technische universität
dortmund

BAG OF FEATURES FOR TEXT DETECTION IN NATURAL SCENE IMAGES

Master Thesis

Rashik Thalappully
June 8, 2015

Supervisors:
Prof. Dr.-Ing. Gernot A. Fink
Leonard Rothacker, Dipl.-Inf.

Fakultät für Informatik
Technische Universität Dortmund
http://www.cs.uni-dortmund.de

CONTENTS

1

INTRODUCTION

Visual text is one of the most important methods of communication used by human beings and is widely used in our daily life. Hence, interpreting these textual information is of great significance. Human beings inherently get the ability to detect and recognize the textual content in their surroundings whereas it is a challenging problem for computer systems. The field of text detection focuses on detecting text embedded in images and videos with the help of computer systems. Researchers have made significant progress in detecting the text from images of machine printed documents, on the other hand detecting the text in natural scenes is till a young topic for research. Text detection in natural scenes is difficult because of the variation in fonts, backgrounds, lighting conditions and textures occurring in these images. Also, the location of text in natural scenes is highly randomized. Detecting the embedded text information in natural scene images is of great significance for image understanding, content-based image retrieval and navigation. On account of these challenges, text detection in unconstrained images is considered as far from solved.

Over the past decade, researchers have invested much time and effort in solving these challenges. To explain the variabilities present in natural scenes, it is extremely difficult to depend on rule-based approaches for developing algorithms to solve text detection in unconstrained images. Thus, machine learning is one of the options available to perform this task. Machine learning focuses on building or constructing algorithms that learn from data. Such algorithms generate a model by carefully examining the input examples provided. Machine learning uses these inputs as training data to initiate the learning process. Ideally, the learning process will benefit if there is high variability in the provided input examples. Generally, if the input samples are directly represented, that will lead to a high dimensionality. For e.g., if the images are represented using an array of intensity values, the dimension of the representation will be high. In order to solve this problem, the samples are transformed into features. A feature contains information about an input example. Using these features, the learning algorithm is trained to generate a model for performing the particular task. Hence, the primary objective of this thesis is to solve the problem of text detection in natural scenes using machine learning techniques.

In natural scenes, text can occur in any part of the image. To detect these locations using machine learning methods, the general methodology followed is to evaluate the

presence of text in subpatches extracted from images. The extraction of subpatches is performed by scanning a window of fixed dimension over the image at regular intervals. This method is referred to as sliding window approach. These extracted patches are represented using features. Hence, in any text detection system, features play a pivotal role in detecting text regions. One of the important design choices in generating the features is the selection of feature representation method. The most common method is to use a-priori known representation method. These methods are referred as hand-crafted feature representation methods. In these methods, the transformation function from the input data to features is already known. One of the obvious drawbacks of this approach is that most of these features are task specific. Hence, the researchers have proposed the learned feature representation method. In this, unsupervised feature learning methods are used to determine a transformation function from the input data to features. In this thesis, these methods are used to solve the problem of text detection. One of the popular learned feature representation method is Bag-of-Features (BoF). This method is widely used for text detection. One of the objective of this thesis is to compare the performance of learned against hand-crafted features for text detection.

The generated features using hand-crafted and learned feature representation are applied to the machine learning algorithm to create a model to differentiate text and non-text samples. The generated model is evaluated on unseen images to predict the text and non-text regions. In natural scenes text regions occurs in different sizes. In order to detect the text regions of variable sizes, rescaling of the images is performed. The evaluation is carried out in a sliding window fashion on these rescaled images. The scaling factors for rescaling are predetermined. The evaluation process is highly time-consuming, as the generated model is evaluated at all predetermined scales. Hence, a novel idea to assess the model on rescaled images where the scaling factors are determined using Maximally Stable Extremal Regions (MSER) is proposed in this thesis. An MSER is a region in an image with homogeneous intensity values surrounded by distinctive intensity values. The detector that detects the MSER in an image is widely used for detecting text regions. Since, the proposed method determines the scales based on the character sizes, it is likely to reduce the number of scales on each image. This might result in improved performance as well as reduced evaluation time. Hence, another objective of this thesis is to conduct a detailed analysis of the performance of the newly proposed method against the standard sliding window approach using predetermined scales.

This thesis focuses on the following: Chapter 2 describes the fundamentals of text detection system. Chapter 3 describes the generation of local image features. In Chapter 4 the bag-of-feature representation technique is discussed in detail. Chapter 5 reviews

various text detection methods using hand-crafted and learned feature representation techniques. The newly proposed method of scale detection using maximally stable extremal regions is introduced in Chapter 6. The text detection system using learned as well as hand-crafted features along with newly proposed method are evaluated in Chapter 7. Chapter 8 gives a final conclusion.

2 TEXT DETECTION ESSENTIALS

The textual content embedded in an image is a rich source of information about the environment. Human beings inherently have the ability to recognize the textual information around us and most of us do not realize the astounding complexity of the process of text detection and recognition. Enabling computer systems to acquire the ability to detect and understand the text is a challenging problem as computer systems directly cannot discriminate between text regions and non-text regions in an image.

The fundamental problem of text detection in images is discussed in Section 2.1. Section 2.2 explains the challenges in detecting text in natural scenes. Section 2.3 describes a typical text detection system. Section 2.4 focuses on different learning methods employed to train a typical text detection system.

2.1 TEXT DETECTION IN IMAGES

The field of text detection and text recognition focuses on detecting and recognizing text in images. In order to perceive the textual content from an image, computer systems have to solve two subproblems, one being text detection and the other text recognition. Text detection enables the computer to identify the location of text regions in an image whereas text recognition decodes the detected text. In Figure 2.1.1 text detection algorithm identifies the region where 'Peacock' is written and text recognition algorithm recognizes the detected text region as a text string 'Peacock'.

A lot of varieties of text are found in images. The two primary classes of text in images are digital text and natural scene text. Digital text refers to machine-printed text, for instance, those found in digital born documents. Natural scene text refers to the text captured in natural environments which include text on signs or boards, packages and clothing in natural scenes, including handwritten materials [KSU+13]. Figure 2.1.2 shows a comparison between a scene text and a digital text.

Optical Character Recognition (OCR) is one of the most successful technologies in the field of computer vision, artificial intelligence and pattern recognition. OCR converts printed or typewritten text into machine-coded text. Images of machine printed documents usually have typical fonts, structured text and they are captured

Text detection - Where is the text?

Text recognition - What does text say?

Figure 2.1.1: Text detection and recognition in images

under controlled settings. Hence, state-of-art OCR methods have achieved recognition rates higher than 99% [WLMH09].

2.2 TEXT DETECTION IN NATURAL SCENE IMAGES

The problem of text detection in the natural scene images has generated a lot of interest in the research community over the past decade. The problem is quite similar to the well-researched area of optical character recognition. The complexity of environments, malleable image acquisition styles and a huge variation of text contents poses numerous challenges in natural scene images. Few of them are summarized below.

- **Background complexity**: In native environments, numerous artificial objects such as buildings, paintings, boards and symbols appear which may have designs and appearances similar to text. Hence, the challenge of the scene complexity is that the surrounding environment makes it difficult to discriminate text from the non-text regions.
- **Extreme lighting variations**: For images captured in natural environments, uneven lighting is typical due to illumination differences or uneven responses from the sensor devices. This is likely to introduce deterioration of features and color distortion which subsequently incorporates false detection and recognition results.

November 25, 1980: (388th Day !!!)
Got "wallpaper" bread this morning for breakfast and again blueberry pie filling in place of jam which I refused. Lights stayed on this morning. Later in the morning they brought in a tape recorder with four tapes of modern music by "Stevie Wonder", "Van Morrison", "Phoebe Snow", etc. Also a box of Rice Krispies we can have for breakfast with powdered milk, of course. No mail again today. Did get a small, but fresh apple for supper tonight. No hot water for showers or laundry.

This afternoon were told we could watch TV. They started to show us an "Osmond Family" tape we had seen so I looked around and found two tapes of a movie "The Night They Raided Minsky's" so put on Part I. It apparently had been videotaped by someone in Los Angeles who sent it over for the hostages as it was "The 8 o'Clock Movie" and was complete with local commercials and news (1978 news). We were about 10 to 15 minutes into the movie when one of the terrorists stuck his head into the room, watched a bit of the movie and then disappeared.

In a few seconds another terrorist came in, turned off the video machine and, with no further explanation or offering another tape for us to see, abruptly informed us that we couldn't see that film! Later I asked him why we couldn't see that film he said it was written on the can in Persian. When I protested and asked who made that decision his only reply was "I don't know,"and when I asked why the tape was in the TV room if we weren't allowed to see it he again replied only "I don't know", which is one of their usual lies. I told him I was sick of having our mail and now our TV censored and reminded him that we weren't children, for all the good my complaining does! I just wonder how long our gov't is going to let us be subjected to this crap?

November 26, 1980: 389th Day !!!)
Had hot shower today and washed underwear in hot water. Another fresh apple for dinner tonight but had the egg and mashed potato combination for breakfast again! Received mail this evening all dated latter part of Sept. except one from my sister July 18; another from Switzerland July 25; one from cousin in Syracuse Aug. 1. Did get two from my wife Sept. 20 and 25. Stamps had been taken off my letter from Switzerland by whichever thief censored my mail.

Figure 2.1.2: Figure on left is an example of scene text and figure on right is an example of machine-printed text.

- **Degradation and blurring**: Due to unstable working conditions, blurring and defocusing of scene images can occur which forms a hurdle for facilitating the legibility of text in images. Image compression and decompression may also result in degradation of the quality of the text in images. The influences of these factors reduce the sharpness which makes tasks such as segmentation difficult.
- **Aspect ratios and distortions**: Texts seen in the natural scene images may have different aspect ratios. In addition, perspective distortion occurs when images are captured with a camera at non-perpendicular optical axis.
- **Fonts**: There always exists a possibility of appearance of italic characters in between script fonts which creates problems or difficulties in performing segmentation tasks. Appearance of artistic fonts also makes it difficult to perform accurate recognition.
- **Multilingual environments**: Depending on the languages, certain languages have tens of character classes while others have thousands, comprising of different shapes, styles and sizes.
- **Random appearances**: The location of text in natural scenes is highly randomized. The text can occur in any part of the image. Thus, heuristic assumptions based on the location cannot be applied to natural scene text detection.

Subsequently, because of these challenges, the detection rates are lower (often less than 80%) [YYHH14]. Hence, there is sufficient scope for improving text detection in

the natural scene images. This thesis concentrates on text detection in natural scene images.

Over the past few decades, there has been a wide range of relevant text detection related applications which include content-based image retrieval, visual accessing and industrial applications. Detecting and recognizing text in natural scene images can give relevant keywords which can be used to retrieve multimedia content related to obtained keywords. Various devices and text detection prototypes are developed enabling people who are visually impaired to recognize text in both indoor and outdoor scenes. Development of mobile devices with digital cameras and embedded modules is also widely used for information extraction as they facilitate the user to use the application more comfortably and efficiently. Numerous language translation applications are also of great interest, helping people to overcome language barriers. Automatic text detection is used in various industrial applications such as identification of addresses in mail sorting systems or classification of products in large-scale production line and logistics. Autonomous navigation of robots and automobiles are also of prime research interest.

2.3 TYPICAL TEXT DETECTION SYSTEMS

The objective of the text detection systems is to discriminate between text and non-text regions in an image. Several methods for text detection from images have been proposed in recent decades. They can be broadly classified into heuristics-based methods and machine-learning based methods.

- **Heuristics-based methods**: They rely on heuristic rules to differentiate text from non-text regions. As discussed above, text in the natural scene images is prone to high variability. This makes it more complicated to define rules for including all different type of texts present in natural scenes. Typical heuristic methods involve region-based and texture-based methods. Region-based methods involve segmenting an image into regions with similar properties using connected-components or edges. These segmented regions are analyzed and based on syntactic rules these regions are classified into text region. For e.g., Epshtein et al. use stroke width transform ([EOW10]) to detect the initial segmented regions while Neumann and Matas use maximally stable extremal regions ([NM11]) to detect the segmented regions. These regions are classified as text or non-text, based on syntactic rules. Texture based methods work under the assumption that certain texture properties are more pertinent in differentiating text from background [GLR+13].

- **Machine-learning based methods**: These methods use computer systems to learn a model to discriminate text and non-text regions. The model learned based on the presented input examples to the system. These models are referred as classifiers. The classifier should assign a numerical value to input based on the "textiness" of the input. The system should give more positive value if it finds that the input region contains texts and vice versa. Neural network based classification was utilized in [WWCN12] and support vector machine ([CV95]) based classification was used in [CCC+11]. The main drawback of using the methods involving machine learning techniques is that it is of high computational complexity.

Hybrid techniques amalgamate the efficiency of both heuristics and machine learning based techniques are also being proposed in [YYHH14] with remarkable success. The fine tuning of syntactic rules is a complex procedure and these rules will vary according to different languages. Thus, researchers use machine-learning approaches to generate statistical models for defining the syntactic rules that are used to segregate text from non-text connected components. In [YYHH14], Yin et al. use an AdaBoost classifier ([FS97]) for defining the model for segregating the text and non-text connected components determined using maximally stable extremal regions ([MCUP04]). The use of statistical models improve the adaptivity of the method significantly. Hence, the text detection system proposed by Yin et al. reported state-of-art results on multi-lingual dataset.

This thesis will focus on machine-learning based approaches for text detection. In machine-learning based approaches, computer system by itself learns to differentiate text and non-text samples. The algorithm that helps to learn the model is known as learning algorithm. In order to initiate the learning process, the computer system needs examples of text and non-text samples. These input samples are used by the learning algorithm to generate a model that can automatically detect text samples. In general, the input samples cannot be represented directly because of its high dimension. When the dimension increases, the number of input samples required for learning also increases. Hence, researchers have proposed the idea of features. Features represent the information about the input sample in lower dimension. These features are provided to the learning algorithm in order to generate the model. The model is a mathematical function which describes the relation between the feature vector and the group it belongs to. A typical text detection systems using machine-learning approach consist of the following:

- **Image acquisition**: A pictorial scene with text is captured and recorded in digital form.

- **Pre-Processing**: This step involves several techniques which could aid in improving the performance of the classifier. Depending on the use cases, images would require distinct preprocessing techniques. For instance, some of the basic techniques are converting images into grayscale, scaling or rotation.

- **Feature detector**: A feature detector detects regions that have higher probability for finding text. The regions can be detected using simple sampling strategies or specialized detectors known as region detectors. These region detectors detect visually salient regions.

- **Feature descriptor**: It describes the detected region. It can be a numerical value or a vector. Feature representation can be heuristically chosen or can be learned from data. Ideally, the feature descriptors of samples of the similar input examples should have similar values and should have dissimilar values with dissimilar input samples. The feature description process generate features that will be used by the learning algorithm to generate the statistical model. Thus, the generated features play a pivotal role in the creation of the model.

- **Classifier**: The fundamental idea of a classifier is to glean knowledge from a set of examples that are known to belong to a particular class and use this knowledge to decide on unknown examples as to which classes it belongs. The process of classification requires a trained classifier to determine the class of the unknown content. Training a classifier is an iterative process by which it is possible to build a suitable model for a particular dataset and classification is a one-time operation carried out to group the unknown samples. For text detection system, the input samples consist of text and non-text examples. These examples are transformed into features and that will be utilized by the learning algorithm to generate a model to discriminate text and non-text samples. The examples used to by the training algorithm to create the model can be classified into two: training data and validation data.
  - **Training data**: A training dataset consists of data with known classification. They should contain sufficient amount of samples that includes all possible variety of data to be classified. The training dataset is the seen data that estimates the model parameters. The performance of a model on a training dataset shows that it learns the seen data.
  - **Validation data**: In order to avoid over fitting of data, a validation dataset is necessary to tune the model in addition to training dataset. Over fitting occurs when the model performance is high on training examples and

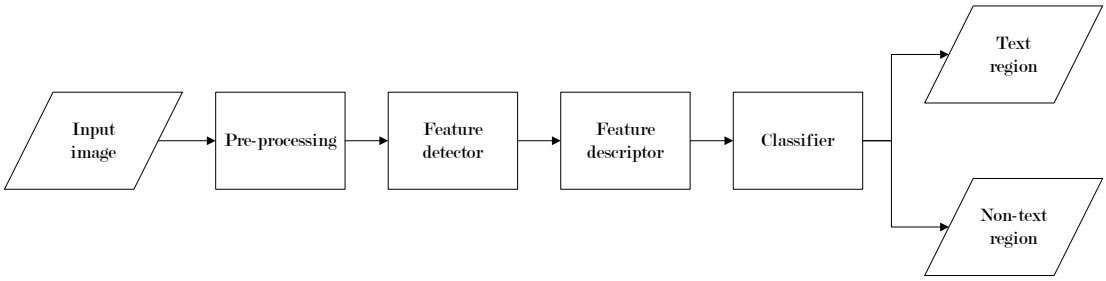


Figure 2.3.1: Typical text detection system

the performance degrades on unseen examples. For e.g., to find the most suitable classifier for a problem, the candidate algorithms are trained on the training dataset and the validation dataset is used to compare their performance and select the finest algorithm. Hence, a validation dataset validates or assesses the dependent predictive ability of the model. Validation dataset is used to find the prediction error of the model which can be used improve the meta parameter related to the trained model. The meta-parameters of the model will be adjusted so that the performance of the model is increased. After scrutinizing the performance of the model on the validation data, the best performing model is chosen. In text detection system, that model is chosen which can predict most input samples correctly as text or non-text.

Along with the training data, a third dataset namely test dataset is necessary to evaluate the performance of the trained model.

- **Test data**: Once a trained model is obtained, it is necessary to measure its performance on unseen data or the test dataset. The performance of the model on the test dataset shows the ability of the model to generalize the data that has no significance in selecting the model. It shows an independent assessment of the predictive ability of the model.

All the three datasets mentioned above should be disjoint and independent of each other.

- **Output**: The output of the model enables us to discriminate between text and non-text regions. Depending on the values of the input features, the classifier assigns a particular value to the input sample. In text detection system, the output depends on the "textiness" of the input sample. The classifier outputs a positive value if it detects the input sample as text and vice versa.

2.4 LEARNING APPROACHES

Section 2.3 described in detail the architecture of a typical text detection system. The system described uses leaning algorithms to generate the statistical model. The learning algorithms can be broadly classified into two, namely unsupervised and supervised learning algorithm.

- **Supervised learning algorithm**: A Supervised learning algorithm requires input-output pair examples and the algorithm determines a mapping function from the input to output. This learning algorithm generally consists of two stages: in the initial stage, training examples are used to determine the model parameters. Then, testing samples are provided, for which the algorithm provides the prediction based on the incoming samples and the model parameters that are learned from the training examples. One of the major issues for the supervised learning algorithms is that, it is not easy to obtain ample labeled training samples. As the input dimension increases the data required to learn a relation between the input-output pairs also increase drastically. Overfitting is also one of the disadvantages of supervised learning algorithm. Overfitting occurs when the algorithm starts to memorize the input examples instead of learning. This will lead to a good performance on the training data but reports a poor performance on test dataset.
- **Unsupervised learning algorithm**: When compared to supervised learning, unsupervised learning algorithms does not require explicit target outputs. The input data will be unlabeled or not annotated. Using unlabeled data, the algorithm forms clusters or natural groupings of the input data. The major advantage of unsupervised learning is that, the input data does not have to be labeled. The most common unsupervised learning method is clustering. Clustering is the process of grouping input samples into clusters. A member in a cluster has similar properties to other members in the same cluster and dissimilar properties to members of another cluster.

2.5 CONCLUSION

Text detection in natural scene is an interesting problem. The challenges for text detection in natural scenes was discussed in detail in Section 2.2. Section 2.3 described the possible approaches followed for text detection in unconstrained images. They are heuristics-based and machine learning-based approaches. In heuristics-based

approaches, they rely on heuristic rules to differentiate text from non-text regions. It is highly complicated to define rules for all the variabilities present in natural scenes. Machine learning-based methods use computer systems to derive a model to differentiate text and non-text regions. Section 2.3 described a typical text detection system based on machine learning methods. Section 2.4 described the basics of learning algorithms that aids the computer systems to generate the model to differentiate between text and non-text regions.

3 LOCAL IMAGE FEATURES

As described in Chapter 2 the input sample images need to be transformed into features. The generated features play a vital role in learning the statistical model. The features can be computed for the whole image. Such features that represent the whole image are known as global image features. On the other hand, features can computed for local image patches. These features are called local image features. Key requirements for a good local feature are invariance to scale, rotation, change in illumination, image distortions and minor changes in viewing directions. Also the local image feature must be highly distinctive.

