THE REPUBLIC OF TURKEY
ISTANBUL SABAHATTIN ZAIM UNIVERISTY
INSTITUTE OF GRADUATION EDUCATION
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING (30% ENGLISH)
COMPUTER SCIENCE AND ENGINEERING

DEEP LEARNING BASED TURKISH VIDEO INDEXING AND RETRIEVAL SYSTEM

Ph.D. DISSERTATION

Jawad RASHEED

Istanbul
January – 2021

THE REPUBLIC OF TURKEY
ISTANBUL SABAHATTIN ZAIM UNIVERSITY
INSTITUTE OF GRADUATION EDUCATION
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING (30% ENGLISH)
COMPUTER SCIENCE AND ENGINEERING

DEEP LEARNING BASED TURKISH VIDEO INDEXING AND RETRIEVAL SYSTEM

Ph.D. DISSERTATION

Jawad RASHEED

Supervisor
Assistant Prof. Dr. Akhtar JAMIL

Istanbul
January – 2021

THESIS APPROVAL

To the Directorate of Institute of Graduate Education,

This is to certify that we (jury) have read this thesis and we find it fully adequate in scope, quality and content, as a thesis for Ph.D. degree in Computer Science and Engineering.

Thesis Supervisor: Assist. Prof. Dr. Akhtar JAMIL

Member: Assist. Prof. Dr. Ali HAMİTOĞLU

Member: Assist. Prof. Dr. Amani YAHYAOUI

Member: Professor. Dr. Naim Mahmood Musleh AJLOUNI

Member: Assist. Prof. Dr. Muhammad ILYAS

Approval

I approve that the signatures above belong to the academic members specified here.

Prof. Dr. Ali GÜNEŞ

Director

Institute of Graduate Education

DECLARATION OF SCIENTIFIC ETHICS AND ORIGINALITY

This is to certify that this Ph.D. dissertation title "**Deep Learning based Turkish Video Indexing and Retrieval System**" is my own work and I have acted according to scientific ethics and academic rules while producing it. I have collected and used all information data according to scientific ethics and guidelines on thesis writing of Sabahattin Zaim University. I have fully referenced, in both the test and bibliography, all direct and indirect quotations and all sources I have used in this work.

Jawad RASHEED

January – 2021

ACKNOWLEDGEMENT

Thanks to everyone who helped and prayed for me during this journey, including the Creator of this universe, my parents, advisor, brother, and wife.

Jawad RASHEED

January – 2021

Dedicated to my son *Jasim*…

ABSTRACT

DEEP LEARNING BASED TURKISH VIDEO INDEXING AND RETRIEVAL SYSTEM

Jawad RASHEED
PhD Dissertation, Computer Science and Engineering
Supervisor: Assistant Prof. Dr. Akhtar JAMIL
January-2021, 142 + XVIII Pages

The continual technological advancement of handheld devices and personal computers over past few decades has reshaped the world's communication system by enabling the humans and robots to capture and share images and videos in digitized form at large. Practically, annotation-based video indexing and retrieval systems are widely being used to maintain the ongoing growth of multimedia content. These systems grant multimedia content retrieval using textual annotations, but are limited to predefined annotation/keywords. The online multimedia content libraries require manual annotation of video while uploading, which is a hectic and time-consuming assignment that sometimes even does not align with the visual content. This limits the searching capacity, as user may be unable to retrieve video because of incomplete video description at the time of annotation. Therefore, it strongly requires an efficient and sophisticated video indexing and retrieval system. To accomplish it, content-based video indexing is an optimal solution by detecting text appearing in videos.

This dissertation demonstrates a new text detection system based on advance deep learning approach to bridge the gap by building an automatic and efficient content-based video indexing and retrieval system for Turkish videos. The text appearing in videos provides useful information that can be exploited for developing automatic video indexing and retrieval system. Therefore, this study integrates heuristic and deep learning-based approaches that utilizes CNN for automatic text detection and extraction. To train the proposed CNN-based model, a new dataset is generated by collecting videos from various Turkish channels related to News, financial and business, sports and cartoon channels. The dataset is fed to proposed model that first generates features maps and then classifies the image as textual or non-textual class. Extensive trails and experiments are carried out with different structural combination of convolutional layers, thus ended up with a best model out of three proposed models that can accurately detect the text.

Next, the extracted text is fed to publicly available Tesseract OCR for recognition, which is then indexed in database along with video information such as file storage location. Lastly, a web-based user interface is provided for querying purposes. For each user query, the proposed system retrieved the most relevant videos based on its textual content appearing inside. Besides displaying the retrieved videos in provided user interface, the system also informs the user about the appearance time of queried words inside each retrieved video so that user can directly jump to the point of interest by using sleek bar. All basic functionalities are provided to play, pause, maximize, minimize, and download the retrieved video with additional controls for volume and sleek bar.

Moreover, various conventional machine-learning algorithms such as SVM and LR, and few state-of-the-art image classification models (including VGG16, ResNet50 and DenseNet121) are also implemented and trained with identical dataset. The proposed models outperformed the prior state-of-the-art deep learning frameworks and machine learning classifiers.

Keywords: Convolutional neural network, deep learning, machine learning, text detection, text recognition, video indexing, video retrieval.

ÖZET

DERİN ÖĞRENMEYE DAYALI TÜRKÇE VİDEO İNDEKSLEME VE BİLGİ GETİRİMİ SİSTEMİ

Jawad RASHEED
Doktora, Bilgisayar Bilimleri ve Mühendisliği
Tez Danışmanı: Dr. Öğr. Üyesi Akhtar JAMIL
January-2021, 142 + XVIII Sayfa

Son zamanlarda el cihazlarının ve kişisel bilgisayarların teknolojik gelişimi, insanların ve robotların görüntüler ve videoları yakalayıp büyük ölçüde dijitalleştirilmiş biçimde paylaşılmasını sağlayarak dünyanın iletişim sistemini yeninden şekillendirmiştir. Pratikte, açıklama tabanlı video indeksleme ve bilgi erişim sistemleri günümüzde büyüyen multimedya içeriklerinin sürdürebilirliğini devam ettirmek amacıyla kullanılmaktadırlar. İlgili sistemler, metin notlarını kullanarak multimedya içeriğinin elde edilmesini sağlamaktadırlar, ancak ek açıklamaları tanımlamada sınırlıdırlar. Çevrimiçi multimedya içerikli kütüphanelerde, video yükleme işleminde videonun açıklaması manuel olarak gerçekleşmesi gerekmektedir. Bu durum, zaman gerektiren bir işlem olduğu gibi bazı durumlarda video açıklaması görsel içerik ile uyuşmamaktadır. Aynı zamanda, videolarda oluşacak eksik açıklamalar nedeniyle kullanıcıların arama yapma kapasitelerini sınırlamaktadır. Bu nedenle, verimli ve sofistike bir video indeksleme ve erişim sistemi gereklidir. Bu problemi çözüme kavuşturmak için, videolarda mevcut olan metinleri tespit ederek içerik tabanlı video indeksleme sistemi geliştirmek en uygun bir çözümdür.

Bu tez çalışmasında, Türkçe videolar için otomatik ve verimli içerik tabanlı video indeksleme ve bilgi erişim sistemi oluşturmak amaçlı derin öğrenmeye dayalı yeni bir metin algılama sistemi geliştirilmektedir. Videolarda görünen metin, otomatik video indeksleme ve bilgi erişim sistemini geliştirmek için kullanılabilecek faydalı bilgiler sağlamaktadır. Dolayısıyla, bu çalışma otomatik metin algılama ve çıkarma işlemlerini gerçekleştirmek için Evrişimsel Sinir Ağlarından yararlanarak, sezgisel ve Derin Öğrenmeye dayalı yaklaşımları bütünleştirmektedir. Önerilen Evrişimsel Sinir Ağı tabanlı modeli eğitmek için, Haberler, finans ve iş, spor ve çizgi ile ilgili çeşitli Türk televizyon kanallarından videolar toplanarak yeni bir veri kümesi oluşturulmuştur. Oluşturulan veri seti, ilk aşamada önerilen modele özellik haritalarının elde edilmesi amacıyla beslenmektedir, devamında önerilen model görüntüyü metinsel veya

metinsel olmayan sınıf olarak sınıflandırmaktadır. Evrişimsel katmanların farklı yapısal kombinasyonları ile kapsamlı deneyler yapıldıktan sonra önerilen üç modelden metni en doğru bir şekilde algılayabilen model elde edildi.

Devamında, çıkarılan (elde edilen) metnin tanınması için Tesseract OCR'a beslenir, ve dosya depolama konumu gibi video bilgileri ile birlikte veri tabanında indekslenir. Son olarak, sorgulama amacıyla web tabanlı bir kullanıcı arayüzü geliştirilir. Her kullanıcı sorgusu için, önerilen sistem görüntü içinde görünen metin içeriğine göre en alakalı videoları kullanıcıya sunmaktadır. Buna ek olarak, sistem kullanıcın sorguladığı kelimeleri videonun hangi süreleri arasında geçtiğine dair bilgi vermektedir, dolayısıyla kullanıcı arama çubuğunu kullanarak doğrudan ilgi alanına gidebilir. Sunulan videoyu oynatmak, duraklatmak, büyütmek, küçültmek ve indirmek için ek ses ve kontrol çubuğu kontrolleriyle birlikte tüm temel işlevler sağlanmıştır.

Ayrıca, Destek Vektör Makineleri ve Lojistik Regresyon gibi çeşitli geleneksel makine öğrenimi algoritmaları ve birkaç son teknoloji görüntü sınıflandırma modeli (VGG16, ResNet50 ve DenseNet121) aynı veri setiyle uygulanır ve eğitilir. Önerilen modeller, son teknoloji derin öğrenme modeleri ve makine öğrenimi sınıflandırıcılarından daha iyi performans göstermiştir.

Anahtar Kelimeler: Evrişimsel Sinir Ağları, derin öğrenme, makine öğrenmesi, metin tespiti, metin tanıma, video indeksleme, video sunma.

TABLE OF CONTENT

THESIS APPROVAL ..... **i**

DECLARATION OF SCIENTIFIC ETHICS AND ORIGINALITY ..... **ii**

ACKNOWLEDGEMENT ..... **iii**

ABSTRACT ..... **v**

ABSTRACT IN TURKISH ..... **vii**

TABLE OF CONTENTS ..... **ix**

LIST OF TABLES ..... **xii**

LIST OF FIGURES ..... **xiii**

ABBREVIATIONS ..... **xvii**

CHAPTER 1

INTRODUCTION **1**

1.1. Background and Motive 1

1.2. Aims and Objectives 5

1.3. Contributions and Significance 6

1.4. Outline and Organization of the Dissertation 6

CHAPTER 2

LITERATURE REVIEW **8**

2.1. Prior Work Related to Other Languages 9

2.2. Prior Work Related to Turkish Language 37

2.3. Findings 41

CHAPTER 3

MATERIALS AND METHODS 42

3.1. Dataset Overview 46

3.2. Image Preprocessing Techniques 50

3.2.1. Sobel Edge Detector 51

3.2.1. Dilation as Morphological Operation 54

3.3. Classification Methods and Frameworks 56

3.3.1. Deep Learning-based Methods and Frameworks 57

3.3.1.1. Convolutional Neural Network 57

3.3.1.2. VGG16 66

3.3.1.2. ResNet50 68

3.3.1.2. DenseNet121 70

3.3.2. Machine Learning-based Methods 72

3.3.2.1. Support Vector Machine 72

3.3.2.2. Logistic Regression 76

3.4. OCR for Text Recognition 78

3.4.1. Tesseract OCR Engine 80

CHAPTER 4

EXPERIMENTAL SETUP AND RESULTS 82

4.1. Data Generation and Image Preprocessing 83

4.2. Performance Metrics 91

4.3. Classification 92

4.3.1. Proposed Models 93

4.3.1.1. Model-A 93

4.3.1.2. Model-B 97

4.3.1.3. Model-C 102
4.3.2. Other Deep Learning-based Frameworks 106
4.3.2.1. VGG16 106
4.3.2.2. ResNet50 108
4.3.2.3. DenseNet121 111
4.3.3. Other Machine Learning-based Frameworks 113
4.3.3.1. Support Vector Machine 113
4.3.3.2. Logistic Regression 114
4.4. Text Recognition using Tesseract OCR Engine 116
4.4.1. Why Separate Text Detection Method? 117
4.5. Video Indexing and Retrieval 119

CHAPTER 5

CONCLUSION 124
5.1. Discussion 124
5.2. Future Work and Limitations 126

REFERENCES 127
RELATED PUBLICATION DETAILS 137
CURRICULUM VITAE 138

LIST OF TABLES

CHAPTER 3

Table 3.1: Overview of generated dataset 50

CHAPTER 4

Table 4.1: Generated dataset split as train and test sets 90

Table 4.2: Software and hardware specification for classification experiments 93

Table 4.3: Network topology of proposed Model-A 95

Table 4.4: Performance evaluation of proposed Model-A on test set 97

Table 4.5: Network topology of proposed Model-B 99

Table 4.6: Performance evaluation of proposed Model-B on test set 100

Table 4.7: Network topology of proposed Model-C 104

Table 4.8: Performance evaluation of proposed Model-C on test set 104

Table 4.9: Performance evaluation of VGG16 on test set 107

Table 4.10: Performance evaluation of ResNet50 on test set 109

Table 4.11: Performance evaluation of DenseNet121 on test set 111

Table 4.12: Performance evaluation of support vector machine on test set 113

Table 4.13: Performance evaluation of logistic regression model on test set 115

CHAPTER 5

Table 5.1: Overall performance comparison of implemented frameworks and proposed models 125

LIST OF FIGURES

CHAPTER 1

Figure 1.1: Sample image frame, containing scene text as well as artificial text 4

CHAPTER 3

Figure 3.1: Schematic diagram for proposed Turkish text detection, video indexing and retrieval system 43

Figure 3.2: Workflow of preprocessing steps on image 46

Figure 3.3: Dismantled frame of a downloaded Turkish News channel video, (a) image with three-color channels (red, green, and blue), (b) image with a grey channel 48

Figure 3.4: Sample image containing multiple lines of horizontally aligned text, each line varies in length 49

Figure 3.5: Samples of generated dataset, (a) textual images, and (b) non-textual images 50

Figure 3.6: Filtering derivative graphs, showing the first order derivative and second order derivative of a grey level profile image 51

Figure 3.7: Image of 3×3 size-region 53

Figure 3.8: Sobel filter masks, (a) horizontal mask, and (b) vertical mask 53

Figure 3.9: Some famous structuring elements used for morphological operation ... 54

Figure 3.10: Image probing with 3×3 square-shape structuring element (hit, fit, or none) 55

Figure 3.11: Image dilation using 3×3 cross-shaped structuring element to join two textual objects 56

Figure 3.12: General overview of CNN architecture 58

Figure 3.13: Visual representation of neurons connectivity and weights in CNN..... 59

Figure 3.14: Visual representation of kernel convolving with single input feature maps in CNN..... 61

Figure 3.15: Visual representation of kernels convolving with 3-channel input image in CNN..... 62

Figure 3.16: Graphical representation of various activation functions used in CNN, (a) ELU, (b) leaky ReLU, (c) ReLU, (d) sigmoid, and (e) tanh..... 63

Figure 3.17: Visual representation of kernel movement in multi-dimensional image..... 64

Figure 3.18: Visual representation of applying 2×2 average pooling on extracted feature map for dimensionality reduction..... 64

Figure 3.19: Visual representation of applying 2×2 max-pooling on extracted feature map for dimensionality reduction..... 65

Figure 3.20: General overview of VGG16 network architecture..... 67

Figure 3.21: General overview of ResNet50 network architecture..... 69

Figure 3.22: General overview of DenseNet121 network architecture..... 71

Figure 3.23: Support vectors and decision boundary in simple linear support vector machine classifier..... 73

Figure 3.24: Non-linearly separable data samples in support vector machine 74

Figure 3.25: Kernel tricks in support vector machine classifier for non-linear complex cases, (a) transforming the space to higher dimensions (3D representation) to find optimal hyperplane, and (b) 2D representation after finding hyperplane..... 74

Figure 3.26: Soft-margin cases in support vector machine..... 76

Figure 3.27: Graphical representation of sigmoid function in logistic regression..... 77

CHAPTER 4

Figure 4.1: Sample image, (a) image having 3-channels, and (b) greyscale conversion of image in (a)..... 84

Figure 4.2: Edges detection by projection analysis on image (a) after horizontal projection analysis, and (b) after vertical projection analysis. 85

Figure 4.3: Resultant image after applying sobel filter 86

Figure 4.4: Resultant image after performing dilation operation 87

Figure 4.5: Image with unwanted noise in bounding boxes before applying geometrical and heuristics constraints 87

Figure 4.6: Image with potential candidate text in bounding boxes after applying geometrical and heuristics constraints 88

Figure 4.7: Samples of text images obtained, (a) normal text, and (b) inverse text .. 89

Figure 4.8: Image formatting; a text image of 55×1247 pixels is adjusted to height of 32 pixels and divided into sub-part with a width of 128-pixels. As the last sub-part has size of 95 pixels that is less than 128 pixels, thus a pure black image of 33 pixels width is added at the end 90

Figure 4.9: Overview of proposed Model-A architecture 94

Figure 4.10: Training loss and accuracy curves of proposed Model-A 96

Figure 4.11: Testing loss and accuracy curves of proposed Model-A 96

Figure 4.12: Confusion matrix of proposed Model-A 97

Figure 4.13: Overview of proposed Model-B architecture 98

Figure 4.14: Training loss and accuracy curves of proposed Model-B 100

Figure 4.15: Testing loss and accuracy curves of proposed Model-B 101

Figure 4.16: Confusion matrix of proposed Model-B 101

Figure 4.17: Overview of CNN-based proposed Model-C architecture 102

Figure 4.18: Training loss and accuracy curves of proposed Model-C 105

Figure 4.19: Testing loss and accuracy curves of proposed Model-C 105

Figure 4.20: Confusion matrix of proposed Model-C 106

Figure 4.21: Training loss and accuracy curves of VGG16 107

Figure 4.22: Testing loss and accuracy curves of VGG16 108

Figure 4.23: Confusion matrix of VGG16 108
Figure 4.24: Training loss and accuracy curves of ResNet50 109
Figure 4.25: Testing loss and accuracy curves of ResNet50 110
Figure 4.26: Confusion matrix of ResNet50 110
Figure 4.27: Training loss and accuracy curves of DenseNet121 111
Figure 4.28: Testing loss and accuracy curves of DenseNet121 112
Figure 4.29: Confusion matrix of DenseNet121 112
Figure 4.30: Confusion matrix of support vector machine 114
Figure 4.31: Confusion matrix of logistic regression 115
Figure 4.32: Sample text file generated after text recognition operation through Tesseract optical character recognizer 116
Figure 4.33: Sample of original image having multiple textual lines, each line has different font and sizes 117
Figure 4.34: Partially recognized text after passing the image directly to Tesseract optical character recognizer without passing through proposed method 118
Figure 4.35: Sample CSV file containing recognized text and other video information 120
Figure 4.36: Database table ‘video’ for storing video links 121
Figure 4.37: Database table ‘word_data’ for storing extracted words and video information 121
Figure 4.38: User interface for querrying and retrieving the relevant videos 122
Figure 4.39: Special feature ‘picture in picture’ for user convenience 123
Figure 4.40: Most relevant videos obtained for ‘türk’ as keyword 123

ABBREVIATIONS

ANN	: Artificial Neural Network
ASR	: Automatic Speech Recognition
Bi-LSTM	: Bi-directional Long Short Term Memory
CAE	: Convolutional Auto-Encoders
CNN	: Convolutional Neural Network
CPU	: Central Processing Unit
CSAE	: Convolution Sparse Auto-Encoder
CSV	: Comma-Separated Values
CTC	: Connectionist Temporal Classification
DBN	: Deep Belief Network
EAST	: Efficient and Accuracy Scene Text
FCN	: Fully Convolutional Network
GB	: Gegabyte
GPUs	: Graphical Processing Units
HMM	: Hidden Markov Model
HWHF	: Haar Wavelet High Frequency
HP	: Hewlett-Packard
ID	: Identification
ICDAR	: International Conference on Document Analysis and Recognition
ILSVRC	: ImageNet Large-Scale Visual Recognition Challenge
LR	: Logistic Regression
LSTF	: Line Segment Texture Feature
LSTM	: Long-Short Term Memory
MB	: Megabyte
M-LSTM	: Multi-Dimensional Long-Short Term Memory
MT	: Machine Translator
NER	: Named Entity Recognizer
NMS	: Non-Maximum Suppression

NN	: Neural Network
OCR	: Optical Character Recognition
RAM	: Random Access Memory
RBF	: Radial Basis Function
RF	: Random Forest
RGB	: Red, Green, Blue
RGPN	: Recognition Guided Proposal Network
RNN	: Recurrent Neural Network
ROI	: Region of Interest
SVM	: Support Vector Machine
SWT	: Stroke Width Transformation
TB	: Terabyte
TBL	: Transformation-Based Learning
TF	: Temporal Redundancy
VGG	: Visual Geometry Group
YOLO	: You Only Look Once

CHAPTER 1
INTRODUCTION

1.1. Background and Motive

The technological advancement in last few decades reshaped the world's communication system by enabling the humans and robots to capture images and record videos in digitized form at large. Before the evolutionary turn, big size analogous cameras were mainly used to record videos while huge number of cables and television were essential for editing and watching the recorded video. In that time, the public had very limited access to those videos due to lesser and expensive devices beside slower medium of transmission. However, with the advent and continual advancement of personal computers and other devices has made the capturing source much cheaper, smaller and accessible. At the same time, the invention and improvement of high-speed internet with stable and reliable networking techniques have switched the technological world to smart and wireless access approach. The evolution of lightning-fast internet and technological advancements in handheld devices encouraged the consumers to generate large amount of multimedia data on regular basis.

Nowadays, these compact, low-priced and long battery powered smart devices embedded with high-performance cameras along with high-speed internet are the common source to produce and transfer huge amount of structured/unstructured information. Beside this, the proliferation of Television broadcasting via internet also contributed towards the outbreak of accessible visual content. This increase in generation and utilization of multimedia data and information is because of its content richness. Since the world has become a global village, the social media websites such

as Facebook[1], free or commercial video hosting platforms like Dailymotion[2], Vimeo[3] and YouTube™[4] have provided online data sharing opportunities among public or closed groups.

The utilization of such platforms can be imagined from this, that YouTube being the world's biggest video hosting library and sharing portal alone have more than 2 billion logged in users monthly[5]. Such amount of users accounts one-third of the total internet users, and each day, these users watch more than 1 billion hours of videos, while 70% of its watch-time is from smart devices. According to Statista report in May 2019, more than 30000 hours of visual content is uploaded on YouTube in every single hour that equates to 500 hours/min[6]. In October 2018, Netflix accounts 26.6% while YouTube accounts more than 21% of global video streaming traffic[7].

These online video repositories, sharing portal and websites provide nice platform to publish the generated or captured videos. While uploading and publishing the multimedia and visual content, user add the corresponding description manually to annotate the video. However, this manual annotation is a hectic and tedious job that may occasionally ends as unaligned description with respect to the visual content. Consequently, it narrows down the searching capabilities by considering the insufficient video details and description. In order to maintain and support the growth

---

[1] Facebook: https://www.facebook.com/

[2] Dailymotion: https://www.dailymotion.com/

[3] Vimeo: https://vimeo.com/

[4] Youtube™: https://www.youtube.com/

[5] YouTube About: https://www.youtube.com/intl/en-GB/about/press/

[6] Statista – Hours of video uploaded to YoutTube every minute 2007-2019: https://www.statista.com/statistics/259477/hours-of-video-uploaded-to-youtube-every-minute/

[7] Statista – Video streaming traffic distribution worldwide 2018: https://www.statista.com/statistics/267191/video-streaming-traffic-distribution-worldwide/

of structured and unstructured multimedia data, an alternate optimal solution is required in form of automatic content-based video indexing and retrieval system.

The utmost exclusive feature of a video is its capability of presenting a fertile semantic demonstration by synchronizing video, audio and textual contents over a certain duration. In the early days of research regarding content-based video indexing and retrieval systems, majority of measures were directly acquired and enhanced mechanism and techniques from sound and image retrieval. Despite the limited success of this approach, it is insufficient for most of the applications as videos fall under different genre that have their own semantic, syntax, formats and rules. Such as an object-shape-color based indexing is not feasible for searching a particular forex values in business videos. On the other hand, after years of research, the approaches used to detect or recognize the text in documents-based images, obtained through traditional scanners, attained highest level of accuracy. Yet, these developed methods and approaches failed to handle the textual content emerging in videos. Thus, content-based video indexing and retrieval system requires supporting such kind of applications.

Experts have suggested various methods and techniques to fulfill the demand of such application by automatically extracting various features from videos and images. Domain-specific is among those techniques that utilizes ordinary features of videos to construct the ultimate powerful engine that extracts the content and then index it accordingly. Therefore, beside audio-visual content, textual material present inside visual frames and images comprises of important source of high-level semantics, thus yields a meaningful clue that can be utilized for automatic video indexing and its retrieval.

