF (+1.00).

(b) Result of LTP-RF-SVM-CRF-Gray.

(c) Grayscale Segmentation.

(d) Input.

(e) Result of LTP-RF-SVM-CRF (+1.00).

(f) Result of LTP-RF-SVM-CRF-Gray.

(g) Grayscale Segmentation.

(h) Input.

(i) Result of LTP-RF-SVM-CRF (+0.39).

(j) Result of LTP-RF-SVM-CRF-Gray.

(k) Grayscale Segmentation.

(l) Input.

Figure 5.7: Qualitative comparison of LTP-RF-SVM-CRF and LTP-RF-SVM-CRF-Gray.

5.1.6 Influence of Word Splitting

The influence of the proposed word splitting method is assessed by comparing the method with a version which uses a word-splitting model with distance over component-height as presented in Section 3.4 (LTP-RF-SVM-CRF-H) The LTP-RF-SVM-CRF-H is outperformed by LTP-RF-SVM-CRF by 5.4% (6.4% [99]). The reason for this is varying letter spacing of fonts. By clustering distances by k-means into two clusters, word gaps are assigned to the larger cluster independently of the letter-spacing. To avoid splitting single words, this model uses centroid-distance normalized by the median distance between components as one of its features. Hence, this feature is invariant to letter-spacing.

In Figure 5.8 a qualitative comparison of splitting models is shown. The k-means method has less errors on samples where the distance of non-word gaps normalized by the character height exceeds the threshold found by LTP-RF-SVM-CRF-H for splitting words.

Method	P	R	F	P (Wolf)	R (Wolf)	F (Wolf)
LTP-RF-SVM-CRF	0.825	0.675	0.742	0.81	0.661	0.728
LTP-RF-SVM-CRF-H	0.748	0.638	0.688	0.627	0.706	0.664

Table 5.7: Influence of the proposed word splitting method.

(a) Result of LTP-RF-SVM-CRF (+1.00).

(b) Result of LTP-RF-SVM-CRF-H.

(c) Input.

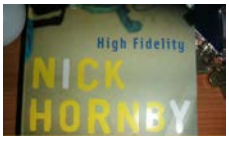
(d) Result of LTP-RF-SVM-CRF (+0.34).

(e) Result of LTP-RF-SVM-CRF-H.

(f) Input.

(g) Result of LTP-RF-SVM-CRF (+0.33).

(h) Result of LTP-RF-SVM-CRF-H.



(i) Input.

Figure 5.8: Comparison of LTP-RF-SVM-CRF and LTP-RF-SVM-CRF-H.

5.1.7 Detection Results with Different Text Confidence Maps

The system is evaluated with and without text-confidence maps and with ground-truth text-confidence maps. The parameter g_c of the detector and the text-confidence-map threshold is swept out to compute a precision-recall curve which is shown in Figure 5.9. Results are shown in Table 5.8. There are several things to observe.

First, the detector achieves competitive results without text-confidence maps. The reason for this is that each step of the detector pipeline is designed to be robust against errors of the previous step.

Next, the biggest improvement in F-score is achieved by using a text-confidence map on any of the feature sets evaluated. The ordering of the feature-sets according to F-score corresponds to the ordering found in the validation procedure in Section 3.1.4.

Furthermore differences between text-confidence are between 0.5-1.5% on F-score. Finally, the performance saturates at 77% (75.8% on the metric proposed by Wolf et al. [99]) when applied to ground-truth responses. Best thresholds of g_c for ground-truth feature-maps are at 0.5, for text-confidence-maps at 0.6-0.7 and without text-confidence maps at 0.8. The reason for this is that in ground-truth regions are less non-text components than in the whole image, or in regions detected by the AdaBoost classifier. Hence, the detection system does not need to have a high textline confidence g_c of 0.6-0.8 to achieve an F-score of 77% (75.8% [99]).

Method	P	R	F	P (Wolf)	R (Wolf)	F (Wolf)
No Response Maps	0.784	0.655	0.713	0.798	0.666	0.726
Ground Truth Response Maps	0.827	0.72	0.770	0.818	0.706	0.758
HOG [68, 69, 70]	0.767	0.702	0.733	0.787	0.689	0.734
ChnFtrs [21]	0.809	0.678	0.735	0.807	0.675	0.735
MACeLBP [2]	0.827	0.668	0.739	0.780	0.690	0.732
Proposed	0.816	0.686	0.745	0.832	0.666	0.74

Table 5.8: Best possible ICDAR 2003 results by sweeping out g_c and the threshold of the response-map on the test set.

5.1.8 Detection Errors

In 40 of the 251 images on the ICDAR 2003 dataset the proposed LTP-RF-SVM-CRF system fails to detect any text in the images on the metric proposed by Wolf et al. [99]. A random sample of 4 such pictures is shown in Figure 5.10. Errors can be divided into non-detected text of the sliding window classifier, segmentation errors in the labeling system, errors due to too small text and errors due to boundary conditions of the evaluation metric.

Since no pixel ground-truth segmentation of the dataset is available, images in which some words or part of words are not retrieved are manually counted. In 78 images text is not retrieved due to segmentation errors, and in 54 images the sliding window classifier fails to detect a part of the textline. In 36 images both, the segmentation and the detection fails to detect text. Hence, errors are correlated, since in images with specular highlights and text with weak contrast both, the sliding window classifier and the segmentation method, fail to detect text. In 27 images

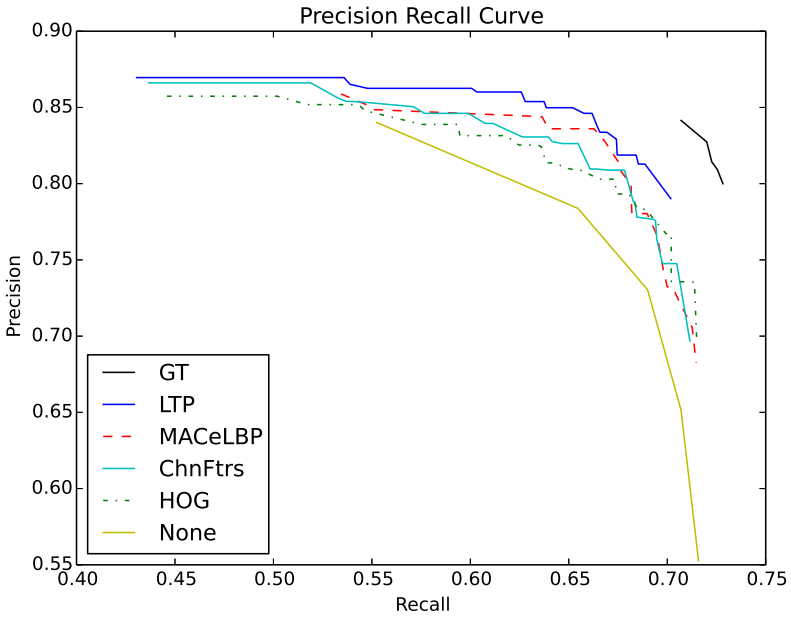
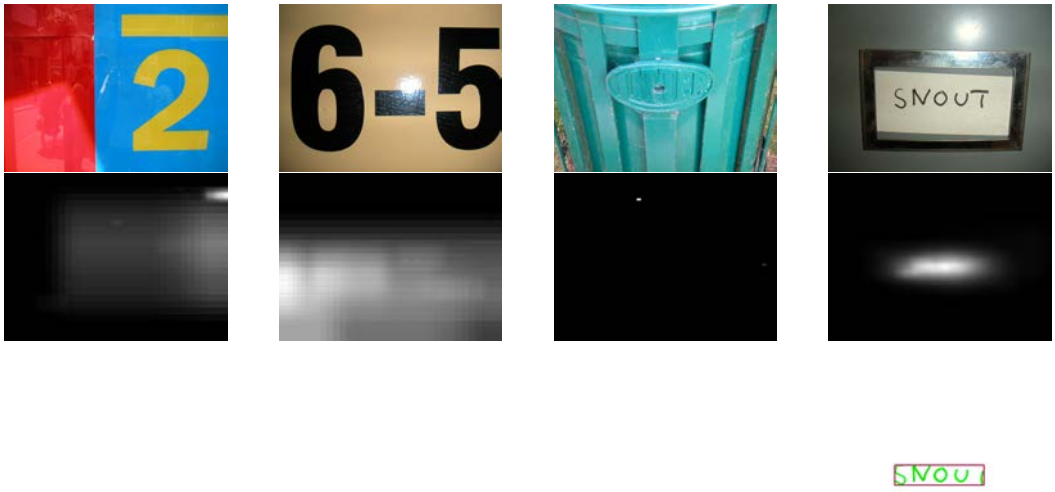


Figure 5.9: Precision recall curves computed on the ICDAR 2003 metric by sweeping out g_c and the threshold of the response-map on the test set.

errors are due to short text consisting of two or less components which are removed by the grouping heuristic. 3 images contain handwritten text which cause segmentation errors. In images where text can be segmented, 3 images cause grouping errors and in 6 images errors due to misclassifications in the labeling stage occur. On segmented components word-splitting fails in 11 images.