The input sample images are pre-processed to make them more suitable for further processing (Section 3.1). The process of detecting interest areas in an image is known as feature detection (Section 3.3). The regions can also be determined by using sampling strategies (Section 3.2). The detected regions obtained either by using sampling strategies or using feature detectors form the local image patches. These detected regions are represented using feature descriptors (Section 3.4).

3.1 PRE-PROCESSING

Pre-processing methods are those methods that are applied to improve the data quality of the image to enhance the performance for subsequent operations. These methods make the image more suitable for particular application than the original image. Pre-processing techniques should be carried out without the loss of relevant information. No pre-processing techniques suit for all the applications. Hence, the pre-processing method suited for a particular application is selected after empirical evaluation. Pre-processing methods are applied to reduce the unwanted variability in image data. All the pre-processing techniques that operate directly on pixel intensity value are referred to as spatial domain methods.

3.1.1 *Normalization*

Normalization is a pre-processing technique that transforms the range of pixel intensity values. There are different normalization techniques available in literature. The

following normalization technique transforms the input data to data with zero mean and unit variance. If the normalization technique is applied on an image patch, then it normalizes the brightness and contrast of the image. Often a small value ϵ_{norm} is added to the variance to avoid division by zero and to suppress unwanted random intensity value changes (noise) [CN12]. Let f_{ij} is the original intensity value in the image patch, then the normalized intensity h_{ij} is given by as follows

$$h_{ij} = \frac{f_{ij} - \mu}{\sqrt{\sigma^2 + \epsilon_{norm}}} \tag{3.1.1}$$

where

$$\mu = \frac{1}{\sum_{ij} 1} \sum_{ij} f_{ij} \qquad \sigma^2 = \frac{1}{\sum_{ij} 1} \sum_{ij} (f_{ij} - \mu)^2 \tag{3.1.2}$$

The mean and variance of the normalized intensities h_{ij} is 0 and 1 respectively.

3.1.2 *Whitening*

Whitening is one of the pre-processing steps. The process of whitening involves decorrelation of data followed by variance normalization. Whitening is otherwise known as sphering. Each dimension of the whitened data will be decorrelated with one another and the variance in each dimension is 1. For any machine learning system, it is highly undesirable to have highly redundant input components. The adjacent pixel values of an image patch are highly correlated and this redundancy can be removed by whitening. Whitening the input data will simplify the subsequent model building. Consider an input data matrix X of size N-by-M, each data sample is of dimension N and M data samples are arranged column-wise. The process of whitening the input data X is enumerated below:

1. N-dimensional mean vector, i.e. mean of the values for each dimension across all the examples is calculated. Then the mean vector is subtracted from the input data. This step ensures that the input data is mean-free.

2. Since, the input data X is mean-free then the N-by-N dimensional covariance matrix is given by

   $$\Sigma = \frac{1}{M} X X^T \tag{3.1.3}$$

   The element in covariance matrix in (i, j) position is the covariance between the vectors along the i^{th} and j^{th} dimension of input data X.

3. The eigenvalue decomposition of the covariance matrix Σ is calculated where V will contain all the eigenvectors column-wise and diagonal entries of D contains all the eigenvalues.

$$VDV^T = \Sigma \tag{3.1.4}$$

4. Zero component analysis (ZCA) whitening transform for input data X is defined as per Equation 3.1.5. A small constant is added ϵ_{zca} to the eigenvalues before taking the square root. This step ensures that the values won't tend to infinity even if the eigenvalues are numerically close to zero.

$$X_{zca} = V * (D + \epsilon_{zca} I)^{\frac{-1}{2}} * V^T * X \tag{3.1.5}$$

where $(D + \epsilon_{zca} I)^{\frac{-1}{2}}$ is obtained by taking the square root of all the elements of the matrix $(D + \epsilon_{zca} I)$ and I is identity matrix of size same as the dimension of D.

Each dimension of X_{zca} will be decorrelated and the variance across each dimension is one. For detailed analysis on whitening, refer [Fin14], [DHS12] and [CN12].

3.2 DENSE SAMPLING

In dense sampling, local image patches are extracted at regular intervals. These extracted local image patches will be used for subsequent processing. The two important parameters required for dense sampling are receptive field and stride. The size of the patch in pixels is known as receptive field and the regular interval at which the patches are extracted is known as stride. Consider Figure 3.2.1, from an M-by-N image with a receptive field equal to r and stride equal to s would result in $\frac{M-r}{s} + 1 - by - \frac{N-r}{s} + 1$ r-by-r patches. If $s < r$, patches are overlapping and if $s > r$, patches are non-overlapping. It is evident from the figure that dense sampling results in a good coverage of the entire image provided a low value is selected for the stride parameter.

3.3 FEATURE DETECTOR

The feature detector is responsible for detecting interest regions in an image. These interest regions are denoted by single point coordinates known as keypoints. Keypoints are also known as interest points. A keypoint is represented by a pixel coordinate in

Figure 3.2.1: Red boxes denote the extracted image patches. The size of the box in pixels is known as receptive field and the interval at which the patches are extracted is called stride. Generally the stride will be less than the receptive field. However, in the above picture, intentionally the stride is chosen to be greater than the receptive field for better visualization.

an image. Keypoints are located in visually salient regions of an image. The keypoint detectors are also referred as interest point operators. Some of the popular interest point operators are Scale Invariant Feature Transform (SIFT) [Low99], Harris-Affine detector [MS04] and Maximally Stable Extremal Regions (MSER) [MCUP04]. Nowat et al. [NJT06] have performed a detailed comparative study of deterministic sampling strategies and various interest point operators for image classification task. The results indicate that the performance depends on the number of patches extracted. Thus, when extracting sufficient patches, the performance of dense sampling methods outperforms more complex methods based on interest point operators. Figure 3.3.1 shows a comparison between the interest points detected by dense sampling method and interest point operator. Interest point operators leave large part of the image unsampled as shown in the figure. In this section, interest point operator maximally stable extremal regions will be discussed in detail.

3.3.1 *Maximally stable extremal regions*

The Maximally Stable Extremal Region (MSER) detector was proposed by Matas et al. [MCUP04]. An MSER is characterized by a region of connected components with

(a) (b)

Figure 3.3.1: Dense Sampling Vs MSER Detector. Figure (a) shows the interest points detected by dense sampling. Here the stride is selected to be greater than the receptive field for better visualization. The detected keypoints are denoted by black dot. Figure (b) shows the keypoints detected by MSER detector from same image. MSER regions detected are denoted by green ellipses. and the keypoints are represented by green 'plus' symbol. It is evident that in case of interest point operators large part of the image is unsampled.

similar intensity values bounded by contrasting backgrounds. Consider a grayscale image, now segment the image at multiple thresholds $T_1, T_2, ..., T_n$. The difference between the thresholds $(T_n - T_{n-1})$ is known as delta. The thresholds range from 0 to 255. During the thresholding process, all the pixels with intensity value below the threshold are assigned black color and white color is assigned to rest of the pixels. Initially at threshold T_1 equal to zero, then there are no pixels where the intensity value is less than zero, hence a white image is obtained as shown in Figure 3.3.2. Subsequently when the threshold increases, black regions starts to appear. As the thresholding intensity is increased, the area of the black regions will also grow and the final image will turn entirely black. The process is represented graphically in Figure 3.3.2. During the whole process, the set of all the connected components that are colored black form the extremal regions. The stability of the extremal region is given by the change in the area of the connected component region over the range of thresholds. An extremal region is said to be maximally stable if the change in the area is minimal over the range of thresholds. MSER have properties such as affine transformation invariance of the intensity function, multiscale detection and a measure of stability [MCUP04]. Since the MSER detector can detect ares of varied sizes, the detector can be used for multiscale detection. This property is exploited by text detection methods that use MSER to detect charter candidates of different sizes. Also each region has stability score based on the change in the area over the range

of thresholds. An MSER detector performs well on images containing homogeneous regions distinctive boundaries [MTS+05]. Hence, the detected MSER can provide relevant interest points for character candidate detection as generally characters in images have homogeneous intensity values bounded by distinctive backgrounds. This fact is also evident from the example shown in Figure 3.3.2. In Figure 3.3.1, MSER regions detected are marked with green ellipses.

3.4 FEATURE DESCRIPTOR

The image patches extracted by using dense sampling or the visually salient regions detected by the interest point operators form the local image patches. Each local image patch is denoted by single pixel coordinate known as keypoints. These patches are represented by using the feature descriptor. Ideally, feature descriptor should be invariant to irrelevant transformations, aids in discriminating different classes and insensitive to noise. These features extracted from local image patches are known as local image features or local image descriptors. The feature descriptor defines the transformation from the local image information to features. Generally local image features is a vector of numerical values.

3.4.1 *Hand-crafted feature descriptors*

Hand-crafted feature descriptor utilizes a-priori selected representation methods. For e.g., an image patch of receptive field 8-by-8 can be represented by its mean of all the intensity values in the patch. The method of representing the region with its mean intensity value is a-priori selected. Such class of feature descriptors where the representation method is a-priori known, can be classified into hand-crafted feature descriptors. The resultant features are known as hand-crafted features or hand-crafted descriptors. One of the most popular hand-crafted feature descriptor is Scale Invariant Feature Transform (SIFT) descriptor [Low04]. In the following section, SIFT descriptor is discussed in detail.

SIFT Descriptor

Scale Invariant Feature Transform (SIFT) detector [Low99] and descriptor [Low04] were proposed by Lowe. The SIFT detector is region detector based on response of Difference-of-Gaussian filter. The SIFT descriptor consists of histogram of oriented gradients. Studies have shown that SIFT descriptor outperforms other hand crafted descriptors in many use cases [MS05].

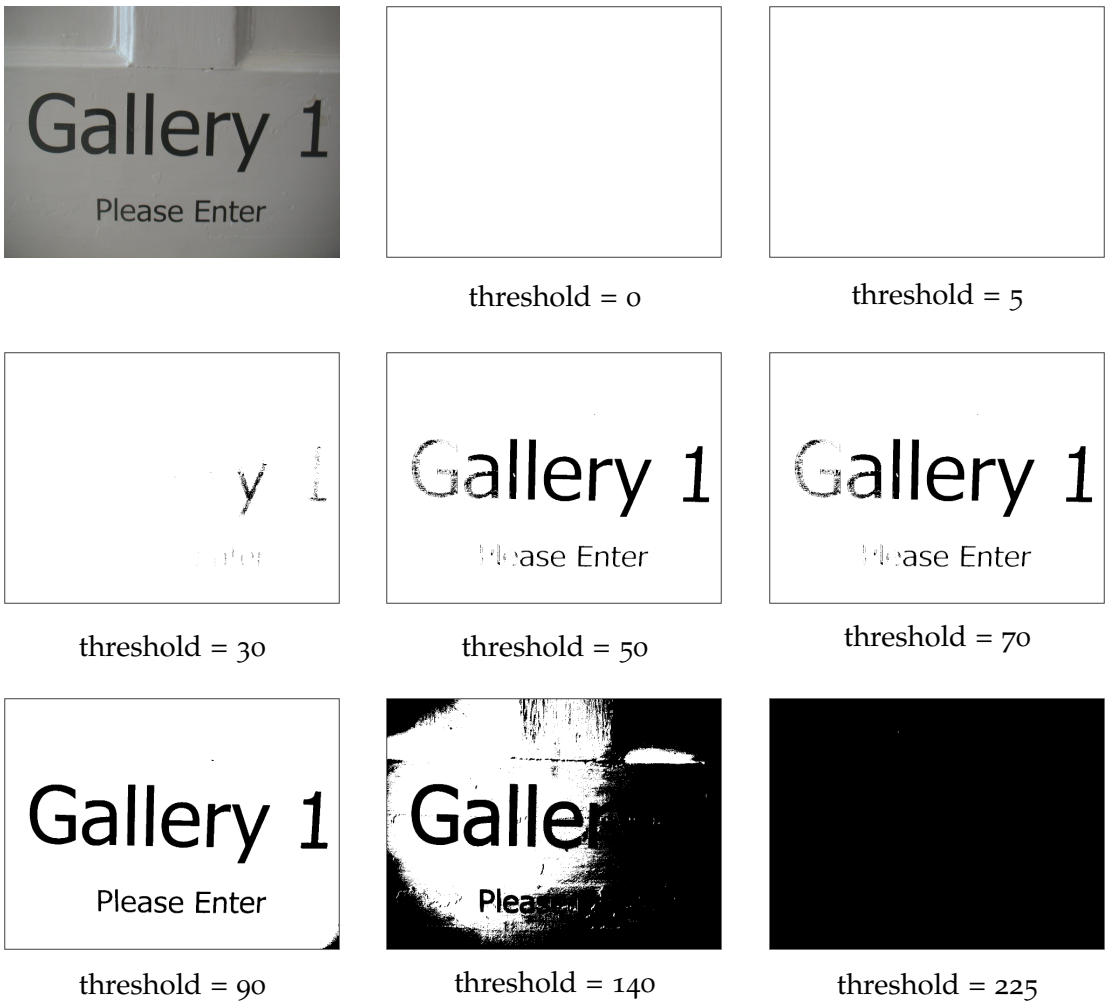


Figure 3.3.2: Figure explains the process of MSER detection. Initially, the original image is thresholded at multiple thresholds. In this example, the image is thresholded at $0, 5, ...30, ...255$. The thresholded image is also shown at different thresholds. Note there are connected component regions which are stable over large range of thresholds. The region where "1" is present in the image remains constant from threshold = 50 until threshold = 90. Such regions will be classified as extremal regions and those extremal region with minimal change in area over the range of thresholds are known as maximally stable extremal regions. The change in area is the stability parameter for the MSER region.

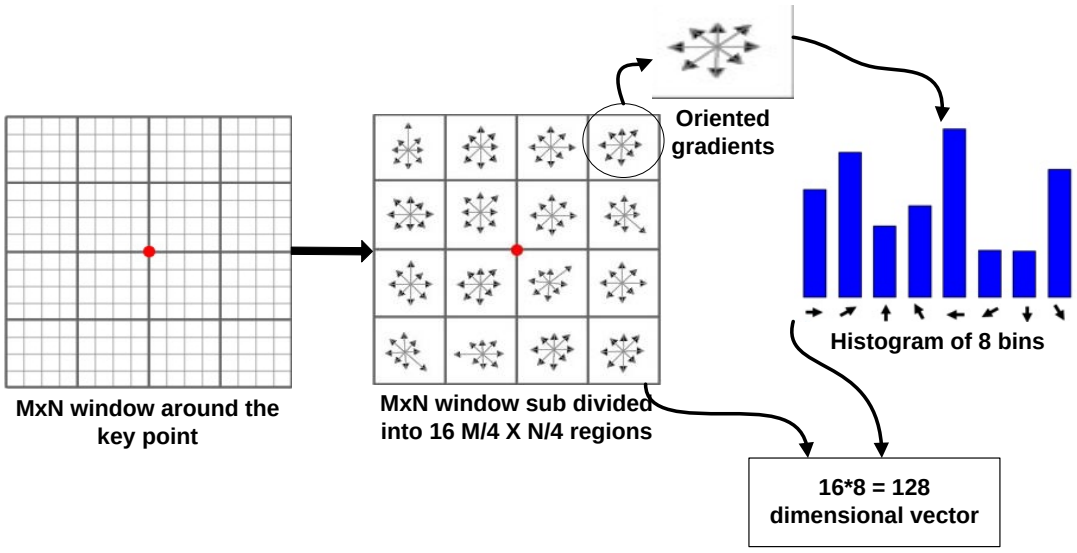


Figure 3.4.1: Scale Invariant Feature Transform Descriptor. SIFT descriptor is 128-dimensional vector based on histogram of oriented gradients. The window around the keypoint can be configured. Image based on [Low04].

First, around the keypoint location, the image gradient magnitudes and orientations are sampled. The coordinates of the descriptor location and gradient orientations are rotated with respect to the keypoint orientation. This makes the descriptor orientation invariant. The magnitude of the gradient is weighted with a Gaussian function. Thus, the magnitude and orientation of the gradients around the keypoint are obtained. Lowe has experimentally determined and reported that the keypoint neighborhood subdivided into 4-by-4 subregions gave the best results. For each subregion, histogram of oriented gradients is generated with eight bins each. Finally, the keypoint neighborhood is represented by $4x4x8 = 128$ values. Thus, the SIFT descriptor has a dimension of 128. The process is depicted in Figure 3.4.1. For detailed study about SIFT detector and descriptor, refer [Low99] and [Low04] respectively.

3.4.2 *Learned feature descriptors*

The hand-crafted feature described above is one of the common choices for descriptor representation. However, these descriptors do not adapt based on application. Hence, researchers have come up with other class of descriptors which will adapt based on

the application domain. These descriptors are classified as learned feature descriptors. The learned feature descriptor learns the representation method from the input data. Hence, they are more specialized to the application domain and need not be engineered by hand. The transformation function that converts the local image patch information to features is learned from input data by using unsupervised learning algorithms. The features obtained are known as learned features or learned descriptors. Due to recent advances in machine learning, learned feature descriptors are considered the current state of the art.

3.5 CONCLUSION

Local image feature generation process was discussed in this chapter. The images are pre-processed to make them more suitable for subsequent processing. The pre-processing algorithms introduced in this chapter are normalization (Subsection 3.1.1) and whitening (Subsection 3.1.2). Normalization technique reviewed in this chapter transforms the input data to data with zero mean and unit variance. Application of this technique on image patch results in brightness and contrast normalization of the image patch. Whitening is also a pre-processing method that decorrelates the input data and subsequently normalizes the variances across each dimension of the input data. Local image patches are extracted from preprocessed images using dense sampling methods (Section 3.2) or detected using interest point operators such as maximally stable extremal region (Subsection 3.3.1). These generated local image patches are described using the hand-crafted feature descriptor (Subsection 3.4.1) or learned feature descriptor (Subsection 3.4.2), thereby generating the local image features.

The performance of learned and hand-crafted feature descriptors need to be evaluated. Chapter 2 substantiated the reasons for text detection in natural scene images being a challenging problem. Thus, the performance of learned and hand-crafted feature descriptors will be scrutinized with their performance in detecting text in natural scenes. The following chapter will discuss in detail bag-of-features paradigm which is the most widely used feature representation method to generate learned descriptors.

4

BAG-OF-FEATURES

Bag-of-Features (BoF) methodology is widely used in the field of computer vision for image classification, object detection, image retrieval and autonomous navigation of robots [OD11]. It represents an image as an orderless collection of local image features. Bag-of-features methods do not store any spatial information of the local image features.

The basic concept of bag-of-features methodology is explained in Section 4.1. Bag of features use local image descriptors (Chapter 3) for image representation. The local image features generated from input images are grouped to generate the representatives (Section 4.2). The histogram of representatives in an image gives the learned feature vector that is used by the classifier (Section 4.3) for classification.

4.1 BAG-OF-FEATURES

Bag-of-features approach was inspired from bag-of-words method, that is used for textual document representation [OD11]. Bag-of-words represents a document as an orderless collection of words. First the document is parsed and all the words are extracted. During this process, all the common words such as 'a', 'and', or 'the' are neglected as these words are commonly found in every document. Then the words are replaced with their stem word. For e.g., the words 'trying', 'tried' will be replaced by their stem word 'try'. The count of stem words occurring in the document is calculated and subsequently, the document is represented as the histogram of relative or absolute frequency of stem words. The resultant representation is known as term vector. Evidently, the term vector does not store the spatial location of the word in the document, hence, the approach is known as bag-of-words.