For textual content-based video indexing and retrieval system, text detection is the first most important challenging step due to textual variations and its different types. The textual content emerging inside videos intuitively belongs to one of the following categories:

- Scene text
- Artificial text

Scene text is the text that is captured naturally while capturing an image or filming and recording the video. Whereas, artificial text is the graphical text that is manually added

during the editing process of a video or image, to give more semantics. Figure 1.1 depicts an image from a video that contains both scene text as well as artificial text. In this image, the text written on the road-sign boards, taxi, bus and microphone is the scene text (marked with circle in black), whereas text appearing as captions (marked with blue rectangle) is an artificial text that is embedded after recording the video. Besides categorizing text as scene or artificial text, it is also classified as inverse or normal text. Normal text is that text that appears with low intensity as compared to it background, whereas inverse text is denoted with high intensity pixel values than its background. Such as the text 'GÜZERGAH VE YOLCU SEÇENLERE CEZA YAĞDI', appearing in Figure 1.1, is a normal text. Contrarily, the text 'ÖZEL HABER' and 'İSTANBUL' is considered as inverse text.

Figure 1.1: Sample image frame, containing scene text as well as artificial text

Text emerging in videos varies in sizes, fonts, style, positioning and alignment, such as scene text occurs in any arbitrary direction while captioned text (artificial text) typically appears in horizontal or vertical direction. The diversities in scene text and its unpredictable nature makes the detection a challenging job. Contrarily, the high contrast of artificial text enhances the users' readability thus comparatively easier to detect.

This study presents a deep learning based approach to detect horizontally aligned artificial text present inside video images. Even though, detection of text is language independent, but this dissertation specifically focused on an under-stressed area of Turkish text detection from videos. For this study, News videos are preferred as the fundamental application as it attracts more viewers across the globe, thus maximizes the benefits of utilizing a content-based video indexing and retrieval system. Compared with rest of video genres, News videos have more challenges such as News videos are broadcasted with various styles, fonts and sizes of text.

The goal is to create a video indexing and retrieval system, for which text detection of these varied sizes and fonts is the first step. The dataset was prepared by capturing text in most popular Turkish News, business and sports channels. Firstly, to obtain candidate text regions, each frame of video was pre-processed by applying the morphological processing, and some heuristics and geometrical constraints. Finally, the detected regions trained the CNN and its performance is tested on unseen dataset. Later, the trained models are used to detect text, which is then extracted and passed to OCR for recognition, subsequently indexed in database. A web-based UI is built for querying, and relevant videos retrieved and presented base on queried keywords.

1.2. Aims and Objectives

The dissertation aims to explore the deep learning based techniques for text detection, which is then used for semantic annotation. An efficient and convenient text detection and extraction without plenty of human intervention is the basic requirements of the system to semantically annotate the videos for proper indexing. Moreover, to satisfy the user-application demand and necessity a precise and accurate retrieval system is one of the crucial aspects that demonstrate the success of indexing technique.

The main objectives are to design, implement, develop and test

- Automatic graphical text (artificial text) detection using deep learning approaches
- Turkish text recognition
- Automatic video indexing and retrieval
- Text extraction and transformation
- Retrieval of most relevant videos based on queried keywords

- Proper web-based user interface

The focus of the dissertation is to detect the horizontally aligned Turkish artificial text via deep learning framework, moreover the proposed approach might detect and recognize some scene text as well, but it is not of major concern if not detected. Each of the proposed and implemented deep learning and machine learning based text detection frameworks will be trained and tested by largely generated dataset that contains text of various fonts.

1.3. Contributions and Significance

The key contributions of this work and its significance are as following:

- Design and implementation CNN-based framework for Turkish text detection from videos
- Comparison with previous state-of-the-art image classification frameworks
- Development of complete video indexing and retrieval system for Turkish News, business and sports channels
- Development of a ground truth labelling system for accuracy assessment
- Efficient searching and retrieval of videos from database based on automatically extracted keywords.

1.4. Outline and Organization of the Dissertation

The rest of the dissertation enlists four more chapters that are organized in following manner.

Chapter 2 provides an extensive analysis of prior works done in the field of video indexing and retrieval system. It discusses their achievements for segmenting and modeling the video, performing content analysis, text detection and extraction, The first half of this chapter focuses on examining various approaches utilized to detection text of languages other than Turkish, while the second half presents the prior work related to Turkish text detection, and video indexing and retrieval system.

Chapter 3 explains the schematics of proposed methodology, and provides an in-depth overview of the methods and algorithms employed at each step to accomplish the aim

of this study. It also presents the overview of data generated to train and test the proposed system. As the major focus in this dissertation is inclined towards the text detection using deep learning approach, thus it elaborates the basic functionalities of CNN based models along with few machine learning traditional classifiers such as SVM and LR. It also includes the discussion about text recognition through commercially available OCRs.

Chapter 4 presents the detailed experimental setup and results of proposed framework for text detection and extraction. The proposed framework is first trained and later tested over a largely generated dataset, aimed to proved the robustness of the system. Different variations of the proposed framework is also evaluated to come up with best possible frameworks for the problem at hand. Later, results of other state-of-the-art deep learning frameworks and few machine-learning classifiers on generated dataset are also presented. Furthermore, it outlines the issues and performance of OCR when images are feed to OCR with and without passing it through the proposed system. Moreover, it introduces the post recognition process for video indexing and retrieval system.

Lastly, Chapter 5 provides the comparative analysis of all proposed model with other implemented deep learning frameworks and machine learning classifier. It summarizes the achievements and concludes with the future work.

CHAPTER 2
LITERATURE REVIEW

Until today, computer vision experts and artificial intelligence researchers have suggested different approaches to detect, localize, and extract the text from images and video frames. Broadly, these text localization and detection approaches can be classified into two main categories; sliding window based techniques and connected component based approaches. These sliding window based techniques also includes supervised methods such as (Bouaziz, Amara, & Mahdi, 2017; Rasheed, Dogru, & Jamil, 2020; Rasheed, Jamil, Dogru, Tilki, & Yesiltepe, 2019), whereas connected components based approaches includes unsupervised methods such as (Jamil, Rasheed, & Bayram, 2019; Rasheed, Jamil, Yahyaoui, & Madey, 2020)

Generally, the sliding window based techniques or supervised approaches extract various features that includes color, edge or texture for each sliding window. The extracted features are later utilized for effective training of machine learning and deep learning based classifiers such as ANN, CNN and SVM. Based upon the trained classifier, the network distinguishes each sliding window frame as textual or non-textual region. Contrarily, connected component based methods or unsupervised approaches to detect the text in images are normally based on techniques like SWT and maximal stable extremal regions. The unsupervised approaches exploits temporal and statistical properties of text that joins the characters to form words, and words to form textual lines.

Mostly, the well-established sliding window based approaches and connected components based techniques have been utilized particularly for detection of Arabic, Chinese, English, French, German, Urdu and Uyghur languages in images and video frames. As this study is mainly concerned with supervised techniques, therefore a more in-depth overview of prior studies related to supervised approaches is presented in this section with little focus on unsupervised techniques.

This section provides the most recent work, published over last decade, related to supervised and unsupervised approaches and techniques employed for text detection and extraction in videos. Later, it also presents the work specifically done for Turkish language.

2.1. Prior Work Related to Other Languages

Experts have designed various unsupervised approaches to deal the text detection task such as (Phan, Shivakumara, & Tan, 2009) exploited Laplacian operator to detect text. To get the candidate text regions in an image, it calculated the difference of maximum gradient for all the pixels and later used K-means clustering to annotate these as textual or non-textual data. In the end, text line boundaries are found using Sobel edge detector. Similarly, another approach is proposed by authors in (Jamil et al., 2019) to detect multilingual artificial text that horizontally aligned in video frames. Based on the local neighborhood, the authors determined the local entropy of each frame. Later, the maximum gradient difference value is calculated and averaged with entropy information. Finally, it detected the textual lines using horizontal projection analysis and employed some empirically geometrical and heuristic constraints to eliminate noise. Afterwards, the identified candidate textual regions are extracted with the help of adaptive thresholding.

Beside these unsupervised techniques, several supervised techniques have also been proposed that evolved with the advancement in technology. For example, authors in (Bouaziz et al., 2017) developed a video browsing system based on its content. Author proposed an approach to extract static superimposed videotext regions and then classify it into semantic classes using deep CNN. Spatiotemporal approach is proposed for text extraction and localization process that consists of identifying edge maps as the square root of vertical & horizontal Sobel edge map of each frame, obtained edge frames are binarized and difference is calculated between consecutive frames to do further splitting on bases of edge density. Then similar density blocs are merged to form probable text regions, which are later filtered to remove false positive using TF and LSTF features. Finally CNN, consisting of 8 layers (1 input, 5 hidden convolutional and pooling layers for automatically feature extractions and reducing feature dimensionality, 1 fully connected layer with softmax activation function to act as classifier, 1 output classifier layer) to predict associated semantic class.

Authors of (Pan, Hou, & Liu, 2011) performed text detection, localization, extraction, enhancement, and recognition in complex background of natural scenes using a hybrid approach of region and connected component (CC) based information. Firstly, text

existence confidence is estimated by a text region detector, which also generates the scale information. Following that, the candidate text elements were produced with help of scale-adaptive local binary pattern. The non-text elements were removed by the use of conditional random filter model during the connected component analysis phase. Following the extraction of text elements, they were sorted into lines using a learning based minimum spanning tree. The hybrid model was evaluated on ICDAR 2005 competition dataset as well as multilingual image dataset. Proposed model where provides benefit of estimation of parameters by supervised training and else have its drawbacks which fails to account for the hard-to-segment text.

The authors of (Park et al., 2010) designed a Korean text detection and translation system for the tourists with minimum or no Korean language skills. The system's novelty was to handle the obstructions originated by distinct styles and size of font, with the illumination variation into the bargain. With the use of edge-histogram, the text from the image is obtained to be segmented and binarized with the help of fuzzy c-mean clustering. Following that, the characters from the extracted text region were then segmented for individual character recognition by connected component information, noise removal, and lastly merging the vertically and horizontally adjacent components. Minimum distant classifier is then used for the recognition of these individual characters. Shape-based statistical feature uses non-linear mesh and four directional segment values for the character recognition and generating the candidates, then characters were rearranged with the help of pairwise classifier. The candidates were then translated and run across the database of the shop names to find suitable result. The dataset used for the experimentation includes 445 images taken from Samsung smartphone run across the 2350 Korean text characters, where the text recognition accuracy can be seen up to 85.97%.

A method for text detection and recognition in videos was introduced by the authors of (Xu, Shivakumara, Lu, Tan, & Uchida, 2016) to classify the different type of texts, such as graphics-scene, 2D, and 3D texts in videos. Firstly, the graphic and scene texts were categorized as static and dynamic clusters respectively based on their velocity, thus generating the text candidates for both clusters. The static texts gets low deviation between pixels when compared between the frames, whereas the dynamic text gets higher deviation thus classifying them into two different clusters with the help of k-mean clustering. Due to uncertainty of static and dynamic cluster having graphic and

scene text only respectively, the gradient values of the clusters were computed to classify them into sub categories using again the k-mean clustering, thus generating the text candidates, which were of high gradient values. The candidates were then segregated using the values provided by ring radius transform to remove the false candidates from potential candidates. Following this, text region were segmented based on neighboring potential text candidates and the graphic and 2D-3D text were classified using skeleton difference components and variance of medial axis of gradient direction respectively. The method also proposes to classify the potential text candidates by tracking with use of Kanade–Lucas–Tomasi algorithm and generate stable text candidates. The data set used by the authors comprises of 500 2D and 300 3D video clips, which includes 960 static and 1298 dynamic texts. In addition to author's personal dataset, IDAR 2013 video data was also used as a standard dataset. The proposed methodology was then compared for text region detection and character recognition with other methods before and after classification of static, dynamic, 2D, and 3D text, with more than 83% classification rate.

Authors of (González & Bergasa, 2013) propose a two stage method for localization and recognition of text in natural images. In these stages, the model extracts the candidates, performs connected component analysis, text line aggregation and classification and then separates the words to perform character recognition, and using the language model to recognize the words. A combination of Maximally Stable Extremal Regions and locally adaptive threshold method of image decomposition were proposed for the localization of text candidates, which were then filtered, grouped and classified as text or non-text. Text line classifier were used to reject the mistakenly accepted text lines, and lastly the text were separated into words based on the distance between the characters in the word compared to other words, as the characters in same word were equidistant while the distance of adjacent characters of two words were comparatively higher. In last stage, i.e. text recognition, the characters in the words were recognized by Direction Histogram using the help of K-Nearest Neighbors and gradient direction features followed by an application of language model for rectification of spelling mistakes caused by OCR. The authors used multiple datasets for experimentation and comparison of results, first being the ICDAR 2003 Reading Competition dataset comprising of 258 images for training set, summing up to 5438 characters and 251 images for test set. Secondly, the data set used in ICDAR 2005

competition was same yet the authors compared their results with new challenge results. Lastly, the second data set used by author comprises of 420 images for training set and 102 images for test set, with a total of 3583 and 918 words respectively. The proposed methodology presents a precision of 81% and 89.23% on ICDAR 2003 and ICDAR 2011 datasets for text localization, 85.8% match on individual character recognition and 47.43% of correct word recognition for ICDAR 20036 data set.

A framework for detection and removal of embedded video text was proposed by authors of (Mosleh, Bouguila, & Hamza, 2013) by developing a text detection algorithm with the help of SWT, which generates connected components, and unsupervised classification. The feature vectors for connected component were used for segmentation of text and non-text components by k-mean clustering. After text locations were certain, CAMSHIFT algorithm were used on each frame of video to minimize it in tracked text objects. Authors used effective bandlet-based edge detector for edge detection and recreation of absent data to help the SWT where latter was done by spatio-temporal regularities. The novelty of the proposed framework was its ability to detect text regions before tracking. Authors used a wide data set of videos extracted from TV and movies etc. ICDAR 2003 and 2005 text location competition data set were also used a benchmark for evaluating the text detection of proposed method, and the results suggest a precision of 0.76 for the text detection.

Authors of (Koo & Kim, 2013) proposed a three stage text detection algorithm that includes generation and normalization of candidates and filtering out the non-text region in the scene using ML classifiers. The text candidate regions were created using the clusters that were classified by the connected components with the help of Maximally Stable Extreme Region (MSER) algorithm. The clusters of connected components were classified with the help of a trained AdaBoost classifier. In the second stage, i.e. text candidate region normalization, the text candidate region were normalized by geometric normalization and binarizing by approximating text and background colors. The normalized images were then filtered for removing the non-text region by dividing them into blocks and using a classifier developed by authors to allocate textness magnitude, which were used to classify the genuine block. Multilayer perception learning was used with the help of 2568 images for training while using one hidden layer comprising of twenty nodes. Authors used ICDAR 2005 text locating competition and ICDAR 2011 robust reading competition data set for experimentation

purposes. The results suggest a precision of 0.764, 0.791, 0.678, and 0.778 for Y, Lab, HSV, and YCbCr color spaces respectively for ICDAR 2005 dataset, whereas a precision of 0.812, 0.815, 0.707, and 0.814 for Y, Lab, HSV, and YCbCr color spaces respectively for ICDAR 2011 dataset.

The authors of (Gonzalez, Bergasa, & Yebes, 2014) proposed a framework to detect and recognize the text on traffic panels firstly by extracting the descriptors from key points, which were retrieved with the help of Harris-Laplace salient point detector. The descriptors were extracted by the application of blue and white color segmentation by Otsu's segmentation and MSER method respectively, using Bag of Visual Words approach for the detection of traffic panel. The extracted features were then clustered with the help of k-mean clustering and the cluster centers comprising of visual word represented the image with its histogram. The images of training dataset were classified either with simple probabilistic classifier 'Naïve Bayes' or supervised learning model called SVM. A two-stage method for localization and recognition of text was applied to the extracted images where traffic panels were detected. A combination of Maximally Stable Extremal Regions and locally adaptive threshold method of image decomposition were proposed for the localization of text candidates, which were then filtered, grouped and classified as text or non-text. For the sake of efficiency, the authors also proposed a modification in the algorithm as to utilize localization only on the regions given by color masking performed previously to detect traffic panel. Text line classifier were used to reject the mistakenly accepted text lines, and lastly the text were separated into words based on the distance between the characters in the word compared to other words. The characters in the words were recognized by Direction Histogram using the help of K-Nearest Neighbors and gradient direction features followed by an application of language model that also includes geographical dictionary. The data set used by the authors were the images from Google Street View in the country of Spain, which allowed the language model to create the language model dictionary by reverse geocoding service. The train data set comprised of 5514 images that were retrieved from Street View, whereas the testing data set contained 10763 images. The ROI for the traffic panel were above the road and on the side, whereas two types of boards, blue and white, were included in dataset. The panel detection rate for blue lateral and upper panels was as high as 95.23% and 97.77% respectively using TCH descriptor, whereas panel detection rate for white

lateral and upper panels was as high as 91.66% and 94.53% respectively using TCH descriptor. Text detection and recognition rate for all short distance lateral panels was 85.48% and 79.25% for words, 62.07% and 48.61% for numbers, 43.33% and 71.79% for symbols respectively. Text detection and recognition rate for all short distance upper panels was 92.92% and 79.19% for words, 83.44% and 61.42% for numbers, 46.27% and 90.32% for symbols respectively.

An exhaustive segmentation model of text detection was proposed by the authors of (Wei et al., 2017) in which the adaptive method extracts the character regions candidates by conversion of gray scale image to binary image with a bounded grey value, which yields the connected regions as candidates. These candidates are then subjected to filtering for pruning non-character candidates, in which the first layer comprises of a simple filter that eliminates evident non-character candidate regions, whereas the second layer uses SVM with RBF kernel to filter more. Following this, the output of filtering is subjected to Density Based Spatial Clustering of Application with Noise algorithm to avoid candidate region overlap. Finally, the text lines are then formed from the regions by creating a fully connected graph of candidate regions, and subjecting it to SVM model that eliminates the non-adjacent edges. The authors used ICDAR 2013 (229 training, 233 testing images) and SVT (350 images) datasets for experimentation. The results suggest a precision of 83.5% for text detection on ICDAR2013 dataset and 18.2% on SVT dataset.

Author of (Lu, Sun, Chu, Huang, & Yu, 2018) presented a methodology for detection and recognition of artificial text on videos. Firstly, the video is decoded by OpenCV library, and for the detection of multilingual text, the authors proposed corner response feature map approach to generate candidate regions including artificial text. After that, the candidate text regions are processed and binarized by grey scale morphology and adaptive threshold, respectively. The candidate text region may contain multiple lines, for which line localization for partitioning is performed with two different proposed methods, either by projection analysis onto contours or by projection analysis by FCM-based separation and conversion in grey-scale. At the final stage, developed transferred-deep-CNN classifier removes false positives. The proposed model is transferred from VGG16, Resnet50, and InceptionV3 by removing the few top layers and using it as deep feature extractor. After the removal of false lines, the true text lines FCM separation, Otsu binarization, and binary morphological dilation are used

to enhance contrast, remove burrs etc. to make it ready for OCR binary text. Authors used personal dataset of 2000 high-resolution frames acquired from random videos, as well as Microsoft common dataset, TV news, and YouTube dataset. The results show a precision of 0.83, 0.82, and 0.82 using proposed models of TVGG, TRESNET, and TINCEPTION model on authors' personal test dataset respectively. The proposed model also shows a precision of 0.96, 0.96, and 0.88 on Microsoft common dataset, TV news, and YouTube dataset respectively using TVGG model.

(Ansari, Shah, Yasmin, Sharif, & Fernandes, 2018) proposed a methodology for the text detection and recognition by contrast enhancement, MSER detection, and CNN. In the first step of text detection, maximally stable extremal region method is used for text region segmentation, which is applied on L channel of enhanced images, contrast wise using LUV color space. The resultant regions are then used to extract LBP and T-HOG, which in turn are then used for supervised learning classification of text-non text regions by feature fusion technique with the help of SVM, followed by character bounding. After the text region detection, the image is split into 26×26 patches, and text is recognized and labelled with the help of a supervised multilayer CNN trained by stochastic gradient learning. To reduce the error in the recognition, the distance between the labelled characters are measured by lexicon search, and with hamming distance true and false positives are separated and in case of false positive, an optimized word combination is suggested. Author used Char74k, IIIT 5K, SVT, and ICDAR 2003 dataset for training the model, whereas for the test dataset, more than 4000 images were taken from same datasets. The results suggest that the proposed model showed an accuracy of 97.1, 89.5, and 85.2 for text detection and 84.1%, 81.3%, and 82.9% for character recognition on ICDAR 2003, SVT, and IIIT5K dataset respectively.

A framework for cross-language word recognition and spotting is proposed by the authors of (Bhunia, Roy, Mohta, & Pal, 2018) for handwritten Indic script which is written in upper, middle, and lower cases. The proposed method relies on the similarity index of the source (training dataset) and the target (test dataset) character mapping. At the first stage, feature extraction from both the source and target script is performed for character modeling. Pyramid of Histogram of Gradient approach is used in SVM and HMM for feature extraction of upper/lower case modifiers and middle case components respectively, where the cases actually represent the zone of the written

script. The middle zone script is aligned and the information of middle zone components and upper/lower zone modifiers is combined for recognition of target characters using the mapping from source characters by look-up-table. Finally, in the recognition stage, the Lexicon is used in two stages, first one directly for upper/lower zone and the second one as modified for the middle zone components. Authors used 11,253 Bangla, 12,385 Gurumukhi, and 3,981 Devanagari words of Indic script for dataset and split it into train and test datasets. Authors experimented with the three above-mentioned languages in Indic script by replacing source and target script and calculated an accuracy as high as 82.18 with same source and target language and as low as 57.49 in cross-language word recognition.

For Chinese language, authors in (Ren et al., 2017) proposed a novel text detection algorithm using CNN. It uses an unsupervised CSAE to extract complex features and spatial pyramid layer to detect texts in multiple scales. In the end, a fully connected DBN is used to ensure that extracted features are comparable. it further boost. The proposed algorithm shows a significant 10% performance improvement over the baseline CNN algorithms. The author of (Yang et al., 2018) proposed a framework for the detection and recognition of Chinese characters. The novelty of the method was the implementation in historical documents in which localization is a difficult task due to dense scattering of characters. Firstly, text lines are segmented by vertical projection of grey scale images, after which the authors proposed a novel method called Recognition Guide Detector for detection of characters comprising of two trained CNN models. The first model furnishes the clarifying context of the text in image, which is then used in the second network for the localization of the characters. The former one, titled as RGPN, helps in recognizing by pre-training with the help of convolutional, fully connected multiple, and CTC layer. The latter network of character-level detection uses the output of former RGPN to accurately map and deduce the position of Chinese characters. The authors used Tripitaka Koreana in Han and Multiple Tripitaka in Han data set for training and testing purposes, which included 323,491 and 197,886 characters respectively out of 1500 images. The results of the experiment suggests an accuracy of 97.25% for the recognition of Chinese characters in historical documents.

The authors introduced a novel method in (Shi, Bai, & Yao, 2017), an end-to-end system for image-based sequence recognition and tested it on scene text recognition

with help of hybrid model of Recurrent NN and Deep CNN, named as CRNN. It uses a convolutional layer to extract feature sequence for each input image, on which recurrent network makes prediction and finally transcription layer translates the predictions into a label sequence. The author's model consumes less storage space by removing the fully connected layers of Deep-CNN and only using convolutional and max-pooling layers for sequence features extraction. It uses deep bidirectional RNN to get contextual cues for sequence recognition which is stored for long period of time with help of LSTM units in RNN. In the end transcription layer convert the prediction into label sequence.

A method was proposed by (Yan et al., 2018) for text detection of a minority language Uyghur for intelligent vehicles to be used in identification of traffic prompts. The authors propose a channel enhanced MSER for component candidate detection, from which the regions with no characters are eliminated with the help of two-layer filtering using a simple filter for elimination of repetition and an SVM classifier trained with HOG features to prune non-text regions. Rest of the text regions, word candidates, are then connected and extended to attach the omitted MSERs. Lastly, another two-layer filter, chain identifier and RF classifier, is used to eliminate the non-text chains by firstly eliminating chains of unreasonable aspect ratio and then identifying the remaining chains by the help of classifier. Authors used a dataset called IMAGE570 comprising of 370 training and 200 testing images with majority of text in Uyghur and the rest in Chinese or English. The results suggest an accuracy of 85.3% for the detection of Uyghur text using the proposed method.

To detect the languages such as English and Chinese, authors in (Zhou et al., 2017) proposed fast and accurate algorithm, named EAST detector. It detects the text of arbitrary orientations using FCN model that helps in eliminating the intermediate steps (candidate aggregation and word partitioning) and efficiently output dense per-pixel predictions of text lines or words. The model consists of three parts: feature extractor stem, feature merging branch and output layer. The FCN model generates multiple channels of pixels-level text score map and geometry, on which thresholding is applied to get valid predictions. The predicted words or lines are then send to NMS to get the end results. By incorporating proper loss functions in this single light-weighted NN algorithm reveals promising test results that outperformed many stat-of-the-art methods in speed and accuracy. The author made combination of proposed algorithm

with VGG16 base network, PVANET and PVANET2x lightweight networks for feature extractor to test on different dataset. VGG16 base network having small receptive field in last convolutional layer thus limiting the capability of network to predict longer text regions.

The author of (Zhao et al., 2019) suggest a method for the text detection and segmentation by assigning weights to the stoke pixels of text box regions. The authors propose a method called Elite Loss, which is a modification of classification loss function, where it assigns less weight to the in-box not-on-stroke pixels to increase the outcome of training and computation efficiency. The low confidence score of not-on-stroke pixels due to its resemblance with scene helps identify the elite and non-elite which helps down weighing the not-on-stroke pixels. The authors used EAST detector for the experimentation to prove the result of Elite Loss efficacy on data set of ICDAR 2015, COCO-Text, and MSRA-TD500, which suggests an F-score of 0.855, 0.425, and 0.819 respectively.