The detector system consists of several parts, each of which contributes to a certain fraction to the total error. In order to estimate this fraction, and hence to find out where to optimize the proposed pipeline, ceiling analysis is performed. Since there are no pixel ground-truth masks for the dataset, a sample of 30 images from the ICDAR 2011 dataset drawn and manually masked. Furthermore, for these experiments the minimum number of CCs which are required to form a textline is reduced from 3 to 1, since this heuristic causes on good segmented images too many errors.

On these samples the performance of the proposed LTP-RF-SVM-CRF detector is measured. Then, the sliding window responses are replaced by ground-truth responses and the performance of the system is measured (GT responses). This gives an upper bound on the maximal improvement which can be achieved by improving the sliding window system. Next, the MSER ground-truth binary masks are added to the MSER segmentation and the performance is measured (GT segmentation + MSER). This is an indicator of how good the classifiers are filtering non-text CCs. Then the MSER segmentation is removed, and the system is executed on ground-truth binary masks, fixing the probability-predictions of the classifiers to 1.0 (GT segmentation). Finally, grouping heuristics are relaxed by disabling the linear SVM and setting minimum vertical



(a) Too short text.

(b) Too short text.

(c) Non-detected text by the sliding window classifier.

(d) Errors due to evaluation metric.

Figure 5.10: Errors of the detection system.

overlap of two components to 0.08 and maximum height ratio to 6.0 (GT segmentation relaxed).

Results are shown in Table 5.9.

From these results it can be concluded that by improving the sliding window detector, a maximum improvement of 3.8% (4.6%) is possible. By further including a method which segments the characters perfectly a maximum improvement of 3.8% (6.2%) is possible. The lower score on the ICDAR 2003 metric is due to word-splitting and grouping errors, which are differently penalized in the metric proposed by Wolf et al. [99]. Word splitting and grouping errors occur due to misclassified CCs on the MSER segmentation which are grouped to textlines and cause errors in the following steps in the pipeline. If the classifiers correctly classify all segmented components a maximal improvement of 8.6% (6.5%) can be achieved. Finally, by disabling the linear SVM and other grouping constraints an improvement of 1.7% (1.2%) can be achieved.

The remaining errors are due to grouping and word splitting errors, which are illustrated in Figure 5.11.

The errors in the combined MSER-grouping phase are due to misclassified components, which form single-component textlines as illustrated in Figure 5.12. Hence, by removing the MSER segmentation, a bigger improvement is achieved.

Furthermore, the segmentation step and the classification step cannot be separated easily. With a better segmentation method with which character components do not stick together with background noise or are split into two CCs, classification becomes easier. On the other hand a perfect labeling stage can correctly identify degenerate characters and characters which are split due to partial occlusions into several CCs. Hence, to improve the performance of the detector

both, the segmentation and the classification step, must be improved 15.3% (12.7%). With improvements in this stage, word splitting and grouping becomes less error-prone.

Method	P	R	F	P (Wolf)	R (Wolf)	F (Wolf)
LTP-RF-SVM-CRF	0.810	0.644	0.717	0.811	0.664	0.73
GT responses	0.720	0.793	0.755	0.807	0.747	0.776
GT segmentation + MSER	0.775	0.812	0.793	0.830	0.847	0.838
GT segmentation	0.873	0.899	0.886	0.908	0.898	0.903
GT segmentation relaxed	0.921	0.886	0.903	0.925	0.905	0.915

Table 5.9: Error analysis for the text detector.

(a) R:0.65, P:0.6, F:0.624

(b) R: 0.8, P: 1, F: 0.889

(c) R: 0.778, P: 0.875, F: 0.824

(d) R: 0.72, P: 0.7, F: 0.71

(e) R: 0.8, P: 0.8, F: 0.8

Figure 5.11: Detector errors on ground truth after relaxing heuristics.

The last experiment measures the behaviour of the system on a perfect segmentation and CC labeler with the LTP detector confidence map. This is a measurement of how many components are missing due to non-detected text of the classifier. Results are shown in Table 5.10. From this results it can be concluded that there is room for improvement for the classifier, since by replacing the responses with the ground-truth, an improvement of the F-score of 2.7% (4.9%) is possible.

Method	P	R	F	P (Wolf)	R (Wolf)	F (Wolf)
LTP segmentation	0.866	0.832	0.859	0.888	0.822	0.854
GT segmentation	0.873	0.899	0.886	0.908	0.898	0.903

Table 5.10: Influence of the response map on a perfect segmentation compared to ground-truth responses.

Figure 5.12: Errors due to single-component textlines.

5.2 Word Recognition Results

In this section an overview of the word recognition results is given. The section is divided in Section 5.2.1 where the proposed neural network is evaluated on cropped character patches from the ICDAR 2003 dataset and in Section 5.2.2 where performance is evaluated on the cropped word recognition problem with varying dictionary sizes.

5.2.1 Cropped Character Recognition

The cropped character recognition performance of the model proposed is evaluated on the ICDAR 2003 cropped character dataset. For evaluation the 62-way misclassification accuracy is measured. The proposed method achieves a 62-way classification accuracy of 86.5% and a 36-way accuracy of 90.4%. Quantitive results for 62-way accuracy are shown in Table 5.11. CNNs outperform handcrafted features on this dataset. Furthermore, by training a deep CNN with backpropagation as proposed, a CNN with an unsupervised layer with no supervised "fine-tuning" [96] is outperformed.

Method	Accuracy
Proposed	**0.865**
unsup. features + CNN [96]	0.839
unsup. features + SVM [15]	0.817
TSM [84] [a]	0.779
HOG + Ferns [94]	0.64

[a] 49-way accuracy

Table 5.11: Cropped character recognition on the ICDAR 2003 dataset.

28.5% of all errors are due to confusions between lower and uppercase letters. The reason for that is, that no textline information for character-patches is available. Hence, character scales

and positions vary in the cropped box. Character-patches are cropped from cropped words which are shown in Figure 5.13. The word 'on' has a lowercase 'o' which is as big as the height of the cropped box. The reason for that is that the word 'on' has no letters with ascenders and descenders. The word 'homogenic' contains an 'o' which is in the middle of the word-patch and smaller than the 'o' in 'on', since the word 'homogenic' has letters with descenders. Finally, the word 'Somerton' has no letters with descenders, hence the 'o' is aligned at the bottom of the word-patch. Hence, the cropped character classifier has to learn to recognize letters invariant to scale and position, which makes it impossible to distinguish uppercase letters with the same shape from lowercase letters with 100% accuracy.

Figure 5.13: Cropped words containing letter 'o' on different positions and in different scales.

Since 28.5% of all errors are due to confusions between lower and uppercase letters, a 36-way confusion matrix of the classifier is presented in Figure 5.14 and the top 10 errors in the matrix are listed in Figure 5.15a. The most dominant errors are misclassified '0's, which are recognized as 'o'. This is also the case for 62-way confusions (see Figure 5.15b) and due to same shape of both objects. Since there are more samples labeled as 'o' in the training set than samples labeled as '0' (see Figure 5.16), ground-truth '0'-patches are classified as 'O'-patches.