Analogous to documents, images also can be represented as a collection of local image features. Bag-of-features image representation can be summarized as follows:

- **Generate vocabulary**: Local image features are extracted from all the training images. All the local image features are grouped by the process of clustering. Clustering is defined as the method of organizing objects into various groups based on certain similarities. Each group is referred to as a cluster which is a collection of objects that are similar in their group and dissimilar to the objects

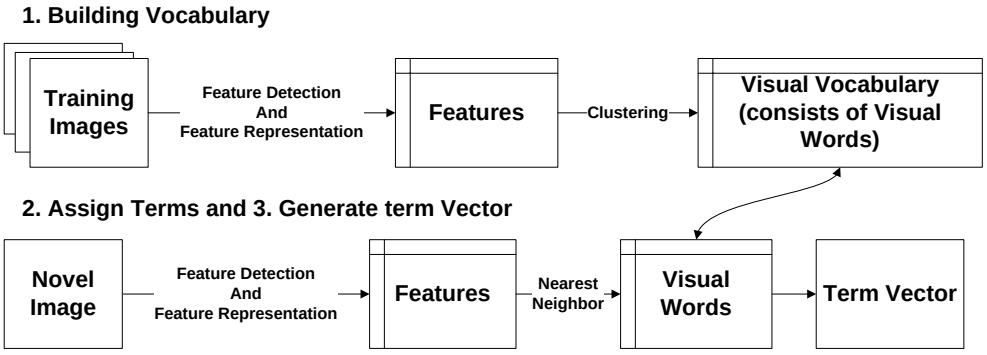


Figure 4.1.1: Bag-of-features approach for image representation involves generation of a visual vocabulary and the features for a novel image is generated using the visual words from visual vocabulary. Image based on [OD11].

in other groups. The resultant clusters obtained after clustering the local image features constitute a visual vocabulary or visual codebook. Each item in the visual vocabulary is known as visual word. Clustering is analogous to stemming in bag-of-words approach.

- **Generate features for a test image**: The local image descriptors are extracted from a novel image and for each descriptor the most similar visual word from the visual vocabulary is assigned.
- **Generate the term vector**: The number of visual words in the novel image are used to generate the normalized histogram of visual words. The term vector dimension is the same as the number of visual words in the visual vocabulary.

The steps involved in BoF process are represented pictorially in Figure 4.1.1. For a detailed overview and review of bag-of-features method, refer [OD11].

4.2 CLUSTERING

Clustering is considered as the most important unsupervised learning task that determines a structure in a collection of unlabeled samples. Clustering is the process of natural grouping of input patterns. i.e. to identify useful clusters of input examples.

Based on the cost function used in the clustering algorithm, different clusters are generated. The number of centroids is experimentally determined before executing the clustering algorithm. In bag-of-features algorithm, vector quantization or clustering is used to build the visual vocabulary. The process of assigning the feature descriptors to the representatives is known as quantization. The collection of these representatives constitutes a codebook. In bag-of-features terminology, the codebook is referred to as visual vocabulary. Each representative in the codebook is also termed as centroid. There are many clustering algorithms available, the following section explains in detail the method of Lloyd's algorithm and spherical k-means.

4.2.1 *Lloyd's algorithm*

Lloyd's algorithm was developed by Stuart P. Lloyd [Llo82]. The algorithm requires the data to be clustered and the number of clusters. Lloyd's clustering algorithm determines the cluster centers or centroids such that the squared Euclidean distance between the centroids and data points is minimum as described by Equation 4.2.1. If cluster centroids are denoted by $D^{(k)} \in \mathbb{R}^n$ and each input data point is denoted by $x^{(i)}$, then Lloyd's algorithm finds out the cluster centroids $D(k)$ such that

$$\underset{D}{\text{minimize}} \quad \sum_{i} \|D^{(k)} - x^{(i)}\|^2 \tag{4.2.1}$$

where $D^{(k)}$ is the cluster center assigned to the data point $x^{(i)}$.

Lloyd's algorithm can be summarized as follows:

1. **Initialization**: Consider a sample dataset $x^{(1)}, x^{(2)} \cdots x^{(n)}$ and cluster this dataset into k clusters with centroids $D^{(1)}, D^{(2)} \cdots D^{(k)}$. A suitable initial codebook is chosen based on heuristic methods, as no optimal assumptions are possible at the initial part of the algorithm. Hence, initial centroids are randomly chosen. The codebook is denoted as Y^m, where m denotes the iteration count. The initial centroids constitute the initial codebook (Y^0). This step also initializes the iteration count $m = 0$. Figure 4.2.1 shows result of the initialization process of randomly generated data.

2. **Assignment**: Each data sample is assigned to the nearest neighbor of the current codebook Y^m. The nearest neighbor is selected based on minimum Euclidean distance between the current data sample and the centroids in the codebook.

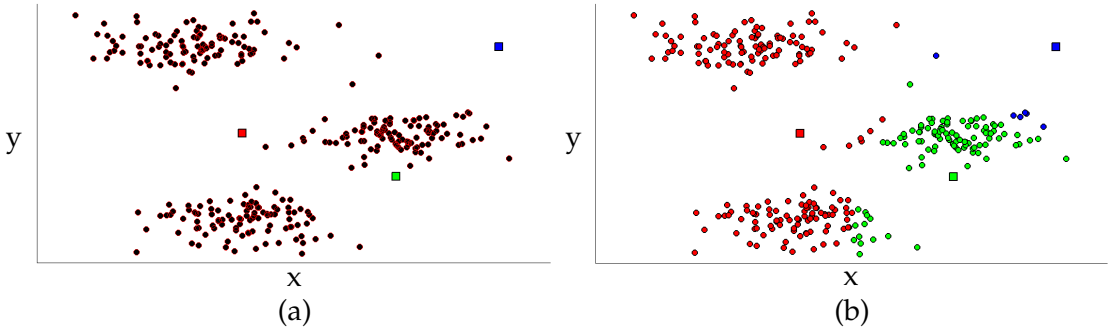


Figure 4.2.1: Figure (a) shows the initialization process. The centroids are randomly initialized. The data points are represented by black dots and centroids are represented by squares. Figure (b) represents the assignment step. During the assignment each data point is assigned to one centroid. The assignment is based on minimum Euclidean distance. The data points are color coded according to their assignment.

This step is computationally expensive because of the iteration through all the data points. Figure 4.2.1 depicts the result of the assignment process.

3. **Centroid update**: After the assignment step, the codebook is updated. The new codebook contains the centroids that define the current partition of the input space. The code book update tries to minimize the distance between data points and centroid. Iteration counter is incremented by one ($m = m + 1$).
Repeat step 2 and step 3 until convergence of Equation 4.2.1.

Once the quantization error is minimized, the resultant clusters are obtained. From the above process, it is quite evident that the resultant clusters are highly dependent on the initialization step. In order to reduce the dependency on initialization step, Lloyd's algorithm is generally executed with few random initializations and select the clusters with minimum quantization error. Figure 4.2.2 shows the updated centroids as well as the resultant clusters obtained.

4.2.2 *Spherical k-means*

Spherical k-means algorithm was proposed by Dhillon et al. [DM01]. Similar to Lloyd's algorithm, spherical k-means also requires data to be clustered and the number of clusters. Spherical k-means is also referred to as gain shape vector quantization.

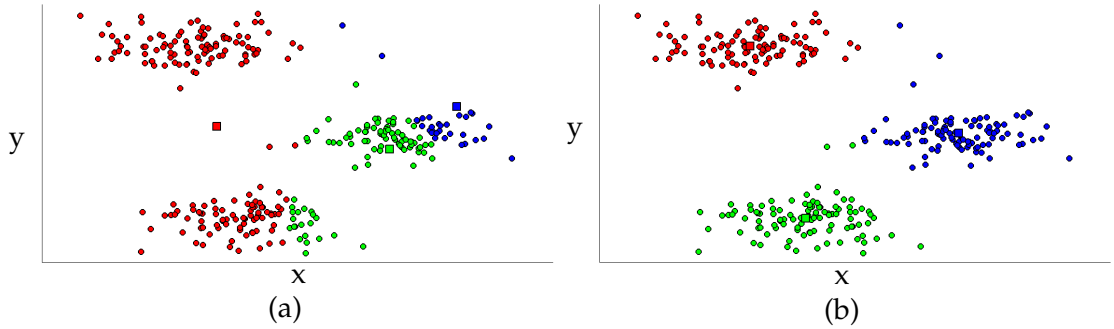


Figure 4.2.2: Figure (a) shows the updated centroids and figure (b) shows the resultant clusters obtained after Lloyd's algorithm converges or quantization error is minimized.

Spherical k-means will also determine the cluster centers such that it minimizes Equation 4.2.2.

$$\begin{aligned} \underset{D,z}{\text{minimize}} \sum_{i} \left\| Dz^{(i)} - x^{(i)} \right\|^2 \\ \text{subject to } \left\| D^{(k)} \right\| = 1, \forall k \\ \text{and } \left\| z^{(i)} \right\| \leqslant 1, \forall i \end{aligned} \tag{4.2.2}$$

where the vectors $z^{(i)} \in \mathbb{R}^k$ are called code vectors and each code vector has at-most one non zero element. These code vectors are also referred as one hot encodings. Each data point will have a corresponding code vector or one hot encodings.

The steps involved in spherical k-means is similar to Lloyd's algorithm. Spherical k-means algorithm can be summarized as follows:

1. **Initialization**: Similar to Lloyd's algorithm, the initial centroids are chosen during this step. Since optimal assumptions cannot be drawn during the initial phase of the algorithm, the initial centroids are chosen based on heuristic methods. Thus, initial centroids are randomly chosen. The centroids are always normalized to the unit length. Figure 4.2.4 shows randomly chosen data points with random initial centroids.

2. **Assignment**: During the assignment step, data points are assigned to one of the centroids in the codebook. Figure 4.2.4 shows the result of assignment step. Spherical k-means uses the projection value between the centroids and the data points and it assigns the data point to that centroid where the projection value is

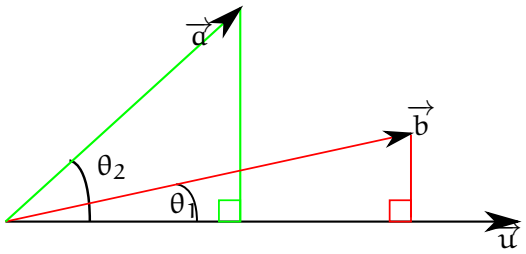


Figure 4.2.3: Figure shows the projection of two vectors onto a unit vector. It shows the fact that smaller the angle between vector and the unit vector, higher the projection value onto the unit vector.

maximum. Consider Figure 4.2.3, it shows the projection of two vectors ($\vec{a}$, $\vec{b}$) onto a unit vector ($\vec{u}$). The projection value of $\vec{a}$ onto $\vec{u}$ is given by $\|a\|\cos(\theta_2)$ and projection value of $\vec{b}$ onto $\vec{u}$ is given by $\|b\|\cos(\theta_1)$. Since $\theta_2 > \theta_1$, then $\cos(\theta_2) < \cos(\theta_1)$. Therefore, the projection value of $\vec{a}$ onto unit vector $\vec{u}$ is smaller than the projection value of $\vec{b}$ onto unit vector $\vec{u}$. Thus, smaller the angle between the data point and centroid, higher the projection value.

3. **Centroid update**: After the assignment operation, the data points are partitioned differently. Hence, the centroid update is performed as per Equation 4.2.4.

$$z_k^i := \begin{cases} D^{(k)T} x^{(i)} & \text{if } k == \arg\max_j D^{(j)T} x^{(i)} \\ 0 & \text{otherwise} \end{cases} \qquad \forall i, k \tag{4.2.3}$$

$$\begin{aligned} D &:= XZ^T + D \\ D^k &:= D^k / \left\| D^k \right\| \forall k. \end{aligned} \tag{4.2.4}$$

where Z is the column-wise concatenation of the code vectors z^i and X is obtained if the input data points are arranged column-wise. Each iteration determines non-zero element for each z^i in order to minimize Equation 4.2.2. Calculate the absolute value of the projection between each data point vector (x^i) and all the k centroids and select the centroid with max absolute projection value. The obtained value gives the non-zero index of z^i and non-zero value is equal to projection value obtained with data point (x^i) and selected centroid. The process

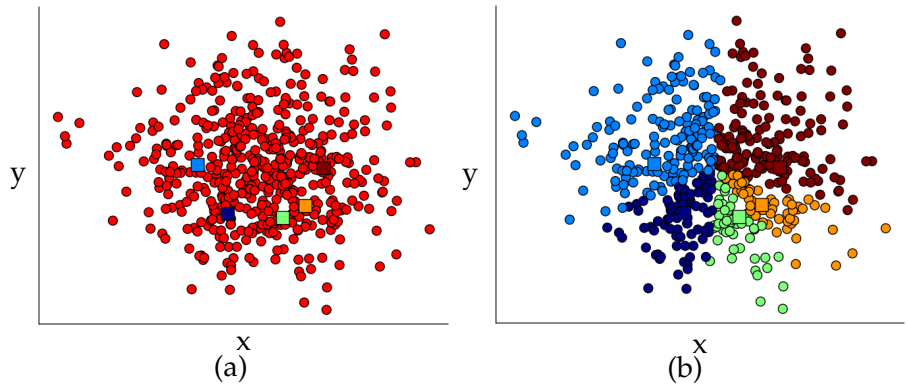


Figure 4.2.4: Figure (a) shows the initialization step. The centroids are randomly initialized from the standard normal distribution. The data points are represented by red dots and centroids are represented by squares. The centroids are normalized to the unit length. Figure (b) shows the assignment step. Each data point is assigned to one of the centroids from the codebook. The assignment is based on projection value of the data point onto the centroid. The data points are color coded according to their assignment.

is iterated over all data points to yield Z which is used to perform the centroid update using Equation 4.2.4. The centroids are normalized to unit length.

Repeat step 2 and step 3 until convergence of Equation 4.2.2.

Once the algorithm converges, the resultant clusters are obtained. Figure 4.2.5 shows the resultant clusters obtained after the execution of spherical k-means. The basic difference between Lloyd's algorithm and spherical k-means is that, the former uses Euclidean distance to determine the nearest neighbor while the latter uses the projection between the centroids and data points to determine the nearest neighbor.

4.3 CLASSIFICATION

In bag-of-feature approach, classifiers are used to classify the generated features into different groups. Moreover, these classifiers take into account, a set of known input data and the corresponding outputs. Based on this information, the classifier learns to classify the input data based on a set of rules or algorithms. All the input samples are converted into features based on selected feature detection and feature representation methods. These generated features are used to train the classifier to derive the relation between features and output labels. Basically, the goal of a classifier is to partition

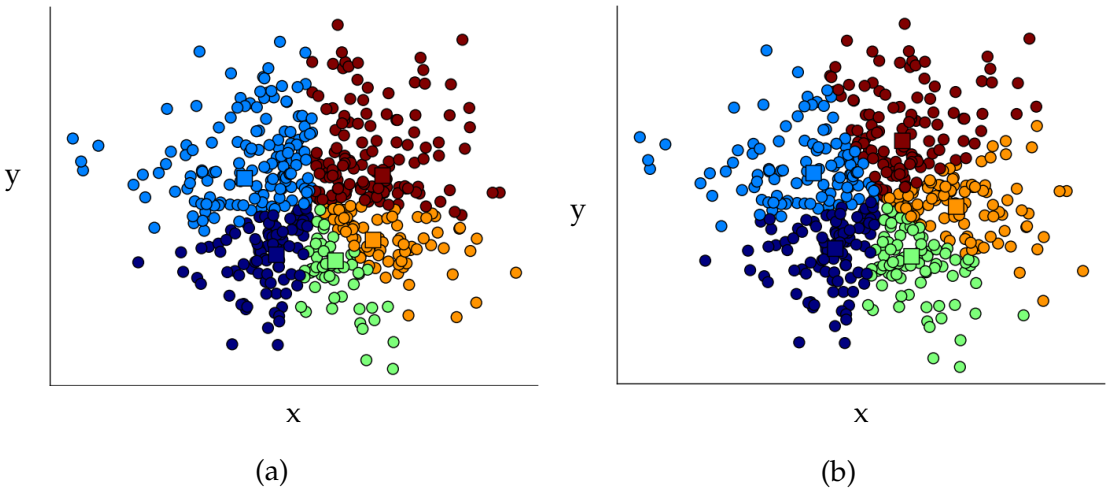


Figure 4.2.5: Figure(a) shows the updated centroids and figure (b) shows the resultant clusters obtained after the spherical k-means algorithm converges.

feature space into labeled decision region. The borders between these decision regions are called decision boundaries.

Some common classifiers are nearest neighbor classifier, naïve Bayes classifier and support vector machines. The Nearest Neighbor classifier assigns the output labels based on the output label of the closet neighbor in the feature space from the training data. K-nearest neighbor classifier predict the output labels based on the majority vote among the outputs predicted by k nearest neighbors from the training data. Naïve Bayes classifier, based on Bayesian theorem, is one of the popular classifiers. It is generally used when the input dimensionality is high. The parameter estimation for the naive Bayes model uses maximum likelihood method. The main advantage of naïve Bayes classifier is that it requires less amount of training data for parameter estimation. The Bayesian classification is a supervised learning method, based on Bayes theorem proposed by Thomas Bayes. This classification method assumes a probabilistic model and captures the uncertainty in the model by determining the probabilities of the outcome. It is highly robust to noise in the input data. Support vector machines is used in this thesis for classification of text and non-text samples. Hence it is described in detail in the following section.

4.3.1 *Support vector machine*

The Support Vector Machine (SVM) was proposed by Vapnik and Cortes in [CV95]. It is one of the popular classifiers used for supervised learning and performs classification by constructing an N-dimensional hyperplane that optimally separates the data into two groups. It constructs a maximum margin separator which enables them to generalize well. Margin is defined as the distance between the closest sample of either class to the separating hyperplane. The data points lying on the hyperplane are known as support vectors. Support vectors and hyperplane separating the data are shown in Figure 4.3.1. The training examples transformed to feature space are used to generate the hyperplane, Consider an input example represented by $\mathbf{x}$, the classification function generated by a linear SVM is given by

$$f(\mathbf{x}) = w^T\mathbf{x} + b \tag{4.3.1}$$

where w and b are the parameters determining the linear hyperplane. w and b are determined using the training data provided to the SVM. The SVM will predict one class if $f(\mathbf{x}) > 0$ and predict the other class if $f(\mathbf{x}) < 0$. The class labels are predicted based on where the input sample is located with respect to defined hyperplane. In many practical scenarios, the problem at hand will not only require the class of the input sample, but also a score to denote how positively the input sample belongs to a class. For e,g. in case of text detection, the score should tell the "textiness" of the input sample. If the input sample is a text, the score should be positive and if the sample is non-text, the score should be negative. For determining the score, the distance from the input sample to the hyperplane is calculated. In this thesis, the SVM score denotes the distance to the hyperplane of the input sample.

In case of SVM, a small penalty term is added to penalize the wrongly classified samples. The process is called as regularization. The SVM modes with different regularization parameter values are trained and these models are evaluated on validation dataset to select the best suited model. The model is chosen that can classify most samples correctly.

Often the data will not be linearly separable, in such cases SVM uses kernels to transform the data into higher dimensional space where it will be linearly separable. The commonly used kernels are radial basis function, polynomial and sigmoid [HCL+03]. Support vector machines are naturally two-class classifiers. To enable SVM to perform multi-class differentiation, different approaches are followed. The most common technique is one-versus-all classifier that chooses the class that classifies the test data with the largest margin. Another approach is to train several one-versus-one

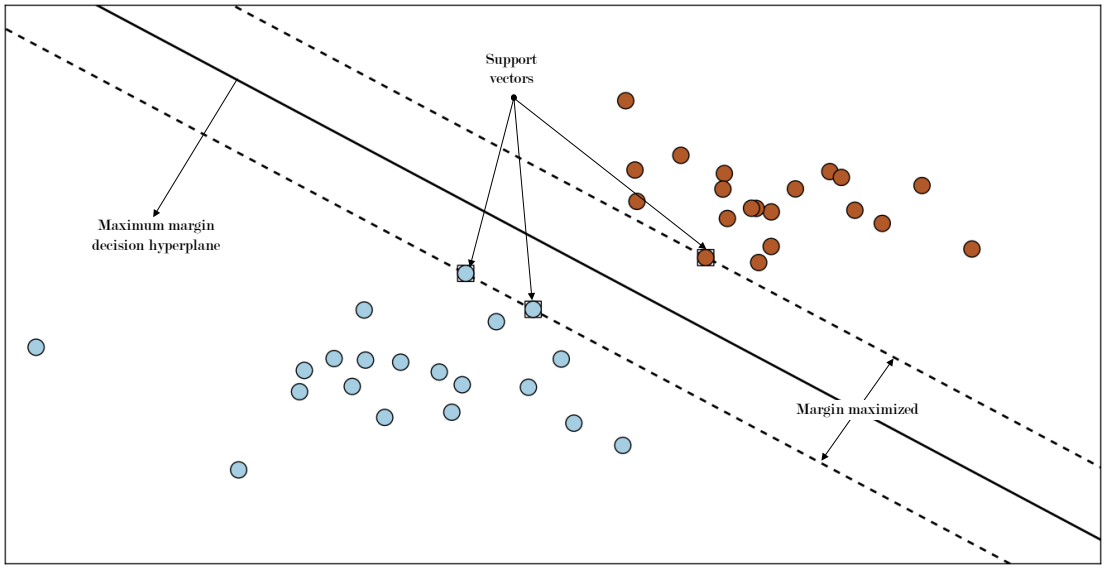


Figure 4.3.1: SVM constructs hyperplane separating the data. The support vectors and the separating hyperplane are appropriately marked in figure.

classifier and choose the class selected by the most classifiers. For detailed overview regarding SVM, refer [MRS08], [DHS12], [RNI95] and [HCL+03]

4.4 CONCLUSION

In this chapter, the basics of bag-of-features approach was described. Bag-of-features is a popular feature representation method widely used for image classification. They represent images as orderless collection of local image features. Chapter 3 discuss in detail about the local image features. Local image features are generated for the training images. These generated features are clustered using Lloyd's algorithm (Subsection 4.2.1) or spherical k-means (Subsection 4.2.2) to generate the visual vocabulary. Based on the generated visual vocabulary, the feature representation for a new image is defined. The classifiers are used to differentiate different patterns in the input data. These classifiers are trained using the generated feature representations. The support vector machine was discussed in detail in Subsection 4.3.1.