A text detection by regression model for multi-lingual and multi-oriented detection was presented by the authors of (W. He, Zhang, Yin, & Liu, 2018) where the text region candidates are generated by down sampled segmentation whose boundaries are then regressed. The detection is performed by a bi-task prediction model, which classifies the text and non-text based on pixels, and regress based on pixel to evaluate the vertex coordinates. Firstly, the model performs the convolution feature extraction of the input image by VGG-16, ResNet-50, and S-VGG, followed by multi-level feature fusion to create feature maps. After the feature fusion, the output of multitask learning module are then subjected to post processing. Word level annotations are subjected to NMS, whereas the line level annotations are subjected to Line segment grouping method. Authors used ICDAR 2013, ICDAR 2015, and MSRA-TD500 data set for training the model by stochastic gradient descent for English text detection. Similarly, RTW-17 and CASIA-10K dataset were utilized for the training and testing of Chinese text detection, where training dataset comprised of 229, 1000, 300, 8346, and 7000 images, respectively, and testing dataset consists of 233, 500, 200, 4229, and 300 images. The results suggest a precision of 0.95 for ICDAR 2013, 0.89 for ICDAR 2015, 0.91 for MSRA-TD500, and 0.81 for CASIA-10K dataset.

Another multi-oriented text detection framework was proposed by the authors of (Wei, Shen, Zeng, Ye, & Zhang, 2018) which used the character region extraction method proposed by (Wei et al., 2017). The adaptive method extracts the character regions candidates by conversion of gray scale image to binary image with a bounded grey value, which yields the connected regions as candidates. These candidates are then subjected to filtering for pruning non-character candidates by implementation of a 7-layer CNN model applying a kernel size of 5×5 per layer. The last step of detection has two stages for cutting of the non-adjacent edges to group the text lines. In first stage, a connected graph of candidate regions is created, and in second stage, a trained algorithm eliminates the non-adjacent edges. The authors used ICDAR2013, and USTB-SV1K dataset for training, which comprises of 15600, and 20000 images respectively. In addition to that, the model was tested on ICDAR2011, ICDAR2013, USTB-SV1K, and OSTD where the results suggested a precision of 86.9%, 87.3%, 54.1%, and 75.4% respectively. The latter two datasets mentioned were oriented scene text datasets.

Authors of (Raghunandan et al., 2019) suggested a framework for multi-oriented text detection and recognition in digital images. The text detection is a three-fold approach in which firstly the components are detected by bit plane slicing from which the candidate planes are spotted with a proposed iterative nearest neighbor symmetry method. This method evaluates inadequacy of individual components by sketching convex and concave boundaries and finding the similarities in the shapes of these sketches. Following that, a new method called mutual nearest neighbor pair is put forward by the author that which determines the symbolic pairs in individual bit plane, which in turn aids the restoration and detection of text. Lastly, recognition of text is divided into two stages. In first stage, a window size is perceived to draw out the text attributes in whichever direction with the help of HWHF sub-band, and in the second stage the features are used to find characters with SVM as well as HMM for text of any orientation. Authors used the ICDAR2011, ICDAR2013, ICDAR2015, YVT, SVT, and MSRA-TD500 dataset for training and testing of the proposed model and the results suggest precision of 76.3%, 84.5%, 62.8%, 78.8%, 60.4, and 67.2% respectively.

A method for text detection and recognition in video by classifying them into different classes for enhanced efficiency was proposed by (Roy et al., 2018). The main frames

and the adjacent temporal frames are obtained from the video to categorize them based on the kind of video and canny edge images are extracted for edge component detection. Based on the geometric shape of the edge component, the video frames are classified into eight groups with the help of a proposed rough set and fuzzy logic. The former helps in gauging lower and upper bounds, whereas the latter helps the complete recognition of shape by pruning the overlapping of shapes. Following this, distinct features are evaluated by investigating the gradient direction by dividing them into planes and locating local distribution to obtain correlation between strokes. Lastly, the features of a plane within and with other planes are chained to form a matrix, analyzed with neighboring frames, which are then classified into categories based on type of frames with the help of a NN classifier. The text is recognized by OCR. Authors used 27,000 images for the training of the model, and used random internet available videos for the testing purpose due to lack of available categorizing dataset. A total of 30,000 frames were used for the testing purposes which were categorized into 10 different classes. The authors applied the classification method by overlapping on already present almost 15 methodologies to test the results, which suggested a significant increase in text detection performance of present methods.

The authors of (Y. Tang & Wu, 2018) propose a method for text detection where the candidate character regions are extracted a developed method called superpixel-based stroke feature transform in which regions are obtained by dividing the image into multiple regions with the help of superpixel-based clustering and simple linear iterative clustering followed by pruning the extra regions by relying on already established standards, filtering the rest of regions by structured random decision forest to form edge probability and gradient orientation map, and calculating the stroke width map. In the second stage, the output of the first stage is subjected to deep learning based region classification where character regions are determined by obtaining low level features by Histogram of Gradients, geometric features, and high level features and meld by fully connected networks in order to categorize the regions. The high-level features are extracted by deep CNN modified from VGGNet-16, which along with fully connected network are trained together. The character regions determined in the second stage are then joined together to construct the candidate text region, out of which the actual text is detected. Authors used ICDAR2011, ICDAR2013, and SVT datasets comprising of a total of 629 training images and 737 testing images for the

experimentation purpose. The results suggested a text detection precision of 0.906 for ICDAR2011, 0.911 for ICDAR2013, and 0.541 for SVT dataset.

Researchers in (Pei et al., 2018) proposed a methodology for text tracking in videos using relationship between trajectories in graph mapping by tracking text region in each frame. The prediction of text trajectory is performed firstly by detecting the text block using a detector and then creating forecast for the coming blocks using a tracker and current trajectories. Following that graph, matching is utilized to connect current trajectories and candidates. Affinity matrix and few-to-many graph-matching methods are utilized to decide either to keep the tracking using the existing trajectory or stop it. Meanwhile, the non-text trajectories are pruned by text discriminator. Re-weighted Random Walk Graph Matching is used to match text candidate in existing frame and trajectories. The authors use ICDAR2015 and MINETTO with Multi-Object Tracking Precision and Multi-Object Tracking Accuracy to evaluate the results, which suggest a precision of 72.11 and accuracy of 48.19 on ICDAR2015 dataset, whereas precision and accuracy as high as 0.821 and 0.971 respectively for MINETTO dataset.

A text detection method presented by (Zhang, Gao, & Tian, 2018) utilizes color prior guide to gather the character candidates. The methodology used Maximally Stable Extremal Region to extract textures that are similar to stroke with a redesigned Stroke Width Transform to overcome the noise issue. The input image is subjected to edge detection and segmentation prior to Stroke Width Transform, and single-link clustering is used stroke filtering and clustering. The extracted candidates are then classified using two CNN trained on grey scale characters and binary characters, later fused together to enhance the validity. The authors performed experimentation on ICDAR 2013 and ICDAR 2015 dataset, which provided a precision of 89.07% and 55.74% respectively.

The authors of (Jiao, Wu, Wang, & Bie, 2018) proposes a method for recovery of absent characters in detected text from images. In the given method, the false text occurrence in an image is identified with the help of hybrid Bi-LSTM (two bidirectional layers with 256 units) and CNN (comprising of leaky ReLU, max-pooling and batch normalization layers) architecture. After the identification of corrupt text, the characters are then deduced with the application of Bayesian posterior probability. Following that, the potential position of absent character is estimated with the application of statistical histogram resulting into text recovery by placing the inferred

character into estimated location. The authors experimented their method on images with corrupt text instances on white background, textured background, textured background with colorful words, and real world images with single and multiple missing characters. The results of the experimentation were not represented in form of numerical figures but were portrayed in form of result images of recovered text. Multiple images, all shown in article, depicted very promising results in recovering the missing characters.

Author in (W. He, Zhang, Yin, & Liu, 2017) used FCN plus one-step NMS for multi-oriented scene text detection. The proposed architecture comprised of convolutional feature extraction, multi-level feature fusion and multi-task leaning. Based on direct regression, it successfully localize quadrilateral boundaries of scene text. Researchers in (Tian, Lu, & Li, 2017) introduced a slight variant in this technique by incorporating regression based deep network and named it as weakly supervised scene text detection (WeText) method for character based approach that trains model by learning from unannotated or weakly annotated data. The architecture is based on standard VGG-16 with slight modification of last two layers as convolutional layers instead of fully-connected layers to speed up computation and with help of additional convolutional layers at end of base network to predict at multiple scales. Finally, NMS is applied to reduce detection boxes. The proposed detector is pre-trained on a small number of character level annotated text images and performance is enhanced by running the model on larger amount of weakly annotated word level annotated images to get candidate characters and bounding box at word level. On resultant, predefined confidence threshold is used to identify positive samples which are then combined with annotated image set to train detector more accurately. This proposed weakly supervised learning performs well than many state-of-the-art methods but fails to handle handwritten texts or ultra-low contrast text.

A scene text detection model titled TextBoxes++ was introduced by the authors of (Liao, Shi, & Bai, 2018) which detect the text in scene of any direction with a simple NMS afterwards. The proposed method is an end-to-end CNN evaluating the discretionally oriented text, which is a modified form of Single Shot Detection algorithm by portraying the discretionally oriented text using either quadrilaterals or triangles. The network then forecasts the regression standard boxes to text portrayed by the shapes. To get along with the text that can be closely packed in some region,

the standard boxes of network are also crowded with vertical offsets. Since the proposed model is modification of an object detection algorithm, convolution kernels are accommodated the lengthy text lines represented as lengthy objects.

Authors of (Li, Wang, & Shen, 2019) proposed a method for license plate text detection and recognition for vehicles in a single forward pass where detection and recognition is performed with a single unified deep NN, which eliminates any heuristic process, segmentation, and clustering etc. Firstly, the low-level features are extracted using VGG-16 comprising of 13 convolutional layers and ReLU, max-pooling layers etc. These low-level features are then fed to region proposal network, trained end-to-end with back propagation and Stochastic Gradient Descent, which in gives potential bounding boxes that are plate proposals passed through rectangular convolutional filters later. 256 anchors used on training of Region Proposal Network help in the detection and recognition, as the positive anchors of input image provide the proposal after processing and the extracted features from the proposal and convolutional feature map are used for ROI pooling in a single channel. In order to rule out any uncertainty of ROI not being a plate, two fully connected layers are used to obtain certain features of ROI and are provided to two distinct linear transformation layers for classification and regression for identifying the possibility of ROI being a vehicle's license plate or not as well as creates the coordinate offsets. Bi-directional recurrent neural network along with CTC is utilized for labelling the plate. Following this, two convolutional layers with ReLU are introduced with a rectangular pooling window in between them, which aids in the recognition of the license plate. The authors used four datasets, firstly CarFlag-Large comprising of 322000 training and 138000 testing images, and secondly Application Oriented License Plate comprising of 4098 testing and 2049 testing images, thirdly Caltech-cars1999 dataset with 1626 testing and 126 testing images, and lastly PKUData with 3977 images. The results of the experimentation suggest 97.13% end-to end performance and 98.33% detection only performance on CarFlag-Large dataset. For the Application Oriented License Plate dataset, the results suggest as high as 95.59% end-to end performance and 99.12% detection only performance. For the Clatech-cars the end-to end performance was 94.12% and detection only performance was 98.04%. Results for PKUData were reported as 100% for detection performance.

A detection method for lengthy text was proposed by the authors of (X. Gao, Han, & Luo, 2019). The authors use a NN for text detection that utilizes single-shot detector, which uses feature maps along with an added layer to detect horizontal texts. The output is then subjected to NMS to remove unnecessary bounding boxes and provide initial detection results. Larger aspect ratio and default boxes with vertical offsets are used proposed for enhance modification of distribution characteristics of candidate areas. The text detection layer feeds input to NN for the localization of text. Following which a text verification model to prune the false positive text areas and recognize the true ones is applied which is reliant on encoder-decoder network comprising of CNN model and Bidirectional Gated Recurrent Unit, where the former is used to obtain features of the input to verification model and latter is used for encoding those attributes. Following to that the decoder comprises of Gated Recurrent Unit to decode the attributes, a fully connected layer to recognize the text, which is then generalized by softmax function. The authors used ICDAR2017 for Chinese text reading (RCTW2017). The results suggested a precision of 83.0 for text detection on the aforementioned dataset.

The authors of (Cheng, Cai, & Wang, 2019) propose application of positive-sensitive segmentation for the pruning of false positives when direct regression is utilized for the text detection. The authors suggest that extra knowledge related to comparative location of text is supplied by training positive-sensitive segmentation, which enhances the conveying quality of network. A spatial pyramid of suggested segmentation is also familiarized in the framework due to dissimilarities in dimensions and feature proportion of text in the image. The deduction capability of the framework was enhanced by the addition of positive-sensitive corner area of interest. The feature extraction as the first step is performed with FCN comprising of ResNet-50 and bottleneck convolution which are then fed to direct regression module and positive-sensitive segmentation. The former comprises of score maps and geometric maps, and the latter involves the spatial pyramid position-sensitive segmentation maps to prune the false positives and avoid dissimilarities in dimensions and feature proportion. In the last stages, the unnecessary candidates produced by regression are pruned by locality-aware NMS, which are then subjected to positive sensitive corner area of interest pooling to gather the attributes of candidates and compute individual score to prune false positive candidates. Authors used datasets of ICDAR2015 with total 1500

images, MS-COCO comprising of 63686 images, and MLT-17 consisting of 18000 images, with 51886 training images, 1800 validation, and 29500 testing images. The results suggested an F-score of 0.86, 0.47, and 0.68 for the aforementioned datasets respectively.

Another method for the detection of multi-lingual license plate detection and recognition was presented by (Kessentini, Besbes, Ammar, & Chabbouh, 2019). The framework includes two deep NN, in which the first one utilizes YOLOv2 for the detection of license plates and the second one is exhausted for the recognition. The send network follows two different methods for recognition, among which the first one the network recognizes the license plate without foregoing segmentation of image and the second method uses component level analysis for recognition. To make the framework more efficient, the authors also put forward a semi-automatic annotation procedure. The YOLOv2 module first detects a license plate in the image and then the cropped image of license plate is subjected to segmentation free CRNN recognition system and YOLOv2 recognition system. In the former model, the sequential attributes of image are extracted with CNN (based on VGG) which are subjected to recurrent layers using Bi-LSTM trained for recognizing features while avoiding gradient vanishing followed by CTC whose output is decoded for the recognition. The latter method of YOLOv2 which carries out segmentation as well as recognition as a function of detection in which the input image maps a grid of ground truth for after detection and bounding box of highest confidence is used as the output of the system. Authors used GAP-LP dataset and Radar datasets comprising of 9175 and 6448 images where 1115 images were used for training, 906 images for validation, and 3602 images for testing purposes. The results suggest detection and recognition rates between 90-98% on GAP-LP and Radar data set using CRNN and YOLO methods.

Authors of (Y. Liu, Jin, & Fang, 2020) proposed a text detection methodology for arbitrary shaped text using Mask Tightness Text detector. The proposed detector utilizes tightness prior, polygonal boundary, and text frontier learning to intensify prediction and localization. Firstly, a Region Proposal Network, trained by Resnet50-fpn, is utilized to produce region proposal, which are then primarily subjected to tightness prior and secondarily to ROI align which uses bilinear interpolation. The output of these are used for the training of region based CNN, which comprises of direct regression curve text detector and mask prediction, where the former is modified

by the knowledge of border of text regions. Authors used ICDAR2015, RCTW-17, CTW1500, and MSRA-TD500 datasets for the experimentation with a precision of 86.6%, 77.2%, 79.7%, and 885.7% respectively.

A method presented by (Chaithanya, Manohar, & Bazil Issac, 2019) detects and classifies the text using MSER and stroke width. Primarily, the input image is subjected to preprocessing using median filter for pruning the noise, followed by segmentation of image with thresholding method to obtain RoI and then comes locating the Maximally Stable External region to classify text and non-text regions. After that, the non-text region is removed with the help of stroke width variation, and CNN is utilized to attain the features and bounding boxes are expanded to identify the text in the image. Lastly, OCR is used recognize the text that has been detected by the CNN. Authors used ICDAR2017 for training and testing purposes and reported an accuracy of 92.31% for detection and classification of text.

Authors of (Y. Liu, Jin, Zhang, Luo, & Zhang, 2019) presented a framework for the detection of curved text using a polygon based curve text detector. The authors built a dataset titled CTW1500 comprising of 1500 images to draw up the detector. The model comprises of convolutional layers based on Reduced Resnet-50 and VGG-16, regional proposal network, and a regression model. Regional proposal network produced candidates for recollecting text with the help of rectangular anchors and regression model tuned the candidates to assemble it tighter using curved locating points, which are stimulated by EAST and DMPNet. Recurrent transverse and longitudinal offset connection is used for the localization of curved regions, and the RNN is connected to positive sensitive ROI pooling to calculate probabilities and offsets. False positives that do not fall under the shape of polygon are eliminated by non-polygon suppression followed by NMS on the results. Authors used CTW1500, Total-text, and MSRA-TD500 dataset for experimentation. The results suggest a precision of 77.4% detection on the entire set of CTW1500, but a precision of 57.1% detection on curvet set. Furthermore, precision of 74% and 84.5% were reported for text detection on Total-text and MSRATD-500 datasets.

A framework for the retrieval of textual images with text detection was proposed by (Unar, Wang, Zhang, & Wang, 2019). The authors used MSER for the extraction of text regions, which uses eccentricity, extent, solidity, Euler number, and size of the image to eliminate the non-text objects, followed by pruning false positives by stroke

width transform. Bounding boxes are used for localize the candidate regions from where the text is recognized by OCR and the image is retrieved using the keywords detected with neural probabilistic language model. The keywords are assessed based on Cosine similarity, Euclidean, Canberra, and Manhattan distance to measure the resemblance among examined and index images. Authors used ICDAR2003 dataset comprising of 251 training images and 249 testing images. In addition to that, authors put forward a new dataset called Sindh, comprising of 1000 random images taken from internet, social media, and natural scenes. Precision for text detection was reported as 0.74 and 0.75 for ICDAR2003 and Sindh dataset respectively, whereas the performance of retrieval for ICDAR2003 and Sindh dataset was reported as 0.63 and 0.71 mAP respectively.

Authors of (Y. Huang, Sun, Jin, & Luo, 2020) proposed an end to end trainable encoder decoder network for text recognition with the extraction of high level features with a Text Image Encoder. They put forward an effective part attention CNN comprising of text image encoder that transforms multidirectional attributes map into single dimension vector that are attached to Bi-LSTM for encoding. In the second stage, the character effective part decoder evaluates the location of characters with the help of novel glimpse network by assigning weights to features, followed by creating a mask using refinement network to slowly restrain the pinpoint location of effective parts of character. The mask is created by the help of a refinement network. Authors used Synth90k and SynthText synthetic datasets for the training the model which comprised of 90000 words and multiple images. The testing was performed on IIIT5K-Words, SVT, ICDAR2003, ICDAR2013, CUTE80, SVT-Perspective, and ICDAR2015 where the results suggested an accuracy rate of 94.0, 88.9, 95.0, 94.5, 82.6, 79.4, and 73.3 for respectively.

A text detection model was presented by the authors of (F. Liu, Chen, Gu, & Zheng, 2019) working in combination of CNN and RNN. Feature pyramid network is utilized in the former for deriving the different dimension attributes to get a feature pyramid. It is then encoded using Bi-LSTM, which is connected to fully connected layer that detects specific global contributions of features to generate candidates, whose output is subjected to ROI pooling for classification and regression to generate coordinates and scores of provided text candidates that are chained together in a line with a text-connector. Authors used ICDAR2013, ICDAR2015, and USTB-SV1K for the

experimental purposes where the results suggest a precision of 0.932 and F-measure of 0.925 for the first dataset, a precision of 0.682 and F-measure of 0.728 on second dataset, and a precision of 0.614 and F-measure of 0.626 on the third and last dataset.

A method for text detection by the use of contextual attention module trained by gradient-inductive module was presented by the authors of (Cao, Zou, Yang, & Liu, 2019). The former module restrain the unnecessary background and call attention to the primary attributes, whereas the latter module produces feature layers. The framework allows the implementation of the presented module in the U-net for segmentation. Firstly, the attributes of the image are obtained by VGG-16 network, followed by U-shaped feature fusion network, which is upgraded by the previously mentioned modules, for the purpose of segmentation. After passing the CAM and GIM, classification begins as text and non-text as well as the linkage forecasting to produce connected components candidates. In the post-processing, the pixel score map and link score map are processed into segmentation map and then subjected to bilinear interpolation. The authors used ICDAR2013, ICDAR2015, and MSRA-TD500 where the results suggest a precision of 92.8%, 89.1%, and 86.3% with an F-measure of 92.1%, 87.3%, and 81.4% respectively.

Authors of (Zhong, Sun, & Huo, 2019) proposed a text detection method by increasing the accuracy of text localization with the use of LocNet module in images. In normal region proposal networks, the regression module instantly forecasts the bounding box coordinates of text candidate, whereas in the proposed model, authors expand the proposal to generate an explore region where a conditional probability is assigned to each rank of the region which in turn can be utilized to precisely deduce the relevant bounding box. Authors used Fast Region based CNN comprising of region proposal network that generates rectangular RoI and secondly a detector for classification and refinement. Online hard example mining algorithm is utilized to prune the difficult background regions. Authors also proposed detection of multi-oriented text that transforms the detection problem into a horizontal detection by deducting rectangular bounding boxes as well as text from de-rotated regions. Authors experimented on multiple datasets with horizontal orientated and multi-oriented text. For the horizontally oriented text datasets, the results suggest precision and F-measure as 89.82% and 89.41% for ICDAR2011, precision and F-measure as 87.53% and 90.61% for ICDAR2013, and precision and F-measure as 84.96% and 84.70% for

MULTILIGUL dataset respectively. In addition to that, for multi-oriented text, precision and F-measure for ICDAR2015 dataset were reported as 80.12% and 83.78% respectively, whereas precision and F-measure for MSRA-TD500 dataset were recorded at 81.39% and 79.66% respectively.

Unlike all other approaches discussed previously, a text detection method on the live stream of a camera was proposed by the authors of (Francis & Sreenath, 2019a) by filtering the live image to determine if the image is worth the text detection or is it false positive. The authors modified the scene text recognition by adding an extra layer of preprocessing the image for reducing the false positives after the image cleansing. The overall framework of the method includes Train function that charge up the training dataset, i.e. CIFAR100, COCOText, ICDAR Born Digital, ICDAR Focused Scene, and IIT5K, followed by derivation of bottleneck value and determination of hyper-parameters, which are used to train the model that in turns provide the final layer as argument to be transformed into flat buffer. This model is launched into the mobile phone, from where the second function called Sleuthing starts by transforming live stream frames from the camera into Bitmap. These images are then subjected to the trained model to evaluate the probabilities of text and non-text images and of the ones with high probability, above the threshold, are accepted for the segmentation and then subjected to character recognition. SVT, MSRA500, and KAIST datasets were used for the validation of the proposed methodology and results suggested a precision 99.31%, 99.68%, and 99.56% respectively whereas F-measure was reported as 99.25%, 99.69%, and 99.49% respectively.

The authors of (Yuchen Zheng, Iwana, & Uchida, 2019) proposed a method to enhance the performance of CNN based framework of text recognition by extracting displacement features from the pooling layers. During the down sampling in max pooling critical spatial information is lost, which the authors claim to avoid by extracting displacement features, which register the position of maximal value, and then merging them with the mx pooling features to evaluate the systemic distortion. In addition to that, the cosine features are also extracted from the displacement features as well as principal component analysis for class-wise similarity, all of which are then combined at the end to indicate the position of maximal value. The authors used HASY and Char74K-font dataset for the training and testing of model. The results suggest an

accuracy of 0.823 for classification on HASY dataset, whereas an accuracy of 0.9145 on Char74K-font dataset.

Author of (Luo, Jin, & Sun, 2019) proposed a text recognition framework based on multi-object rectification trained using attention based sequence recognition network. The former is developed to pinpoint images with irregular text whereas the latter reads the irregular text. The former actually refine the images with irregular text, such as slant text becomes horizontal, and the latter then takes the refined images and generates the predicted words. Authors used deformable convolution network for the rectification of irregular text in the images and prediction of offsets. Attention based sequence recognition network is a Bi-LSTM. The framework starts with pooling before convolution and prepares offset maps, multi-object rectification network prepares the basic grid and resized offset maps. These all are combined to form the sampling which is then passed on to second stage of method that is attention based sequence recognition network where the rectified image is subjected to BLTSM followed by the decoder that has the knowledge of labels and targets ends up into the output as recognized text. Authors used IIIT5K-Words, SVT, ICDAR2003, ICDAR2013, SVT-Perspective, CUTE80, and ICDAR2015 dataset for horizontal and multi-oriented text with 3000, 647, 251, 1015, 645, 80, and 2077 images respectively for testing purpose. Based on an evaluation matrix developed by the authors, the proposed methods was reported to have a performance rate of 91.2 for IIIT5K-Words, 88.3 for SVT, 95.0 for ICDAR2003, 92.4 for ICDAR2013, 76.1 for SVT-Perspective, 77.4 for CUTE80, and 68.8 for ICDAR2015.