To compare the effect of different network sizes on accuracy, two CNNs with convolution layers of size 32-64-64 and 16-32-32 are trained. Each network makes 5 guesses for a given sample. The error curves after each guess are shown in Figure 5.17. Bigger networks perform better on the test set, which is the same result Wang et al. [96] shows with CNNs trained with a fixed unsupervised feature layer. 62-way accuracy saturates at 94% for the best network (96.7% on 36-way accuracy). The accuracies of the networks are 86.57%, 85.61% and 84.57%. The differences are 0.96% and 1.04%. Furthermore, 5 random patches which are misclassified by the best network after 5 guesses are shown in Figure 5.18. These samples represent the hardest samples for the network. Misclassifications are due to too narrow text, too blurred text, handwritten text and mislabeled samples.

The effect of different training set sizes is shown in Figure 5.19. Networks of size 32-128-128 are trained on training sets of 60,000 (proposed), 30,000 and 15,000. The hyperparameters found for the network of size 32-128-128 with 60,000 are kept fixed due to computational constraints. The accuracy of 5 guesses are shown in Figure 5.19a and the best accuracy is plotted against the training set size in Figure 5.19b. Networks with more training samples perform better than networks with less training samples. The accuracy of the three networks is 0.8247, 0.8472 and 0.8657 for training set sizes of 15,000, 30,000 and 60,000 samples. The differences are 2.3% and 1.8%, indicating that there is still room for improvement for a larger training set size.

To show the effectiveness of the synthetic training data a 32-128-128 CNN is trained and the 62-way accuracy is measured. The accuracy of the network is 82.65%.

A random sample of misclassified patches in shown in Figure 5.20. The predicted label is shown in the lower left corner of each patch and the ground-truth label is shown in the lower right corner of each patch. Due to missing textline information and label noise, the case of

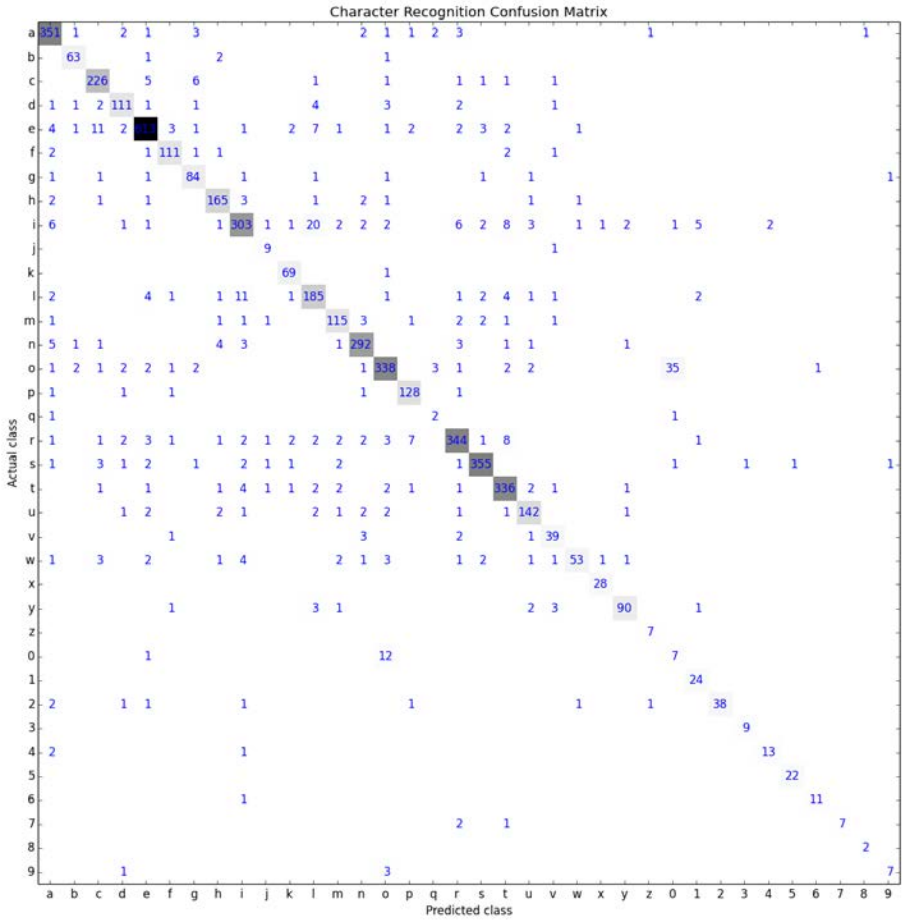
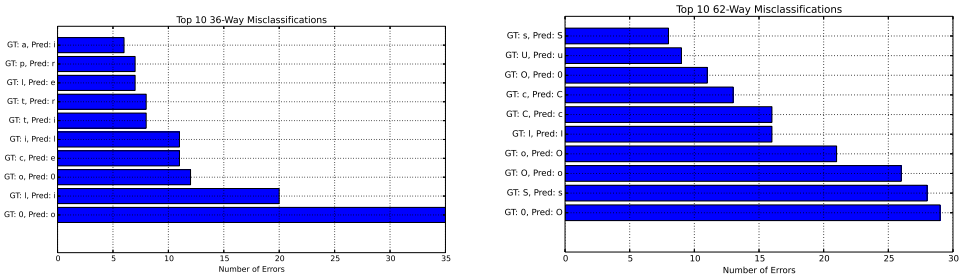


Figure 5.14: 36-way confusion matrix on the ICDAR 2003 cropped character test set.



(a) 36-way errors. (b) 62-way errors.

Figure 5.15: Top 10 errors

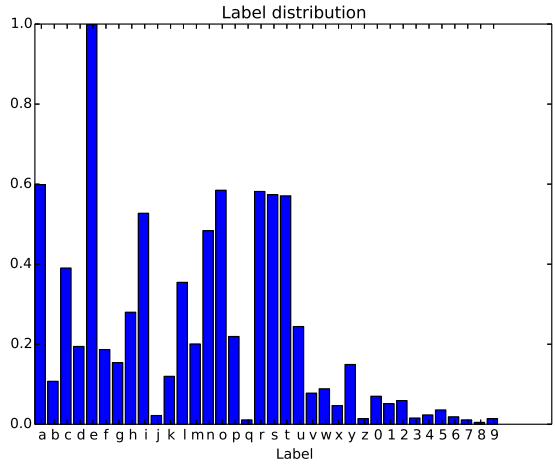


Figure 5.16: Class label distribution

several classes like ’O’ and ’o’, ’S’ or ’s’ can not be determined for each sample. Further errors are due to blur, label noise, partially occluded text, narrow text, misaligned text and text with weak contrast.

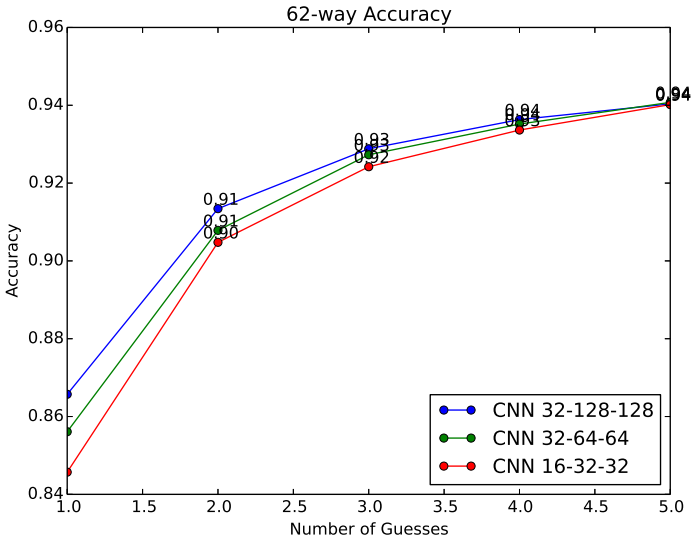


Figure 5.17: Accuracy of CNNs of different sizes after 1, 2, . . . , 5 guesses.