5 TEXT DETECTION IN NATURAL SCENE IMAGES

Text detection in natural scene images is a challenging problem because of the unpredictability of the occurrence of text in any part of the image. Over the past decade, researchers have made great improvements in the field of text detection in natural scene images. As described in Chapter 2, for a typical text detection system, feature representation play a pivotal role. Hand-crafted and feature learned representation methods were discussed in Subsection 3.4.1 and Subsection 3.4.2 respectively. Bag-of-features paradigm is one of the widely used feature representation technique which represents an image as an order less collection of local image features. This paradigm is widely applied to generate learned features.

In Section 5.1 the methods proposed to solve the problem of text detection in scene images using learned feature descriptors are discussed. In Section 5.2 the method for text detection using hand-crafted feature descriptors is reviewed. Section 5.3 explains the text detection methods using the region detector Maximally Stable Extremal Regions (MSER).

5.1 TEXT DETECTION USING LEARNED FEATURE REPRESENTATION

Learned feature approaches are characterized by the absence of a-priori selected representation methods. The representation is derived from the input data. Hence, these representations are highly specialized according to the input data. Due to the recent advances in machine learning, learned feature representations have gained a lot of popularity. Two methods discussed in this section use unsupervised feature learning. These methods use the visual vocabulary to define their feature representation and they never store the location of the feature responses. Hence, these methods follow bag-of-features methodology for feature representation.

5.1.1 *Single layered feature representation*

In [CCC+11], a method for text detection using learned features from unlabeled data was proposed. The advantage of learned feature representation from the data is that, the generated features are highly specialized. This method also follows bag-of-

features paradigm for feature representation, as discussed in Chapter 4. The emphasis of this paper is that learned feature representation can achieve the state-of-the-art performance in scene text detection. The feature representation is defined using the generated visual vocabulary. The process of creating the visual vocabulary involves the use of spherical k-means clustering algorithm and also does not require any annotations for the data to be clustered, hence this method is defined as feature learning using unsupervised algorithm. Based on the learned feature representation, the training samples are transformed into features. A supervised classifier learns to discriminate text and non-text regions using the generated features. The proposed method is described in detail.

The primary step for any text detection system is to properly define text and non-text samples from the training dataset. In this paper, a definition of the text and non-text region is formulated. Gray scale image patches of size 32-by-32 are extracted from the training images and each of these image patches is labeled as text or non-text, based on its overlap with the character bounding boxes in the ground truth of the training images. The ground truth contains the location and the size of text regions in the training images. A 32-by-32 gray image patch is labeled as text region, only if it satisfies the following conditions:

1. 80% of the image patch region is within the text region defined by the ground truth.

2. Character bounding box width or height is within 30% of the width or height of the image patch region respectively i.e. the character bounding box width or height should be in the range of $0.7 * 32$ to $1.3 * 32$, where 32 is the height and width of the image patch considered for training the classifier.

The 32-by-32 gray scale image patches that do not satisfy the above-mentioned conditions are labeled as non-text regions. The above logic detects characters of size similar to the image patch considered which is 32 in the current scenario. The text regions are regarded as positive samples and non-text regions as negative samples. Hence, the training dataset are generated from the training images based on the above logic. The training dataset and the validation dataset are generated from the training images. Both these datasets consist of gray scale images of size 32-by-32. Figure 5.1.1 shows a few examples of text and non-text samples.

Once the training dataset is generated, a representation needs to be defined for each of these 32-by-32 image patches. Analogously, for any bag-of-feature method, the primary step is to generate the visual vocabulary or dictionary. Even though the patches for dictionary creation are extracted only from text samples, these patches do

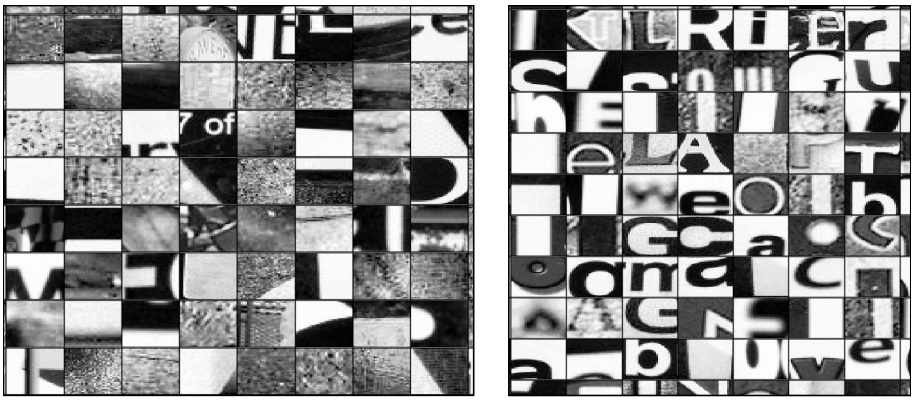


Figure 5.1.1: The figure on the left shows few examples of non-text samples and the figure on right shows some examples of text samples. These patches are used for training the classifier. All the training patches are of size 32-by-32.

not require any annotations for the subsequent clustering process using spherical k-means. Hence the feature learning process explained in this section uses unsupervised learning techniques. The steps for creating the visual vocabulary is enumerated below and also in Figure 5.1.2.

1. A set of m image patches of size 8-by-8 is extracted from the text samples in the training data. Each 8-by-8 patch is regarded as 64-dimensional vector $(8 * 8 = 64)$ where each element corresponds to the pixel intensity value of an 8-by-8 image patch. Thus, a set of m vectors of $\tilde{\mathbf{x}}^{(i)} \in \mathbb{R}^{64}, i = 1, ..., m$ is obtained.

2. Each vector $\tilde{\mathbf{x}}^{(i)}$, is brightness and contrast normalized by subtracting the mean and diving by the standard deviation of all the 64 pixel values (Subsection 3.1.1). A small offset is added to the variance before division to avoid division by zero which is equal to 10 ($\epsilon_{norm} = 10$). The value is selected based on the findings from the paper [CN12]. The m normalized vectors are arranged column-wise.

$$\hat{\mathbf{x}}^{(i)} = \frac{\tilde{\mathbf{x}}^{(i)} - mean(\tilde{\mathbf{x}}^{(i)})}{\sqrt{var(\tilde{\mathbf{x}}^{(i)}) + \epsilon_{norm}}} \tag{5.1.1}$$

3. The normalized vectors $\hat{\mathbf{x}}^{(i)}$ are ZCA whitened to obtain $\mathbf{x}^{(i)}$ as explained in Subsection 3.1.2. Thus, m whitened vectors of dimension 64 are obtained. Since 8-by-8 image patches are used to create the visual vocabulary, the value for whitening constant is chosen as $\epsilon_{zca} = 0.1$ as per the findings from [CN12]. Do note that the

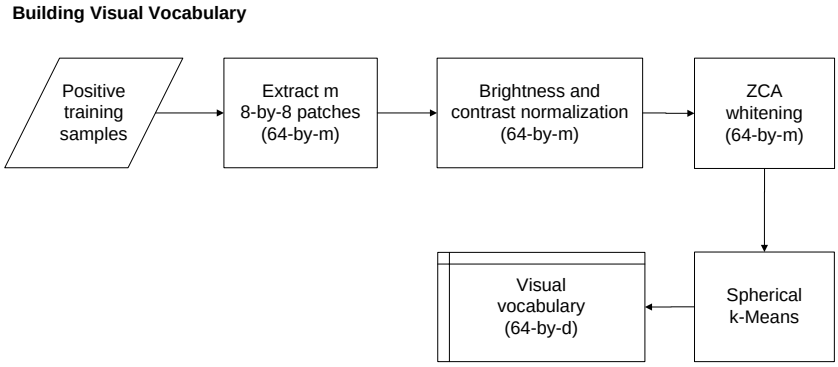


Figure 5.1.2: Generation of the visual vocabulary: Extract m samples of 8-by-8 patches from the training samples labeled as text. Each extracted patch is represented by 64-dimensional vector by using all the intensity values in the patch. Each patch is brightness and contrast normalized. Subsequently, after normalization ZCA whitening is performed. All the m whitened vectors are clustered using spherical k-means to obtain the visual vocabulary. The visual vocabulary is of size 64-by-d, where d is the number of clusters.

whitening transform matrix is stored. This whitening transform matrix will be used to whiten the 64-dimensional brightness and contrast normalized vectors extracted from a test image.

4. The spherical k-means algorithm (Subsection 4.2.2) uses the ZCA whitened vectors ($\mathbf{x}^{(i)}$) to generate the visual vocabulary. The one hot encoding or the code vectors generated during the visual vocabulary creation is discussed in detail in Subsection 4.2.2.The visual vocabulary is denoted by $D \in \mathbb{R}^{64 \times d}$, where d is the number of clusters. Each vector in the visual vocabulary is normalized to unit length. Each column of D corresponds to a centroid or a visual word.

Based on the generated visual vocabulary, the feature representation for an image patch is defined. Consider a gray scale image patch of size 32-by-32, extract all the 8-by-8 patches as the visual vocabulary is built using 8-by-8 patches. All the extracted patches are processed using the same pre-processing algorithms used during the visual vocabulary creation. Hence each 8-by-8 patch is brightness and contrast normalized and then whitened using the whitening matrix saved during dictionary creation. The resultant vector is denoted by $\mathbf{x}$. The projection or inner product between whitened

vector ($\mathbf{x}$) and each visual word (each column of D) is encoded into a feature vector as per the Equation 5.1.2.

$$\mathbf{z} = max\{0, |D^T\mathbf{x}| - 0.5\} \tag{5.1.2}$$

where D is the learned visual vocabulary in which each visual word is arranged column-wise, $\mathbf{x}$ is the 64-dimensional column vector obtained after whitening and normalization of each 8-by-8 patch extracted from 32-by-32 and $\mathbf{z}$ is the feature representation for the 8-by-8 patch under consideration.

During this step, the dimension of vector is modified from 64 to d where d is the number of visual words in the vocabulary. If the angle between the two vectors is small, then the projection value will be higher and if the angle between two vectors is higher, then the projection value will be lower. Hence, all the vectors lying in similar direction as the visual words in the vocabulary will give higher values. For details, refer Subsection 4.2.2.

Hence, a 32-by-32 image patch yields a feature representation of dimension 25-by-25-by-d. Spatial pooling is widely used technique in computer vision to reduce the dimensionality of an obtained representation. In this technique, the feature responses of multiple locations are combined into a single feature response [BBLP10]. Figure 5.1.4 depicts the process of average pooling. Here, the 25-by-25-by-d array is divided into a 3-by-3 grid and all the feature responses in a single grid are summed up to generate d-dimensional feature vector for each grid. Thus, for a 32-by-32 image, the final feature vector is $9d$-dimensional. Similar to any bag-of-feature representations, this generated final feature vector does not store spatial location of the feature responses.

The generated training and validation dataset are converted to features using the feature extraction method described above. The training dataset consists of 30000 samples labeled as text and 30000 samples labeled as non-text. Each sample is transformed into a feature vector of dimension $9d$. The training data is used to train the linear Support Vector Machine (SVM). The SVM generates a hyperplane separating the text and non-text samples. The regularization parameter of the linear SVM is determined through cross-validation as mentioned in [HCL+03]. The classifier is capable of predicting if the given 32-by-32 image patch contains text or not. The SVM is trained to predict a positive score if it finds the input patch to be text and it will predict a negative score if it finds the input patch to be non-text.

The performance of the trained classifier is evaluated by using it to differentiate text and non-text regions in an unseen image (image from testing dataset). The trained classifier is evaluated over the test image in sliding window fashion as shown in Figure 5.1.5. In [CCC+11], the window size is 32-by-32 and the stride at which the window is moving is equal to one. Coates et al. extract all the 32-by-32 image patches

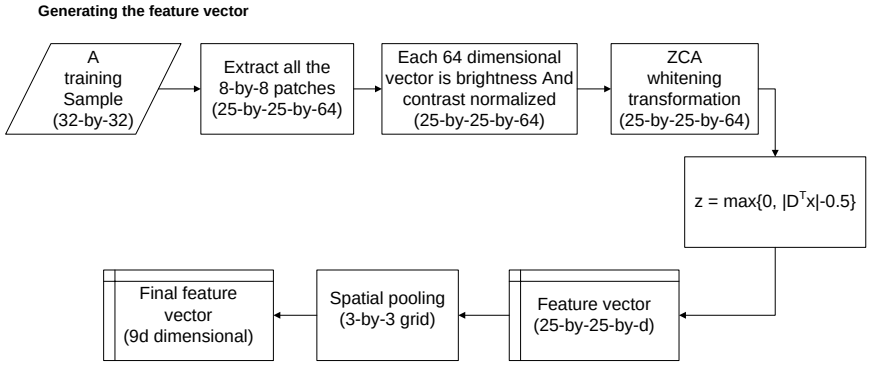


Figure 5.1.3: First, extract all the 8-by-8 patches from 32-by-32 image patch. Each 64-dimensional vector is normalized and ZCA whitened. Each of these whitened vectors is projected onto each visual word resulting in 25-by-25-by-d array. This feature vector is spatially pooled over 3-by-3 grid to reduce the dimension. The final feature vector is 9d-dimensional, where d is the number of visual words in the vocabulary.

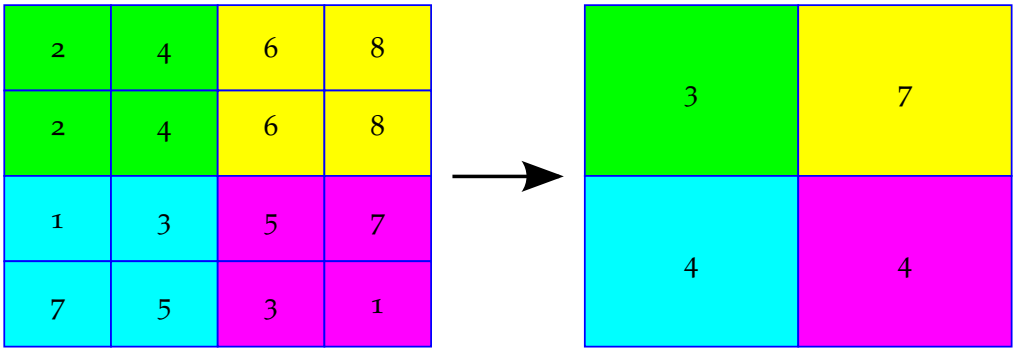


Figure 5.1.4: The figure shows how the feature responses shown on the left is combined spatially to generate pooled feature responses shown on the right.

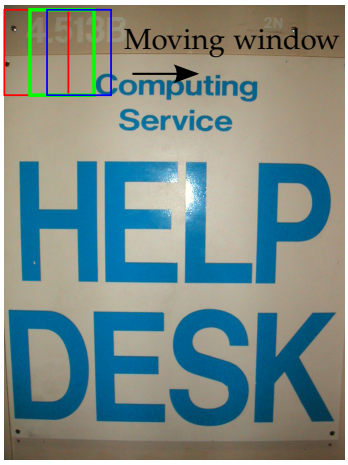


Figure 5.1.5: The trained classifier is evaluated in sliding window fashion. Size of the window is 32-by-32. The stride at which window moves can be configured. The image is taken from ICDAR 2003 dataset [LPS+03].

from the test image. The extracted image patches are converted into features of 9d dimension. The features are given to the classifier and the classifier generates a text detector score for each 32-by-32 image patch. Score depends on the distance of feature vector from the hyperplane. The text detector score will be positive if the SVM predicts the input as text and it will be negative otherwise.

The SVM is only trained to detect text of sizes similar to 32-by-32. However, in the real world the size of text is unpredictable. Hence, the classifier is unable to detect text of different sizes if evaluated on a single scale. In order to solve this inability, the test image is rescaled to different scales and then the classifier is evaluated in sliding window fashion on the rescaled image. Coates et al. have used 13 different scales, ranging from 0.1 times to 1.5 times the size of the original image. For e.g., a 32-by-32 sliding window over a rescaled image of 0.8 scale is equivalent to a 40-by-40 sliding window over the original image. At each scale, the method produces a 2-dimensional array of text detector scores. The size of the array depends on the stride at which the sliding window moves. Consider a rescaled image of size M-by-N, a sliding window of size 32-by-32 and a stride equal to one, then the output of the sliding window operation will be a 2-dimensional array of text detector scores. Each text detector score in the array represents the prediction for a region in the original image. The size of the region depends on the scale under consideration. For e.g., each text detector score corresponds to a 32-by-32 region at scale 1.0 and corresponds to a 64-by-64 region at scale 0.5. Hence, the prediction for the image at each scale is generated and finally the

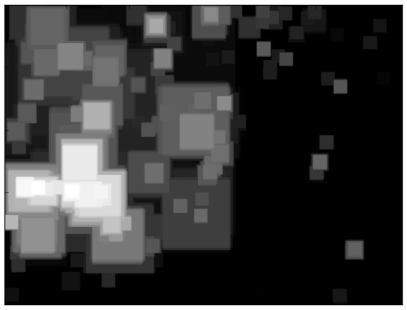


Figure 5.1.6: The above figure shows a sample output. The 'white' regions denote the text areas and the 'black' regions denote the non-text areas. The white rectangular boxes of different sizes represent the prediction from different scales.

maximum text detector score across all the scales is taken to obtain the final per-pixel prediction for the image. Figure 5.1.6 shows a sample output from the text detector algorithm proposed by Coates et al.

5.1.2 *Multi layered feature representation*

Wang et al. [WWCN12] also propose a text detection method based on learned features. In this method, the training data generation and visual vocabulary generation are the same as [CCC+11] which is explained in detail in Subsection 5.1.1. The visual vocabulary is generated by using spherical k-means on 8-by-8 normalized and ZCA whitened image patches. The algorithm uses 8-by-8 grayscale image patches for creation of the visual vocabulary and 32-by-32 grayscale image patches for text detection. For text detection, Wang et al. use 96 visual words in the visual vocabulary.

With the generated visual vocabulary, the representation for a new 32-by-32 image patch is defined as follows:

- All the 8-by-8 image patches are extracted and each image patch is converted into a vector of 64-dimension. Each vector is brightness and contrast normalized and then ZCA whitened. The resultant vector ($\mathbf{x}$) is encoded to feature as per the Equation 5.1.2. Until this step the process followed by Wang et al. is the same as the method proposed by Coates et al. in [CCC+11]. Spatial pooling technique [BBLP10] is used to reduce the dimension of the resultant 25-by-25-by-96 matrix. Wang et al. use a grid of 5-by-5 instead of 3-by-3 grid used by Coates et al.. After spatial pooling, the yielded feature array is of dimension 5-by-5-by-96. Then another layer of convolution and average pooling is added on top of the outputs

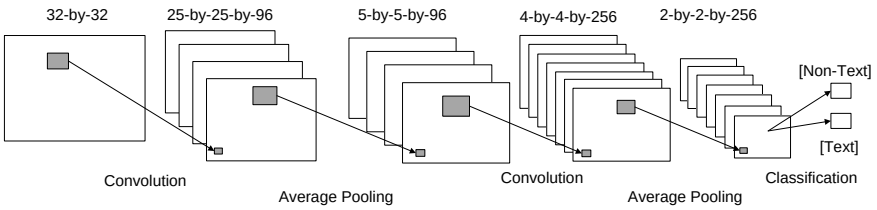


Figure 5.1.7: Primarily a 32-by-32 grayscale image is convolved with 96 filters of size 8-by-8 yielding 25-by-25-by-96 feature array. This feature array is spatially pooled over 5-by-5 grid to generate a feature array of size 5-by-5-by-96. Again, this is convolved with 2-by-2-by-256 filters to generate 4-by-4-by-256 feature array. The resultant feature array is again pooled spatially over 2-by-2 grid to yield the final feature representation of size 2-by-2-by-256 for each 32-by-32 grayscale image. Image based on [WWCN12].

from the previous layer. In this layer, each of 5-by-5-by-96 feature arrays yielded from the first layer, is convolved with 2-by-2-by-256 filters to generate a feature array of dimension 4-by-4-by-256. Again, a spatial pooling is performed over 2-by-2 grid to yield a feature array of dimension 2-by-2-by-256. The process is represented graphically in Figure 5.1.7. This additional layer of convolution and spatial pooling are the significant differences in the method of Wang et. al from the feature representation proposed by Coates et al. in [CCC+11].