An encoder-decoder method for the text recognition was proposed by the authors of (Zuo, Sun, Mao, Qi, & Jia, 2019). The model starts with extracting features and creating a feature map with help of CNN. The feature map is then subjected to Map-to-sequence operation to form feature sequence and a bidirectional analysis is done on it with a multilayer Bi-LSTM, and the image features are then encoded along with. At the second stage of the framework. The authors presented an integrated Connection Time Classification-Attention mechanism, which incorporates one way LSTM to decode the text sequence encoded previously during sequence analysis. The end-to-end training model used Synth90k dataset for training purpose, whereas multiple publically available datasets such as ICDAR2003, ICDAR2013, SVT, and IIIT5K with 3000, 251, 647, and 1015 images respectively were used for testing purposes. The

results for the experiments suggest that when the 50 words dictionary were assigned to each image, the accuracy of proposed system was reported to be 98.2 for ICDAR 2003, 95.6 for SVT, and 98.0 for IIIT5K. Furthermore, when 1K words dictionary was assigned to each image, the accuracy was recorded as 97.1 for ICDAR2003, and 97.1 for IIIT5K. In addition to that, when no word dictionary was assigned to any image, the accuracy of the proposed system was reported as 91.9 for ICDAR2003, 91.0 for ICDAR2013, 84.5 for SVT, and 85.4 for IIIT5K dataset.

Authors of (Lei et al., 2019) proposed a text extraction strategy of Wagon text using pre-trained models as base detection network, that includes CTPN, EAST, and TextBoxes++ by retraining these models on wagon text. The authors use unsupervised learning model of Defect-Restore Adversarial network trained on synthetic images to enhance robustness, composed of Generator and Discriminator for extracting distinctive features and recognizing the Wagon text. In the proposed network, the images are first subjected to random mask imaging to dial down the vulnerability of encoder. The network is also composed of encoder-decoder-encoder apprehend diffusion of input. RNN composed of two Bi-LSTM layers to recognize the text sequence which is then transformed into text at transcription stage. When the proposed method was used to refine the already existing models, the results suggest the F-score of CTPN to increase from 0.5367 to 0.9107, F-score of EAST increased from 0.2635 to 0.6089, and finally the F-score of TextBoxes++ increased from 0.3385 to 0.8106 from pre-trained model to refined model.

Authors in (Zayene et al., 2016) proposed a novel text detection approach for Arabic news videos, composed of CC-base heuristic approach using adapted version of original SWT algorithm and deep learning-based approach based on CAE as feature extractor and SVM as text line verification method. Firstly, in heuristic-based phase, connected component text candidates are extracted using SWT operator, filtered by applying geometrical constraints and grouped by text line formation method. Then in machine learning-based phase, CAE automatically produces features that have been learned from previously obtained unlabeled text line candidates. Later SVM classifier uses these features to classify text line candidates as text or non-text. Authors approach showed some promising results as compared with two other state-of-the-art algorithms, but still needs improvements. Future task is to replace the combination of

CAE and SVM by stacking a NN on top of the auto-encoder to fine-tune the features for classification task.

Same year, authors in (Iwata, Ohyama, Wakabayashi, & Kimura, 2016) proposed a complete OCR system for video indexing and retrieval of Arabic News videos with moving news caption, and it also detects the frame transition by applying inter-frame text difference technique. The image is binarized using Otsu's threshold method and non-text connected components are removed by size comparison with image. Text lines are detected by applying 1-demensional difference of Gaussian filter (DOG) to vertical profile analysis to overcome problem of over separation of diacritics and adherence of lines. In next step, the noise from moving news caption is removed by shifting odd number of scan lines to left by a pixel. False text lines are removed by averaging the eccentricity of connected components and words segmentation is done by Otsu's discriminant thresholding analysis to classify horizontal gaps between connected components. Finally, transition fames are detected with help of inter-frame text difference technique. The resultant OCR outperformed the commercial ABBY FineReader OCR by huge with recall from 23.82% to 96.29% for interlaced moving news caption images of AlJazeera broadcasting programs.

Another deep learning based approach for Arabic character detection and recognition is presented in (Ahmed, Naz, Razzak, & Yousaf, 2017). Author presented Arabic text detection and recognition method with Convolutional NN (ConvNets) as deep learning classifier. They use Bi-LSTM network as a classifier to learn the detected scene images. The input image is converted into grayscale and fed with five various orientations, as Arabic is cursive scripts, to convolution layers with ReLU activation function to remove the non-linearity and features are extracted from max pooling. Finally, the features are passed to fully connected layers to do the character classification.

A segmentation free method for Arabic hand written text recognition from out of vocabulary words was proposed by the authors of (Jemni, Kessentini, & Kanoun, 2019). Three lexicon-operated methods were used in the recognition system. Firstly, Word Lexicon Driven method helps in hypothesis generation based on Word Statistical Language Model, followed by PAW Lexicon Driven, and Morpheme Lexicon Driven method for the recognition. The framework comprises of M-LSTM and CNN layers with max-pooling which produce feature sequence of grey image,

along with HOG feature reliant recognition framework to validate the efficacy of proposed system. Out of the vocabulary words were detected based on the lexicon driven methods, firstly by calculating the confidence score hypothesis, secondly by distinctions among word hypothesis, and lastly by sub-words modeling. Finally, the words are recognized with an integration of weighted finite state transducers and CTC. Authors used two datasets called KHATT comprising of 690 training pages, 141 test pages, and 148 validation pages, and AHTID/MW consisting of 2819 training hand written lines and 887 test lines. Multiple experimentation results were presented using three different lexicon driven methods and two different recognition methods. The results suggest word error rate as high as 46.42 with morpheme lexicon driven method with CNN and MDLSTM, and as low as 21.96 with same method for KHA database. In AHTID/MW dataset, using CNN-MDLSTM the WER is as high as 25.63% and low as 18.13% with Word lexicon driven method and Dynamic Word Lexicon Drive respectively.

Authors of (Y. Gao, Chen, Wang, Tang, & Lu, 2019) proposed a method for reading text by using stacked convolutional layers and a feature encoder. The former efficiently encapsulates the long-term reliance between components of image, and the latter prunes the noise and improves the front text. The authors utilized stacked convolutional layers and CTC in their approach. First part of the model is attention feature encoder comprising of dense blocks and residual attention for computing the feature sequence, which are then sketched on a 2D map. In the second stage of model, convolutional sequencing modeling network takes the map, which is the output of first stage, and encapsulates the efficacious knowledge and memorize the dependency, which provides back the feature map with individual component transformed into probability distribution. At the last stage, connectionist temporal classifier renders the probability distribution as the recognized text. The model was trained on synthetic dataset comprising of eight million images. For the testing purposes, SVT, IIIT5K, ICDAR2003, and ICDAR2013 dataset are used with a total of 5522 cropped text images. The results suggested an accuracy of 97.4 and 86.9 for SVT dataset with lexicon 50 and 1k respectively, whereas an accuracy of 97.9 and 81.8 for IIIT5K dataset for lexicon 50 and 1K respectively. In addition that, the accuracy for ICDAR2003 and ICDAR2013 dataset was reported as 92.9 and 93.9 with unconstrained recognition.

A real-time text detection method was proposed by (W. He et al., 2020) where the authors localized and approximated the scale by a scale based region proposal network. The method was used to prune non-text images and produce the region proposals. The features are extracted by VGG-16 layers, followed by feature fusion. After the scale estimation, the network also performs boundary regression to fuse standard range on all the irregular sized text and then provides the region proposals by following text region split, rough box generation and grouping, and lastly the proposal refinement. In the second stage, the output proposals are subjected to a FCN that localizes the words with higher precision for small scale since the non-text regions are pruned already previously. Two different text detectors, one altered version of the self-developed network (SPRN) for the first stage and second fine-tuned VGG-16 network, were used for performance evaluation followed by NMS as post-processing of the output. Experimentation was carried out on ICDAR2015, ICDAR2013, and MSRA-TD500 where the results suggest an F-measure of 0.7966 (SPRN), 0.8540 (VGG) and precision of 0.8524 (SPRN), 0.92 (VGG) for ICDAR2015 dataset. Furthermore, an F-measure of 0.848 (SPRN), 0.8817 (VGG) and precision of 0.8638 (SPRN), 0.9254 (VGG) was recorded for ICDAR2013 dataset. Lastly, an F-measure of 0.7665 (SPRN), 0.8072 (VGG) and precision of 0.8357 (SPRN), 0.8454 (VGG) was recorded for MSRA-TD500 dataset.

A text recognition model with a modified twin-SVM was proposed by the authors of (Francis & Sreenath, 2019b). The twin-SVM is accessorized with ambient and intrinsic regularizer that helps in the generation of an undisturbed function. The model first amplify the quality and contrast of the input image by Total Variation and Otsu method, followed by feature extraction with the help of HOG and Least Square-SVM for segmentation. An extra stage proposed by the author, revalidation utilizes manifold twin SVM to prune non-text objects. The next in the line is the text recognition part, which again uses manifold twin SVM to recognize characters, which are re-arranged using reconstruction with the help of pool of coordinates. Public dataset such as ICDAR2015, MSRA, and SVT were used for the experimentation of proposed methodology, where the results suggested an accuracy of 84.91, 84.21, and 86.21% respectively.

A multi-oriented text detection model was presented by the authors of (Yuqiang Zheng et al., 2020). The first stage, feature extraction block comprises of 5

convolutional block comprising of ResNet50, followed by the feature refining block incorporated with residual convolutional unit and chained residual pooling where the former one fine-tunes the weights and the latter one pool features. It also contains the fusion unit that chains the feature map in 3-dimension. Lastly, the prediction block comprising of multiple convolutional layers which generates the angle, class, as well as the geometry with the help of Axis-Aligned Bounding Box with maps that have higher score than threshold, followed by locality-aware NMS for the scale robust text detection. Authors used ICDAR2003, 2015, COCO-TEXT, and MSRA-TD500 datasets for the experimentation and the results suggest an F-score and precision of 83.1% and 88.9% for ICDAR2015, 79.2% and 87.6% for MSRA-TD500, 85.9% and 90.2% for ICDAR2013, and 52.9% and 59.3% for COCO-Text respectively.

A video text detection tracking system with novel data association method and deep text detection was proposed by (Wang, Feng, & Xia, 2019). The author used semantic segmentation module and VGG-16 with convolutional layers along for feature extraction along with transfer connection blocks to discover multiple sized text. The high level and low-level features are merged with three iterations of convolutional layers and transfer block connections, which are then used for text region detection and segmentation. In the second stage of text prediction, which relies regression models of on anchor prediction and bounding box prediction. The former modifies the position and proportion of anchors for refinement, whereas the latter regresses the actual region followed by a NMS. Text sensitive segmentation is used to enhance the learning of models. Authors used public datasets and the results suggested an F-measure and precision of 0.801 and 0.836 for ICDAR2015 dataset, 0.762 and 0.836 for MSRA-TD500 dataset, 0.70 and 0.73 for USTB-SV1Kdataset, and 0.91 and 0.93 for Minetto dataset when experimented for text detection. In addition to that, F-measure of proposed method increased from 0.788 to 0.701 and precision increased from 0.825 to 0.836 when integrated with segmentation module on ICDAR2015 dataset. Furthermore, multi-object tracking accuracy was reported 0.76 and the precision was 0.79 when applied on Minetto datasets.

Authors of (J. Tang et al., 2019) proposed a detection method for dense multi-oriented text by instance aware component grouping. The method that authors put forward overcome the barriers of segmentation by the use of attractive and repulsive links and instance aware loss. Lastly, the detection of text is performed by an altered Minimum

Spanning Tree. A VGG-16 architecture is used for the feature extraction with additional layers to compute deeper features. After the extraction of features, the instance awareness stage start with the forecasting of attractive/repulsive links via softmax layer. For the text detection of multi-oriented and curved text, the ground truths interpreted as different geometrical shapes are calculated by angle of orientation. Text components are then grouped together by altered minimum spanning tree to form instances, then the non-classified are filtered and the rest are then converted oriented rectangle by polygon combination. Lastly, polygon NMS prunes minute and unnecessary detections. Authors implemented the proposed methodology on DAST1500, Total-Text, CTW1500, ICDAR2015, and MTWI. The results for the text detection suggest an F-measure and precision of 0.794 and 0.796 for DAST1500 respectively, F-measure and precision of 0.820 and 0.837 for ICDAR2015 respectively, F-measure and precision of 0.721 and 0.747 for MTWI respectively, F-measure and precision of 0.815 and 0.821 for Total-Text respectively, and F-measure and precision of 0.813 and 0.828 for CTW1500 respectively.

A text detection and recognition model for medical lab reports was proposed by the authors of (Xue, Li, & Xue, 2020). The authors used fast region-based CNN for as basic framework for detection as VGG-16 extracts the feature maps, which are used to suggest bounding boxes with the help of region proposal network. Based on the position of proposed by the network, ROI pooling computes the features from the map and converts them into prearranged proportion. Which are then elongated to convey through fully connected layers to compute the loss. RNN is used as a basic framework for the detection architecture, which utilizes convolutional layers, rectified linear unit, and Bi-LSTM to provide the predictions. Lastly, NMS is used for the final recognition text. The experimentation for the proposed method was carried out on Chinese Medical Documents Dataset with 238 documents. The experimentation results suggest an F1-measure and precision of 97.0% and 95.3% respectively for text detection of multiresolution test set. In addition to that, the experimentation results for text recognition on multi-resolution set suggest an accuracy of 93.7%.

Authors of (Kanagarathinam & Sekar, 2019) proposed a method for text detection of seven segment delay to contribute towards less computational cost in advanced metering infrastructure. The authors developed a new dataset and labeled it as YUVA EB Dataset that includes a total of 169 images of digital energy meters of four to eight

digits in the display under different lighting conditions. The images in the dataset are divided based on lighting scheme and orientation etc. where tilted images constitutes 50 images along with 20 blurred images. In addition to that, 50 images of energy meters were taken during daytime and 49 at nighttime. For the text detection and recognition, authors proposed maximally stable extremal regions algorithm along with image dilation algorithm to prune noise, remove non-text regions, simplify the background from foreground, and smoothen the illumination etc. non-text regions are proposed to be eliminated by connected component based filters followed by OCR algorithm for the recognition of seven segment display characters. The results suggest that during the training stage, an accuracy of 95.50% was recorded when the model recognized 446 digits out of 467 digits. Furthermore, in the testing stage, the model depicted an accuracy of 93.17% when it recognized 464 digits out of 498 provided digits.

Authors of (Rong, Yi, & Tian, 2020) proposed a method for text instance segmentation utilizing the textual description of the query. In first part, the model extracts the spatial feature map, aided by FCN, by encoding character impression and its association to the environment of frame that helps in factual forecasting of segmentation mask. In second part, query sequence for the referring expression that elaborates the text in the image is subjected to word embedding matrix and LSTM algorithm providing the vector representation. Outputs from first and second stage are then subjected to fully convolutional classifier that evaluates if the position on the map relates to instances of referring expression thus the encoded image data and language query are decoded into saliency map, followed by unsampling to compute segmentation mask. Authors formulated a new dataset titled COCO-CharRef, for the evaluation of unambiguous text segmentation, which was derived from COCO-Text, Ref-COCO, and COCO-Stuff comprising of total 4000 training and 1000 testing images with language expression annotation. With the performance evaluation matrix selected as overall intersection over union, the result was reported at 28.1%

2.2. Prior Work Related to Turkish Language

It is evident from above presented published articles that much of the work is done for languages other than Turkish. On the other hand, a very little research is carried out

for detection, localization, extraction, and recognition of Turkish language from videos and images. Mainly the techniques employed in prior work are based on connected component or unsupervised approaches, such as (Dikici & Saraclar, 2008) suggested a histogram analysis approach to extract text that is sliding over the frames in the video, which is actually artificial text. It extracted the sliding text in Turkish news videos with the help of vertical and horizontal histograms to find out the text band boundaries of binary image. It further segmented the text area by performing connected component analysis to extract the individual characters, which then performed template matching to get actual character with the help of improvement measures based on rules learned through TBL.

A text recognizing approach to get text from Turkish News videos, is presented by authors in (Som, Can, & Saraclar, 2009). They developed a context independent HMMs (speech recognition tool) to model video text characters. It uses HMM-based sliding video text recognizer to recognize sliding text on Turkish news videos for hearing impaired. The model used the results of sliding video text recognition for unsupervised labeling of speech utterances extracted from the same videos. At preprocessing stage, it performed detection, localization and extraction of video text, and then from extracted text language independent features are extracted, which are then recognized by HMM-based glyph model. It used horizontal projection to detect vertical position of sliding text, and assumed it stayed at same position (location) for rest of video. They manually labeled the train set and used extraction algorithm to get words in test set by assuming text is non-overlapping and include maximum amount of complete words in the image. Furthermore, it modeled characters into different HMMs states to get Gaussian representation, keeping in view the inter-character and inter-word spacing. Handmade Bootstrap glyph model is used to segment the video text images in order to train actual glyph model. Then performed three iterations of maximum likelihood training to obtain baseline glyph model. It uses semi-supervised training in two stages. In first stage, glyph models are trained with labeled data only, and then it recognizes and label unlabeled data to generate more samples for training. In second stage, glyph models with expanded labeled data set are used to get final model with 1%-word error rate. As it is character based, thus it is vocabulary independent recognition.

A manually engineered rule based named entity recognizer that extracts the nouns (person or organization name, location) and money, date or time etc is suggested in (Küçük & Yazıcı, 2009). Authors applied set of lexical resources along with pattern bases and then morphological analyzer to validate the candidate named entities. Lexical resources contain dictionary of person names, list of well-known persons, organization and location; whereas pattern base resources include several patterns for extraction of organization or location names or it defines rules for numerical extraction like money, date, time or percentage expression. It is a dictionary learner, which scans, and extract named entities from annotated corpora and calculates its occurrence to find confidence value for that entity. Equipping the initial recognizer with rote learning component. In another paper, same authors proposed the advanced variant by equipping the initial NER with rote learning component to form a hybrid NER (Küçük & Yazici, 2011). The model is based on rule-based recognizer that has the ability to extend its information sources through learning from annotated data. It is a dictionary learner which scans and extract named entities from annotated corpora and calculates its occurrence to find confidence value for that entity and finally upgrades the lexical resources with newly named entities having a confidence value greater than 0.5.

In the same course, authors developed the first fully automated semantic annotation and retrieval system for news video in Turkish (Kuccuk & Yazici, 2010). It segmented the news into stories by detecting silence in audio waveform and then it is feed to sliding text recognizer to get video transcriptions of each segment. The transcripts are then passed through ruled-based NER to excerpt name entities in order to store in database as semantic annotations along with raw video information for retrieval. The author formed a hybrid recognizer (manually engineered and learning), called NER, which enhances its entities information automatically through annotated corpora. The authors added several rules to enhance the extraction of name-surname from text, which was lacking in other paper (Küçük & Yazıcı, 2009). They also improved the lexical resources of recognizer by implementing rote learner that scans training set and calculates statistical features (confidence) for distinct entities and new entities set is added to information source. The video retrieval is done through Boolean queries that combined queried entities, converted it into SQL expression for execution to get the results from database that stores information of raw video, segmented News stories, and extracted named entities in database.

Later, authors merged their previously proposed work (Küçük & Yazıcı, 2009) and (Küçük & Yazici, 2011) in (Küçük & Yazıcı, 2011a) to form an automatic video indexing and retrieval system based on semantic annotation for videos related to Turkish News. It used already implemented audio-based News story-segmenter to find news stories by detecting silence periods. The segmented video parts are then fed to already available text extractor tool (sliding text recognizer). The text bands on images are detected by vertical and horizontal histograms analysis and then individual characters are extracted by connected component analysis, and performed template matching for actual character recognition. NER with normalization, person entity extraction and its co-reference resolution, and event extraction techniques are implied on extracted text to get semantic information. It further enhances the annotation by performing web-crawling for specified time window to find the extracted named entities and calculating its confidence value. Finally, they stored all extracted information along with raw videos in central database, which can be accessed through semantic retrieval interface. The author used two datasets (one of them is manually transcribed) for training and testing to annotate the text in order to create answer key for named entity recognition.

In (Küçük & Yazıcı, 2011b), authors proposed system that performs multilingual video indexing of videos in English while the semantic information extracted is in Turkish. To transcribe the raw videos into English already available PocketSphinx ASR system is used, the resulting transcribed English text is then fed to English-to-Turkish MT to get its equivalent Turkish translation. Finally, semantic information is extracted from Turkish translation to obtain the contents using hybrid named entity recognizer to do video annotation. Text-based semantic indexing is done by storing semantic information and production metadata in database, which is retrieved via user interface with help of Boolean queries. The recognizer was not compromising especially for person name, as foreign person names were not in author's reorganizer, but performed well on location extraction or date/money expression. Once video is retrieved, English transcriptions are also presented along with Turkish.

2.3. Finding

In spite of such extensive studies, according to best of my knowledge and research, scarce amount of work is found in the area of Turkish video indexing and retrieval based on text appearing inside it. Most importantly, still deep learning techniques are not applied for Turkish text detection and extraction from videos. To bridge this gap for Turkish text, this dissertation analyzes the offline video (especially for Turkish language text videos) to detect the artificial text in the video frames, and extract it using deep learning techniques.

CHAPTER 3
MATERIALS AND METHODS

As mentioned earlier, the principal target in this research is to build a system that takes raw videos as input, detect and extract the Turkish text, and generates indexing table in database for video ranking. Later, a user interface is provided where a user gets the relevant videos in ranking order based on the queried keywords, as depicted in semantic diagram shown in Figure 3.1. To achieve the finest video indexing and retrieval system for Turkish videos, the process is carried out in 10 steps, which are as follows:

1- The relevant videos are disintegrated into frames. The video frames from offline dataset, having 3 colored channels (red, green, blue) are feed into the system, which is then converted to gray scale. Laplacian of Gaussian is used to discard the noise. Furthermore, some morphological operations are employed followed by heuristic and geometrical constraints to suppress the unwanted background noise.

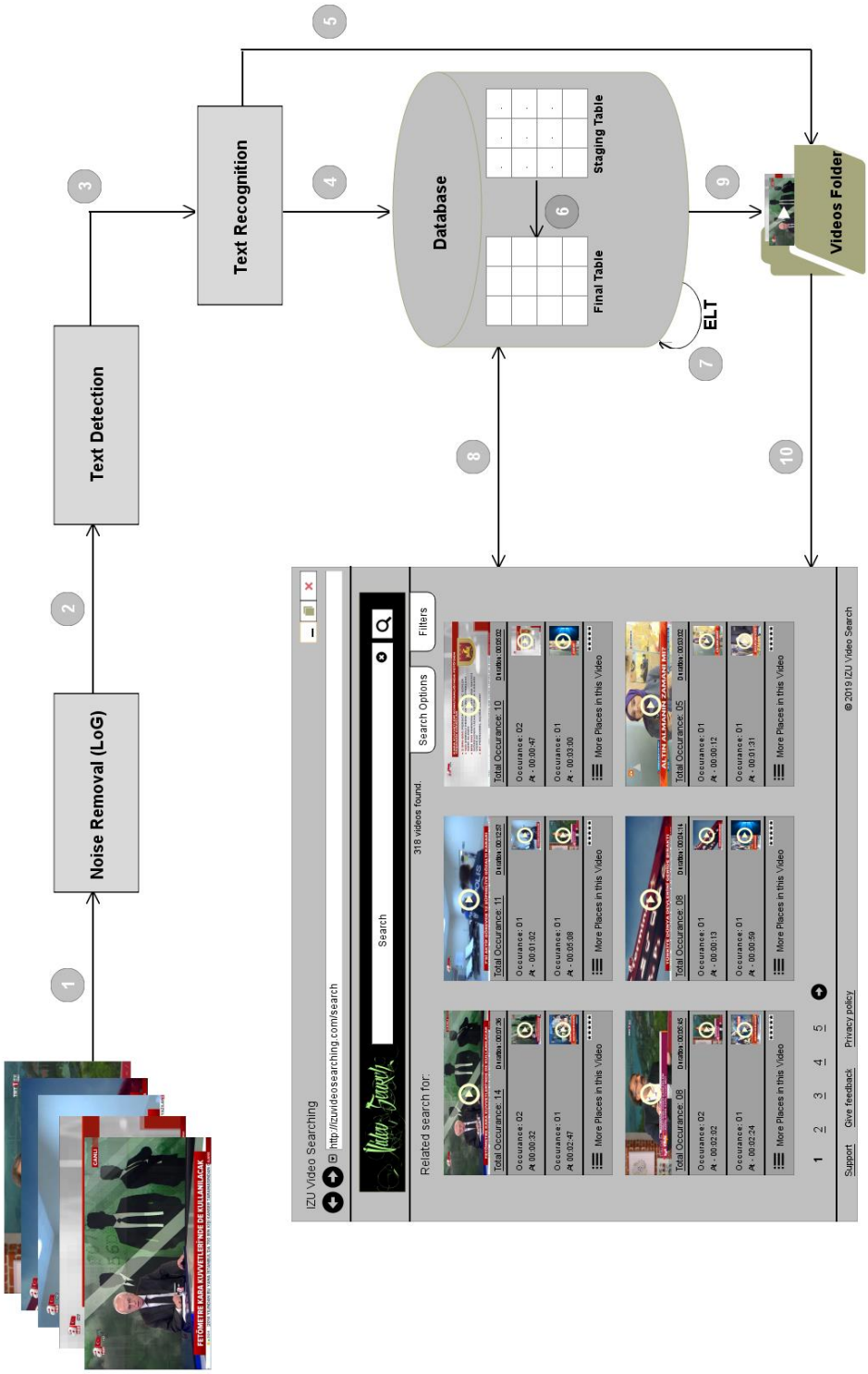
2- These filtered images are feed to the deep-learning frameworks, for Turkish text detection. The candidate region *c*, labeled as 1 if contains text information *t*, otherwise labeled as 0 for non-text region. This trained model is later used on various video images for Turkish text detection and extraction. These labeled regions are further compared with human made reference dataset to check the authenticity of system.

$$Label(c) = \begin{cases} 0, & c \notin t \\ 1, & c \in t \end{cases} \quad (1)$$

3- The detected text regions by deep-learning framework are then cropped from the image frame. These Turkish textual regions are feed to commercially available open-source Tesseract OCR engine for text recognition.