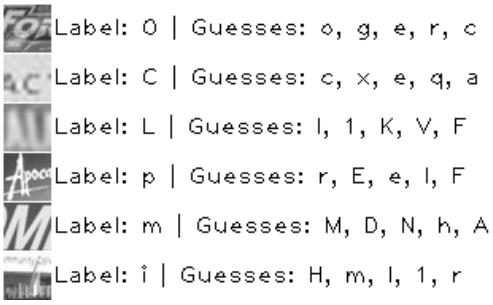
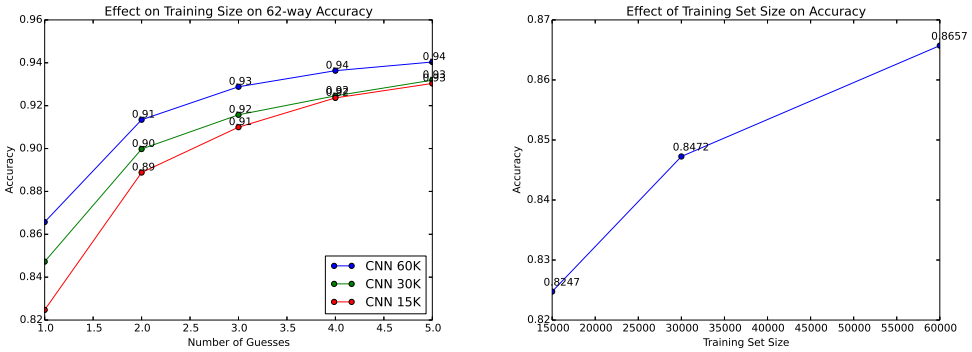


Figure 5.18: Random sample of 5 patches which are misclassified after 5 guesses.



(a) Accuracy after 1, 2, . . . , 5 guesses.

(b) Accuracy vs training set size.

Figure 5.19: Influence of different training set sizes on the accuracy of the network.

(a) Case sensitive false positives. (b) Case insensitive false positives.

Figure 5.20: False positives of the CNN on the ICDAR 2003 cropped character dataset. The predicted label is shown in the lower left corner of each patch, the ground-truth label is shown in the lower right corner of each patch.

5.2.2 Cropped Word Recognition

In this section results for dictionary driven cropped word recognition on the ICDAR 2003 dataset, the SVT-dataset and the ICDAR 2011 dataset are presented. For evaluation the same protocol proposed by Wang et al. [94] is used. Words consisting of fewer than 3 letters and non-alphanumeric letters are excluded from evaluation. Furthermore case-insensitive comparisons are done. The model is evaluated with a dictionary consisting of 50 random distractor words (I-50) and the full ICDAR word list (I-FULL). To achieve comparable results, the same dictionaries used by Wang et al. [94] are used. For the SVT-dataset the dictionaries are already provided by the authors. Since for the ICDAR 2011 dataset no publicly available dictionaries are available, performance is evaluated on the full word list (I11-FULL). The performance of the system is shown in Table 5.12. Due to the better performing CNN the method is about 3% better on ICDAR benchmarks than the method proposed by Wang et al. On the SVT dataset, the difference is higher (12%). The reason for that is that Wang et al. [96] train the unsupervised first layer of the network with synthetic and ICDAR data, and the supervised learned upper layers only with ICDAR data. Hence the method achieves better performance on ICDAR, but worse performance on SVT. Methods with handcrafted features (TSM, HOG + CRF) have to rely on more complex and computationally more expensive graphical models to achieve competitive results.

Method	I-50	I-Full	SVT	I11-FULL
Proposed	**0.929**	**0.871**	**0.82**	**0.871**
unsup. features + CNN [96]	0.90	0.84	0.70	-
Hough Forests [103]	0.857	-	-	-
TSM [84]	0.847	0.793	0.735	0.829
HOG + CRF [58]	0.818	-	0.733	-
PLEX [94]	0.76	0.62	0.57	-

Table 5.12: Cropped word recognition results.

5.3 End-to-End Results

For end-to-end evaluation the predicted word bounding boxes of the detector are used in the recognizer to predict the word. As text-detector the proposed RF-SVM-CRF combination is used. For each detected word the text recognition module predicts a word and a confidence score given a list of candidate words from a dictionary. From these confidence scores a precision-recall curve is swept out as proposed by Wang et al. [96]. Word predictions are thresholded on a linear grid consisting of 40 steps starting at -1 and ending at 50.0 The entry with best F-score is listed in Table 5.13. The system is evaluated with the full ICDAR word list (I-Full), 50 (I-50), 20 (I-20), and 5 (I-5) distractor words, as well as on the ICDAR 2011 dataset with the full word list (I11-Full).

Precision-recall curves are shown in Figure 5.21. If the word-recognition system has a smaller dictionary consisting of less distractor words, it achieves a higher F-score on the test-set.

The AUC in the I-Full, I-50, I-20, and I-5 settings are 0.57, 0.62, 0.63 and 0.63 respectively.

To measure the impact of the network size on the end-to-end performance, the end-to-end performance is measured with the two smaller networks (32-64-64, 16-32-32), trained in Section 5.2.1. The end-to-end F-scores are shown in Table 5.14. Improvements on the cropped character recognition benchmark result in improvements to the end-to-end performance. A CNN trained on synthetic data achieves an F-score 70.2% on I-Full.

Method	I-Full	I-50	I-20	I-5	I11-Full
Proposed	**0.727**	**0.790**	**0.80**	**0.81**	**72.3**
Wang et al. [96]	0.67	0.72	0.74	0.76	-
Wang et al. [94]	0.51	0.68	0.70	0.72	-

Table 5.13: ICDAR 2003 and 2011 end-to-end results (F-score).

Network	F-score (I-Full)
64-128-128	0.727
32-64-64	0.719
16-32-32	0.705

Table 5.14: Influence of the network size on the end-to-end F-score.

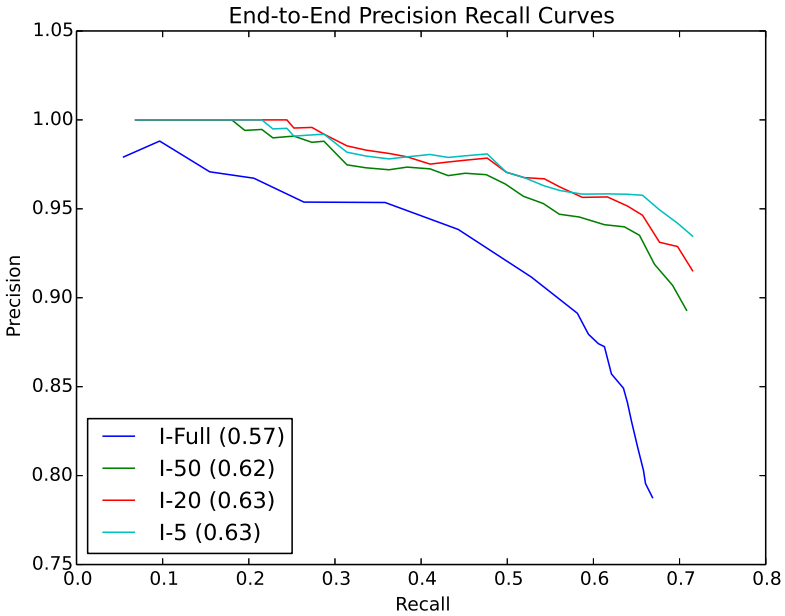
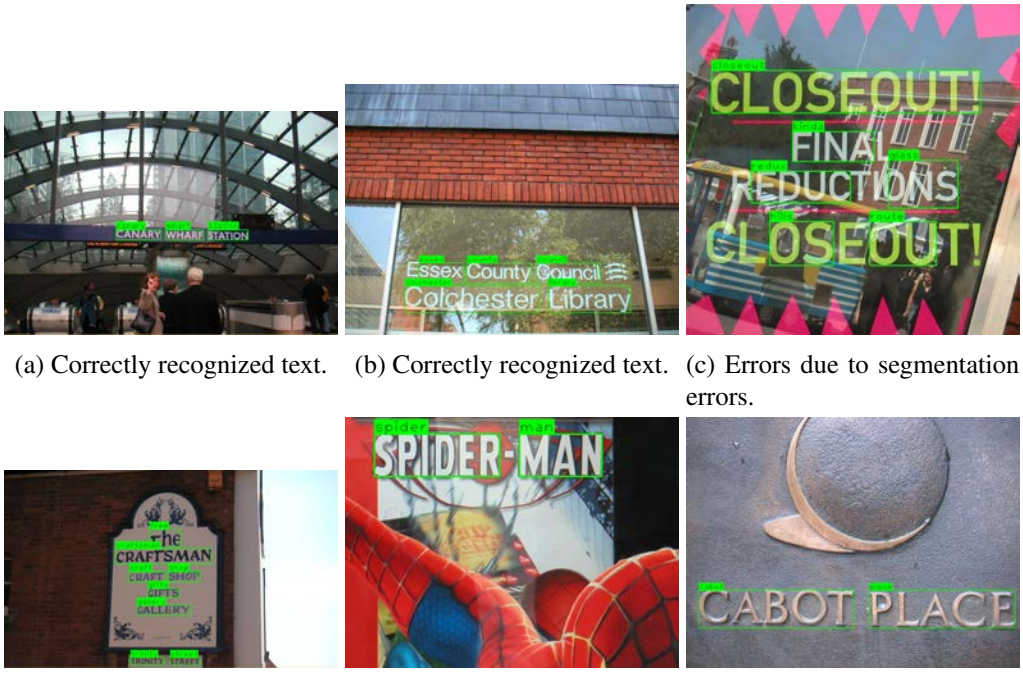