Using the above-mentioned multi-layered feature representation scheme, the training data are converted into features. Using this generated features, a classifier is trained to differentiate text and non-text regions. The classifier is evaluated in sliding window fashion as explained in Subsection 5.1.1. Wang et al. evaluation method is similar to Coates et al. [CCC+11]. Hence, a classifier trained on multi-layered features is yielded to differentiate text and non-text regions.

5.2 TEXT DETECTION USING HAND-CRAFTED FEATURE REPRESENTATION

In the previous section, various methods for text detection using learned feature representations were reviewed. In this section a text detection algorithm using a hand-crafted feature representation will be reviewed. One of the most widely used hand-crafted feature representations is the scale invariant feature transform (SIFT) descriptor proposed by Lowe [Low04]. SIFT descriptor consists of histogram of oriented gradients. The descriptor registers the count of oriented gradients in a particular range of directions and also does not store the location information. When the process of

generating the SIFT descriptor is compared with bag-of-features method, it has certain similarities and dissimilarities. In bag-of-features representation, local image features are generated from the training data and the generated descriptors are vector quantized to generate the visual vocabulary but the concept of clustering the calculated oriented gradients to form representatives is absent in SIFT descriptor calculation. In BoF method, term vector is calculated by counting the number of visual words present in the image. In SIFT descriptor, the number of oriented gradients in particular directions is calculated and the obtained values form the final descriptor. Hence both BoF descriptors and SIFT descriptor follows similar representation methods. Additionally, both these descriptors do not store the spatial location of the feature responses. In BoF, features are calculated for a local neighborhood. Similarly while calculating the SIFT descriptor, the region is subdivided into 4-by-4 grid and each grid can be considered as a local patch. The histogram of oriented gradients are calculated for each grid. Hence, both these descriptors use information from local neighborhood for representation. On account of these similarities, the SIFT descriptor can be considered as a bag-of-feature descriptor.

5.2.1 *Text detection using SIFT descriptor*

In this section no literature has been reviewed. The following method is designed as the control method to benchmark the performance of learned features in text detection. Since the SIFT descriptor is the most popular hand-crafted descriptor [MS05], it is the natural choice to design the text detection system using hand-crafted features with SIFT descriptor. In this method, the training data generation from the training images are the same as [CCC+11] which is explained in Subsection 5.1.1. The details about SIFT descriptor are reviewed in Subsection 3.4.1. Hence, all the samples from the training and validation dataset are converted into 128-dimensional SIFT descriptor. The generated features from the training data are used to train a linear SVM (Subsection 4.3.1) to identify text and non-text regions.

Similar to all the methods reviewed above, the classifier is evaluated in the sliding window fashion as explained in Subsection 5.1.1. This method is used to benchmark the performance of the learned feature representations with hand-crafted feature representations.

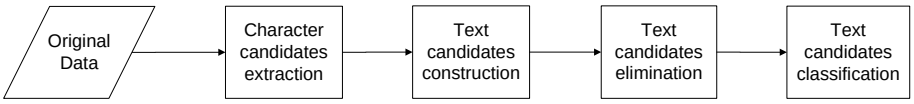


Figure 5.3.1: Flowchart for text detection system using MSER. Image based on [YYHH14].

5.3 TEXT DETECTION USING MAXIMALLY STABLE EXTREMAL REGIONS

The maximally stable extremal regions is a region detector proposed by Matas et al. [MCUP04]. MSER detection procedure was described in detail in Subsection 3.3.1. Studies have shown that the MSER detector performs well on images containing homogeneous regions with distinctive boundaries [MS04]. Typically, the text regions have uniform intensity values surrounded by contrasting backgrounds. Consequently, the MSER detector is widely used for the character candidate selection [YD14].

This section discusses the text detection methods using the MSER detector. In Subsection 5.3.1, the current state-of-the-art method for text detection that uses MSER for identifying the character candidates is reviewed. In Subsection 5.3.2, the details about a text detection method which harvests the advantages of MSER and feature learning techniques is discussed.

5.3.1 *Text Detection using MSER as character candidates*

In [YYHH14], an algorithm is designed that extracts MSER as character candidates. These character candidates are grouped into text candidates by single-link clustering algorithm. This clustering algorithm is an agglomerative hierarchical clustering algorithm where initially each data point is treated as a cluster. Further, it is successively merged until a single cluster is obtained. Then a text classifier segregates text candidates from non-text candidates. The process is explained in Figure 5.3.1.

Character candidates are extracted using MSER algorithm. The detected regions contain lot of repeating components. The repeated components are removed using an efficient pruning algorithm by minimizing regularized variations. The MSER detector detects multiple extremal regions around the local neighborhood of the character. These multiple extremal regions are called as repeating components. As stated in [MCUP04], an extremal region is considered to be more stable if the variation in area of the region is minimum over a range of intensity levels. A stable MSER remains virtually unchanged over a range of intensity values. The MSER regions with lower variations tend to have sharp-edged borders and have a higher probability to be a character region. However, it is not always possible to select a threshold for variation

to separate all the character regions from non-characters. Hence, the method proposes a variation regularization procedure which penalizes MSER with too large and too small aspect ratios (Equation 5.3.1). During this step, the method determines those regions which have a higher probability to be a character.

$$V = \begin{cases} V + \theta_1(a - a_{max}) & \text{if } a > a_{max}, \\ V + \theta_2(a_{min} - a) & \text{if } a < a_{min}, \\ V & \text{otherwise.} \end{cases} \tag{5.3.1}$$

where V is the variation in area, θ_1 and θ_2 are penalty parameters , a_{max} and a_{min} are the max and min aspect ratios. If the aspect ratio is less that a_{min}, then a penalty term is added to the actual value of variation in area. Similarly, the same procedure is applied to MSER with aspect ratio greater than a_{max}. The variation in area is the metric which denotes the stability of an MSER. Higher the variation in area lesser the stability of the particular MSER. Hence, the stability of MSER with too high or too low aspect ratios reduces.

After the pruning algorithm, generated character candidates are clustered using single-link clustering algorithm. Thus, the character candidates are clustered in a similar fashion to generate the text candidates. For more details about text candidate construction and single-link clustering algorithm, refer to [YYHH14].

Text candidates are differentiated from non-text candidates based on a character classifier. The character classifier is trained using features, text region height and width, aspect ratio, stroke-width features and smoothness. Smoothness is defined as the average difference of gradient directions of adjacent boundary pixels. Stroke width features consists of variance and mean of character stroke widths. With the trained character classifier, the posterior probabilities of all the text candidate regions are generated. The text candidate regions with low probabilities will be eliminated.

Final text candidates are identified using an Adaptive Boosting (AdaBoost) classifier. Adaboost classifier was proposed by Yoav Freund and Robert Shapire. It generates a strong classifier from a set of weak classifiers. Initially, a set of classifiers are selected and the training data points are assigned same weight. Starting with a selected classifier from the set and perform the classification process and the weights of misclassified examples are adjusted. Then, in every m^{th} iteration a classifier is selected from the set. The process yields a group of classifiers which can perform better classification. Such an ensemble of classifier is trained to generate a model to predict the text candidate to

be a true text or not in the paper review in this section. Since adaboost classifier is not used in this thesis, it is not described in detail. For details, refer [FS97].

The method proposed in this paper has achieved state-of-the-art performance on a variety of public databases such as ICDAR 2003 [LPS+03], ICDAR 2011 [SSD11] and street view database [EOW10].

5.3.2 *Text Detection using MSER and learned feature descriptor*

In [HQT14], a method is proposed for text detection by combining the MSER detector and feature learning approach. From each image, MSER components are extracted and these extracted components are represented using unsupervised feature learning technique. In standard sliding window method, to detect text of different sizes, the image is rescaled to different scales and then a trained classifier is used to predict the presence or absence of text in the rescaled image. This method proposes that instead of scanning the whole image, only the detected MSER regions are scanned. Hence, the advantage of this method is that it reduces the computational complexity of the standard sliding window technique.

Primarily, MSER components are detected from the image. The MSER detector is configured to a lower threshold so that none of the text characters are missed during this process. The reason is that, if the characters are missed, it is impossible to recover them in the subsequent process.

MSER components with the minuscule number of pixels (e.g. 0.01% of the total number of pixels in an image) are discarded. Then the retained MSER component's aspect ratio is computed by fitting a bounding box around it. If the width of the bounding box around the MSER component is larger than the height, that particular component is resized to size 32-by-32 as the trained classifier recognizes text and non-text samples of size 32-by-32. If the height of the bounding box around the MSER component is larger than the width, a square patch is extracted with size equal to the height of the box as shown in Figure 5.3.2. This extracted square patch is resized into 32-by-32 and is given to the classifier to predict the text detector score. The classifier would predict the high positive value if it recognizes the input as text, else the negative detector score is predicted. Once the detector scores for all the MSER components in an image are determined, the text regions in the image are detected. Also, this method has shown promising results on public databases such as ICDAR 2003 [LPS+03], ICDAR 2011 [SSD11].

Figure 5.3.2: Patch extraction from detected MSER components. Patches of size 32-by-32 are extracted and given to the classifier to generate the detector scores. Image based on [HQT14].

5.4 CONCLUSION

In this chapter, various text detection systems using learned features (Section 5.1) and hand-crafted features (Section 5.2) were discussed. These methods used sliding window approach to evaluate the trained classifier on testing images. The sliding window evaluated on single scale have the inability to detect text of various sizes inherently. In order to solve this inability, these methods perform the sliding window operation at different scales. The scales used are determined experimentally. Figure 5.1.6 shows a typical output from the text detection methods using sliding window at multiple scales. It is evident that each new scale used will introduce wrong predictions. Hence, if the approximate size of text in the image is determined and the corresponding scaling factor is used to resize the image, this will reduce the false predictions as well as the execution time.

Later on in this chapter, text detection methods using the region detector maximally stable extremal regions (Section 5.3) were reviewed. The reviewed methods show that MSER detector is successful in detecting good character candidates which can easily detect the character sizes. Thus, MSER detector is an excellent choice for detecting character regions.

Instead of running the sliding window at experimentally determined scales, a method using MSER detector to detect the character sizes in the image and choose appropriate scaling factors is proposed. This novel twist to sliding window method has few advantages. It will reduce the computation time and also lessen the number of wrong predictions induced by using more number of scales.

6 TEXT DETECTION WITH DYNAMIC SCALE DETECTION

In Chapter 5, various sliding window methods used for text detection in natural scenes were reviewed. In all these methods, the classifier is trained to detect the presence or absence of text in image patches of fixed dimension (32-by-32). The trained classifier is evaluated with experimentally determined scales for detecting text of different sizes which occurs in natural scenes. From the reviewed literature [YD14], it is inferred that the Maximally Stable Extremal Region (MSER) is a widely popular and highly successful region detector used for text detection. In addition, various approaches using MSER for text detection were reviewed and all these methods have produced state-of-the-art results on various public databases such as ICDAR 2003 [LPS+03] and ICDAR 2011 [SSD11]. Consequently, MSER detector is an excellent option to detect character candidates. From the detected character candidates in an image, the scaling factors used for rescaling images in the sliding window method are determined and theses rescaled images are used for text detection.

In this chapter, a novel idea of integrating MSER detector with the sliding window technique is proposed. In Section 6.1 the MSER detector configuration used to detect the character candidates is described. From these character candidates, character representatives are chosen by using Lloyd's algorithm and the gap statistic as discussed in Section 6.2. Finally Section 6.3, a method to determine the scaling factors used for scaling the image using the dimension of determined character representatives is described.

6.1 CHARACTER CANDIDATE SELECTION USING MSER

MSER detector is suitable for finding stable regions in an image. Since text regions have homogeneous intensity values surrounded by contrasting pixel intensity values, the MSER detector is an excellent choice for detecting text regions. The basics about the MSER detector are explained in detail in Subsection 3.3.1. The detected MSER will have different shapes. In order to determine the size of the MSER, a rectangular bounding box is constructed around the detected MSER as shown in Figure 6.1.1. From the bounding box information, the height and width of the detected MSER are determined.

Figure 6.1.1: Detected maximally stable extremal regions in an image. Each detected region is surrounded by a rectangular bounding box. The height and width of the MSER are determined from the height and width of the rectangular bounding box respectively.

A widely used implementation of MSER detector [Braoo] has the following parameters:

1. **Delta:** Delta is defined as the change in the intensity thresholds while performing the MSER detection. In a grayscale image, the image is segmented at various thresholds and the difference between each threshold is delta. The process of MSER generation is explained in detail in Subsection 3.3.1. For e.g., if delta = 5, then the thresholds chosen will be $0, 5, 10, ...255$. An extremal region is considered stable if the variation in the area at different thresholds is minimal. As delta increases, the change in the area of the detected extremal regions will be higher and hence it will reduce the number of stable extremal regions. Delta can take any value from 0 to 255. A small value for delta (delta = 5) is chosen in order to detect low contrast characters in the image. The effect of variation in delta is shown in Figure 6.1.2.

2. **Minimum Area:** This parameter decides the minimum area of an extremal region. All the regions with the area less than this parameter will be discarded. In the experiments, this value is configured based on the training dataset. From the training dataset, the minimum area of a character is determined and the value of this parameter is chosen accordingly.

3. **Maximum Area:** This parameter decides the maximum area of an extremal region. All the regions with the area greater than this parameter will be discarded.

(a) (b)

(c) (d)

Figure 6.1.2: Effect of variation in delta is illustrated here. The detected MSER are marked with black bounding boxes. In figure (a), delta = 5 and the number of detected MSER = 419 whereas in figure (b), delta = 10 and the number of detected MSER = 253. In figure (c), delta = 30 and the number of detected MSER = 39 whereas in figure (d), delta = 50 and the number of detected MSER = 5. It is evident from the example that as delta increases, the number of detected MSER decreases.

In the experiments, this value is configured based on the training dataset. From the training dataset, the maximum area of a character is determined and that value is chosen for this parameter. Also, from the training data the maximum and minimum aspect ratios for the character regions are determined. All the MSER outside the valid range of aspect ratios are also discarded.

4. **Minimum Diversity:** The detector detects large number of repeating components around an extremal region. Based on this parameter value, the similar repeating components (region which cover similar area in an image) are discarded. When this parameter has a lower value, the number of repeating components on an extremal region will be higher and vice versa. Since a lot of repeating components around an actual character for efficient clustering are needed, a low value of 0.1 is chosen for this parameter. The effect of variation in minimum diversity is illustrated in Figure 6.1.3.

5. **Maximum Variation:** An extremal region is defined as maximally stable if the variation in the area is less than the value specified by this parameter. The value for this parameter ranges from 0.1 to 1.0. When the value of this parameter is lower, then the number of extremal regions becoming maximally stable will be less, and if the value for this parameter is higher, the number of MSER detected will be higher. The value of this parameter is chosen in such a way that even the low contrast character regions are also detected. Hence, a value of 0.2 is chosen for maximum variation. The effect of this parameter is shown in Figure 6.1.4.

   Once the MSER detector parameters are configured, the MSER are detected. These detected regions constitute the character candidates. From these detected character candidates, the size of the characters in the image can be determined. The method for finding the character sizes in the image from the detected MSER is described in Section 6.2.

6.2 SCALE DETECTION USING THE GAP STATISTIC

The bounding box information around each detected MSER provides us with the height and width of the MSER. In this step, the data consisting of width and height of the bounding box is considered as input. The scales are determined from the character representatives found by clustering the input data using Lloyd's algorithm (Subsection 4.2.1). Lloyd's algorithm requires the number of clusters as input. However, in the case of natural scenes, the character sizes are highly unpredictable. Hence, determining the number of clusters based on heuristics is a bad choice. In order

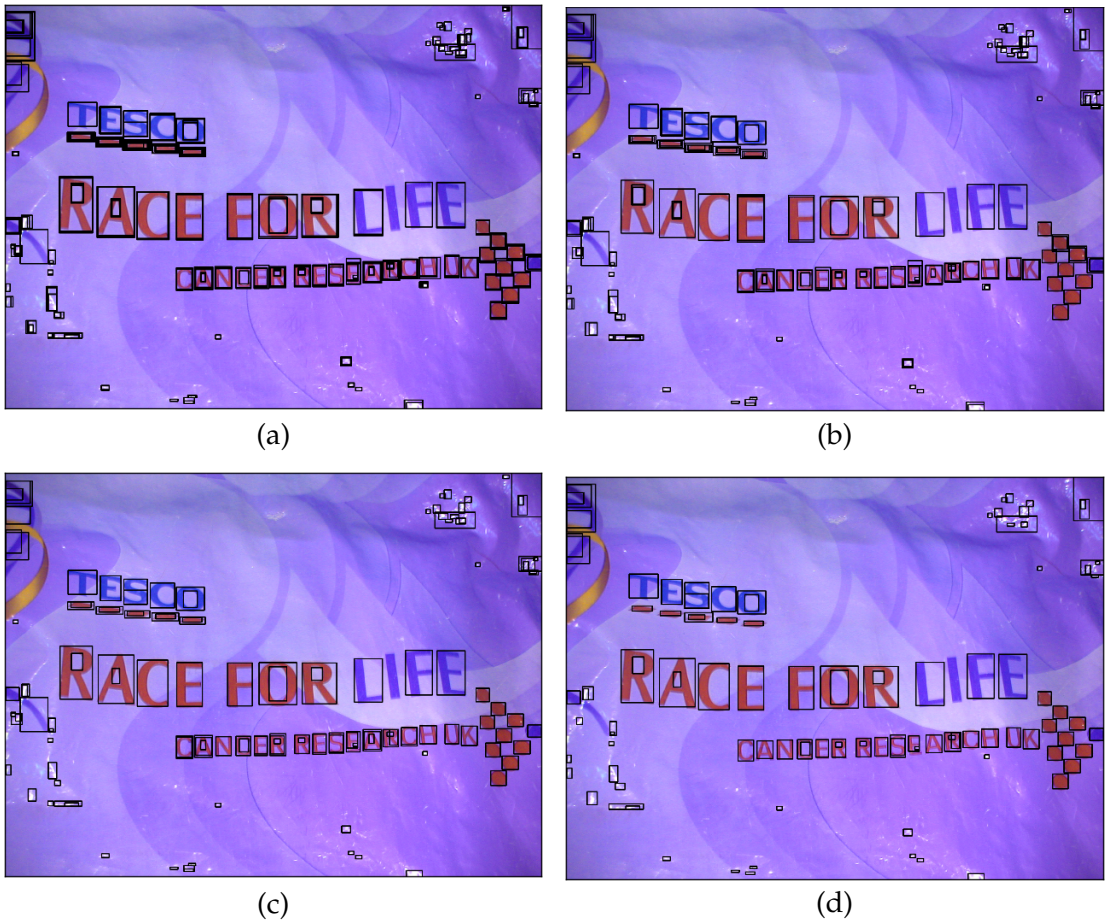


Figure 6.1.3: Effect of variation in maximum diversity is illustrated here. In figure (a), maximum diversity = 0.1 and the number of detected MSER = 389 whereas in figure (b), maximum diversity = 0.3 and the number of detected MSER = 191. In figure (c), maximum diversity = 0.5 and the number of detected MSER = 132 whereas in figure (d), maximum diversity = 0.7 and the number of detected MSER = 107. It is evident from the example that as maximum diversity increases, the number of detected MSER decreases as more similar repeated components are discarded.