4- The recognized Turkish text, obtained from the OCR in previous step, along with the frame production information (regarding occurrence time, raw video name and the file path of archived video under consideration) are then stored in database for video indexing accordingly by annotating corresponding video with extracted textual information.

Figure 3.1: Schematic diagram for proposed Turkish text detection, video indexing and retrieval system



5- The processed videos are then archived in defined location instead of storing it completely in database due to huge and numerous videos resulting in much needed space and size. Therefore, each video file path is inserted into database for retrieval purposes.

6- Later a staging table is used as temporary table to stage the data for temporary purpose just before loading it to final/target table from source table. Each extracted word staged in this temporary table and duplicates are discarded before moving it to final table. It is used for ETL (Extract, Transform, and Load) process, as data might be loaded from more source table. While data is in staging table, steps like cleaning, normalizing and de-normalizing from multiple to final table is performed. Depending on case, it might have several staging tables from different sources are then combined into the resulting Final table.

7- Extract, Transform and Load (ETL) refer to three database functions combined into one tool to consolidate the data spread across multiple source systems. It helps to extract the data from a source system, transforms (converts) the data into required format so that it can be analyzed, and finally stores (loads) it to other system. It actually pushes the processing down to the database to improve the performance.

8- An interactive web interface is built for prospective users to specify their query by entering keywords and literals. The system will process the specified keywords to formulate the query. The formulated query is then executed to search through database for retrieval of relevant videos satisfying the query, thus these resultant relevant videos are shown to user. The users can access and play their video of interest among these retrieved videos through provided interface. Once the user clicks on one of the retrieved videos, the system automatically seek to the point of interest where that word occurred first in the video, without going through whole video. If system fails to directly play the video from point of interest, the interface also mentions queried word's timestamp in the relevant videos so that user can directly seek to the point of interest in that video, without going through whole video.

9- Once the query executes, the relevant videos are retrieved from archived folder based on the file location links stored in the database.

10- The retrieved videos from archived folder then can be played through provided interface.

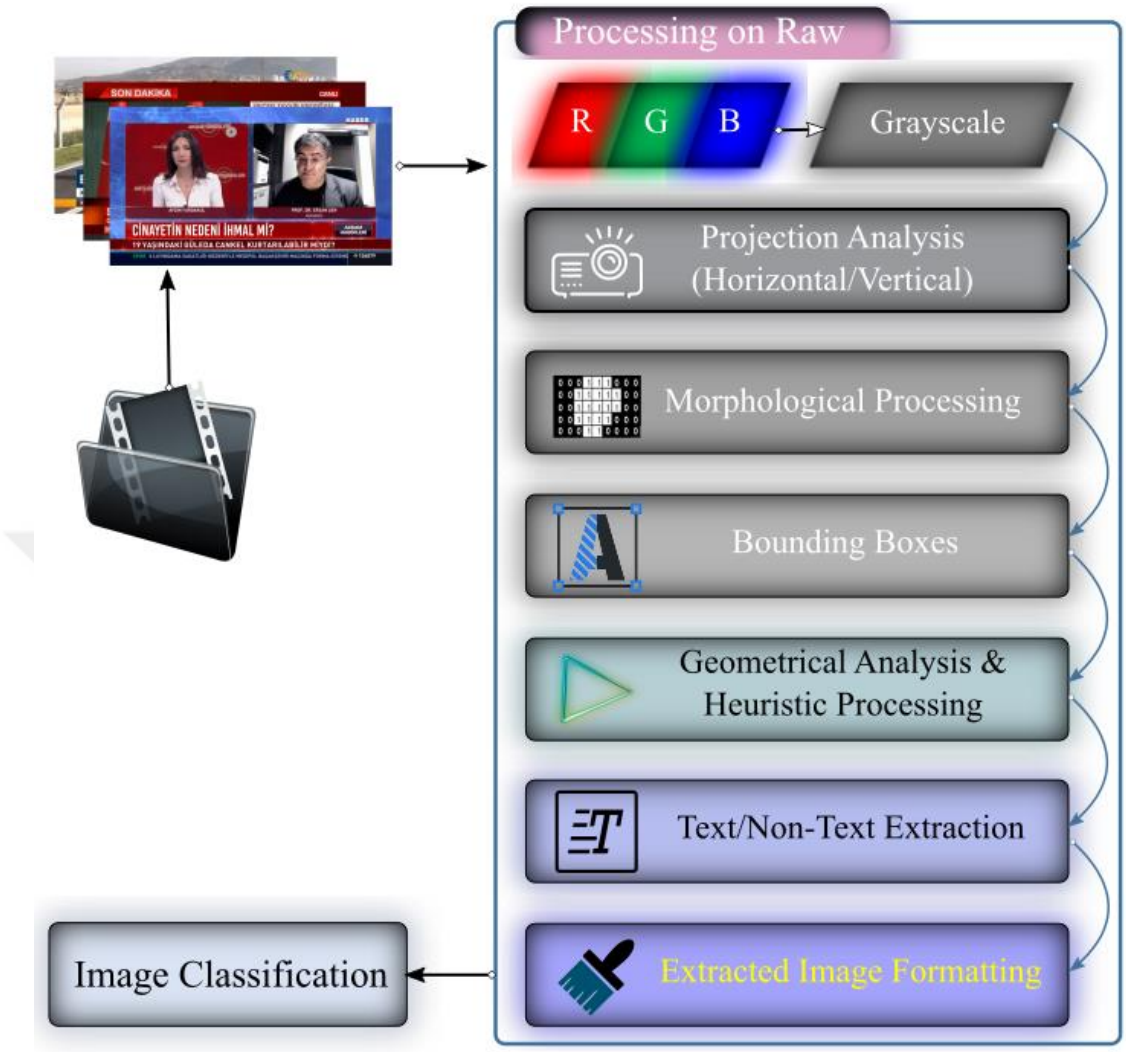
Literature review reveals that a very scarce amount of work is done for textual-content based Turkish video indexing and retrieval system. Due to its infancy state, there is no publicly available benchmark dataset of images containing text from Turkish videos, whereas deep learning models require huge amount of data to effectively train the classifiers so that it can accurately predict the region in future. Therefore, a new dataset based on Turkish videos is prepared for experimental work, which is then used to train and test the classifier in proposed framework.

In order to detect and extract artificial text, this study proposed a deep learning based models, which first requires sufficient amount of data for training. In order to train the models, this study generates the dataset using the process defined in schematic diagram depicted in Figure 3.2. In this proposed approach, a statistical and heuristics based approach is employed for automatic text localization, detection, and extraction from Turkish video frames. The extracted text regions are later fed to proposed deep learning models for training, which are then used to detect the text in the images. Later, the detected text from proposed deep learning models are fed to commercially available OCR. The OCR recognizes the text present in each extracted region from video frames.

For classification of frame region as text or non-text area, this study proposed a CNN based deep learning classifier. In addition to proposed CNN-based framework, the study also investigated various state-of-art deep learning models (such as variants of VGG, DenseNet, and ResNet networks) and conventional machine learning classifiers (LR, and SVM) on compiled Turkish Text dataset.

This section presents the overview of dataset and its compilation process, image pre-processing techniques, various deep learning approaches and machine learning based frameworks used in experimental setup.

Figure 3.2: Workflow of preprocessing steps on image



3.1. Dataset Overview

The research related to Turkish text retrieval from videos and images is in its infancy state thus limits the availability of Turkish benchmark dataset. The literature research conducted for this study depicts that there is no publicly available benchmark dataset of video images and frames that contains Turkish text. On the other hand, generally deep learning based classification models require huge amount of samples to train the classifier and then assess it with test samples. Therefore, prior to implementation of proposed deep learning models for Turkish text detection and extraction, firstly image dataset is formed for the commencement of the experimental work.

As of August 1, 2019 report published by Radio and Television Supreme Council (RTÜK)[8] there are 536 Television channels operating across Turkish, and majority of those are operating in Turkish language. According to statistics issued by Statista[9] in December 2019 that categorizes Turkish channels by genre, 22 channels are operating as News and business channels, 40 channels fall under sports category while 18 are broadcasting cartoons and stuff related to children. Everyday these wide range of Turkish channels are generating enormous amount of visual contents that are constantly being uploaded on several video hosting platforms for later usage.

For this research, a new offline dataset (composed of images) is generated by downloading videos encompassing Turkish text from various sources. In this regard more than 100 videos of different genres that contains artificial text in horizontal direction were downloaded. The downloaded videos spread over diverse genres are mostly from News and business channels, financial and stock channels, and sports related Turkish Television channels. As broadcasted News videos generally span for hours with more textual contents that the key segments could contribute a lot to entire data, therefore News videos are selected as primary domain beside others (sports, stock and financial channels) due to its textual-information richness.

The videos are of various resolution, having three-colored RGB (red, blue, greed) channel, obtained from several open access video hosting platforms such as Dailymotion, YouTube™ and Vimeo. Each acquired video is first disintegrated into frames, as Figure 3.3(a) shows dismantled frame of one video. Later these dismantled frames are converted into grayscale (see Figure 3.3(b)) to automatically obtain textual/non-textual areas (images) by applying procedures defined later in this Chapter and Chapter 4.

---

[8] RTÜK – Radio and Television Supreme Council, Turkey: a state-owned authority established to monitor, regulate and audit the communication and broadcasting organizations and systems, https://www.rtuk.gov.tr/

[9] Statista: a German company provides statistics related to business, industry, finance and media, https://www.statista.com/statistics/437761/tv-channels-by-genre-in-turkey/

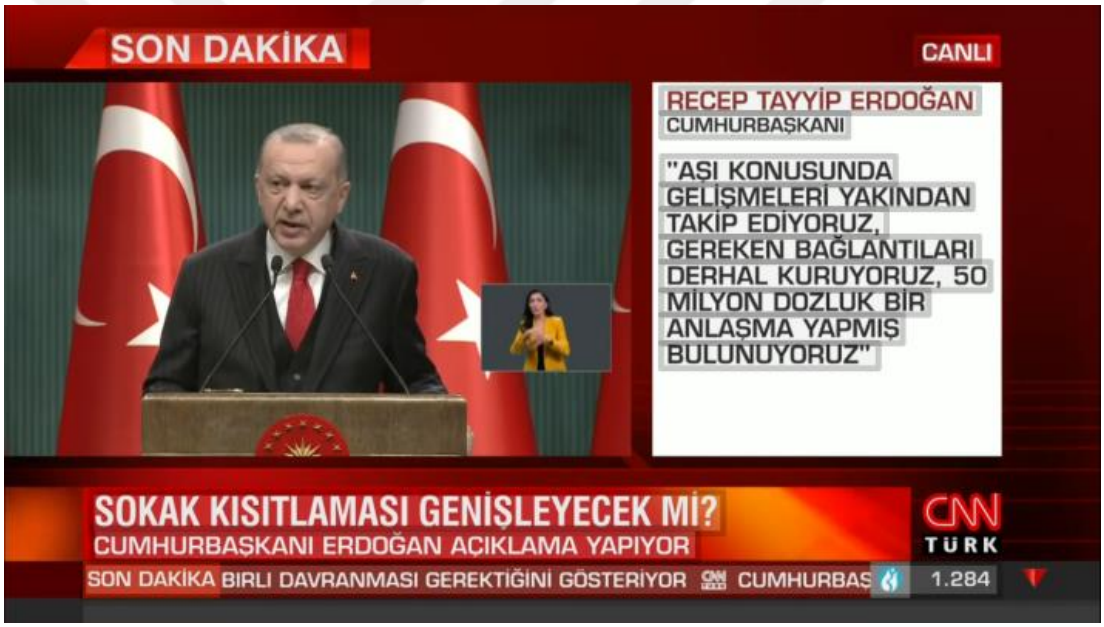
Figure 3.3: Dismantled frame of a downloaded Turkish News channel video, (a) image with three-color channels (red, green, and blue), (b) image with a grey channel

(a)

(b)

The artificial text appeared in horizontally aligned lines in frames of various videos varies in sizes. For Example: the text in video frame depicted in Figure 3.4 appears in 15 different horizontally aligned lines. It can be seen that each text line has different width. Similarly, the non-text images also varies in length and height. Hence, to make a uniform image dataset, the varied length horizontally aligned textual lines are cropped from frames and then sub-divided into fixed length of 128 pixels. Figure 3.5 illustrates some samples of textual and non-textual images generated for the experimental work of this study. The process of obtaining text lines is described in Chapter 4 of this study.

Figure 3.4: Sample image containing multiple lines of horizontally aligned text, each line varies in length



The entire dataset used in this study consists 231622 images generated from more than 100 Turkish videos. Approximately 47.21% out of 231622 images are non-textual samples that equates to 109360 images, and rest of the images refer to textual instances which are 122262 in total, as listed in Table 3.1. Furthermore, to effectively train the proposed machine learning and deep learning classifiers, the dataset is divided with a ratio of 30%:70% to form testing and training sets respectively.

Figure 3.5: Samples of generated dataset, (a) textual images, and (b) non-textual images

(a)

(b)

Table 3.1: Overview of generated dataset

	No. of Instances
Images with textual content	122262
Images with non-textual content	109360
Total	**231622**

3.2. Image Preprocessing Techniques

The principal aim of current study is to construct a system that can detect either the frame in the video has text or not. As depicted in Figure 3.2, the proposed classification model performs some pre-processing to obtain required data, before feeding it to machine learning classifier, that involves various image filtering and morphological techniques for edges detection and removal of imperfections from filtered image. The overview of these techniques that are applied in preprocessing system to acquire meaningful data are presented below.

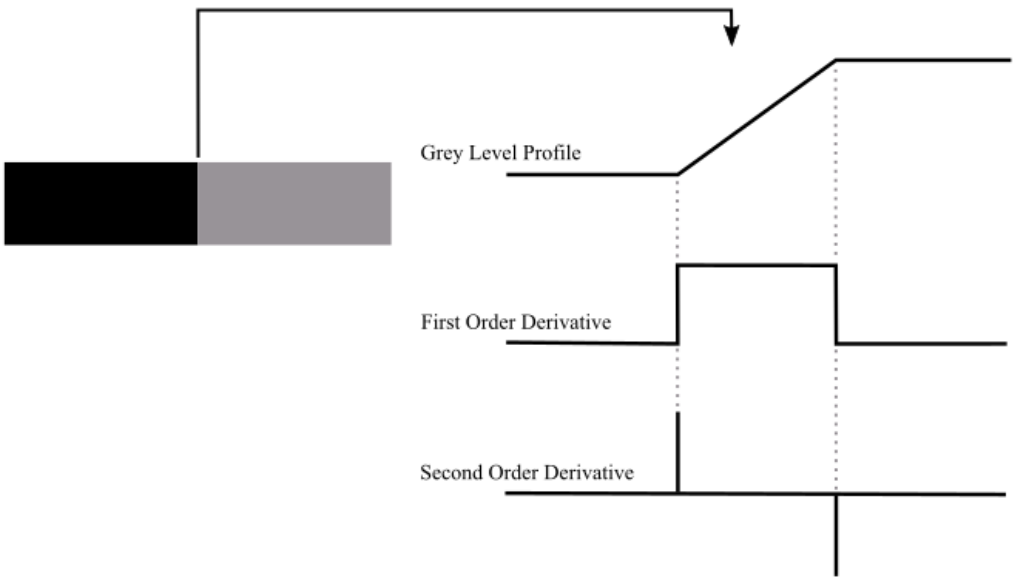
3.2.1. Sobel Edge Detector

Edges are the abrupt transition of discontinuities in a region or area of image. These significant transitions can be in three direction:

- Vertical
- Horizontal
- Diagonal

Edges actually encloses majority of object information of an image. Thus if user is interested in an object or its respective position in the image, edge detection helps in focusing that specific region of the image. It grants the user to examine image features for a transitional shift in gray level, which indicates the end of one area and start of another in an image. Researchers introduced various edge detection techniques and masks (such as Laplacian operator, Prewitt operator, Krisch compass operator, Sobel operator, Robinson compass mask, etc.) to obtain the boundaries of shapes present within images. The main concept behind majority of these approaches is calculation of local derivative operator to determine the gray-level discontinuities. The resultant of first derivative determines the edge presence, while second derivative discovers the transition type (dark-to-light or light-to-dark) as depicted in Figure 3.6.

Figure 3.6: Filtering derivative graphs, showing the first order derivative and second order derivative of a grey level profile image



Basically, image edge detection operators has two categorizes; Gaussian operator and gradient operator. Gaussian operators such as Laplacian of Gaussian or Canny edge detector are based on second-order derivations, while gradient operators, like Prewitt, Robert and Sobel operators, rely on first-order derivation. The gradient operators calculates the first derivative by exploiting gradient magnitude at any point:

$$\nabla f = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} df/dx \\ df/dy \end{bmatrix}, \tag{2}$$

where magnitude is

$$\text{mag}(\nabla f) = \sqrt{G_x^2 + G_y^2} \approx |G_x| + |G_y|. \tag{3}$$

Similarly, the direction of gradient is calculated by the angle:

$$\emptyset(\text{x}, \text{y}) = \tan^{-1}\left(\frac{G_y}{G_x}\right) \tag{4}$$

These techniques are usually employed in the field of image processing, machine and computer vision for data extraction and image segmentation. For this study, sobel operator is adopted for detection of edges due to its popularity in digital image processing such as image morphology, pattern recognition and feature extraction. Besides being simple to implement, it can easily search smooth edges with time efficient computational capability. Being a discrete differentiation operator, it detects the edges in an image by approximating the gradient of intensity function.

At image pixels, it builds corresponding gradient vector or normal to a vector. For an image region of 3×3 (depicted in Figure 3.7), it performs the gradient computation using (5) and (6) to minimize the amount of data while maintaining image structural properties.

$$G_x = (z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3) \tag{5}$$

$$G_y = (z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7) \tag{6}$$

Figure 3.7: Image of 3×3 size-region

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

The masks or kernels shown in Figure 3.8(a) and Figure 3.8(b) are horizontal and vertical masks of Sobel operator, respectively. The horizontal and vertical masks, each of size 3×3, convolves with image to determine horizontal and vertical derivatives, respectively. As the central row has all zeros in horizontal mask, thus it determines edges in horizontal direction. Moreover, -2 and 2 in the second column of first and third row, respectively, enhances the edge intensities by giving extra weightage to values of pixels surrounding edge region, thus makes the edges more prominent. Likewise, vertical mask has all zeros in central column hence without involving the original values edges, it computes left and right pixels intensity differences.

Figure 3.8: Sobel filter masks, (a) horizontal mask, and (b) vertical mask

-1	-2	-1
0	0	0
1	2	1

(a)

-1	0	1
-2	0	2
-1	0	1

(b)

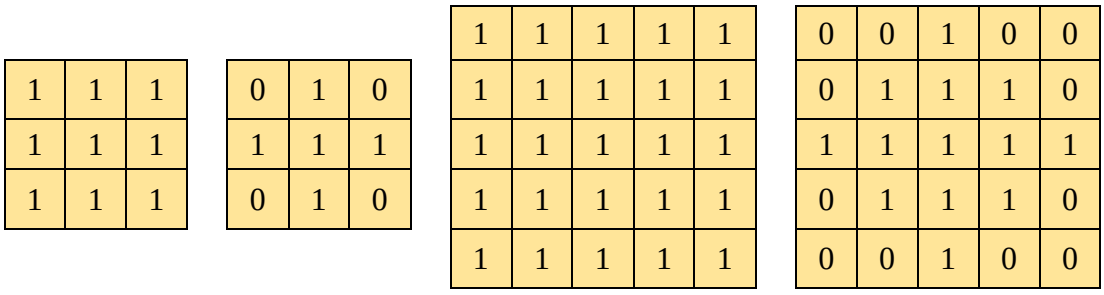
Its derivate mask capabilities of detecting edges in both vertical as well as horizontal direction and smoothing the image beside just providing the differences, gives an extra advantage in digital image processing field.

3.2.2. Dilation as Morphological Operation

In image processing, morphological operation is a broad set of non-linear operations (such as erosion, dilation, hit and miss transform, thickening, thinning, opening, closing, etc.) that does not depend on pixels' numerical values, but considers their relative ordering (Sundararajan, 2017) to process an image. Thus makes it suitable for binary as well as grayscale images with unknown light transfer functions. Sometimes image filtering or thresholding introduces distortion and noises. These imperfections can be removed with the help of morphological image processing operations that account for the structure and form of the image. Morphological operations are suitable for eliminating tiny objects, filling miniature objects, destroying objects until centerlines, detecting patterns, grayscale contrast enhancement, isolating tiny and less bright objects, and finding intensity troughs in an image.

Dilation is the most fundamental morphological operation that is mostly utilized in computer vision and image processing area to adjoin boundaries of various objects by adding a pixels layer to outer and inner boundaries of regions present inside an image (Fisher, Perkins, Walker, & Wolfart, 1997). A structuring element, formed of small template or shape having a matrix of pixels with binary values, is used to probe an image. The size of structuring element is equal to its matrix dimensions, while shape is determined by patterns of zeros and ones in the matrix, whereas origin is any of its pixels. Figure 3.9 shows some sample structuring elements with different shapes and sizes.

Figure 3.9: Some famous structuring elements used for morphological operation



3×3 Square Template (a)

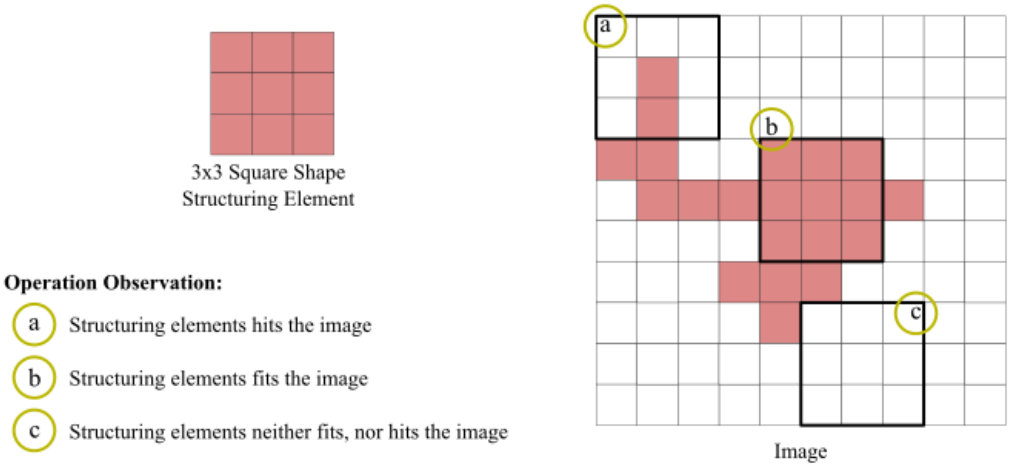
3×3 Cross-shaped Template (b)

5×5 Square Template (c)

5×5 Diamond-shaped Template (d)

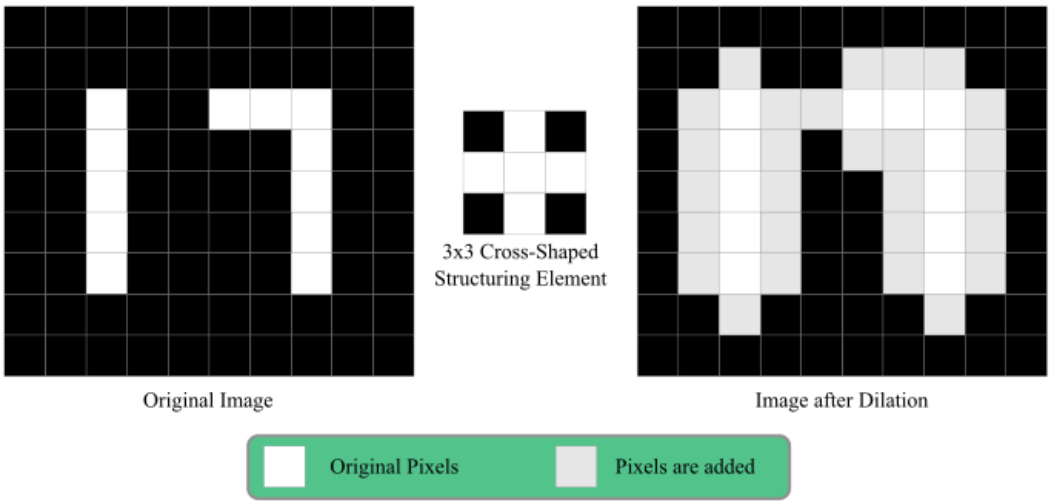
The structuring template is placed over all pixels in an image to determine whether it 'fits', 'hits' or 'none of these' within the neighborhood. It is a 'fit' if structuring element fits in the image, or a 'hit' if structuring element partially intersects the image, otherwise it is none, as shown in Figure 3.10.