Figure 5.21: End-to-end precision-recall curves.

Qualitative results are shown in Figure 5.22. Errors are either due to detection-errors or due to recognition-errors. Detection errors are due to segmentation errors, which are responsible for errors shown in Figure 5.22c, or misclassification errors of the detector. The text-recognition algorithm compensates for errors in the detection module in Figure 5.22b, since the most plausible word in the dictionary for the letters “ssex” is “essex”.



(a) Correctly recognized text. (b) Correctly recognized text. (c) Errors due to segmentation errors.

(d) Misclassification error at the first box. (e) Correctly recognized text. (f) Correctly recognized text.

Figure 5.22: End-to-end text recognition results

5.4 End-to-End Ceiling Analysis

In this section ceiling analysis is done. The proposed system is a machine learning pipeline consisting of several steps (see Figure 5.23). In the first step a sliding window classifier creates text confidence maps. In the second step CCs are labeled within the detected regions. Finally, in the last step word recognition on retrieved word-boxes is performed. Hence, in order to assess which module contributes the most to the end-performance, ceiling analysis is done.

Each module starting from top to bottom is replaced by the ground-truth data, and the end-to-end performance of the remaining pipeline is measured by executing the whole system on ground-truth data. Results are shown in Table 5.15. Replacing the responses of the sliding window classifier with the ground-truth responses improves the end-to-end performance by 2.1%. This implies that with a perfect detector the end-to-end performance can be maximally improved by 2.1%.

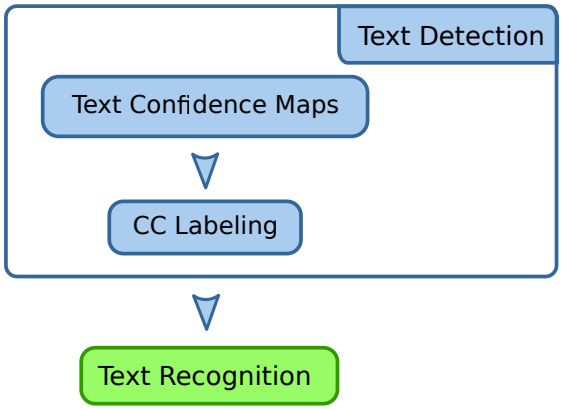


Figure 5.23: System architecture.

Replacing the detected word-level bounding-boxes with ground truth word-level bounding-boxes is the cropped-word recognition task. Hence, the system achieves an F-score of 87.1% which is an improvement of 12.6% on end-to-end performance.

By improving cropped word recognition a maximum improvement of 12.9% can be achieved.

Hence, most improvement can be achieved by improving performance of the word-recognition step and the CC labeling step.

Method	F-score
End-to-end system	0.727
Ground-truth detector responses	0.748
Ground-truth boxes	0.871
Ground-truth	1.0

Table 5.15: Performance of the system when replacing previous steps in the pipeline with ground-truth data.

5.5 Summary

In this section text-detection and text-recognition systems were evaluated. For the text-detection system the contribution to detection-performance on the ICDAR evaluation metrics for several submodules of the system proposed were shown. The use of CRFs in combination with RF and SVM achieved a difference of about 1% (3% [99]) on F-score compared to other combinations. Detecting MSER in the U and V channels of the LUV image improved performance by 1% (3.5% [99]) on F-score. Without a grouping SVM the performance decreased by 1.4% (0.9% [99]). Disabling grouping thresholds had a higher effect on LTP-RF-SVM than LTP-RF-SVM-CRF. The proposed method needed a word splitting method which works with variations in letter spacing. Otherwise the performance decreased by 5.4% (6.4% [99]). The influence of the feature set of the sliding window classifier was 1.5% (0.8% [99]). Without sliding window classifier

the performance decreased by 3.2% (1.4% [99]). With a perfect sliding window classifier the performance improved by 2.5% (3.2% [99]). The text detection system achieved an F-score of 74.2% on ICDAR 2003, 76.7% on ICDAR 2011.

28.5% of all errors of the CNN were due to misclassifications of lower- and uppercase letters. Case-insensitive errors are due to too narrow, too blurred, handwritten text and mislabeled samples. Smaller CNNs achieve less accuracy on cropped word recognition. Furthermore less data results in a reduced accuracy. In cropped word recognition an F-score of 87.1% was achieved.

The system proposed achieved an F-score of 72.7% on the ICDAR 2003 dataset and 72.3% on the ICDAR 2011 dataset on dictionary driven end-to-end scene text recognition. By improving the performance of the CC labeling stage and the word recognition step end-to-end performance was influenced most.

From these results several things can be concluded. First, with an F-score of 72.7% (72.3%) on dictionary driven end-to-end scene text recognition there is still much room for improvement for bridging the gap to human performance. End-to-end error analysis shows that errors are due to the detection system and the cropped word recognition system.

Improving the sliding window detector without improving the labeling stage will not improve the end-to-end performance on ICDAR 2003 by more than 2.1%. Hence, resources should be spent on optimizing the labeling stage and the word recognition step. The word splitting method (6.5%), the integration of color channels (3.5%) and improvements of the labeling classifier (3%) have biggest impact on the ICDAR 2011 metric. Since error analysis shows that there is room for improvement in the segmentation, labeling, grouping and word-splitting stage, better methods are needed in these areas to improve performance.

In the word recognition stage experiments show that learned features outperform handcrafted methods by a large margin. A CNN trained on synthetic data only is able to perform better on cropped character recognition than methods based on handcrafted features. From the conducted experiments it can be concluded that more data and bigger networks improve the performance of these networks. Furthermore, since better language models increase performance for handcrafted classifiers [94, 58, 84], it is reasonable to assume that these models improve the performance for CNNs as well.

CHAPTER 6

Conclusion

In this work a competitive dictionary driven end-to-end scene text recognition system is presented. The system is divided into a text detection and text recognition part. For text detection an AdaBoost sliding window classifier detects text in multiple scales. The result of this step is a text confidence map. Within detected regions MSER are detected, labeled and grouped to textlines. Textlines are split into word-level bounding boxes. The text recognition system uses a one-dimensional sliding window in detected word bounding boxes. Responses are used in a Viterbi-style algorithm to predict the final word.

Compared to other methods, the proposed system combines a sliding window classifier on an adapted feature set with a MSER labeling stage to improve detection performance. As feature-set a combination of MACeLBP and LTP features is used, which achieve better performance than other proposed feature-sets. For text recognition a CNN regularized by Dropout [40] is trained, to learn local features which outperform methods based on hand-crafted features and CNNs with fixed unsupervised learned layers [96].

The system is evaluated on the ICDAR datasets [86, 80], outperforming state-of-the-art methods in text detection, recognition and end-to-end dictionary driven scene text detection.

Hence, the research question can be answered: it is possible to achieve competitive results with a combination of local features, region based approches and learned local features.