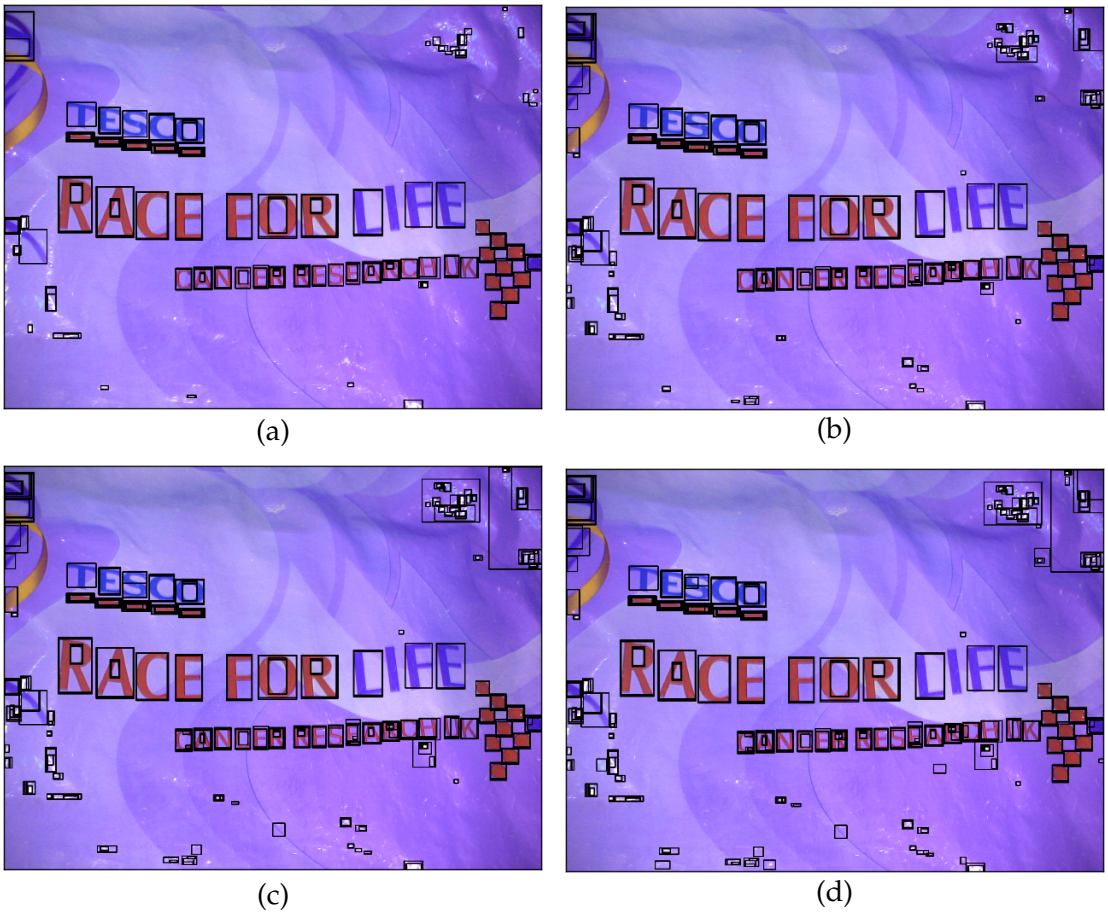


Figure 6.1.4: Effect of change in maximum variation is illustrated here. In figure (a), maximum variation = 0.1 and the number of detected MSER = 277 whereas in figure (b), maximum variation = 0.3 and the number of detected MSER = 437. In figure (c), maximum variation = 0.5 and the number of detected MSER = 478 whereas in figure (d), maximum variation = 0.7 and the number of detected MSER = 500. It is evident from the example that as the maximum variation increases, the number of detected MSER increases as more extremal regions will be classified as maximally stable.

to determine the number of clusters automatically, the gap statistic proposed by Tibshirani et al. [TWH01] is used. According to Tibshirani et al., gap statistic is a good method for identifying well separated clusters. Hence it can give good estimate of character candidate representatives in images with characters of similar sizes and images with character of dissimilar sizes. Also studies have shown that gap statistics is one of the best performing cluster selection algorithm [BHEG01]. Based on these inferences gap statistic is chosen for identifying the number of clusters in data

6.2.1 *The gap statistic*

The gap statistic is one the widely used methods for the automatic selection of the number of clusters in any dataset. This technique is applied to the output of a clustering method such as Lloyd's algorithm. It is designed in such a way that, this method can be applied virtually to any clustering algorithm. The simulations carried out showed that the gap statistic outperform other cluster selection methods [TWH01].

The steps involved in computing the gap statistic is enumerated below:

1. Consider input data x_{ij}, $i = 1, 2, ..., n$, $j = 1, 2, ..., m$, where n is the number of samples and m is the dimension of each sample. Let d_{pq} be the squared Euclidean distance between two observations p and q. Cluster this data with varying the number of clusters $k = 1, 2, ..., K$. Let D_r be the sum of pairwise distances of all points in a cluster, given as:

$$D_r = \sum_{p,q \in C_r} d_{pq} \tag{6.2.1}$$

   where C_r represents the cluster r. Let W_k be the within-dispersion measure defined as

$$W_k = \sum_{r=1}^{k} \frac{1}{2n_r} D_r \tag{6.2.2}$$

   where n_r is the number of data points in cluster r. W_k gives the compactness of the cluster. The value for D_r will be smaller for densely populated cluster and will be higher for sparsely populated clusters. For densely populated well separated cluster, the value of W_k will be lower when compared to clusters of sparsely located data points.

2. The basic idea of the gap statistic is to compare these generated clusters with clusters obtained by using a reference dataset. The reference dataset is generated

by uniform sampling over a range of values as proposed by Tibshirani et al. The range is from the minimum value to the maximum value of the data for a particular dimension. Now generate B reference datasets using uniform sampling and cluster each reference dataset yielding within-dispersion measure W^*_{kb}, $b = 1, 2, ...B$, $k = 1, 2, ...K$. Then the gap statistic is evaluated as follows:

$$Gap(k) = (1/B) \sum_b log(W^*_{kb}) - log(W_k) \tag{6.2.3}$$

3. The within-dispersion measures for the actual dataset W_{kb} and within-dispersion measures for B reference datasets W^*_{kb} are calculated. Since the reference dataset is generated by uniform sampling, the value of W^*_{kb} will be high. At the most suitable number of clusters, the W_{kb} will decrease steeply. In this step, the standard deviation of $log(W^*_{kb})$ is computed.

$$sd_k = [(1/B) \sum_b (log(W^*_{kb}) - \bar{l})^2]^{1/2} \tag{6.2.4}$$

where

$$\bar{l} = (1/B) \sum_b log(W^*_{kb}) \tag{6.2.5}$$

4. Step 3 is performed for all number of clusters $k = 1, 2, ...K$ and then S_k is defined as

$$s_k = sd_k \sqrt{(1 + 1/B)} \tag{6.2.6}$$

5. Select the smallest k such that

$$Gap(k) \geqslant Gap(k+1) - s_{k+1} \tag{6.2.7}$$

After the execution of the above-mentioned steps, the algorithm selects the number of clusters suitable for the input data. Figure 6.2.1 shows how to determine the number of clusters using the gap statistic for an example dataset.

6.2.2 *Process of scale detection*

In the previous section, the method of using gap statistic to determine the number of clusters automatically for an input data was explained. During the process of scale detection, the size of the character candidates is determined in an image using MSER. The steps involved in the process are enumerated below and also Figure 6.2.2 show the outputs obtained after each step for a sample image.

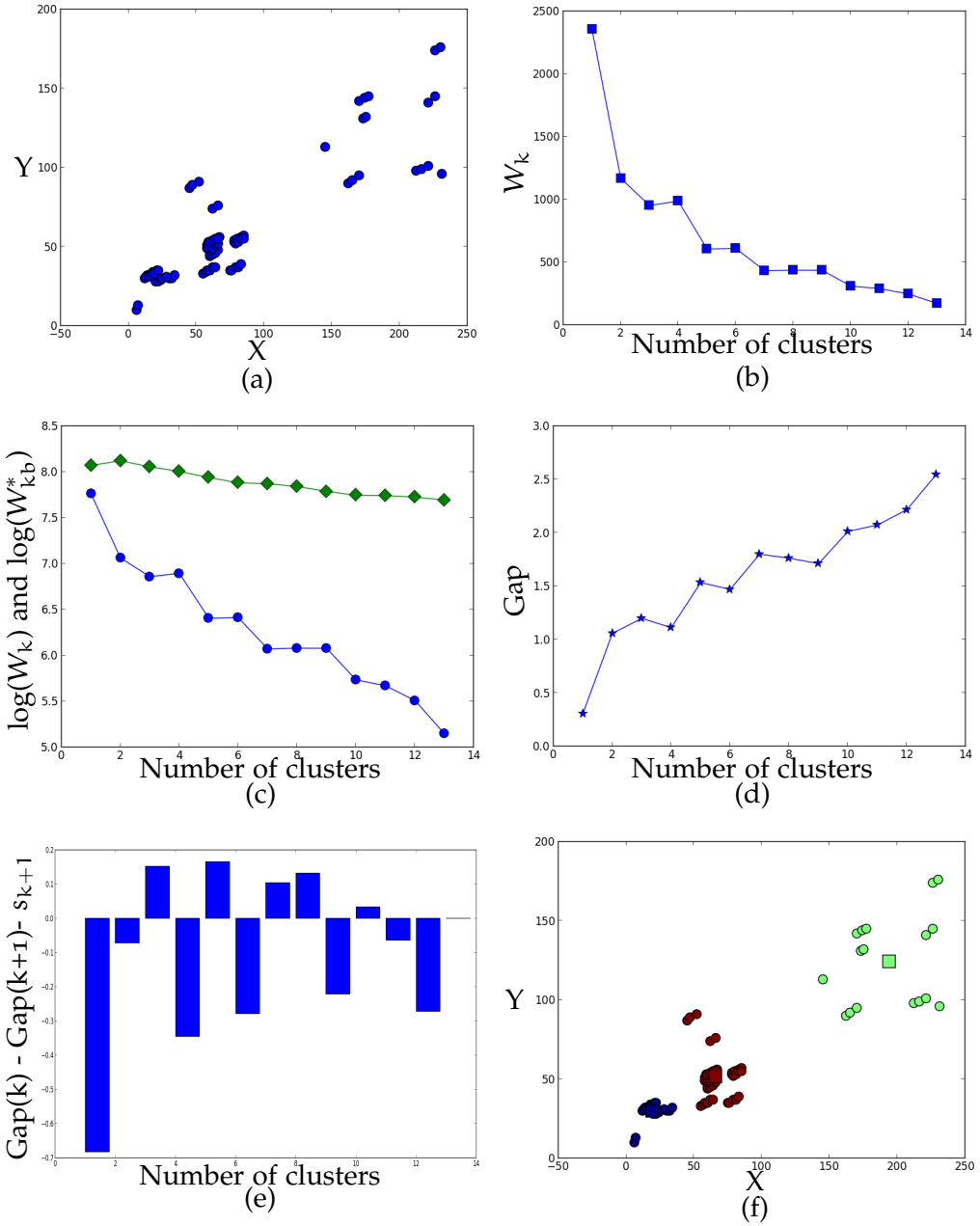


Figure 6.2.1: The figure illustrates various steps in calculating the number of clusters using the gap statistic. Figure (a) shows the data to be clustered. This data are clustered with varying number of clusters starting from 1 to 13. Figure (b) shows the variation of W_k across the number of clusters. The variation of $\log(W_k)$ with respect to the number of clusters is shown by the blue colored line in figure(c) whereas the green colored line shows the variation of $\log(W^*_{kb})$ when the number of clusters is varied. The variation of the gap statistic is depicted in figure (d). Finally, the smallest value for the number of clusters is selected where the metric Gap(k) - Gap(k+1)- s_{k+1} is positive as shown in figure (e). In this example, the value is 3. Figure (f) shows the clustered input data using 3 clusters. The centroids are denoted by a square.

1. The input image is converted into the grayscale image and then given to the configured MSER detector as explained in Section 6.1.
2. The MSER detector detects the character candidates in the input image. Each detected MSER is surrounded by a rectangular bounding box. From the dimension of the bounding box, the size of the MSER is determined.
3. The height and width of all the detected MSER are considered and this data are clustered using Lloyd's algorithm (Subsection 4.2.1) with varying number of clusters starting from 1 to 13. The maximum number of clusters used in the algorithm is chosen to be 13 as the reference method explained in Subsection 5.1.1 uses 13 different scales. The number of input data points is small (in the order of hundreds). Thus, clustering this data multiple times using Lloyd's algorithm is fast. The selection of the best number of clusters is performed using the gap statistic as explained in the previous section.
4. A cluster centroid represents the dimension of the character candidate representatives. Hence, the approximate sizes of the characters is determined in the given input image. Using these character sizes the image scaling factors are determined which is discussed in detail in the next section.

6.3 SLIDING WINDOW USING MSER SCALE DETECTION

The proposed scale detection method described above is integrated into the sliding window method. In Subsection 5.1.1, the reference sliding window method for evaluating the trained classifier on a test image was described in detail. In the reference method, the test image is scaled with scaling factors determined experimentally whereas in this proposed method scaling factors are determined using the MSER detector. The method is described below:

1. Consider an input grayscale image of size M-by-N. The image is given to the MSER detector and the representative character candidates are figured out using the gap statistic as explained in Subsection 6.2.2.
2. Let the character candidate representative of the input image is denoted by (w, h), where w is width and h is the height and the image patches used for the training the classifier is of dimension P-by-R. Then the scaling factors are determined as follows:

$$s_w = \frac{R}{w} \qquad s_h = \frac{P}{h} \tag{6.3.1}$$

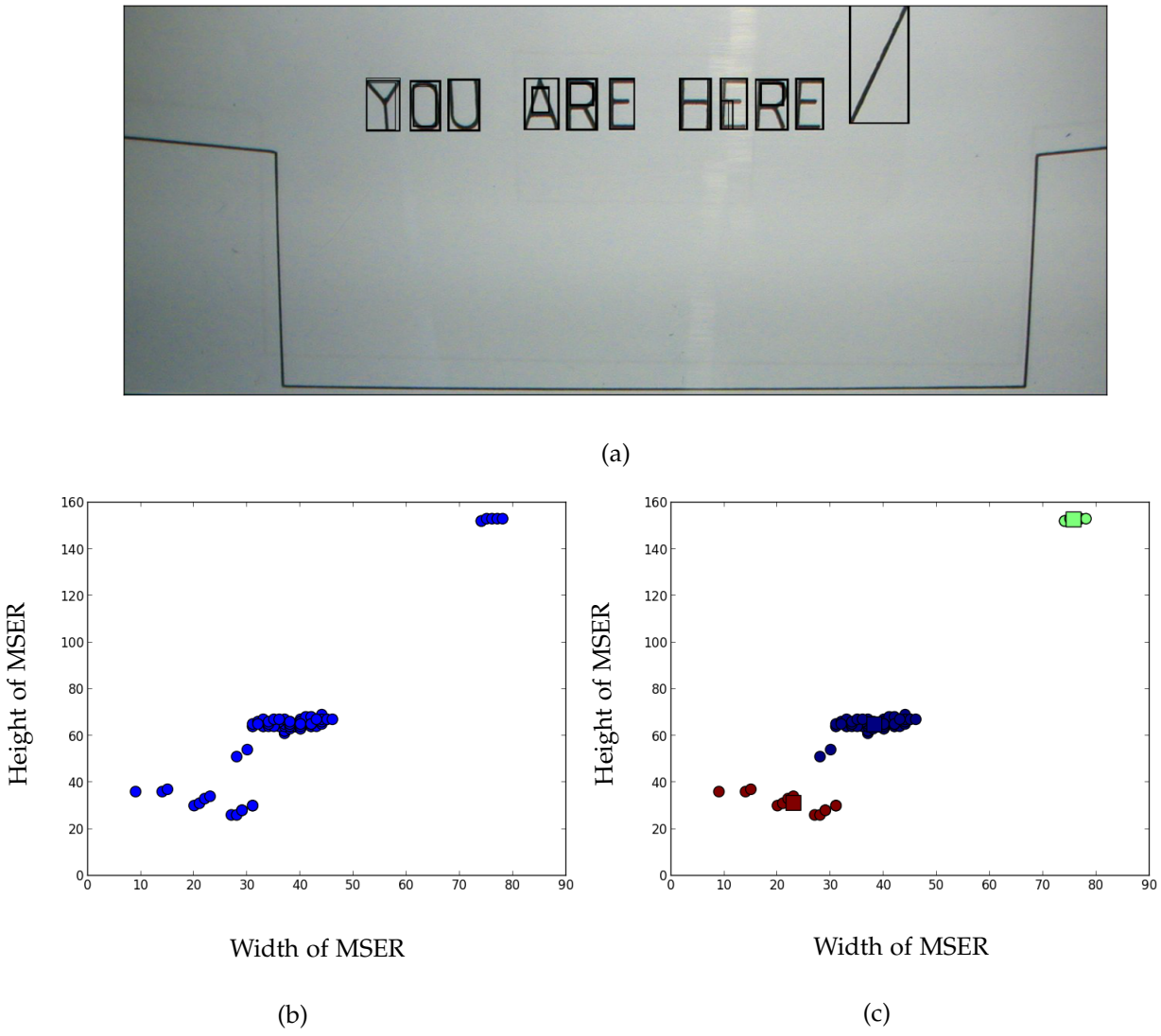


Figure 6.2.2: The figure shows the output obtained after each step during the scale detection process. The input image is given to the configured MSER detector to yield the output as shown in figure (a). The height and width of all the detected MSER are shown in figure (b). This data are clustered using Lloyd's algorithm and the best cluster is chosen using the gap statistic which is 3 in this example. Figure (c) shows the input data clustered into 3 clusters. The centroids provide the size of the representative character candidates.

where s_w and s_h are the scaling factors along the x and y directions. They are determined such that in the rescaled image the character candidates would have sizes which can be detected by the classifier trained on patches of size P-by-R.

3. Using the scaling factors the image is rescaled into an image of size $M * s_h$-by-$N * s_w$. The classifier trained using the proposed methods in Subsection 5.1.1 and Subsection 5.2.1 is evaluated in the sliding window fashion as explained in Subsection 5.1.1.

Thus, this proposed method ensures that the image is resized according to the size of character regions present in the image.

6.4 CONCLUSION

In this chapter, an improvement for the widely popular sliding window method used for text detection was introduced in Section 6.3. The method uses MSER detector to determine the character candidates. The character candidates dimension are clustered using Lloyd's algorithm (Subsection 4.2.1). The number of clusters is determined using gap statistics (Subsection 6.2.1). Hence, the character candidate representatives are obtained. Based on the size of the character candidate, the scaling factor that is used for rescaling the image is determined. The rescaling factor depends on the size of the character found in the image. Thus, the rescaling process is highly specialized according to each image instead of using experimentally determined scale factors for all the test images. The newly proposed method will be evaluated with classifiers trained using learned and hand-crafted feature representations. The evaluation results will be benchmarked with reference methods proposed in Subsection 5.1.1 and Subsection 5.2.1. The evaluation methodology and results will be discussed in Chapter 7.

7 EVALUATION

Text detection systems using learned feature and hand-crafted feature representations have been discussed in Subsection 5.1.1 and Subsection 5.2.1. These methods will be benchmarked against each other to study the performance of different feature representation methods. The text detection system proposed by Coates et al. using the learned feature representation was implemented in Python using Numpy [VDWCV11] and Scipy [JOP14]. The hand-crafted feature representation used in this thesis is the SIFT descriptor, it was calculated with the python bindings of VLfeat library [VF10]. This thesis proposes a novel idea of integrating MSER scale detection in a sliding window approach as described in Section 6.3. The performance of the newly proposed method will be evaluated against the sliding window approach without scale detection. The MSER detector used for scale detection was implemented using OpenCV [Braoo] python bindings.

Section 7.1 gives an overview about the datasets used for evaluation. Afterwards, a description about the experiments conducted in order to evaluate the performance of the text detection systems is given in Section 7.3. These methods are evaluated based on metrics described in Section 7.2. Subsection 7.3.1 explains an experiment conducted to study the performance of the text detection system using learned features. It also evaluates the performance of the system when the number of visual words is increased. Subsection 7.3.2 describes an experiment for comparing the performance of a text detection system with hand-crafted features as well as learned features. Subsection 7.3.3 describes an experiment to compare the performance of newly proposed MSER integrated sliding window method with the standard sliding window method.