Figure 3.10: Image probing with 3×3 square-shape structuring element (hit, fit, or none)



More pixels are added to the objects' boundaries by a structuring element in such a way that a new formed image has same size as of input image. The structuring element *se* dilates the input image *im* to form an equally sized output image *oi*, by positioning *se* at all pixels *(x,y)* of im, so that *oi(x,y)* = 1 if *se* hits *im* otherwise oi is 0. Figure 3.11 shows the example of dilation operation that fills the gaps between two objects within an image with a 3×3 cross-shaped structural element. It can be observed from Figure 3.11 that boundaries of two objects; '1' and '7' are joined together by dilation operation. A new layer is added to these boundaries, thus it formed a single object in new image.

Figure 3.11: Image dilation using 3×3 cross-shaped structuring element to join two textual objects



Therefore, dilation operation is used in this study to combine the different objects, specifically text/Turkish alphabets present inside a video frame are joined to form lines.

3.3. Classification Methods and Frameworks

The dominant paradigm of symbolic artificial intelligence ends in early 1990s, when new artificial intelligence-based approaches such as machine learning and deep learning started flourishing that can intellectually automate more complex and fuzzy tasks like voice recognition, object and image classification. Unlike symbolic artificial intelligence approach in which a programmer handcrafts enough rules to solve problem, these new machine learning and deep learning automatically craft set of information-processing rules by observing the data (Chollet, 2018). These engineering oriented approaches aims to handle huge and complex datasets to execute problem at hand.

In recent years, these approaches were employed to make sense of visual data, which became the field of computer vision. Literature is full of countless articles related to image classification, text detection and prediction using deep learning and machine learning based classifiers. Most of these classifiers require expected output along with

the input image to discover statistical structures among these to devise explicit data-processing rules for later prediction. The following sub-sections provides a detailed overview of all the deep learning and machine learning-based methods or frameworks used in experimental work for classification of regions as text and non-text images.

3.3.1. Deep Learning-based Methods and Frameworks

3.3.1.1. Convolutional Neural Network (CNN)

The advancements in field of computer vision took a new turn when in late 1980s and early 1990s, when Yann LeCun, a research at Bell Labs developed an intelligent check-recognition tool that recognizes handwritten digits[10]. Later in 1998, he along with other authors presented an image recognition technique known as ConvNets (Lecun, Bottou, Bengio, & Haffner, 1998). They introduced CNN model, inspired by the working of human visual cortex. The ConvNet architecture replicates the working of neurons and their connectivity patterns in hominid brain, in which each neuron only reacts to stimuli in its corresponding receptive field. Receptive fields are collection of confined areas of visual field that covers complete visual area when combined together. CNN is developed on phenomena of neuron working in human brain, and it is perfected with time, that it became the heart of most of deep learning frameworks constructed for various image recognition, video analysis and classification contests. Few of those popular frameworks are listed below:

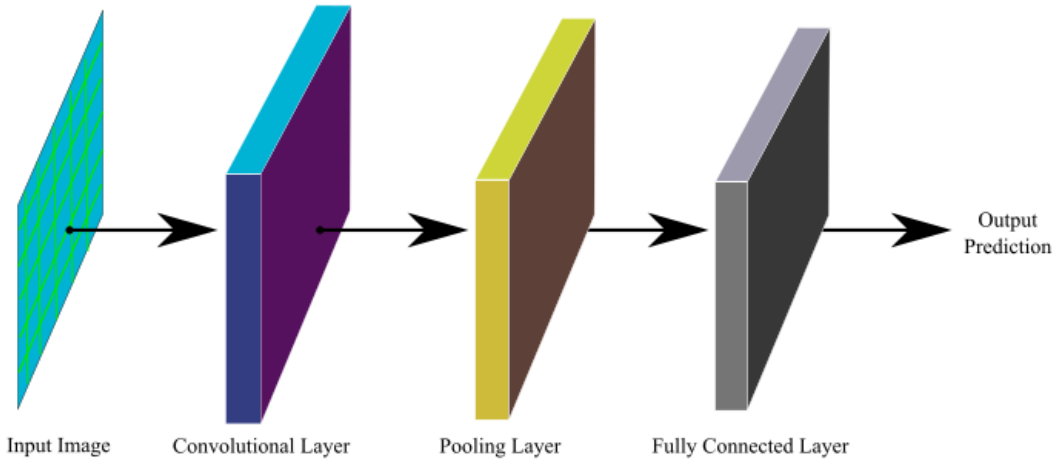
- AlexNet (Krizhevsky, Sutskever, & Hinton, 2017)
- DenseNet (G. Huang, Liu, Van Der Maaten, & Weinberger, 2017)
- GoogLeNet (Szegedy et al., 2015)
- LeNet
- ResNet (K. He, Zhang, Ren, & Sun, 2016)

[10] CNN Demo 1993 – Youtube video: https://www.youtube.com/watch?v=FwFduRA_L6Q&ab_channel=YannLeCun

- VGGNet (Simonyan & Zisserman, 2015)
- ZFNet (Zeiler & Fergus, 2014)

It is a deep learning based classification method that constitutes of linked layers each multiple perceptron, which takes an image as input, sets priority (assign biases and weights) to different features of an image in order to distinguish between each other. The filters in primitive techniques are hand-engineered and requires significant training, but CNN filters learns these characteristics automatically. Similarly, the prior neural networks mostly takes the input in form of vector, whereas CNN is capable of taking a multi-channeled image as input. Generally, there are three different layers' type in CNN architect. Convolutional layer is first among those types. Next comes pooling layer and lastly a dense layer as illustrated in Figure 3.12.

Figure 3.12: General overview of CNN architecture

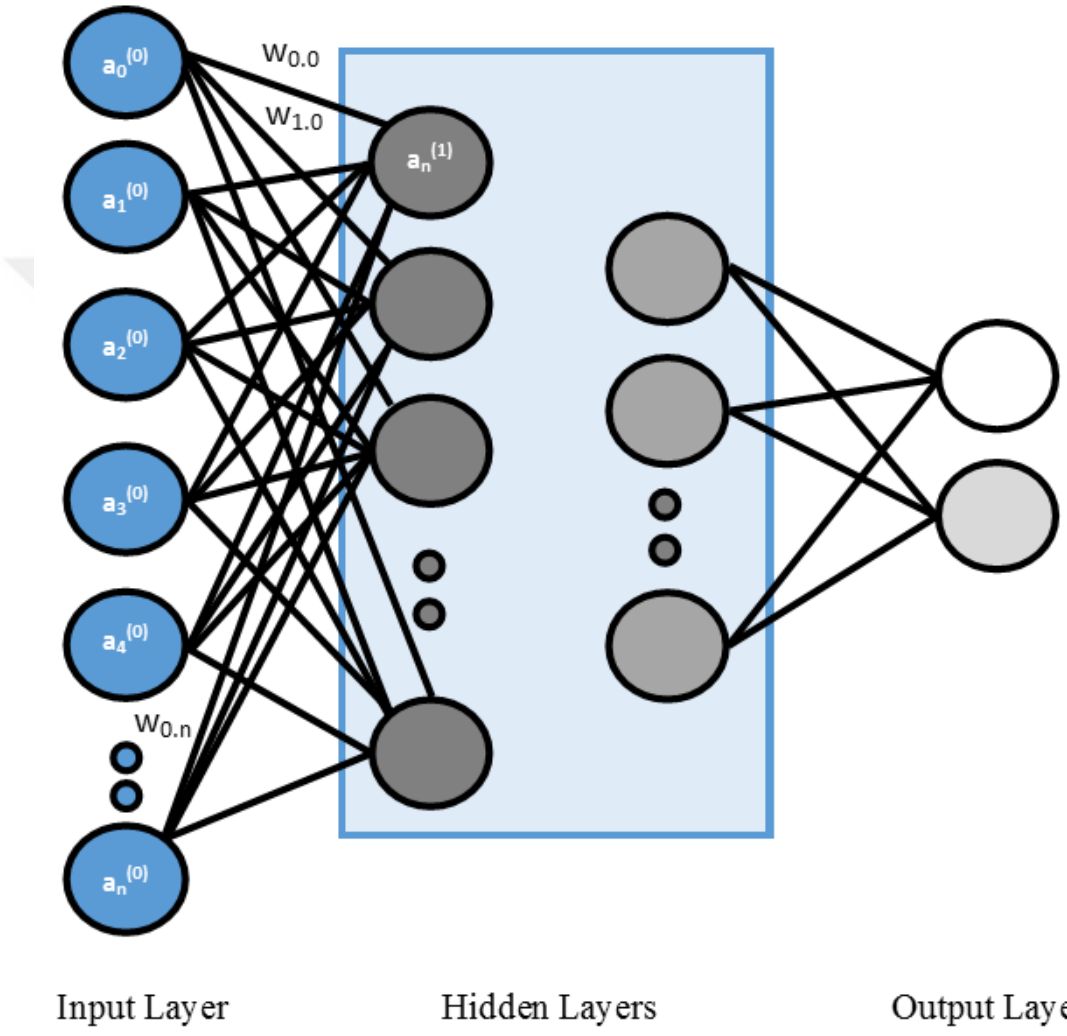


Usually ConvNet has several convolutional layers that are considered as primary building blocks. It aims to distill relevant features at each stage of multiple layers. The concept is based on local connectivity and parameter sharing. For example, as depicted in Figure 3.13, each neuron of first convolutional layer is linked with limited/all neurons of input layer to extract low-level features from input image such as color or edges. This neuron connectivity with sub-set of previous layer is called local connectivity. The additional convolutional layers empower the network to fully understand the images by learning high-level features. Every connection between neurons of succeeding and preceding layers shares a weight 'w', called parameter

sharing based on their common characteristics. The network then calculates the sum of these weights along with entire activations 'a':

$$w_{0,0}a_0^{(0)} + w_{0.1}a_1^{(0)} + \cdots w_{0,n}a_n^{(0)} \tag{7}$$

Figure 3.13: Visual representation of neurons connectivity and weights in CNN



The network then normalizes the resultant of calculated weighted-sum to be a value between 0 and 1 with the help of a squishing function such as sigmoid (σ). Later it activates or deactivates a neuron by including an appropriate bias 'b'. For instance, the meaningful activation of third neuron in first convolutional layer is expressed by

$$a_2^{(1)} = \sigma\left(w_{2,0}a_0^{(0)} + w_{2,1}a_1^{(0)} + \cdots + w_{2,n}a_n^{(0)} + b_1\right) \tag{8}$$

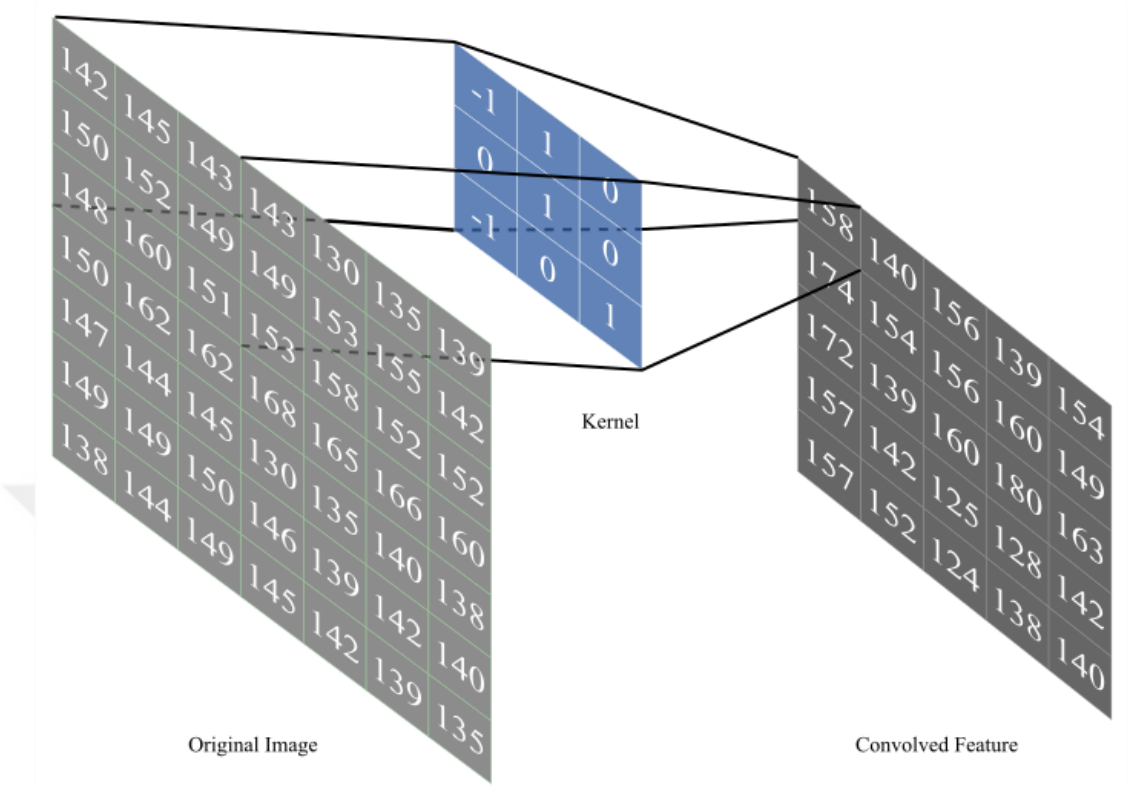
Later, the network computes the activation of succeeding layer by transposing the weights into matrix, while biases, and calculated neurons' activations of individual layer into column vector. Every row in equation below depicts the connectivity between preceding layer and specific neuron of succeeding layer.

$$\sigma\left(\begin{bmatrix} w_{0,0} & \cdots & w_{0,n} \\ \vdots & \ddots & \vdots \\ w_{k,0} & \cdots & w_{k,n} \end{bmatrix}\begin{bmatrix} a_0^{(0)} \\ \vdots \\ a_n^{(0)} \end{bmatrix} + \begin{bmatrix} b_0 \\ \vdots \\ b_n \end{bmatrix}\right) = \begin{bmatrix} a_0^{(1)} \\ \vdots \\ a_k^1 \end{bmatrix} \tag{9}$$

Likewise, the network learns the appropriate biases and weights for each convolutional layer in order to devise a better classification system. The convolutional layer functionality revolves around kernel that determines the activation maps. Kernel is a filter element that represent a matrix of any size to perform convolution with input feature maps. Figure 3.14 depicts an example, in which a kernel of size 3×3×1 is convolved with a grey-channel input feature map of size 7×7. If the stride (shift) is set to 1, then kernel finishes the convolution of grey-channel image (shown in Figure 3..14) by sliding 25 times over entire channel's input feature map. At each kernel shift, network executes scaler product between kernel and chunk of image over which it hovers.

If stride is set other than 1, the kernel jumps right according to that specific stride value to covers the entire width of feature map. Then kernel jumps down to the extreme left as per stride value and repeats the same procedure until it traverse complete feature map. Similarly, the kernel depth increases with the input image depth, if input image depth is one, kernel depth will also be one, but if input image has three-channels (such as RGB) then kernel depth will also be three. Likewise, the convolution operation will add the resultant of matrix multiplication of entire three channels, as shown in Figure 3.15. In the end, a bias is added and rectified linear unit is used to squeeze the resultant convoluted features.

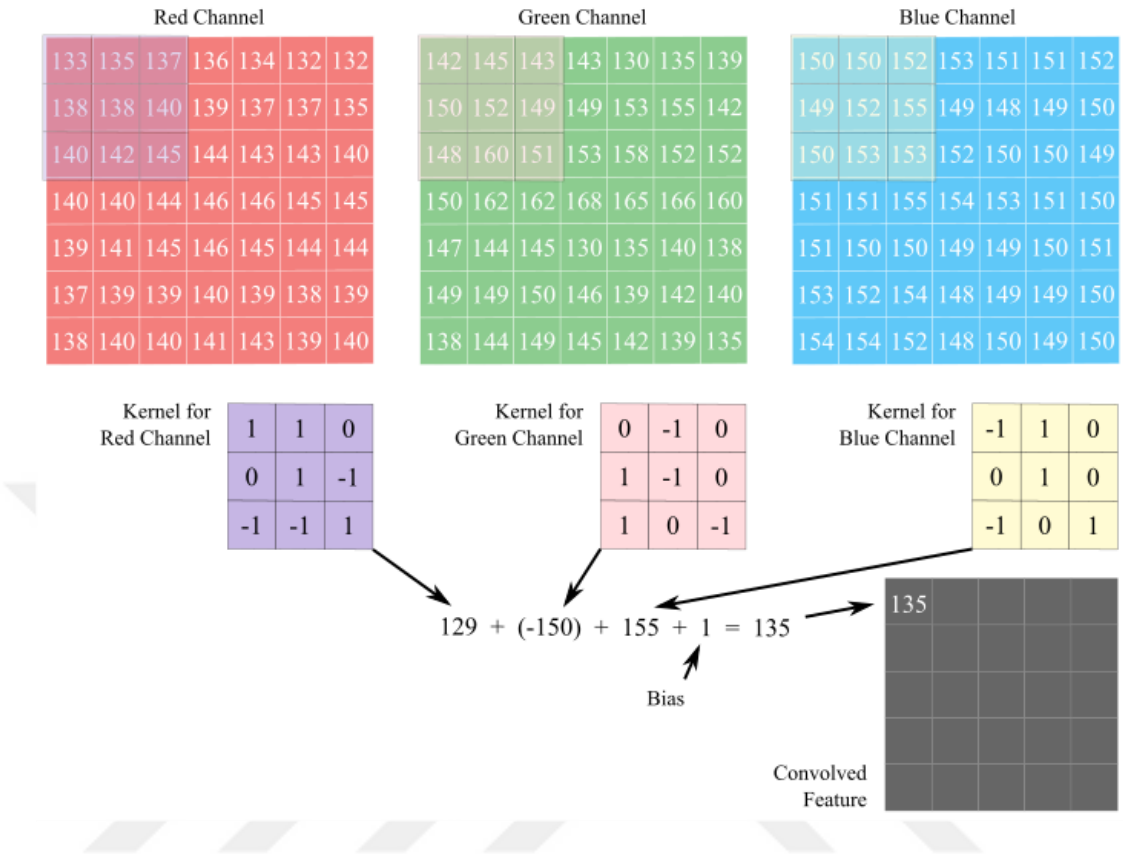
Figure 3.14: Visual representation of kernel convolving with single input feature maps in CNN



Rectified linear unit is an activation function that is mostly employed to encompass the non-linearity in NN. The main functionality of activation function is to squeeze the output values by limiting them between 0 and 1. Following are some widely used activation functions in CNN models (also depicted in Figure 3.16):

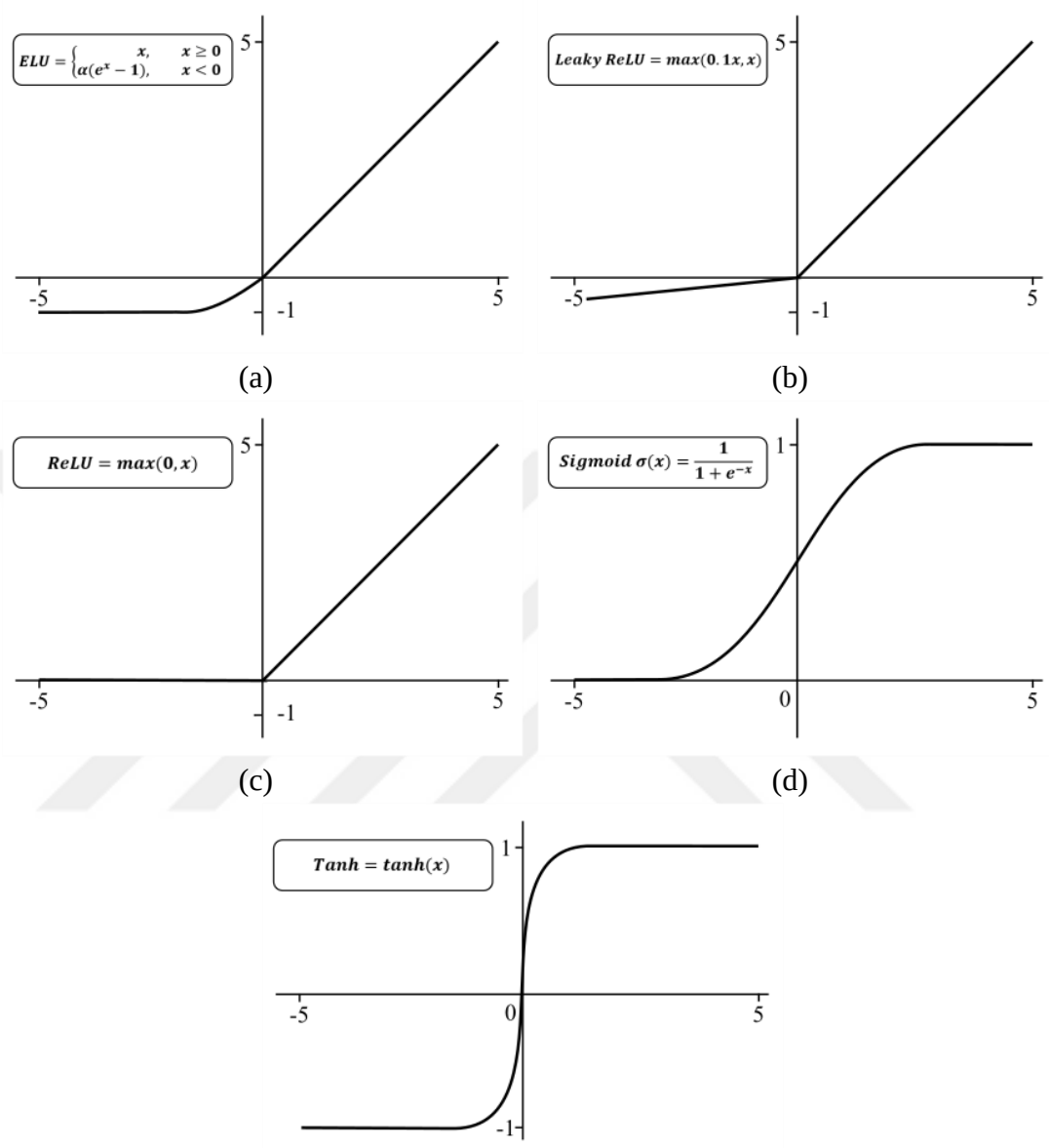
- ELU
- Leaky rectified linear unit
- Maxout
- Rectified linear unit
- Sigmoid
- Tanh

Figure 3.15: Visual representation of kernels convolving with 3-channel input image in CNN



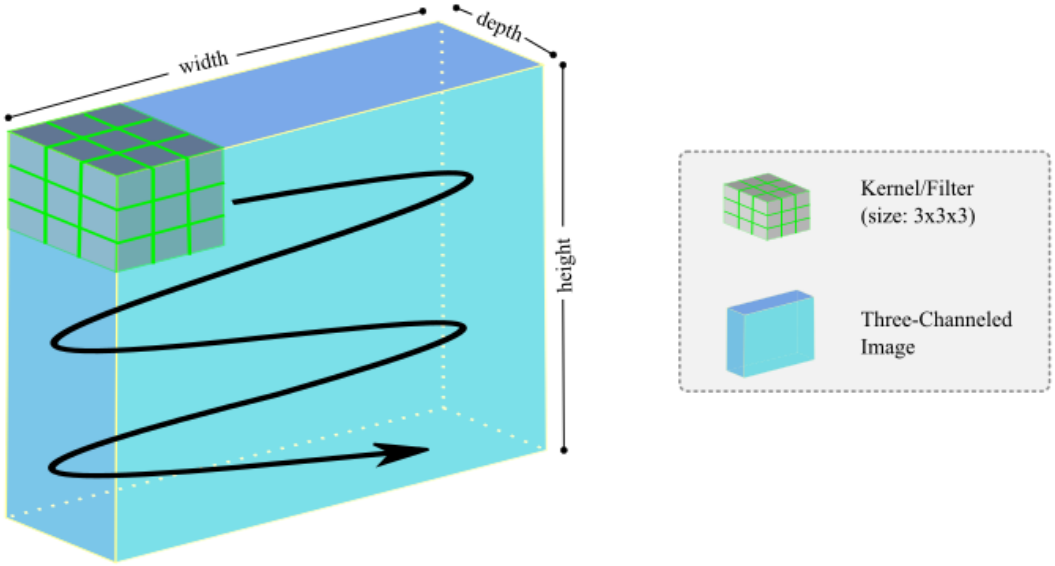
Each filter in convolutional layer is randomly initialized that is learned and adjusted with the passage of time while the network is being trained. For a three-channel image convolution, the kernel movement is shown in Figure 3.17. The convolutional layer retains the dimensionality of resultant feature maps or increases it, if *padding* is set to *same*. On the other hand, if *valid padding* is employed, it reduces the convolved feature maps' dimensionality.