For text detection the influence of different feature-sets is demonstrated. Furthermore, the influence of combining the UV channels of the LUV image with the grayscale channel, the influence of the word splitting method, the word grouping method and the influence the classifier, are assessed in experiments.

For text recognition an error analysis of misclassified patches was conducted, which shows that 28.5% of all errors are due to case-misclassifications. Other errors are due to narrow text, blurred text, handwritten text or mislabeled ground truth.

Finally, ceiling analysis was performed, showing the influence of the end-to-end performance by replacing parts of the system with ground-truth data.

6.1 Disadvantages of the System

The proposed system trades off runtime speed for accuracy. In the text detection step a separate sliding window step is proposed. Since image sizes on ICDAR datasets range from 800×600 to $3,888 \times 2,592$, up to 650,000 classifications for the first scale are done. Furthermore, the binary feature-set used in the CC labeling stage is more expensive compared to the feature-set proposed by Neumann et al. [63]. The binary features can not be computed incrementally in constant time as Neumann et al. [63] proposes. Hence, performance is linear in the number of MSER detected and thus depends on the complexity of the scene. The use of more expensive feature sets makes it computationally more expensive to include additional segmentations like MSER or ERs segmented from the gradient image [63] or segmentations on the stroke width image [24]. Furthermore, since the ICDAR training data only contains horizontal text and grouping models proposed in this method assume that text is horizontal, only horizontal text can be detected.

The text recognition system uses deep CNNs. In the system proposed for each prediction 32 5×5 convolutions are applied in the first layer, 128 5×5 3D convolutions in the second and third layer. On a Intel Core i5 CPU classification of about 5000 patches takes 116 seconds. CNNs, however, take advantage of GPU hardware. On a NVIDIA 770 GTX classification takes 1.3 seconds including host to GPU transfers.

Next, the text detection system relies on segmentation which does not work on images where text is partially occluded, on images where semi-transparent text is shown, on images where text components are stuck together or on images with varying illumination.

Furthermore, the text recognition system does not scale to dictionary sizes consisting of thousands of words. The method, proposed has to evaluate a Viterbi algorithm for each word of the dictionary.

6.2 Advantages of the System

The main advantages of the system proposed are robustness and accuracy compared to other methods. For text detection advantages of texture based approaches and region based approaches are combined to a robust text detection system.

For text recognition a discriminative model is used which achieves a higher accuracy than other methods on the task of cropped character recognition. Hence, to read text, only a graphical chain-model is needed to outperform other methods which use more complex models consisting of arbitrary graphs ([58, 84]).

6.3 Future Work

Ceiling analysis in Section 5.4 shows that improving or replacing the component labeling step and the word recognition step has the biggest potential in improving end-to-end accuracy.

For the text detection module a better performing labeling system is needed. As shown in Section 5.1.8 the by improving segmentation and labeling step an improvement of 15.3% (12.7%) is possible. Hence, in images where segmentation is not possible, a segmentation-free method is needed. Furthermore, based on the conducted experiments, combination of multiple

segmentation methods will improve performance. To improve performance in the labeling stage, better feature sets are needed. For better feature sets more training data is needed. With a better performing labeling stage the grouping and word-splitting tasks will become easier.

For the text recognition module experiments show that a larger training set size and a larger network are beneficial for performance. Furthermore, regularization techniques such as Stochastic Pooling [104] and DropConnect [93] and better activation functions such as Maxout [36] will improve the performance of the CNN. With fast dropout training [95], larger networks can be trained in the same amount of time, improving performance of the method. Next, as Jarret et al. [41] shows, unsupervised pre-training with supervised fine-tuning helps improving performance. Hence regularized Autoencoders [91, 75, 74, 37], RBMs [39], Sparse Filtering [64] Sparse Coding [53] or semi supervised methods as proposed by Sermanet et al. [79] which are fine-tuned with backpropagation help improving performance.

Next, color information is ignored by the system. Color can be integrated as proposed by Sermanet et al. [79] by convolving in the first layer of the network over the three YUV channels instead of over the grayscale image.

Furthermore by comparing the work of Mishra et al. [58] to work of Wang et al. [94], it can be seen that better language models improve word recognition performance.

Bibliography

[1] T. Ahonen, A. Hadid, and M. Pietikainen. Face Description with Local Binary Patterns: Application to Face Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(12):2037–2041, 2006.

[2] M. Anthimopoulos, B. Gatos, and I. Pratikakis. Detection of Artificial and Scene Text in Images and Video Frames. *Pattern Analysis and Applications*, 16(3):431–446, 2013.

[3] Y. Bengio. Practical Recommendations for Gradient-Based Training of Deep Architectures. In *Neural Networks: Tricks of the Trade*, pages 437–478. Springer, 2012.

[4] J. Bergstra and Y. Bengio. Random Search for Hyper-Parameter Optimization. *The Journal of Machine Learning Research*, 13:281–305, 2012.

[5] C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[6] A. Bosch, A. Zisserman, and X. Muoz. Image Classification using Random Forests and Ferns. In *IEEE International Conference on Computer Vision.*, pages 1–8, 2007.

[7] L. Breiman. Random Forests. *Machine Learning*, 45(1):5–32, 2001.

[8] C. JC Burges. A Tutorial on Support Vector Machines for Pattern Recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167, 1998.

[9] J. Canny. A Computational Approach to Edge Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (6):679–698, 1986.

[10] H. Chen, S. S. Tsai, G. Schroth, D. M. Chen, R. Grzeszczuk, and B. Girod. Robust Text Detection in Natural Images with Edge-Enhanced Maximally Stable Extremal Regions. In *International Conference on Image Processing*, pages 2609–2612. IEEE, 2011.

[11] X. Chen and A. L. Yuille. Detecting and Reading Text in Natural Scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.*, volume 2, pages 366–373, 2004.

[12] D. C. Cireşan, A. Giusti, L. M. Gambardella, and J. Schmidhuber. Mitosis Detection in Breast Cancer Histology Images with Deep Neural Networks. In *MICCAI*, 2013 in press.

[13] D. C. Cireşan, U. Meier, J. Masci, L. M. Gambardella, and J. Schmidhuber. Flexible, High Performance Convolutional Neural Networks for Image Classification. In *Proceedings of International Conference on Artificial Intelligence*, pages 1237–1242. AAAI Press, 2011.

[14] D. C. Cireşan, U. Meier, J. Masci, and J. Schmidhuber. Multi-Column Deep Neural Network for Traffic Sign Classification. *Neural Networks*, 32:333–338, 2012.

[15] A. Coates, B. Carpenter, C. Case, S. Satheesh, B. Suresh, T. Wang, D. J. Wu, and A. Y. Ng. Text Detection and Character Recognition in Scene Images with Unsupervised Feature Learning. In *International Conference on Document Analysis and Recognition*, pages 440–445, 2011.

[16] N. Dalal. *Finding People in Images and Videos*. PhD thesis, Institut National Polytechnique de Grenoble, July 2006.

[17] N. Dalal and B. Triggs. Histograms of Oriented Gradients for Human Detection. In *IEEE Conference on Computer Vision and Pattern Recognition.*, volume 1, pages 886–893, 2005.

[18] J. G. Daugman. High Confidence Visual Recognition of Persons by a Test of Statistical Independence. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(11):1148–1161, 1993.

[19] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *IEEE Conference on Computer Vision and Pattern Recognition.*, pages 248–255, 2009.

[20] P. Dollár, S. Belongie, and P. Perona. The Fastest Pedestrian Detector in the West. In *Proceedings of British Machine Vision Conference*, volume 2, pages 1– 11, 2010.

[21] P. Dollár, Z. Tu, P. Perona, and S. Belongie. Integral Channel Features. In *Proceedings of the British Machine Vision Conference*, pages 1–11. BMVA Press, 2009.

[22] P. Dollár, C. Wojek, B. Schiele, and P. Perona. Pedestrian Detection: An Evaluation of the State of the Art. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(4):743–761, 2012.

[23] R.O. Duda, P.E. Hart, and D.G. Stork. *Pattern Classification*. Wiley, 2001.