7.1 EVALUATION DATASETS

The proposed methods need to be evaluated on public datasets in order to compare the performance with state-of-the art text detection methods. The datasets for text detection are often challenging due to the diversity in the text and intricacy of the background in the images. They consist of images from natural scenes which may contain text with different fonts, colors, sizes and orientations. The image backgrounds may include repeating patterns such as bricks and windows that are hard to distinguish from text.

Each evaluation dataset comprises of training and testing images. The information on the location of text in these images are also included. This information is referred to as ground truth. Depending on the dataset, the ground truth may provide the location of a word or a character or both. The location is specified in terms of bounding box coordinates that enclose a word or a character. Due to gaining popularity of challenges for interpreting the text information in unconstrained images over the past decade, the datasets released for these challenges are widely used to benchmark different text detection methods. These challenges are referred to as robust reading challenges. The text detection methods discussed in this thesis use the character bounding box information from the ground truth for creating the positive and negative training dataset as discussed in Subsection 5.1.1. The methods were evaluated on ICDAR 2003 [LPS+03] and ICDAR 2013 [KSU+13] datasets. Both these datasets provide character level bounding box information.

7.1.1 *ICDAR 2003 dataset*

ICDAR 2003 [LPS+03] dataset was used for the robust reading challenge 2003 edition. The dataset comprises of 258 training images and 251 testing images of scene text. The proposed text detection algorithms are tuned on the training images, and the results are evaluated on testing images. The ground truth of each image in the dataset is provided with a set of particulars that gives information about the text in the image. It describes rectangles surrounding each text in the image in image pixel coordinates. It also annotates the word in each rectangular box. The character bounding box information of each word is also provided in the ground truth as shown in Figure 7.1.1. Since, the reference method described in Subsection 5.1.1 uses ICDAR 2003 dataset for performing the experiments, this thesis used ICDAR 2003 as the primary dataset for evaluation.

7.1.2 *ICDAR 2013 dataset*

ICDAR 2013 [KSU+13] dataset includes 229 training images and 233 testing images of scene text. The ground truth consists of word and character bounding box information. The provided ground truth information is similar to ICDAR 2003 dataset. A sample image from the ICDAR 2013 dataset is shown in Figure 7.1.1. The formulated methods are also evaluated on this dataset.

(a) (b)

Figure 7.1.1: The figure (a) shows a sample image from ICDAR 2003 and figure (b) shows a sample image from ICDAR 2013 datasets. In each image, the character bounding boxes are marked in black. This information is derived from the ground truths provided along with the datasets.

7.2 EVALUATION METRICS

The evaluation metrics used should reflect the effectiveness of proposed text detection systems. The text detection system predicts the input sample as text or non-text. Hence, it can be classified as binary classification problem. In binary, the input samples are classified into two groups based on a classification rule. There are several metrics used to evaluate the performance of a binary classification system such as classification accuracy, precision and recall.

Classification accuracy is defined as the percentage of correctly classified samples. For a text detection system, the classification accuracy yields the effectiveness of the system to differentiate between text and non-text samples accurately. The higher the classification accuracy the better is the classifier.

$$\text{Classification accuracy} = \frac{\text{number of correctly classified samples}}{\text{total number of samples}} \tag{7.2.1}$$

Additionally, binary classification problems are evaluated using the confusion matrix between the actual and the predicted class as shown in Table 7.2.1

In text detection systems, True Positives (TP) are defined as correctly classified text samples and True Negatives (TN) are defined as correctly classified non-text samples. False Positives (FP) and False Negatives (FN) give the number of incorrectly classified text and non-text samples respectively.

	predicted class	
actual class	True Positive (TP)	False Negative (FN)
	False Positive (FP)	True Negative (TN)

Table 7.2.1: Confusion-matrix for a binary classifier.

In binary classification, precision and recall are two popular measures for evaluating the performance of the classifier. Precision and recall are defined as follows:

$$\text{Precision} = \frac{TP}{TP + FP} \tag{7.2.2}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{7.2.3}$$

The sum of TP and FP gives number of samples predicted as text by the classifier whereas the sum of TP and FN gives the number of actual text samples in the ground truth. Precision gives the ratio of the correctly detected text samples and total number of samples predicted as text. Recall gives the ratio of correctly detected text samples and total number of actual text samples in the ground truth. The value of precision and recall ranges from 0 to 1. Generally, a high precision system will have low recall and vice versa. The performance of the classifier is evaluated based on both precision and recall values together. Hence, metric like area under P-R curve is defined to reflect the performance of the classifier using a single value. The AUC measures the performance of a system irrespective of any operating point or a particular threshold. Consider the text detection system used in this thesis, the system outputs a score map for an input image as described in Subsection 5.1.1. The score map consists of floating point values which are predicted by the classifier. In order to calculate the precision and recall these predictions need to be converted into binary decisions. Hence various thresholds ranging from the minimum to maximum score in the score map are considered for this purpose. At each threshold, the scores above the current threshold is given as one and score below the threshold is given as zero. Thus, a binary map is obtained at each threshold and subsequently, the precision and recall values are calculated as per Equation 7.2.2 and Equation 7.2.3 respectively. The process is repeated for different thresholds and precision-recall pairs are obtained for each threshold. After plotting the precision-recall pairs, PR curve is constructed and then the area under this curve

is computed. The area under the curve is calculated by using the trapezoidal rule. The area under the curve is subdivided into trapezoids and the ares of each trapezoid is calculated. Sum of the area of all trapezoids give the area under curve. The AUC value ranges from 0 to 1. The higher the AUC value, the better is the performance of the classifier.

For calculating the area under precision-recall curve for the whole dataset, the following steps are performed:

1. True positives, false positives and false negatives are calculated at fixed thresholds for each image.
2. Execute the step 1 for all the images in the dataset.
3. Sum all the true positives, false positives and false negatives across all the images at each threshold. This step yields a single true positive, false positive and false negative value at each threshold.
4. Precision and recall values are calculated as per Equation 7.2.2 and Equation 7.2.3 respectively at each threshold. The precision-recall pairs are plotted to generate the PR curve. The area under the PR curve is calculated using trapezoidal rule as described above. This is reported as the final metric of performance of the text detection system.

Since the reference text detection method [CCC+11] discussed in Subsection 5.1.1 uses the area under precision-recall curve for evaluating their performance, this thesis also uses the same metric for comparison.

7.3 EXPERIMENTS

The following section provides an overview of the experiments carried out on two different datasets, ICDAR 2003 and ICDAR 2013. In Subsection 7.3.2, a detail discussion of the experiment conducted to compare the performance of hand-crafted and learned feature representation is described. Subsection 7.3.1 shows the effect of increasing the number of visual words in the visual vocabulary of the learned feature representation. In Subsection 7.3.3, the outcome of MSER integrated sliding window approach is examined.

7.3.1 *Learned feature representation*

The basic target of this study is to evaluate the effect of increase in the number of visual words in the vocabulary, on the performance of text detection. The training dataset extracted from the training images is transformed into the feature space using the learned feature representation method. Image patches of size 32-by-32 are considered as input. Initially, the learned features are generated with the visual vocabulary of size 32 and then a linear SVM classifier is trained with these features to differentiate between text and non-text samples. The SVM score depends on the "textiness" of the input sample. It predicts a high score for text patches and low score for non-text patches. The score predicted by the SVM depends on the distance from the hyperplane to the feature vector. The trained classifier is evaluated in a sliding window fashion on rescaled testing images. The scaling factors for rescaling the image is experimentally determined. Using the trained classifier, the predictions for test images are generated. These predictions are used to calculate the AUC as described in Section 7.2. Similar steps are followed with visual vocabulary of size 64 and 200. The text detection system using learned feature was described in detail in Subsection 5.1.1.

Table 7.3.1 shows the classification accuracy and AUC values for ICDAR 2003 and ICDAR 2013 datasets for varying number of visual words. The reference method proposed by Coates et al. [CCC+11], reports that with an increase in the number of visual words, the detector performance increases consistently. They have evaluated the detector's performance on ICDAR 2003 dataset for 32, 64, 200, 500 and 1000 visual words and have achieved the values ranging from 0.5 AUC to 0.62 AUC. Similarly, the detector using the learned features evaluated in this thesis also showed identical behavior on ICDAR 2003 dataset with AUC value of 0.45 for 32 visual words and achieved an increasing pattern for increased number of visual words. The AUC values for 64 and 200 visual words are 0.47 and 0.52 respectively. Due to computational complexity, the evaluation with further increase in the number of visual words was not performed. Since the reference method does not report the exact AUC values for 64 and 200 visual words, the values are not tabulated in Table 7.3.1 for comparison. For further evaluation of the performance of the detector used in this thesis, classification accuracy of the SVM classifier is also reported. Since the reference method does not report the classification accuracy of their detector the values are not tabulated here for comparison. Classification accuracy on ICDAR 2003 dataset for 32, 64 and 200 visual words are 83.3%, 84.3% and 87.2% respectively. Classification accuracy denotes the effectiveness of the detector to differentiate between text and non-text samples accurately. Hence, an increase in the classification accuracy indicates increased learning ability of the classifier due to increase in the number of visual words in visual

vocabulary. Increased learning ability of the detector is proved by the increased AUC values with the increase in number of visual words. Figure 7.3.2 shows the visual vocabulary obtained for 32, 64 and 200 visual words. When the number of visual words is increased, more variable patterns are included in the visual vocabulary. Since these patterns are learned from text patches, the detector can differentiate text from non-text regions more efficiently.

The system was also evaluated on ICDAR 2013 dataset. The classification accuracy obtained on ICDAR 2013 dataset for 32, 64 and 200 visual words are 77.2%, 78.1% and 82.4% respectively. The AUC values also increased from 0.36 to 0.41 when the number of visual words were increased from 32 to 200. Similar to ICDAR 2003 dataset, both classification accuracy and AUC showed an increasing behavior with increase in the number of visual words. These results also proves that the detector performance increases when the number of visual words are increased. But when the results of ICDAR 2013 is compared with ICDAR 2003 the performance of the detector is comparatively less. The parameters for the training data generation is tuned for ICDAR 2003 dataset and the same algorithm is used on ICDAR 2013 dataset in oder to evaluate the effect of training data generation algorithm on different datasets. The reason for the lesser AUC values on ICDAR 2013 implies that using a specialized training data generation method for each dataset might have resulted in higher values. Although, the value is less, the fact to be emphasized here is that the behavior is similar to ICDAR 2003.

Figure 7.3.1 plots the precision-recall curves for varying number of visual words for ICDAR 2003 and ICDAR 2013 datasets. Figure(a) depicts the PR curves for ICDAR 2003 dataset. It is evident from the figure that as the number of visual words is increased from 32 to 200 the area under the curve also increased. Similar increasing behavior is also observed on ICDAR 2013 dataset for similar number of visual words which is shown in figure (b).

The number of visual words in visual vocabulary should be selected in such a way that the maximum performance is extracted from the feature learning method without incurring huge computational cost. From the results of this experiment it is inferred that 200 visual words are a good starting point for number of visual words and later the vocabulary size can be tuned according to the performance of the detector.

7.3.2 *Hand-crafted and learned feature representations*

The primary objective of this experiment is to compare the performance of a text detection system using the learned feature representation proposed by Coates et al.

No. of visual words	ICDAR 2003		ICDAR 2013	
	Classification accuracy	AUC	Classification accuracy	AUC
32	83.3%	0.45	77.2%	0.36
64	84.3%	0.47	78.1%	0.38
200	87.2%	0.52	82.4%	0.41

Table 7.3.1: Results for the performance of learned features with varying number of visual words on ICDAR 2003 and ICDAR 2013.

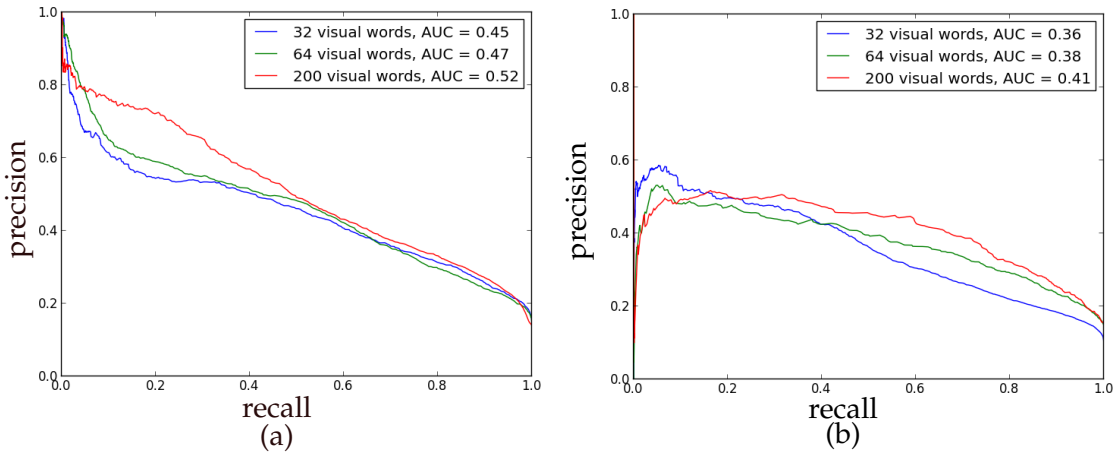
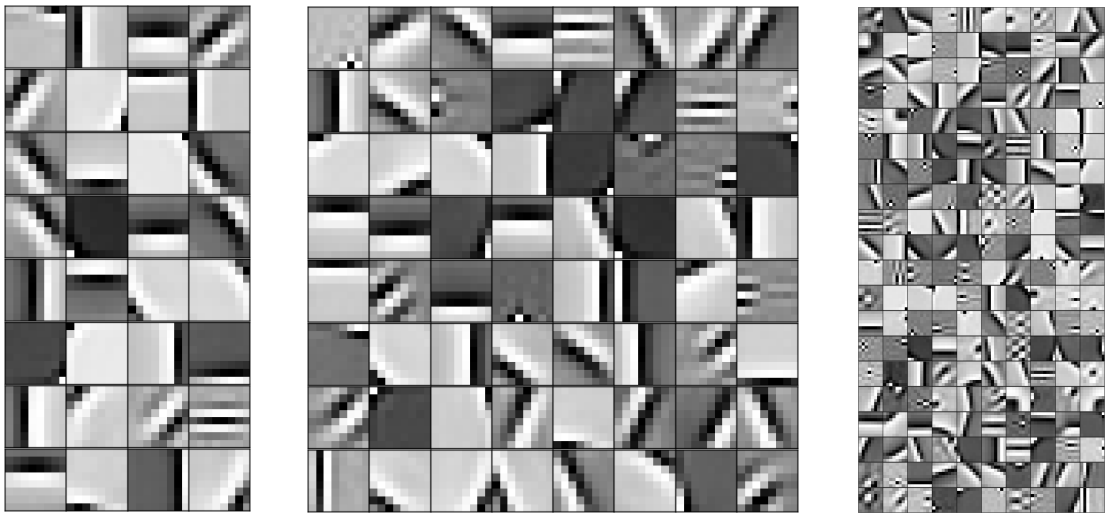


Figure 7.3.1: Figure (a) shows the precision-recall curve for varying number of visual words on ICDAR 2003 dataset whereas figure(b) depicts the PR curves obtained on ICDAR 2013 when the number of visual words is increased. Higher AUC denotes better performance. The classifier using learned features with 200 visual words in the vocabulary has reported the highest performance on both datasets.



(a) (b) (c)

Figure 7.3.2: The figure(a), (b) and (c) depicts the visual vocabulary obtained for 32, 64 and 200 visual words respectively.

(Subsection 5.1.1) and text detection system using hand-crafted feature representation (Subsection 5.2.1). The experimental setup is similar to previous experiment. The training dataset extracted from the training images is transformed into the feature space using the selected feature representation. In both scenarios, image patches of size 32-by-32 are considered as input. In hand-crafted feature representation approach, SIFT descriptor is used and in learned feature representation method, the descriptor representation learned using the visual vocabulary generated by spherical k-means is used. For learned feature representation in this experiment, 200 visual words are chosen in the visual vocabulary since it reported the best performance in the previous experiment.

In ICDAR 2003 dataset, the classifier trained using SIFT descriptor yields a classification accuracy of 73.4% whereas the classifier trained using learned feature representation with 200 visual words resulted in a classification accuracy of 87.2%. These classifiers were evaluated on all testing images and the precision-recall pairs at defined thresholds were calculated. The area under precision-recall curve was calculated as 0.39 and 0.52 for hand-crafted and learned feature representations respectively. Similarly, the experiments were also carried out on ICDAR 2013 dataset. The classification accuracy for hand-crafted and learned feature representation approach on ICDAR

Feature representation	ICDAR 2003		ICDAR 2013	
	Classification accuracy	AUC	Classification accuracy	AUC
Hand-crafted (SIFT descriptor)	73.4%	0.39	70.5%	0.34
Learned (200 visual words)	87.2%	0.52	82.4%	0.41

Table 7.3.2: Results for the performance of hand-crafted and learned feature representation methods on ICDAR 2003 and ICDAR 2013.

2013 was calculated as 70.5% and 82.4% respectively. The AUC values for the same was calculated as 0.34 and 0.41 respectively. The results are shown in Table 7.3.2.

From the results obtained, it is quite evident that the learned feature representation method outperforms the hand-crafted feature representation method. The higher classification accuracy for the learned features indicates that it can discriminate between text and non-text samples more efficiently than the SIFT descriptor. The better AUC value is obtained for learned features on the test dataset indicate that the ability of the detector to classify unseen data accurately. Hence, the above results prove that the learned features are more specialized to the data than the hand-crafted features.

The PR curves for ICDAR 2003 and ICDAR 2013 are shown in Figure 7.3.3. Figure(a) depicts the PR curves for ICDAR 2003 dataset. It is clear from the figure that there is a considerable difference in the AUC values of hand-crafted and learned feature representation method implying that learned features can discriminate between text and non-text samples more accurately. Similar increasing behavior is also observed on ICDAR 2013 dataset as shown in figure (b).

The results of this experiment indicate that the text detection system using learned feature representation method is superior to systems using the SIFT descriptor. These results also show the adaptive nature of learned features. These features are highly specialized to the input data. In [NWC+11] Netzer et al. have evaluated the performance of learned features against the hand-crafted features with respect to digit recognition in natural scenes. The results reported in this experiment are in conformity with the results of Netzer et al.

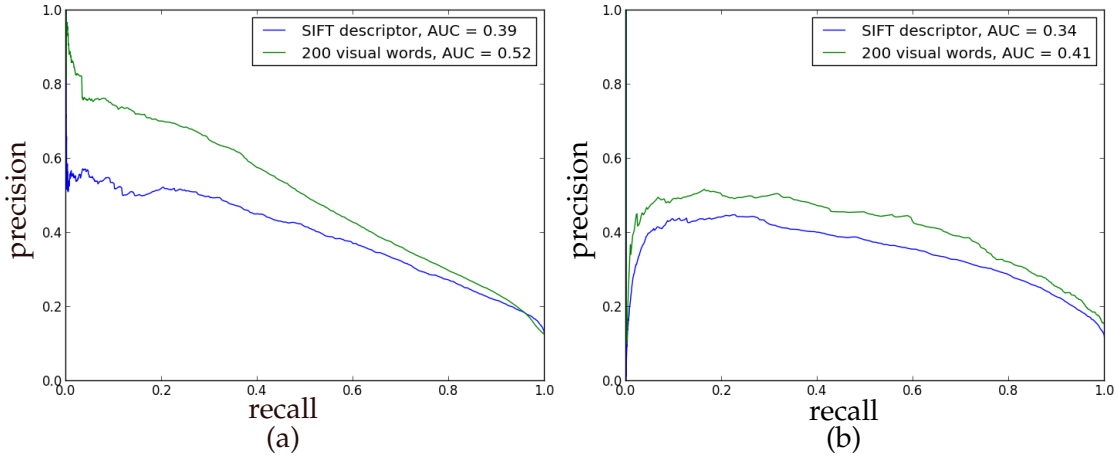
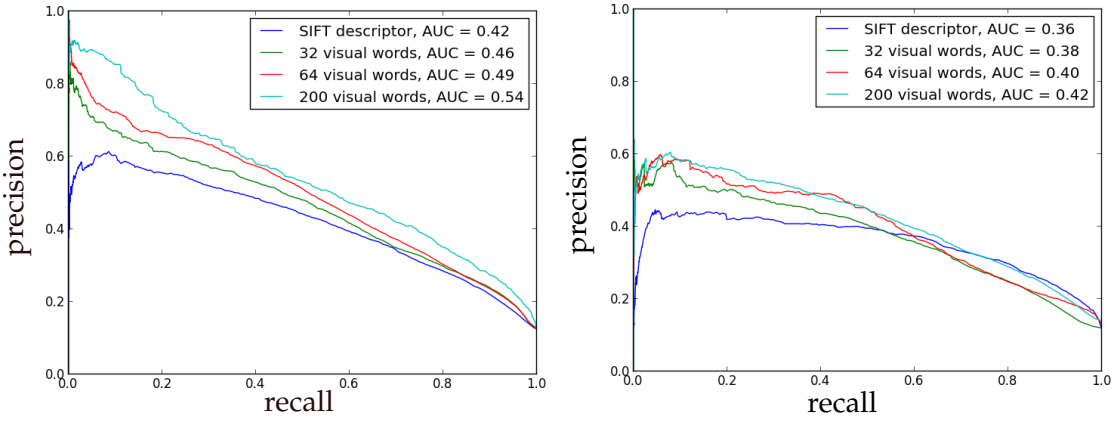


Figure 7.3.3: The precision-recall curves obtained for a detector using SIFT descriptor and learned features on ICDAR 2003 and ICDAR 2013. Figure (a) shows the result from ICDAR 2003 and figure (b) shows the result from ICDAR 2013. Both the P-R curves indicate that the learned features can achieve higher performance than hand-crafted feature descriptors such as SIFT descriptor.