Figure 3.16: Graphical representation of various activation functions used in CNN, (a) ELU, (b) leaky ReLU, (c) ReLU, (d) sigmoid, and (e) tanh



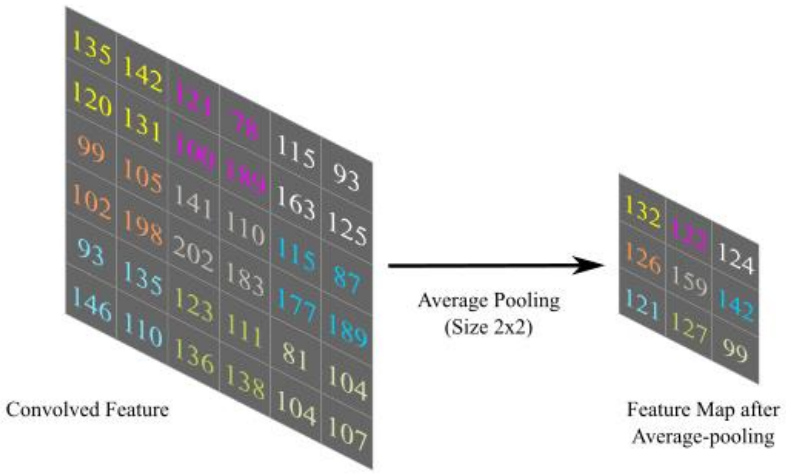
The second most commonly used layers in CNNs are pooling layers. These are specifically placed to minimize the computational power by progressively decreasing the spatial size of resultant features extracted by convolutional layers. Moreover, to train the model effectively, it extracts the positional and rotational invariant dominant representations. The dimensionality reduction through pooling is mainly performed in two ways;

Figure 3.17: Visual representation of kernel movement in multi-dimensional image



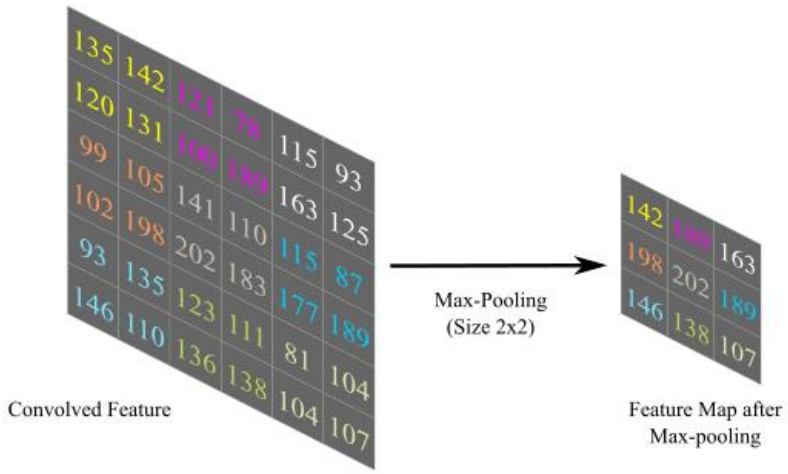
- Average-pooling; In this type of pooling, the values of feature maps on which kernel is placed are averaged as shown in Figure 3.18. It simplifies the activations formed by previous layers by minimizing the number of parameters through averaging operation.

Figure 3.18: Visual representation of applying 2×2 average pooling on extracted feature map for dimensionality reduction



- Max-pooling; It minimizes the dimensionality by selecting the feature that has maximum value in portion map on which kernel is placed. As shown in Figure 3.19, it takes the activation with biggest value in confined window, and discarded rest of activations.

Figure 3.19: Visual representation of applying 2×2 max-pooling on extracted feature map for dimensionality reduction



Beside feature dimensionality reduction, max-pooling is also preferred over average-pooling due to its de-noising capabilities that eliminates the noisy activations as a whole. Generally, a convolutional layer together with pooling layer is considered as single layer of a CNN. Based on the complexities in images, a CNN architect can have multiple convolutional and pooling layers for extraction of low-level information in order to fully understand the data at hand.

The last important type of layer in CNN is fully connected layer, also called dense layer. Similar to traditional NNs, each neuron in a fully connected layer is linked with all the activations of preceding layer to learn non-linear characteristics of all the representations extracted by preceding convolutional layers. It is usually placed at the end of CNN. Finally, extracted relevant features are flattened into a column vector which is then supplied to feed-forward NN for classification using *softmax* as activation function in case of multiple-class problem.

Generally, ConvNet exercises two main tasks: feature/representation extraction through pooling and convolutional layers and prediction/classification via dense

layers. CNN effectively handles the computationally intensive stuff such as 4K or 8K images by reducing high-resolution images into more meaningful representation and discarding irrelevant features that do not contribute in decision-making. The more building blocks (convolutional layers) enables the network to learn features that are more complex. The above-mentioned characteristics and features such as scalability to huge dataset and learning process must be considered while defining a CNN architecture.

3.3.1.2. VGG16

In 2012, the traditional deep learning technique based on ConvNet for image processing and classification advances with the advent of AlexNet (Krizhevsky et al., 2017). This not only revolutionized the CNN technique but also provided a breakthrough in the field of computer vision by successfully detecting objects in massive data. Unlike traditional architect that uses *tanh* as activation function, AlexNet employed ReLU and efficaciously utilized multiple GPUs for object detection. It won ILSVRC competition in 2012 by presenting an innovative methodology of deeper network (includes five, three, and three layers of convolution with ReLU, max-pooling and dense layers, respectively) with dropout technique and data augmentation method to avoid overfitting problem, thus achieved better accuracy.

This pioneering deep ConvNet encouraged the researchers to introduce novel methods, for image processing and classification problem. In 2014, researchers (Simonyan and Zisserman) of Oxford University introduced a more deeper ConvNet architecture called VGG (Simonyan & Zisserman, 2015), comprised of several block of max-pooling and 2D convolutional layers. The uniqueness of VGG is that instead of just focusing on lesser strides and window with smaller size, it also incorporated more weight layers.

VGG has several variants such as VGG11, VGG13, VGG16 and VGG19, each depending on number of weight layers in it, but basic functionality remains the same. For example, VGG13 has 13 weight layers while VGG16 has 16 weight layers in total. Each VGG architecture includes an input layer of size 224×244×3 (for three-channeled image), five blocks of several convolutional layers with 3×3 receptive fields having a

stride of 1, few convolutional filters of 1×1 size for linear transformation (in VGG16 and VGG 199), ReLU as activation function, and at end it has three dense layers.

Figure 3.20: General overview of VGG16 network architecture

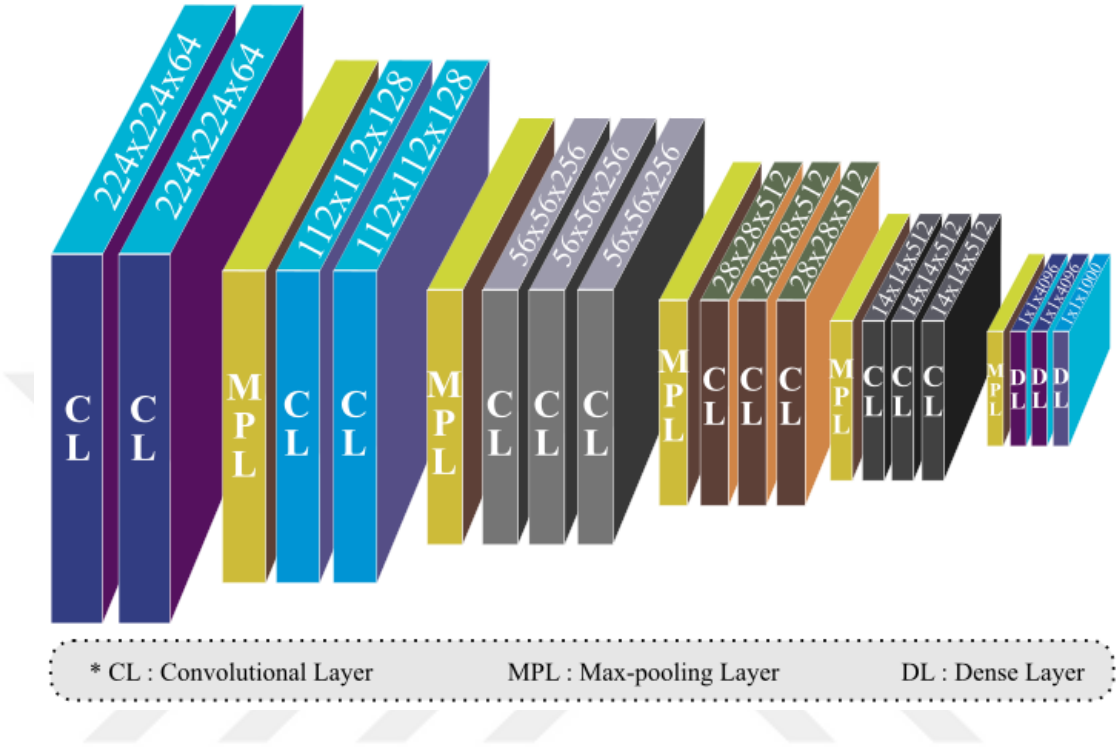


Figure 3.20 depicts an architectural overview of VGG16, having 16 weight layers. VGG16 has five convolutional blocks, each block two convolutional layers with 3×3 receptive fields. The last three convolutional blocks also contain one more convolutional layer having a filter of size 1×1 to preserve the spatial resolutional. Every convolutional layer in first, second, third, fourth and fifth blocks has a depth of 64, 128, 256, 512 and 512, respectively. A 2×2 max-pooling layer, with stride 2, follows each convolutional block. Later, three fully connected layers are placed in the end, where first two layers has 4096 nodes each, while last layer has 1000 nodes because of 1000 classes in ImageNet dataset.

VGG stands out among many deep learning based frameworks due to its several features such as usage of tiny 3×3 receptive fields, fewer parameters, and incorporation of more ReLU units to make an enhance discriminative decision function. Furthermore, the inclusion of convolutional layer of size 1×1 in last few block, turns the prediction processing non-linear.

3.3.1.3. ResNet50

Residual network (ResNet) is extremely vigorous and robust deep NN that accomplished astonishing results for ImageNet localization and detection in competition called ILSVRC-2015 (K. He et al., 2016). Similarly, it outperformed other models while securing better generalization performance on COCO detection and segmentation in Common Objects in Context (COCO 2015) competition.

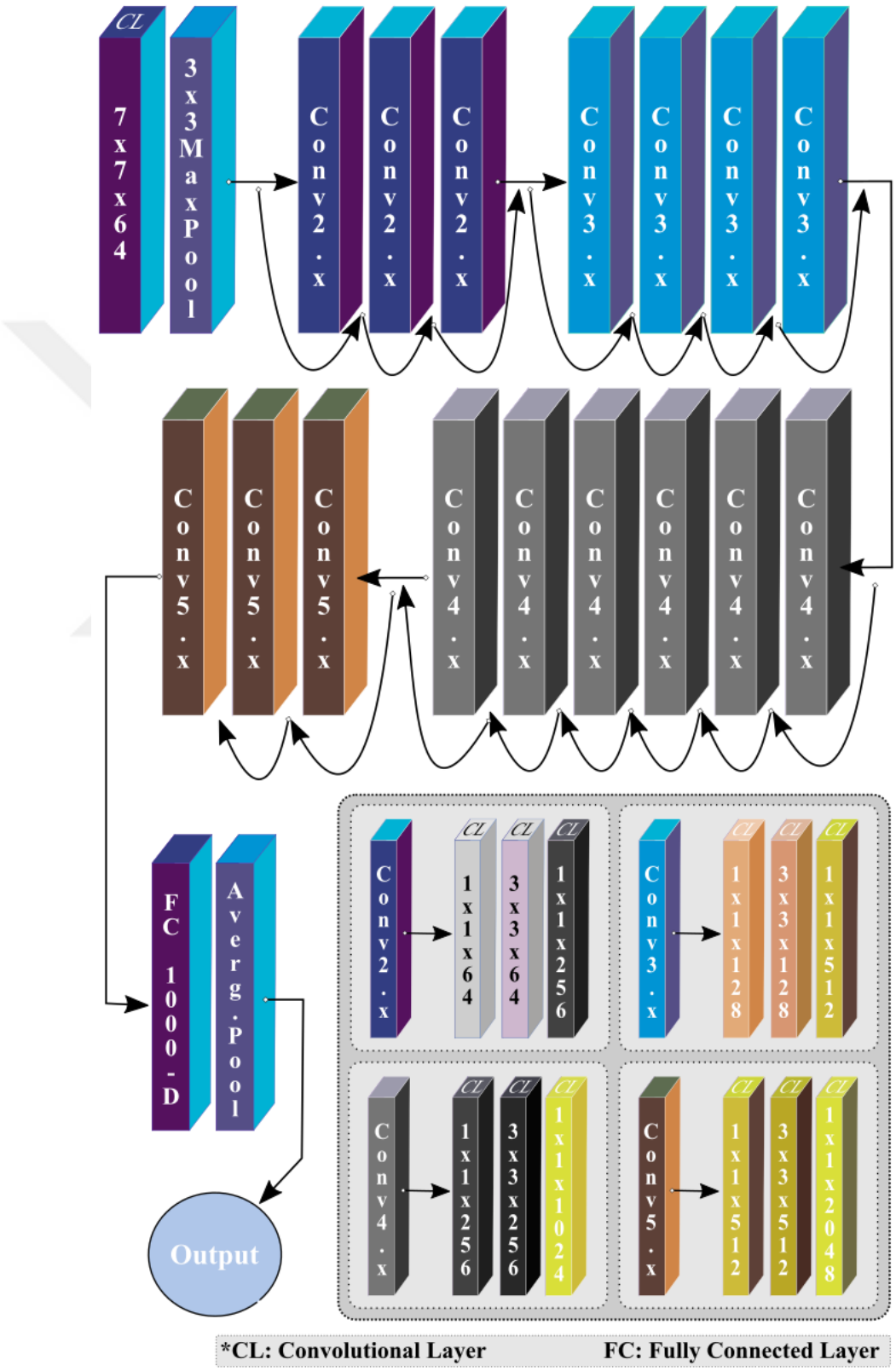
Prior to ResNet, deep learning researcher were developing deep networks by stacking more layers to extract complex features in order to attain high accuracy, but as the network converges, it faces a degradation problem. The increase in network depth, first, saturates the accuracy value, which then swiftly degrades due to vanishing gradient effect. (K. He et al., 2016) addressed the issue by introducing ResNet framework which stacks multiple convolutional layers like any other deep network, but at the same time it establishes identity connections between different layers. Instead of approximating the original mapping between input and output, these additional identity connections optimize residual mapping function by the induction of non-linear convolutional layers. To generate optimal identity mapping, the ResNet model learns the residual function such that it approaches zero. Near to zero residual-value enables the network to extract best case optimal feature map that encompasses pertinent features for accurate classification.

Basically, ResNet model was designed for image recognition and classification (ImageNet classification) but later it is adopted widely for other computer vision tasks. There are many variants of ResNet, but we exploited ResNet50 because of its popularity as vibrant network. It mitigates vanishing gradient effect and covariate shift problem by incorporation of identity connection and batch normalization respectively. The ResNet50 architecture consists of four stages, as depicted in

Figure 3.21. It takes the input image, having a size of 224×224×3 in case of this study, and passes it to first convolutional layer with a 7×7 kernel and then to max- pooling layer having a kernel size of 3×3. The resultant is then fed to first stage has three residual blocks with identity connections between them. Each block has three convolutional layers of 64, 64, and 256 kernels respectively. The second, third and fourth stages have four, six and three blocks, where each block has three convolutional layers of different kernel sizes. After every network stage, the input size reduces while

channel width gets double. Lastly, average-pooling layer is placed followed by a dense layer with thousand neurons.

Figure 3.21: General overview of ResNet50 network architecture



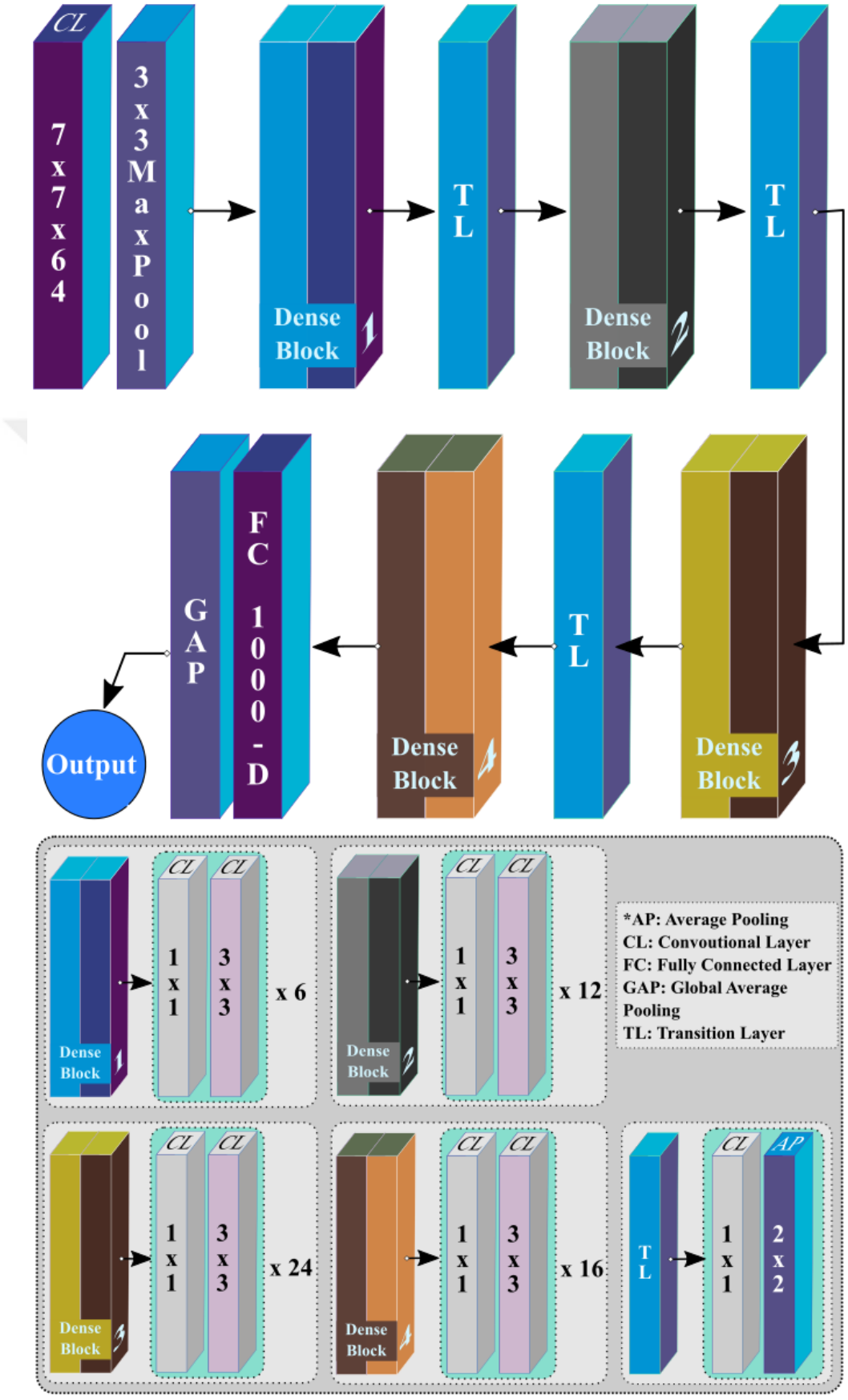
3.3.1.4. DenseNet121

In traditional deeper CNN models, the image features vanish before it reaches the destination due to longer passage between output and input layers. To overcome the issue of gradient vanishing effect, like ResNet, a new way of developing deeper convolutional network was presented by (G. Huang et al., 2017) that uses shorter but direct feed-forward connections between each layer of the network. The layers are placed in such fashion that first layer establishes links with all succeeding layer. Similarly, every next layer attaches itself with other succeeding layers to maximize the information flow and effectively utilize the extracted image features. Rather than learning representational power from deep framework, which requires more parameters, the model enhances the network's potential by reusing the features.

DenseNet is specifically designed to ensure maximum flow of information, thus resembles with ResNet but has few fundamental differences. ResNet adopts an additive method (+) to combine identity of succeeding layer with preceding layer, while DenseNet concatenates the incoming feature maps with output feature maps. For this study, DenseNet121, as name implies it contains 121 layers in total, is used to extract the effective features that can be used for classification while processing further.

Beside basic pooling and convolutional layer, DenseNet usually has series of blocks which are divided into transition layer and dense blocks. Initially, a convolution layer with 64 filters, each having size of 7×7, is followed by max-pooling layer of size 3×3. Next, it contains 4 dense block, each followed by transition layer, except the last dense block which is followed by classification layer. In DenseNet121 (see Figure 3.22), each dense block consists of several pairs of convolutional layers, having a fixed size kernels of 1 × 1 and 3 × 3. The first, second, third and fourth dense blocks have 6, 12, 24, and 16 pairs of convolutional layers. Therefore, the feature maps dimensionality within a dense block remains constant while number of filters varies. The transition layer performs the down sampling by introducing a 1 × 1 convolutional layer and 2 × 2 average-pooling layer. Finally, a 7 × 7 global-average-pooling is performed followed fully connected layer with 1000 neurons.

Figure 3.22: General overview of DenseNet121 network architecture



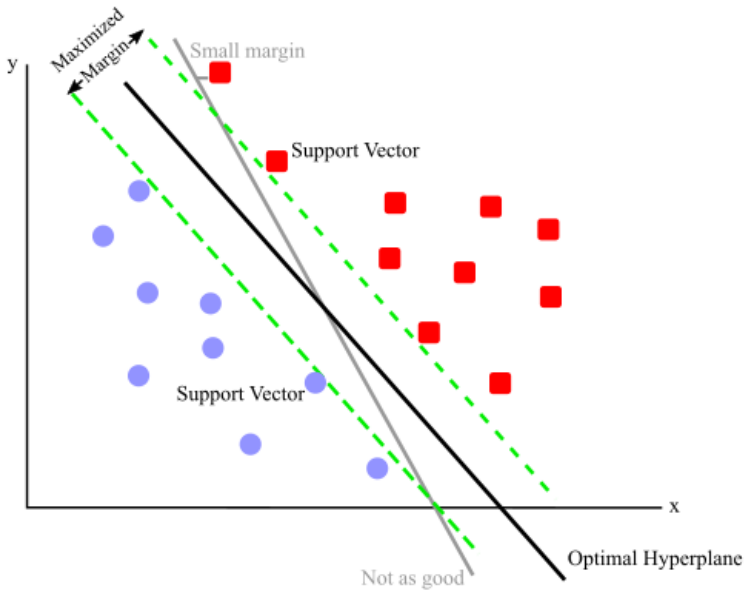
3.3.2. Machine Learning-based Methods and Models

3.3.2.1. Support Vector Machine (SVM)

The ground-work for multilayer perceptron-based NNs was laid by Frank Rosenblatt (Rosenblatt, 1957), who proposed the concept of perceptron based single layer linear model that had only one computational neuron. Years later, a new model called maximal margin classifier was proposed by researchers of AT&T Bell Laboratories (Vapnik and Chervonenkis) that built the foundation of SVM. In 1990s, Vapnik along with co-authors first introduced an idea known as 'kernel trick' that enables SVM to linearly solve non-separable information, and then suggested soft-margin classifier for SVM (Cortes & Vapnik, 1995). Therefore, SVM models evolved with time that includes various versions and features; maximal margin classifier, kernel trick version, soft-margin, and combination of all these three.

SVM is a machine learning-based supervised model utilized to solve regression, classification and outliers' detection tasks. It is widely adopted as discriminative classifier to deal with linear as well as non-linear tasks. In order to correctly classify instances, SVM aims to define a decision boundary that maximizes the separating distance between the training data. Thus incorporates the first feature known as maximal margin classifier. Figure 3.23 shows a decision boundary that tries to find support vectors, instances located on the edge of class-descriptors, so that it can appropriately tag the samples into two classes. It is achieved by dividing the 2-dimensional space with a line such that data points falls on left side of boundary segregates as different class than the ones on the other side. This division can be achieved by infinite amount of lines but what makes SVM outstanding among others (like k-nearest neighbors) is its ability to discover the best optimal possible separation line that has maximum margin between support vectors. This separating line is also known as hyperplane if it is for more than 3-dimensional space.

Figure 3.23: Support vectors and decision boundary in simple linear support vector machine classifier



A margin is double of the minimum distance between hyperplane and closest support vector. Generally, no instance/data-point/sample falls inside the margin, but is not valid for noisy data. The bigger the margin is, the more optimal a hyperplane is. A simple maximum margin classifier based SVM easily chose a decision boundary to classify linearly separable data points, but for non-linearly separable data it is impossible to define a boundary line for classification (see Figure 3.24). Therefore, SVM uses kernel trick, also called generalized dot product, which cleverly maps the features into linearly separable form. According to Cover's theorem, it projects the training data into higher dimensional space through some non-linear transformation functions. Therefore an addition of one more dimension 'z', means transforming to higher dimension (from 2-dimensional to 3-dimensional) as shown in Figure 3.25(a), conveniently enables the model to separates the complex data points with a hyperplane having a circumference of certain radius (see Figure 3.25(b) that represents the mapping it back to 2-dimensional).