[24] B. Epshtein, E. Ofek, and Y. Wexler. Detecting Text in Natural Scenes with Stroke Width Transform. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2963–2970. IEEE, 2010.

[25] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

[26] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The Pascal Visual Object Classes (VOC) Challenge. *International Journal of Computer Vision*, 88(2):303–338, June 2010.

[27] C. Farabet, C. Couprie, L. Najman, and Y. LeCun. Learning Hierarchical Features for Scene Labeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1915–1929, 2013.

[28] L. Fei-Fei, R. Fergus, and P. Perona. Learning Generative Visual Models from Few Training Examples: An Incremental Bayesian Approach Tested on 101 Object Categories. *Computer Vision and Image Understanding*, 106(1):59–70, 2007.

[29] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object Detection with Discriminatively Trained Part-Based Models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, 2010.

[30] R. Fergus, L. Fei-Fei, P. Perona, and A. Zisserman. Learning Object Categories from Google's Image Search. In *IEEE International Conference on Computer Vision*, volume 2, pages 1816–1823, 2005.

[31] Y. Freund, R. Schapire, and N. Abe. A Short Introduction to Boosting. *Journal-Japanese Society For Artificial Intelligence*, 14(771-780):1612, 1999.

[32] J. Friedman, T. Hastie, and R. Tibshirani. Additive Logistic Regression: A Statistical View of Boosting (with Discussion and a Rejoinder by the Authors). *The annals of statistics*, 28(2):337–407, 2000.

[33] X. Glorot, A. Bordes, and Y. Bengio. Deep Sparse Rectifier Networks. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, volume 15, pages 315–323, 2011.

[34] A. Gonzalez, L.M. Bergasa, J.J. Yebes, and S. Bronte. Text Location in Complex Images. In *International Conference on Pattern Recognition*, pages 617–620, 2012.

[35] I. J. Goodfellow, D. Warde-Farley, P. Lamblin, V. Dumoulin, M. Mirza, R. Pascanu, J. Bergstra, F. Bastien, and Y. Bengio. Pylearn2: a Machine Learning Research Library. *Computing Research Repository*, abs/1308.4214, August 2013.

[36] I. J. Goodfellow, D. Warde-Farley, M. Mirza, A. Courville, and Y. Bengio. Maxout Networks. In *Proceedings of the International Conference on Machine Learning*, volume 28, pages 1319–1327. JMLR Workshop and Conference Proceedings, May 2013.

[37] R. Goroshin and Y. LeCun. Saturating Auto-Encoders. In *International Conference on Learning Representations*, April 2013 in press.

[38] R. M. Haralick. Statistical and Structural Approaches to Texture. *Proceedings of the IEEE*, 67(5):786–804, 1979.

[39] G. E. Hinton, S. Osindero, and Y. Teh. A Fast Learning Algorithm for Deep Belief Nets. *Neural Computation*, 18(7):1527–1554, July 2006.

[40] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R.R. Salakhutdinov. Improving Neural Networks by Preventing Co-Adaptation of Feature Detectors. *Computing Research Repository*, abs/1207.0580, July 2012.

[41] K. Jarrett, K. Kavukcuoglu, M. Ranzato, and Y. LeCun. What is the best Multi-Stage Architecture for Object Recognition? In *IEEE International Conference on Computer Vision*, pages 2146–2153, 2009.

[42] J. P. Jones and L. A. Palmer. An Evaluation of the Two-Dimensional Gabor Filter Model of Simple Receptive Fields in Cat Striate Cortex. *Journal of Neurophysiology*, 58(6):1233–1258, 1987.

[43] D. E. King. Dlib-ml: A Machine Learning Toolkit. *Journal of Machine Learning Research*, 10:1755–1758, 2009.

[44] D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.

[45] A. Krizhevsky. Learning Multiple Layers of Features from Tiny Images. Technical report, University of Toronto, 2009. `http://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf`.

[46] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In *Conference on Neural Information Processing Systems*, pages 1106–1114, 2012.

[47] S. Lazebnik, C. Schmid, and J. Ponce. Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pages 2169–2178, 2006.

[48] Q. V. Le, M.'A. Ranzato, R. Monga, M. Devin, G. Corrado, K. Chen, J. Dean, and A. Y. Ng. Building High-Level Features using Large Scale Unsupervised Learning. In *Proceedings of the International Conference of Machine Learning*, pages 81–88, 2012.

[49] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation Applied to Handwritten Zip Code Recognition. *Neural Computation*, 1(4):541–551, Winter 1989.

[50] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-Based Learning Applied to Document Recognition. *Proceedings of the IEEE*, 86(11):2278–2324, November 1998.

[51] Y. LeCun and C. Cortes. MNIST Handwritten Digit Database. http://yann.lecun.com/exdb/mnist/, 2010.

[52] Y. LeCun, F. J. Huang, and L. Bottou. Learning Methods for Generic Object Recognition with Invariance to Pose and Lighting. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 97–104. IEEE, 2004.

[53] H. Lee, A. Battle, R. Raina, and A. Ng. Efficient Sparse Coding Algorithms. In *Advances in Neural Information Processing Systems*, pages 801–808, 2006.

[54] JJ. Lee, P.-H. Lee, S.-W. Lee, A. L. Yuille, and C. Koch. AdaBoost for Text Detection in Natural Scene. In *International Conference on Document Analysis and Recognition*, pages 429–434, 2011.

[55] D. G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[56] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust Wide-Baseline Stereo from Maximally Stable Extremal Regions. *Image and Vision Computing*, 22(10):761–767, 2004.

[57] R. Minetto, N. Thome, M. Cord, J. Stolfi, F. Precioso, J. Guyomard, and N. J. Leite. Text Detection and Recognition in Urban Scenes. In *International Conference on Computer Vision Workshops*, pages 227–234. IEEE, 2011.

[58] A. Mishra, K. Alahari, and C. V. Jawahar. Top-Down and Bottom-Up Cues for Scene Text Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2687–2694, 2012.

[59] V. Nair and G. E. Hinton. Rectified Linear Units Improve Restricted Boltzmann Machines. In *Proceedings of the International Conference on Machine Learning*, pages 807–814, 2010.

[60] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng. Reading Digits in Natural Images with Unsupervised Feature Learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

[61] L. Neumann and J. Matas. A Method for Text Localization and Recognition in Real-World Images. In *Proceedings of the Asian Conference on Computer Vision*, pages 770–783. Springer, 2011.

[62] L. Neumann and J. Matas. Text Localization in Real-World Images using Efficiently Pruned Exhaustive Search. In *International Conference on Document Analysis and Recognition*, pages 687–691. IEEE, 2011.

[63] L. Neumann and J. Matas. Real-Time Scene Text Localization and Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3538–3545. IEEE Computer Society, 2012.

[64] J. Ngiam, Z. Chen, S. A. Bhaskar, P. W. Koh, and A. Ng. Sparse filtering. In *Advances in Neural Information Processing Systems*, pages 1125–1133, 2011.

[65] W. Niblack. *An Introduction to Digital Image Processing*. Strandberg Publishing Company, Birkeroed, Denmark, Denmark, 1985.

[66] T. Ojala, M. Pietikäinen, and D. Harwood. A Comparative Study of Texture Measures with Classification Based on Featured Distributions. *Pattern Recognition*, 29(1):51–59, 1996.

[67] N. Otsu. A Threshold Selection Method from Gray-Level Histograms. *Automatica*, 11(285-296):23–27, 1975.

[68] Y.-F. Pan, X. Hou, and C.-L. Liu. A Robust System to Detect and Localize Texts in Natural Scene Images. In *International Workshop on Document Analysis Systems*, pages 35–42. IEEE Computer Society, 2008.

[69] Y.-F. Pan, X. Hou, and C.-L. Liu. Text Localization in Natural Scene Images Based on Conditional Random Field. In *International Conference on Document Analysis and Recognition*, pages 6–10. IEEE Computer Society, 2009.

[70] Y.-F. Pan, X. Hou, and C.-L. Liu. A Hybrid Approach to Detect and Localize Texts in Natural Scene Images. *IEEE Transactions on Image Processing*, 20(3):800–813, 2011.