7.3.3 *MSER integrated sliding window*

This experiment evaluates the performance of newly proposed MSER integrated sliding window approach discussed in Section 6.3 with the normal sliding window method. In this experiment, no new classifiers are trained. Instead, the same classifiers in experiments described in Subsection 7.3.2 and Subsection 7.3.1 are used. Using these classifiers, the test images are evaluated at MSER detected scales instead of experimentally determined scales. Each test image is given as input to the configured MSER detector (Section 6.1). The dimension (height and width) of the detected MSER regions are clustered in order to obtain the size of the character candidate representative. The best number of clusters suitable for the data is selected using the gap statistic. The selected cluster centroids give the sizes of the character present in the test image. Using these sizes, appropriate scaling factors are determined. These scaling factors are used to rescale the image instead of using heuristically determined 13 different scales in the reference method. On the rescaled image, trained classifier is evaluated in a sliding window fashion. AUC values are calculated from the generated predictions.

Table 7.3.3 shows the comparison of AUC values obtained for standard sliding window approach and MSER integrated sliding window approach on ICDAR 2003 and ICDAR 2013 dataset. The results indicate that determining the scaling factors



(a) (b)

Figure 7.3.4: Figure (a) shows the precision-recall curve obtained on ICDAR 2003 dataset whereas figure(b) depicts the PR curves obtained on ICDAR 2013 using the MSER integrated sliding window approach. The newly proposed approach improves the performance of text detection system using hand-crafted features as well as learned feature.

(a) (b) (c)

Figure 7.3.5: Figure (b) and (c) depicts the prediction results from the classifier using standard sliding window with predetermined scale factors and MSER detected scales respectively for the image shown in figure (a). The per-pixel predictions are overlaid over the original image. The white region denotes the regions predicted as text and black region denotes the region predicted as non-text. In figure (b), there are a lot of false positive predictions when compared to figure(c).

	AUC					
	ICDAR 2003			ICDAR 2013		
	13 scales	MSER scales	Increase in %	13 scales	MSER scales	Increase in %
SIFT descriptor	0.39	0.42	7.7	0.34	0.36	5.9
32 visual words	0.45	0.46	2.2	0.36	0.38	5.6
64 visual words	0.47	0.50	6.4	0.38	0.40	5.3
200 visual words	0.52	0.55	5.8	0.41	0.42	2.4

Table 7.3.3: Results of the performance of standard sliding window and MSER integrated sliding window on ICDAR 2003 and ICDAR 2013. When compared to the performance of standard sliding window using predetermined scales, it is evident that there is an improvement in AUC values for text detection systems using hand-crafted features as well as learned features evaluated using MSER integrated sliding window.

using the MSER detector has improved the performance of the text detection system when compared to the text detection system using experimentally determined scales in the sliding window approach. The proposed method shows an improvement in AUC values in both hand-crafted as well as learned features on ICDAR 2003 and ICDAR 2013. In hand-crafted feature representation, an increase of 7.7% and 5.9% was observed on ICDAR 2003 and ICDAR 2013 respectively. In the case of learned feature representation, an average increase of 4.8% and 4.4% was observed on ICDAR 2003 and ICDAR 2013 respectively. The increase in the performance is due to the fact that the number of false positives decreases as the detector is evaluated on scales according to the size of the character in the image. Figure 7.3.5 shows the comparison of outputs on a sample image using experimentally and MSER determined scaling factors which shows a reduction in number of false positives. The increase in the AUC values for the proposed MSER integrated sliding window approach also evident from the PR curve plot shown in Figure 7.3.4.

The important fact highlighted here is that integrating the MSER detector to the sliding window process has not only improved the performance but also decreased the computational time. One of the major drawbacks of the sliding window method is that time taken for evaluation process depends on the number of scales used. Reducing the scales will result in neglecting characters of particular sizes. On the other hand,

	Time taken for evaluation					
	ICDAR 2003			ICDAR 2013		
	13 scales (s)	MSER scales (s)	Reduction in %	13 scales (s)	MSER scales (s)	Reduction in %
SIFT descriptor	155.8	65.4	58.0	157.3	67.1	57.3
32 visual words	254.7	148.3	41.7	312.2	194.3	37.8
64 visual words	571.3	322.4	43.5	639.3	367.2	43.6
200 visual words	1021.8	547.8	46.3	1147.4	586.6	48.9

Table 7.3.4: Results of the evaluation time of standard sliding window and MSER integrated sliding window on ICDAR 2003 and ICDAR 2013. For the above results, it is evident that the time taken for evaluation using MSER determined scales is less when compared to evaluation using predetermined scales.

increasing the number of scales to include large range of character sizes will lead to increased computational time. As a solution for this problem, the MSER integrated sliding window find the most optimum scales according to the size of the character in the image, hence finding an optimum solution for a better detector performance and computational time. The time taken for the evaluation of the classifier on test images depends on the number of scales detected in the image. In order to obtain an overview about execution time for both sliding window using predetermined and MSER scales, the detector was evaluated on ten random images from both datasets and an average execution time was calculated. The values are tabulated in Table 7.3.4. The execution time was obtained by executing the algorithm on a single core in an Intel i5 processor with clock speed of 2.5 GHz and 8 Gb of RAM. For SIFT descriptor, there was a reduction of 58% and 57.3% in the execution time on ICDAR 2003 and ICDAR 2013 dataset respectively. For learned feature representations, on an average the execution time was reduced by 43.8% and 43.4% on ICDAR 2003 and ICDAR 2013 dataset respectively. The evaluation time for the whole data set has reduced without sacrificing the performance of the system. Also the scaling factors need not be predetermined for each dataset for optimum performance. Thus, the newly proposed method of integrating the MSER detector has improved the performance of the text detection system at reduced execution time.

7.4 CONCLUSION

The performance of various text detection systems were evaluated by performing experiments on ICDAR 2003 and ICDAR 2013 datasets. Initially, performance of the text detection system using learned feature representation was evaluated and the effect of increase in the number of visual words on performance of the system was studied and found to be increasing. Later, an experiment was performed in order to compare the performance of hand-crafted features and learned features where the learned features outperformed the hand-crafted features. Finally, the proposed MSER integrated sliding window approach was benchmarked with the standard sliding window approach. The newly proposed method not only reported higher AUC values but also reduced the execution time. The results obtained from the experiments are quantitatively summarized in the following chapter.

8

CONCLUSION

Text detection in natural scenes is a difficult problem. The difficulty is due to wide dissimilarity in fonts, sizes, color and textures of text regions embedded in natural scenes. The location of the text in the image also follows a random behavior. Text can occur in any part of the image. The presence of repeating patterns and complex backgrounds in unconstrained images increase the difficulty for detecting text in natural scenes.

Encoding all these variabilities in a rule-based approach is extremely challenging. In such situations, researchers tend to resort to machine learning methods to generate a model to discriminate text and non-text region in natural scenes. Feature representation methods play a vital role in model building. The most common choice for feature representation is hand-crafted feature representation methods. One of the caveats of these methods is that the feature values are not specialized to the data. Also, these features need to be engineered according to the task and they lack adaptability. On the other hand, feature learning approaches derive the representation from the input data and adapt according to the input data. In this thesis, a feature learning approach proposed by Coates et. al. was discussed in detail. This method uses spherical k-means to generate the visual vocabulary. Using this visual vocabulary, the feature representations were defined. The performance of the text detection system using learned features was evaluated. Also, the effect of size of visual vocabulary on the text detector performance was analyzed. With increase in the number of visual words, the detector performance also showed considerable improvement in both classification accuracy and area under precision-recall curve. The classification accuracy improved by 4.7% on ICDAR 2003 and by 6.7% on ICDAR 2013 datasets. As the number of visual words were increased from 32 to 200, the area under the precision-recall curve showed an increase of 15.6% on ICDAR 2003 and 13.9% on ICDAR 2013. The reason for this improved behavior is due to the fact that visual vocabulary comprised of more patterns learned from the text patches. The increasing trend was found in both the datasets.

The performance of the detector is comparatively low on ICDAR 2013 when compared to ICDAR 2003. For e.g., for 200 visual words, the AUC value obtained was 0.52 and 0.41 on ICDAR 2003 and ICDAR 2013 respectively. The reason for this difference is that the training data generation algorithm is tuned for better performance on ICDAR

2003. Since the same algorithm was applied to generate the training data for ICDAR 2013, the detector performance is lower when compared to ICDAR 2003. The inference from the ICDAR 2013 dataset is that the training data generation parameters play a pivotal role in the performance of the system while using unsupervised feature learning method. This experiment also emphasizes that superior performance can be achieved with the increase in the number of visual words.

Text detection system using hand-crafted features is judged against the system using learned features. The result shows that the system using learned features has outperformed the text detection system using hand-crafted features. The hand-crafted features where compared against learned features with 200 visual words. The AUC values improved by 33.3% and 20.6% on ICDAR 2003 and ICDAR 2013 respectively. On both the datasets, learned features were found superior to hand-crafted features. The superiority of learned features is due to their adaptive nature and they highly specialized to the input data.

One of the major shortcomings of the text detection system using sliding window approach is that it is computationally expensive. The computational cost is incurred due to the evaluation of the trained detector across predetermined multiple scales. Hence, to solve this problem, this thesis presented a new method of MSER integrated sliding window. The method uses the maximally stable extremal region detector to detect character candidates sizes. The height and width of the character candidates are clustered to generate the size of character representatives. The scales are determined based on the obtained sizes of character representatives. The results indicate that the detector performance has improved with the newly proposed method. In ICDAR 2003, an improvement of 7.7% and 4.8% in AUC values for hand-crafted and learned features respectively whereas in ICDAR 2013 the AUC values showed an improvement of 5.9% and 4.4% for hand-crafted and learned features respectively. The reason for this improvement is due to the fact that there is less number of false positive detections as the detector is only evaluated at scales according to the size of the character in the image. Along with increased performance, the evaluation time has decreased considerably for both datasets. On ICDAR 2003 the evaluation time reduced by an average of 47.4% whereas on ICDAR 2013 the evaluation time reduced by an average of 46.9%. Hence the proposed method has not only improved detection rates but also reduced the execution time.

The MSER detector used in this thesis still detects non-text regions as character candidates especially in images with repeating patterns or vegetations. Besides examining the MSER parameter space, various proposed methods for pruning non-character MSER regions can be investigated. Exploring the feasibility of proposing a guideline for parameter configuration involved in the training data generation, by analyzing the

performance of the text detection system using learned features with various sets of training data is a further research possibility.

BIBLIOGRAPHY

[BBLP10] Boureau, Y-L ; Bach, Francis ; LeCun, Yann ; Ponce, Jean: Learning mid-level features for recognition. In: *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on* IEEE, 2010, S. 2559–2566

[BHEG01] Ben-Hur, Asa ; Elisseeff, Andre ; Guyon, Isabelle: A stability based method for discovering structure in clustered data. In: *Pacific symposium on biocomputing* Bd. 7, 2001, S. 6–17

[Bra00] Bradski, Gary: The opencv library. In: *Doctor Dobbs Journal* 25 (2000), Nr. 11, S. 120–126

[CCC+11] Coates, Adam ; Carpenter, Blake ; Case, Carl ; Satheesh, Sanjeev ; Suresh, Bipin ; Wang, Tao ; Wu, David J. ; Ng, Andrew Y.: Text detection and character recognition in scene images with unsupervised feature learning. In: *Document Analysis and Recognition (ICDAR), 2011 International Conference on* IEEE, 2011, S. 440–445

[CN12] Coates, Adam ; Ng, Andrew Y.: Learning feature representations with k-means. In: *Neural Networks: Tricks of the Trade*. Springer, 2012, S. 561–580

[CV95] Cortes, Corinna ; Vapnik, Vladimir: Support-vector networks. In: *Machine learning* 20 (1995), Nr. 3, S. 273–297

[DHS12] Duda, Richard O. ; Hart, Peter E. ; Stork, David G.: *Pattern classification*. John Wiley & Sons, 2012

[DM01] Dhillon, Inderjit S. ; Modha, Dharmendra S.: Concept decompositions for large sparse text data using clustering. In: *Machine learning* 42 (2001), Nr. 1-2, S. 143–175

[EOW10] Epshtein, Boris ; Ofek, Eyal ; Wexler, Yonatan: Detecting text in natural scenes with stroke width transform. In: *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on* IEEE, 2010, S. 2963–2970

[Fin14] Fink, Gernot A.: *Markov models for pattern recognition: from theory to applications*. Springer Science & Business Media, 2014

[FS97] Freund, Yoav ; Schapire, Robert E.: A decision-theoretic generalization of on-line learning and an application to boosting. In: *Journal of computer and system sciences* 55 (1997), Nr. 1, S. 119–139

[GLR+13] Grzegorzek, Marcin ; Li, Chen ; Raskatow, Johann ; Paulus, Dietrich ; Vassilieva, Natalia: Texture-Based Text Detection in Digital Images with Wavelet Features and Support Vector Machines. In: *Proceedings of the 8th International Conference on Computer Recognition Systems CORES 2013* Springer, 2013, S. 857–866

[HCL+03] Hsu, Chih-Wei ; Chang, Chih-Chung ; Lin, Chih-Jen u. a.: *A practical guide to support vector classification*. 2003

[HQT14] Huang, Weilin ; Qiao, Yu ; Tang, Xiaoou: Robust Scene Text Detection with Convolution Neural Network Induced MSER Trees. In: *Computer Vision–ECCV 2014*. Springer, 2014, S. 497–511

[JOP14] Jones, Eric ; Oliphant, Travis ; Peterson, Pearu: SciPy: Open source scientific tools for Python. (2014)

[KSU+13] Karatzas, Dimosthenis ; Shafait, Faisal ; Uchida, Seiichi ; Iwamura, Masakazu ; Mestre, Sergi R. ; Mas, Joan ; Mota, David F. ; Almazan, Jon A. ; Heras, Lluis P. l. u. a.: ICDAR 2013 Robust Reading Competition. In: *Document Analysis and Recognition (ICDAR), 2013 12th International Conference on* IEEE, 2013, S. 1484–1493

[Llo82] Lloyd, Stuart: Least squares quantization in PCM. In: *Information Theory, IEEE Transactions on* 28 (1982), Nr. 2, S. 129–137

[Low99] Lowe, David G.: Object recognition from local scale-invariant features. In: *Computer vision, 1999. The proceedings of the seventh IEEE international conference on* Bd. 2 Ieee, 1999, S. 1150–1157

[Low04] Lowe, David G.: Distinctive image features from scale-invariant keypoints. In: *International journal of computer vision* 60 (2004), Nr. 2, S. 91–110

[LPS+03] Lucas, Simon M. ; Panaretos, Alex ; Sosa, Luis ; Tang, Anthony ; Wong, Shirley ; Young, Robert: ICDAR 2003 robust reading competitions. In: *2013 12th International Conference on Document Analysis and Recognition* Bd. 2 IEEE Computer Society, 2003, S. 682–682

[MCUP04] Matas, Jiri ; Chum, Ondrej ; Urban, Martin ; Pajdla, Tomás: Robust wide-baseline stereo from maximally stable extremal regions. In: *Image and vision computing* 22 (2004), Nr. 10, S. 761–767

[MRS08] Manning, Christopher D. ; Raghavan, Prabhakar ; Schütze, Hinrich: *Introduction to information retrieval*. Bd. 1. Cambridge university press Cambridge, 2008

[MS04] Mikolajczyk, Krystian ; Schmid, Cordelia: Scale & affine invariant interest point detectors. In: *International journal of computer vision* 60 (2004), Nr. 1, S. 63–86

[MS05] Mikolajczyk, Krystian ; Schmid, Cordelia: A performance evaluation of local descriptors. In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 27 (2005), Nr. 10, S. 1615–1630

[MTS+05] Mikolajczyk, Krystian ; Tuytelaars, Tinne ; Schmid, Cordelia ; Zisserman, Andrew ; Matas, Jiri ; Schaffalitzky, Frederik ; Kadir, Timor ; Van Gool, Luc: A comparison of affine region detectors. In: *International journal of computer vision* 65 (2005), Nr. 1-2, S. 43–72

[NJT06] Nowak, Eric ; Jurie, Frédéric ; Triggs, Bill: Sampling strategies for bag-of-features image classification. In: *Computer Vision–ECCV 2006*. Springer, 2006, S. 490–503

[NM11] Neumann, Lukas ; Matas, Jiri: A method for text localization and recognition in real-world images. In: *Computer Vision–ACCV 2010*. Springer, 2011, S. 770–783

[NWC+11] Netzer, Yuval ; Wang, Tao ; Coates, Adam ; Bissacco, Alessandro ; Wu, Bo ; Ng, Andrew Y.: Reading digits in natural images with unsupervised feature learning. In: *NIPS workshop on deep learning and unsupervised feature learning* Bd. 2011 Granada, Spain, 2011, S. 5

[OD11] O'Hara, Stephen ; Draper, Bruce A.: Introduction to the bag of features paradigm for image classification and retrieval. In: *arXiv preprint arXiv:1101.3354* (2011)

[RNI95] Russell, Stuart ; Norvig, Peter ; Intelligence, Artificial: A modern approach. In: *Artificial Intelligence. Prentice-Hall, Egnlewood Cliffs* 25 (1995)

[SSD11] Shahab, Asif ; Shafait, Faisal ; Dengel, Andreas: ICDAR 2011 robust reading competition challenge 2: Reading text in scene images. In: *Document Analysis and Recognition (ICDAR), 2011 International Conference on* IEEE, 2011, S. 1491–1496

[TWH01] Tibshirani, Robert ; Walther, Guenther ; Hastie, Trevor: Estimating the number of clusters in a data set via the gap statistic. In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 63 (2001), Nr. 2, S. 411–423

[VDWCV11] Van Der Walt, Stefan ; Colbert, S C. ; Varoquaux, Gael: The NumPy array: a structure for efficient numerical computation. In: *Computing in Science & Engineering* 13 (2011), Nr. 2, S. 22–30

[VF10] Vedaldi, Andrea ; Fulkerson, Brian: VLFeat: An open and portable library of computer vision algorithms. In: *Proceedings of the international conference on Multimedia* ACM, 2010, S. 1469–1472

[WLMH09] Weinman, Jerod J. ; Learned-Miller, Erik ; Hanson, Allen R.: Scene text recognition using similarity and a lexicon with sparse belief propagation. In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 31 (2009), Nr. 10, S. 1733–1746

[WWCN12] Wang, Tao ; Wu, David J. ; Coates, Adam ; Ng, Andrew Y.: End-to-end text recognition with convolutional neural networks. In: *Pattern Recognition (ICPR), 2012 21st International Conference on* IEEE, 2012, S. 3304–3308

[YD14] Ye, Qixiang ; Doermann, David: Text Detection and Recognition in Imagery: A Survey. (2014)

[YYHH14] Yin, Xu-Cheng ; Yin, Xuwang ; Huang, Kaizhu ; Hao, Hong-Wei: Robust text detection in natural scene images. In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 36 (2014), Nr. 5, S. 970–983