[71] F. Perronnin and C. Dance. Fisher Kernels on Visual Vocabularies for Image Categorization. In *IEEE Conference on Computer Vision and Pattern Recognition.*, pages 1–8, 2007.

[72] I. Pratikakis, B. Gatos, and K. Ntirogiannis. ICDAR 2011 Document Image Binarization Contest (DIBCO 2011). In *International Conference on Document Analysis and Recognition*, pages 1506–1510, 2011.

[73] S. V. Rice, F. R. Jenkins, and T. A. Nartker. *The Fifth Annual Test of OCR Accuracy*. Information Science Research Institute, 1996.

[74] S. Rifai, G. Mesnil, P. Vincent, X. Muller, Y. Bengio, Y. Dauphin, and X. Glorot. Higher Order Contractive Auto-Encoder. In *Machine Learning and Knowledge Discovery in Databases*, volume 6912 of *Lecture Notes in Computer Science*, pages 645–660. Springer Berlin Heidelberg, 2011.

[75] S. Rifai, P. Vincent, X. Muller, X. Glorot, and Y. Bengio. Contractive Auto-Encoders: Explicit Invariance during Feature Extraction. In *Proceedings of the International Conference on Machine Learning*, pages 833–840, 2011.

[76] D. E. Rumelhart, G. E Hinton, and R. J Williams. Learning Representations by Back-Propagating Errors. *NATURE*, 323(6088):533–536, 1986.

[77] J. Sauvola and M. Pietikäinen. Adaptive Document Image Binarization. *Pattern Recognition*, 33(2):225–236, 2000.

[78] P. Sermanet, S. Chintala, and Y. LeCun. Convolutional Neural Networks Applied to House Numbers Digit Classification. In *International Conference on Pattern Recognition*, pages 3288–3291. IEEE, 2012.

[79] P. Sermanet, K. Kavukcuoglu, S. Chintala, and Y. LeCun. Pedestrian Detection with Unsupervised Multi-Stage Feature Learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, June 2013 in press.

[80] A. Shahab, F. Shafait, and A. Dengel. ICDAR 2011 Robust Reading Competition Challenge 2: Reading Text in Scene Images. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 1491–1496. IEEE Computer Society, 2011.

[81] F. Shahbaz Khan, R.M. Anwer, J. van de Weijer, A.D. Bagdanov, M. Vanrell, and A.M. Lopez. Color Attributes for Object Detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3306–3313, 2012.

[82] Y. Shao, C. Wang, B. Xiao, Y. Zhang, L. Zhang, and L. Ma. Text Detection in Natural Images Based on Character Classification. In *Advances in Multimedia Information Processing*, volume 6298 of *Lecture Notes in Computer Science*, pages 736–746. Springer Berlin Heidelberg, 2011.

[83] L. Shen and L. Bai. A Review on Gabor Wavelets for Face Recognition. *Pattern Analysis and Applications*, 9(2-3):273–292, 2006.

[84] C. Shi, C. Wang, B. Xiao, Y. Zhang, S. Gao, and Z. Zhang. Scene Text Recognition using Part-based Tree-structured Character Detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2013 in press.

[85] P. Simard, D. Steinkraus, and J. C Platt. Best Practices for Convolutional Neural Networks Applied to Visual Document Analysis. In *ICDAR*, volume 3, pages 958–962, 2003.

[86] L. P. Sosa, S. M. Lucas, A. Panaretos, L. Sosa, A. Tang, S. Wong, and R. Young. ICDAR 2003 Robust Reading Competitions. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 682–687. IEEE Press, 2003.

[87] X. Tan and B. Triggs. Enhanced Local Texture Feature Sets for Face Recognition Under Difficult Lighting Conditions. *IEEE Transactions on Image Processing*, 19(6):1635–1650, 2010.

[88] Y. Tang. Deep Learning using Support Vector Machines. *Computing Research Repository*, abs/1306.0239, 2013.

[89] K. E. A. Van de Sande, J. R. R. Uijlings, T. Gevers, and A. W. M. Smeulders. Segmentation as Selective Search for Object Recognition. In *IEEE International Conference on Computer Vision*, pages 1879–1886, 2011.

[90] M. Varma and A. Zisserman. A Statistical Approach to Texture Classification from Single Images. *International Journal of Computer Vision*, 62(1-2):61–81, April 2005.

[91] P. Vincent, H. Larochelle, Y. Bengio, and P. Manzagol. Extracting and Composing Robust Features with Denoising Autoencoders. In *Proceedings of the International Conference on Machine learning*, pages 1096–1103. ACM, 2008.

[92] P. Viola and M. Jones. Rapid Object Detection using a Boosted Cascade of Simple Features. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages 511–518. IEEE, 2001.

[93] L. Wan, M. Zeiler, S. Zhang, Y. LeCun, and R. Fergus. Regularization of Neural Networks using DropConnect. In *Proceedings of the International Conference on Machine Learning*, volume 28, pages 1058–1066. JMLR Workshop and Conference Proceedings, May 2013.

[94] K. Wang, B. Babenko, and S. Belongie. End-to-End Scene Text Recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1457–1464, Barcelona, Spain, 2011.

[95] S. Wang and C. Manning. Fast Dropout Training. In *Proceedings of the International Conference on Machine Learning*, volume 28, pages 118–126. JMLR Workshop and Conference Proceedings, May 2013.

[96] T. Wang, D.J. Wu, A. Coates, and A.Y. Ng. End-to-End Text Recognition with Convolutional Neural Networks. In *International Conference on Pattern Recognition*, pages 3304–3308, 2012.

[97] X. Wang, T.X. Han, and S. Yan. An HOG-LBP Human Detector with Partial Occlusion Handling. In *IEEE International Conference on Computer Vision*, pages 32–39, 2009.

[98] T. Watanabe, S. Ito, and K. Yokoi. Co-occurrence Histograms of Oriented Gradients for Pedestrian Detection. In *Advances in Image and Video Technology*, volume 5414 of *Lecture Notes in Computer Science*, pages 37–47. Springer Berlin Heidelberg, 2009.

[99] C. Wolf and J.-M. Jolion. Object Count/Area Graphs for the Evaluation of Object Detection and Segmentation Algorithms. *International Journal on Document Analysis and Recognition*, 8(4):280–296, 2006.

[100] J. Yang, K. Yu, Y. Gong, and T. Huang. Linear Spatial Pyramid Matching using Sparse Coding for Image Classification. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1794–1801, 2009.

[101] C. Yao, X. Bai, W. Liu, Y. Ma, and Z. Tu. Detecting Texts of Arbitrary Orientations in Natural Images. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1083–1090, 2012.

[102] C. Yi and Y. Tian. Text String Detection from Natural Scenes by Structure-Based Partition and Grouping. *IEEE Transactions on Image Processing*, 20(9):2594–2605, 2011.

[103] G. Yildirim, R. Achanta, and S. Süsstrunk. Text Recognition in Natural Images Using Multiclass Hough Forests. In *Proceedings of the International Conference on Computer Vision Theory and Applications*, volume 1, pages 737–741, 2013.

[104] M. D. Zeiler and R. Fergus. Stochastic Pooling for Regularization of Deep Convolutional Neural Networks. In *International Conference on Learning Representations*, April 2013 in press.

List of Acronyms

ANN Artificial Neural Network

AUC Area Under Curve

CC Connected Component

CNN Convolutional Neural Network

CRF Conditional Random Field

CoHOG Co-occurence HOG

ER Extremal Region

GPU Graphics Processing Unit

HOG Histogram of Oriented Gradients

HF Hough Forest

ICDAR International Conference of Document Analysis and Recognition

LBP Local Binary Pattern

LTP Local Ternary Pattern

MSER Maximally Stable Extremal Region

MACeLBP Multilevel Color edge Local Binary Pattern

msLBP multiscale LBP

MLP Multilayer Perceptron

OCR Object Character Recognition

RBM Restricted Boltzmann Machine

RBF Radial Basis Function

PDF Probability Density Function

ReLU Rectified Linear Unit

RF Random Forest

SIFT Scale Invariant Feature Transform

SVM Support Vector Machine

SGD Stochastic Gradient Descent

SVT Street View Text

SWT Stroke Width Transform

VOC Visual Object Class