Figure 3.24: Non-linearly separable data samples in support vector machine

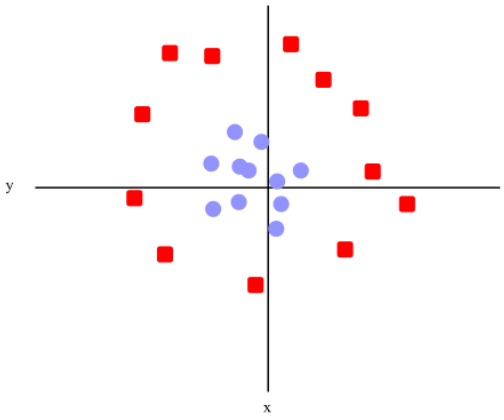
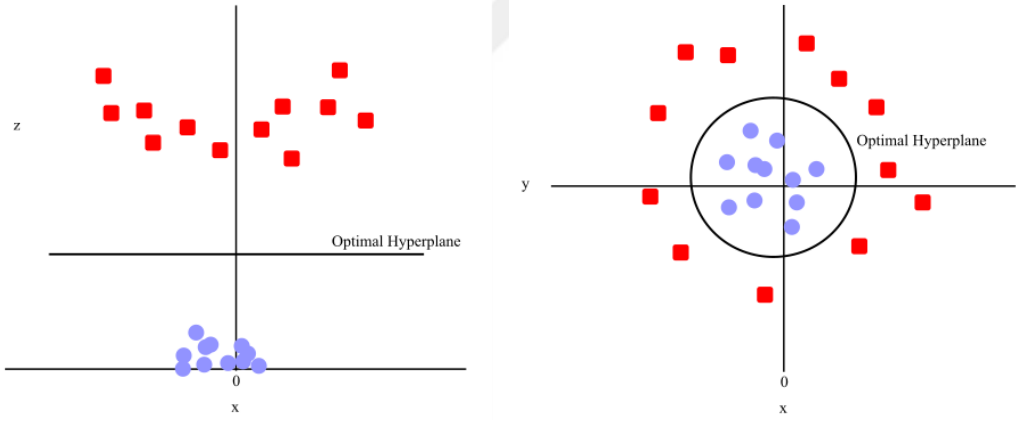


Figure 3.25: Kernel tricks in support vector machine classifier for non-linear complex cases, (a) transforming the space to higher dimensions (3D representation) to find optimal hyperplane, and (b) 2D representation after finding hyperplane



(a) (b)

Instead of transforming the space into higher dimension, that is computationally expensive task for each vector in huge dataset, SVM actually performs dot product of transformed input vectors, therefore it is called kernel trick. Several kernel functions are used according to user's need for boosting the training time, meeting specific error constraints, or parameter tuning requirements. The most famous kernels includes; linear, polynomial, RBF, and sigmoid. A linear kernel is defined by dot product of data

'$\boldsymbol{X}$' (which is to be categorized) with weight vector '$\boldsymbol{w}$' (which user wants to minimize) and addition of linearly estimated coefficient 'b':

$$\text{KernelFunction}(\mathbf{X}) = \boldsymbol{w^T}\mathbf{X} + \text{b} \tag{10}$$

The polynomial kernel generates with the help of polynomial sequences of entire existing features using equation like (11) that depicts polynomial decision boundary for data $\boldsymbol{X_1}$ and $\boldsymbol{X_2}$. Such kernels are not widely adopted due to more computational cost and less prediction accuracy than others.

$$\text{KernelFunction}(\boldsymbol{X_1}, \boldsymbol{X_2}) = (a + \boldsymbol{X_1^T X_2})^b \tag{11}$$

The most used kernel for non-linear data is RBF that computes distances between specific features with all others to generate new features. Gaussian RBF is defined by:

$$\text{KernelFunction}(\boldsymbol{X_1}, \boldsymbol{X_2}) = e^{(-\gamma * \|X_1 - X_2\|^2)} \tag{12}$$

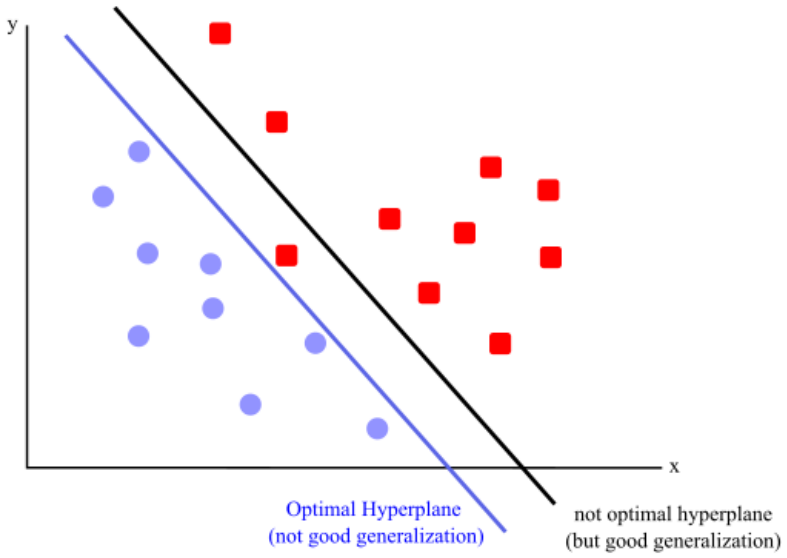
where γ is a learnable parameter. Sigmoid is another kernel, mostly used in NNs but occasionally utilized in SVM models. It is calculated using *tanh* function given in (13)

$$\text{KernelFunction}(\mathbf{X}, \mathbf{y}) = \tanh(\alpha * \boldsymbol{X^T} * y + C) \tag{13}$$

where C and α are off-set value and weight vector, respectively, that accounts the misclassification of data if happens.

The above discussion presents the first two versions of SVM, namely, maximum margin classifier and kernel version. The third version is the soft-margin based SVM. As main objective of SVM is to reduce the amount of misclassification error directly, but this is not applicable for real world problems where most of the time the data is linearly inseparable, and in some cases such a hyperplane is not desired that perform perfect classification as shown in Figure 2.26. The optimal hyperplane (in blue color) completely classify entire training data, but it does not generalizes well on unseen data. Whereas a hyperplane (in black color) with wider margin, do some misclassification but overall generalizes well. Therefore, for such cases soft-margin based SVM plays important role to bypass overfitting issue.

Figure 3.26: Soft-margin cases in support vector machine



In soft-margin based SVM, models are allowed to do few mistakes (misclassifications) that can maintain the margin as wide as possible in order to correctly classify other data points. This is achieved by modifying the objective function in second term defined in (14).

$$L = \frac{1}{2}\|w\|^2 + C(mt) \tag{14}$$

Here, *mt* is the number of mistakes, and C is the trade-off hyperparameter. A large C means that margin is kept small to avoid misclassifications whereas a smaller value of C corresponds to larger margin with few misclassifications. This technique is best for noisy data. A fourth variant of SVM is the combination of kernel version, soft-margin and maximum margin classifier. Therefore, while programming it is usually asked to mention the hyperparameter C and kernel type to administer the impact of soft-margin and kernel trick, respectively.

3.3.2.2. Logistic Regression

LR is a predictive analysis algorithm that is extensively utilized to classify observations into discrete set of classes by transforming the output to get a probability

value (Peng, Lee, & Ingersoll, 2002). It is the most basic machine learning classification algorithm based on the concept of probability. LR is somewhat similar to linear regression, but there is a main different in them. Linear regression uses a simple *linear function* whose resultant values can be smaller than 0 and greater than 1. Unlike linear regression where a linear cost function is used to predict the output, LR exploits a more complex function that returns the probability values. As the model name implies, the cost function is called *logistic cost function*, which is also known as *sigmoid function*. The hypothesis of LR states that this sigmoid function limits the values between 0 and 1, thus the hypothesis expectation is represented by

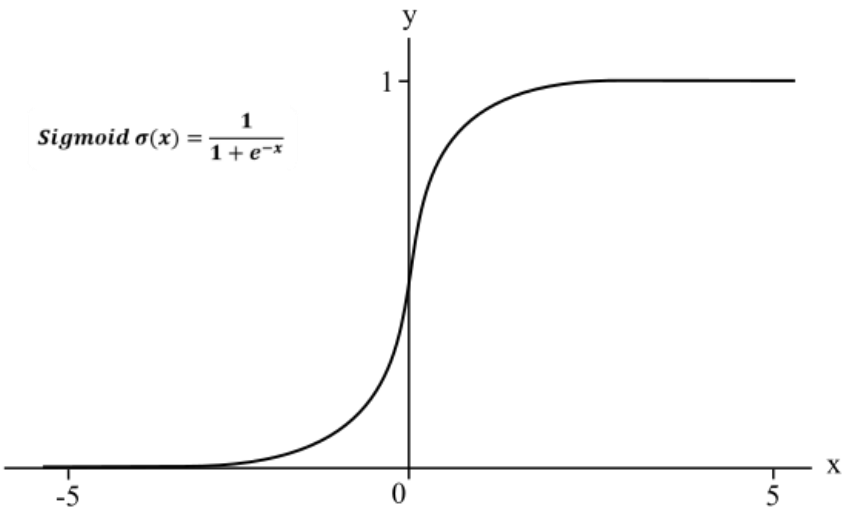
$$0 \leq h_\theta(x) \leq 1 \tag{15}$$

where *x* is the input. This logistic function maps real values to a corresponding value that falls between 0 and 1. Therefore, this sigmoid function is used to map the values predicted by the model to appropriate probabilities. Hence, the sigmoid (σ) function is shown in Figure 3.27 and represented by

$$\sigma(Z) = \frac{1}{1 + e^{-x}} \tag{16}$$

where $Z = \sum w_i x_i + bias$, and *w* is the weight.

Figure 3.27: Graphical representation of sigmoid function in logistic regression



The hypothesis of linear regression is given by

$$h_\theta(x) = \beta_0 + \beta_1 X \tag{17}$$

where βo the intercept term (biases) and β1 refers to weight. Thus for LR it becomes

$$\sigma(Z) = \sigma(\beta_0 + \beta_1 X) \tag{18}$$

As range of values remains between 0 to 1, thus

$$\mathrm{Z} = (\beta_0 + \beta_1 X) \tag{19}$$

$$h_\theta(x) = \sigma(\mathrm{Z}) \tag{20}$$

$$h_\theta(X) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X)}} \tag{21}$$

In order to build an accurate model, a cost function is monitored to minimize the error. The linear regression cost function or error function is represented by

$$\mathrm{J}(\theta) = \frac{1}{2} \sum_i^m (h_\theta(x^{(i)}) - y^{(i)})^2 \tag{22}$$

where $y^{(i)}$ is the predicted output, and $h\Theta(x^{(i)})$ corresponds to the predicted value. Being a non-convex function, it is difficult to reduce the error as this cost function produces many local minimums. As it is challenging to find a global minimum, this cost function cannot be used in LR. Thus LR cost function is defined as

$$\mathrm{Cost}(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)), & if\ y = 1 \\ -\log(1 - h_\theta(x)), & if\ y = 0 \end{cases} \tag{23}$$

It can be rewritten as single cost function

$$\mathrm{J}(\theta) = -\frac{1}{m} \sum \left[y^{(i)} \log\left(h_\theta(x^{(i)})\right) + (1 - y^{(i)}) \log(1 - h_\theta(x^{(i)})) \right] \tag{24}$$

3.4. OCR for Text Recognition

Recognizing an artificial text is actually to automate the exhausting data entry task. These automated task includes reading text from receipts, credit cards, vehicle number plate, business cards, and images for inventory database, bank records, traffic violation

system, trade records, and translation purposes etc. To automate such tasks, experts have introduced various optical character readers that can recognize the text present in an image. OCR is a system that electronically identifies and performs digital encoding of printed or typed text by using a specialized software and optical scanner. This specialized software allow the machine to read images to get the text into an searchable and editable form. OCR software generally consists of three following three steps:

- Opening/Scanning an image or document
- Recognizing the text
- Saving the recognized text in a desired format

In computer vision community, following three OCR software are popular and widely adopted for text comparison and data mining tasks:

- Adobe Acrobat Pro[11]
- ABBYY FineReader[12]
- Tesseract[13]

OCR software packages does not support following features

- Supper tiny fonts (minimum of 6pt)
- Special fonts
- Low contrast text
- Handwritten documents

Most of these OCR software packages support various text languages but are limited to text that is published after 1850 year. Few of other drawbacks of OCR software are

- Poor performance for inconsistent use of font faces.
- Lesser accuracy for irregular font sizes

[11] Adobe Acobat Pro: https://acrobat.adobe.com/us/en/acrobat/acrobat-pro.html

[12] ABBYY Fine Reader: https://pdf.abbyy.com/

[13] Tesseract: https://github.com/tesseract-ocr/tesseract

- Undesired and miserable recognition if image contain multiple elements such as headings, tables, captions, tiny shapes etc.

Majority of OCR engines are commercial, thus this study uses an open-source OCR package called Tesseract to recognize text from video frames. A brief descript of Tesseract is provided below.

3.4.1. Tesseract OCR Engine

An OCR system turns images (containing text) into an editable text file. An initial PhD research project (R. W. Smith, 1987) later extended into a team project at HP Lab that developed an OCR engine, called Tesseract, in duration of 10 years (1984 – 1994). In 1995, the developed product proved its worth by outperformed many commercial engines (Rice, Jenkins, & Nartker, 1995). After its success, in 2005, HP made it an open-source OCR engine (R. Smith, 2007).

At the time of its development, the human-computer interaction was based on command-line interface (CLI), thus Tesseract package is run via CLI. As it has a remarkable lead in accuracy over other commercially available text recognition engines, it is widely adopted by giant companies such as Google uses it for detecting spam images in Gmail emails[14]. Since 2006, Google is sponsoring it development and publicly providing programming knowledge so that users can edit the code for learning and understanding purposes.

Tesseract works in following steps:

1. Input image with appropriate file format
2. Recognizes the text (if there is any)
3. Display the recognized text (if user requires)
4. Writes the recognized text into file
5. Saves the generated text file in user's desired format

---

[14] Tesseract OCR, How Google uses Tesseract OCR: https://opensource.google/projects/tesseract

It supports following types of input image file format

- BMP (bitmap)
- JFIF (file interchange format: image data encoded with JPEG algorithm)
- JPEG (joint photographic experts group)
- JPG (joint photographic experts group)
- PNG (portable network graphic)
- PNM (portable any map)
- TIFF (tagged image file format)

It does not support GIF file format for in input file. On the other hand, it can store the generated text file in following format at desired location.

- hOCR (.html)
- Microsoft Word (.doc, .docx)
- PDF (.pdf)
- Plain text file (.txt)
- TSV etc.

Tesseract is preferred for this study as it recognizes more than 100 languages, including Turkish language as well, with ability to support Unicode data

CHAPTER 4
EXPERIMENTAL SETUP AND RESULTS

Beside the objective of building a content-based video indexing and retrieval system for Turkish videos, the main goal is to detect and extract text appearing in video frames using deep learning approaches. To accomplish this, this study generated a new dataset from Turkish videos and applied some image pre-processing techniques. The research proposed CNN-based deep learning model to identify the region as text or non-text area of an image. The generated dataset were randomly shuffled and divided into train and test sets, and then trained the proposed CNN-based framework. Extensive experiments are carried out with various possible combinations of convolutional and pooling layers to analyze the performance of proposed models on unseen test set.

In addition to proposed model, the study also implemented few state-of-the-art deep learning-based image classification models (VGG16, ResNet50, and DenseNet121) and conventional machine learning approaches (SVM and LR). These models are trained on generated dataset and results were compared with proposed models. Thus, the experimental flow consists of some major stages: firstly, some image preprocessing techniques are applied to raw input videos for data set generation. Next is the classification stage, in which the proposed CNN-based models are trained on generated dataset to classify image as text/non-textual region. Beside the proposed model, other state-of-the-art models are also implemented and compared. In succeeding stage, a commercially available OCR is used to recognize the detected text, and a database is built to store the recognized textual content for indexing. Finally, a web-based user interface as content based video search engine, where user can query and relevant videos are retrieved accordingly.

This chapter presents the experimental setup for dataset generation and image preprocessing, proposed CNN-based model training and testing, results and performance analysis of proposed model along with previous state-of-the-art text detection models. In the end, it provides the overview of database and web-based video searching platform built for this study.

4.1. Data Generation and Image Preprocessing

As stated earlier, the Turkish text detection and extraction is infancy state, and no such dataset is available publicly, therefore a new dataset is generated to train the deep learning-based classifier. Around 100 Turkish videos related to News, sports and financial channels, containing horizontally aligned artificial text were download from different online platforms. Each downloaded video belongs to three-color space (RGB). As proposed in schematic diagram (see Figure 3.2), every video is first dismantled into frames and then the color-space is converted from three-channel space to gray scale. Figure 4.1(a) presents+ one of the sample frame, having three-channels (RGB), from a downloaded video, while Figure 4.1(b) represents the image frame in equivalent gray-scale color space. This color-space conversion eradicates the irrelevant computation by reducing the number of channels and adjusting pixel values in range 0 to 255.

Like other objects in these digitized frames of dismantled videos, the text is also considered as shape/object whose region is dominant by small boundaries in horizontal direction as well as in vertical direction. To extract these boundaries and edges in grayscale image, Sobel filters defined in Figure 3.8 are employed on gray-channel frame to get the objects' boundaries and edges. Each alphabet in Turkish language possess different characteristics, such as some textual characters (alphabets) dominate with horizontal boundaries whereas others prevail with vertical edges. As both horizontal and vertical boundaries contains useful information, hence, both edges are computed independently. To acquire boundaries in horizontal course, a Sobel edge detector with mask shown in Figure 3.8(a) is employed, whereas mask depicted in Figure 3.8(b) is used to get vertical edges of the objects. Figure 4.2 shows the projection analysis on a sample gray-scale image, where image frame in Figure 4.2(a) depicts the edges extracted via horizontal projection analysis, while image frame in Figure 4.2(b) depicts the edges found through vertical projection analysis.

Figure 4.1: Sample image, (a) image having 3-channels, and (b) greyscale conversion of image in (a)

(a)

(b)

Figure 4.2: Edges detection by projection analysis on image (a) after horizontal projection analysis, and (b) after vertical projection analysis

(a)

(b)

Later the resultants (horizontal edge profile and vertical edge profile) of Sobel edge detector are combined to get the boundaries of entire shapes. The implication of Sobel edge detector not only determines the edges of shapes in an image, but it also introduce an undesired background noise, which may affect the training on deep learning classifiers. Therefore, to reduce this unwanted noise, an Otsu's global thresholding technique is employed. The application of Otsu's thresholding requires a user defined thresholding value, thus an empirically calculated value of 130 is set as optimal thresholding filter. Figure 4.3 shows a resultant image frame with combined edges of horizontal and vertical projection analysis, after applying Otsu's global thresholding.

Figure 4.3: Resultant image after applying sobel filter

According to component analysis, the joining of boundaries and edges of various textual elements in an image can establish a bigger element such as line. The density of edges is normally higher in textual regions because of huge amount of small boundaries present in specified regions Thus these edges can be connected to form horizontal regions. Therefore, for this study, the obtained textual boundaries via projection analysis are joined with the neighboring textual edges to embrace a shape of horizontal textual lines using morphological operation. Dilation (described in Chapter 3 of this study) is employed to link edges and a resultant sample image frame is demonstrated in Figure 4.4. A [1, 11] is used as dilation vector, which was calculated empirically.

Figure 4.4: Resultant image after performing dilation operation

Eventually, bounding boxes are formed around these dilated areas to obtain these regions. As illustrated in Figure 4.5, besides just linking the text regions, the dilation morphological operator connects the non-text regions as well. Therefore, some geometrical analysis and heuristic constraints are applied to obtain candidate textual regions while discarding unwanted non-text areas.

Figure 4.5: Image with unwanted noise in bounding boxes before applying geometrical and heuristics constraints

The implication of various empirically determined constraints helped to obtain the horizontally aligned textual areas from each frame under operation. By observing texts

of varying lengths and heights in different videos, it was noted that height of a textual content is too small than its width in general. The observation concluded that width of the text appearing in Turkish videos is at least 1.5 times more than its height, thus equates less than 0.667 as height to width ratio. Through inspecting various videos, the author also noticed that the bounding boxes around textual content are at least 30 pixels and height covers more than 10 pixels. It is also noted that pixel-to-density ratio of Turkish text is more than 50 when compared with the background of its bounding box. By applying this geometrical analysis, the empirically computed heuristics constraints successfully eliminates the noise further as depicted in Figure 4.6.

Figure 4.6: Image with potential candidate text in bounding boxes after applying geometrical and heuristics constraints

Subsequently, the textual content surrounded by the bounding boxes is detected and extracted. Moreover, the regions marked as unwanted noise during heuristic and geometrical constraints are also extracted as non-textual content to form a complete dataset. The extracted textual content in image has different levels of grayscale, such as some textual elements are displayed in white color with black in the background as depicted in Figure 4.7(a). On the other hand, few images contains text and background represented in black and white colors, respectively, as demonstrated in Figure 4.7(b).

Figure 4.7: Samples of text images obtained, (a) normal text, and (b) inverse text

(a)

(b)

The empirical analysis deduced that the textual pixels are less in density than the background pixels, and noted that white-to-black pixel-density-ratio should be less 50. Therefore, the pixel values are switched (from black to white) of those textual content where the white-to-black pixel-density ratio is greater than 50.

As discussed in Chapter 3, the length height of the textual lines not only varies from video to video, but also varies with a frame of a particular video; hence, the extracted textual images are of different sizes. For the sake of uniformity in generated dataset, each extracted image is resized to height of 32 pixels. In the last phase of preprocessing technique, each resized image is vertically divided into desired chunks, where each chunk must has a width of 128 pixels. While dividing each extracted textual line into chunks of images, if the width of last chunk is less than 25 pixels, it is discarded. Contrarily, if last chunk has width greater than 25 but less than 128, then a pure black image is joined at the end of last chunk to adjust the width to desired level of 128 pixels as illustrated in Figure 4.8.

Figure 4.8: Image formatting; a text image of 55×1247 pixels is adjusted to height of 32 pixels and divided into sub-part with a width of 128-pixels. As the last sub-part has size of 95 pixels that is less than 128 pixels, thus a pure black image of 33 pixels width is added at the end

The generated dataset consists of 231622 samples out of which 122262 samples contains the textual content while 109360 samples are non-text images. For training and evaluation purposes of various machine learning and deep learning based classifiers, the dataset is further split into test and train set with a ratio of 30-70%. Thus test set constitutes 85583 text-images and 76552 non-text images, while test set contains 36679 textual images and 32808 non-text images. Table 4.1 shows the detail split of generated dataset into test and train sets for this experimental study.

Table 4.1: Generated dataset split as train and test sets

	No. of Instances	***Train set***	***Test set***
Images with textual content	122262	85583	36679
Images with non-textual content	109360	76552	32808
Total	**231622**	**162135**	**69487**

4.2. Performance Metrics

Obtaining a final prediction is not the end of the classification process, but determining the correctness of prediction is important to investigate the efficacy of the system. Mathematicians and scientists have introduced several measurements to perform this task. Out of various measurements, there is no individual finest measurement to judge the quality of the prediction. The best measurement approach depends on the data at hand and the task domain.

The dataset generated for this research, consists of two-classes; text and non-textual, and each implemented model performs a binary classification In this study, following performance metrics are considered due to their wide usefulness and conceptual simplicity to evaluate the correctness of each implemented model.

- Accuracy
- Confusion Matrix
- F1-Score
- Precision
- Recall

Confusion matrix is not an analytical measurement but this study consider it to present a more concise visual representation of exact numbers of correct vs incorrect predictions. It represents the four important building blocks (true positive, false positive, true negative, and false negative) used in other performance metrics. A true positive (*tp*) is that the network correctly identifies the positive class, while a false positive (*fp*) is that when it predicts a negative-class as positive. Similarly, a true negative (*tn*) is when a network correctly predicts the negative-class, while a false negative (*fn*) is when it misclassify a positive-class as negative. Based on these four parameter, the rest of the metrics are measured.

Accuracy is usually the first metrics that a researcher calculates while evaluating the results to get ratio of correct prediction over total number of predictions by the network, as show below.

$$Accuracy = \frac{tp + tn}{tp + fp + tn + fn} \tag{25}$$

Next is the precision, used to determine the rate of *fp* with help of following

$$Precision = \frac{tp}{tp + fp} \tag{26}$$

Precision is not useful to minimize the *fn*, thus a new metrics called recall is calculated by

$$Recall = \frac{tp}{tp + fn} \tag{27}$$

It is an opposite of precision, and determines on *fn* against *tp*. Some precision and recall yields similar results, thus makes it unsuitable for unbalanced datasets, a fourth metric called f1-mearsure or f1-score is calculated by

$$f1 - score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{28}$$

It is basically a harmonic mean of recall and precision to balance the two objectives; high recall and high precision, thus makes it more resistant to outliers. Therefore, f1-measure corresponds to a balanced performance metric, which properly quantifies the correctness of model across various domains.

4.3. Classification

The main aim of this study is to detect the text present inside a video frame and then extract those textual elements, which are later fed to commercially available OCR for recognition. For text detection and image classification, the study conducted different experiments with various CNN architectural combinations of layers. The experiments are performed separately to test the efficiency of each combination of proposed CNN-based model. The details of each model and their evaluation results are described in following sub-sections along with the other state-of-the-art image classification models.

All the experimental work is carried out using Python 3.6 and Keras packages on Jupyter Notebook (version 6.0.3) having Intel® Xeon® CPU E3-1231 v3 3.40 GHz processor with 32 GB RAM. In addition, the code is executed using graphical processing unit (GPU) NVIDIA Quadro K500. Table 4.2 enlists the software,

hardware and system specification/configuration used to carry out the experimental work.

Table 4.2: Software and hardware specification for classification experiments

Hardware Specification	
Made	HP
Processor	Intel® Xeon® CPU E3-1231 v3 3.40 GHz
Memory	32 GB RAM
Storage	1 TB
Graphics (GPU)	NVIDIA Quadro K500 (512 MB)
Software Specification	
Operating System	Windows 10 Pro 64-bit
Language Distribution Platform	Anaconda (Individual Edition)
Development Environment	Jupyter Notebook (version 6.0.3)
Language	Python 3.6
Libraries/Packages	Keras, and TensorFlow

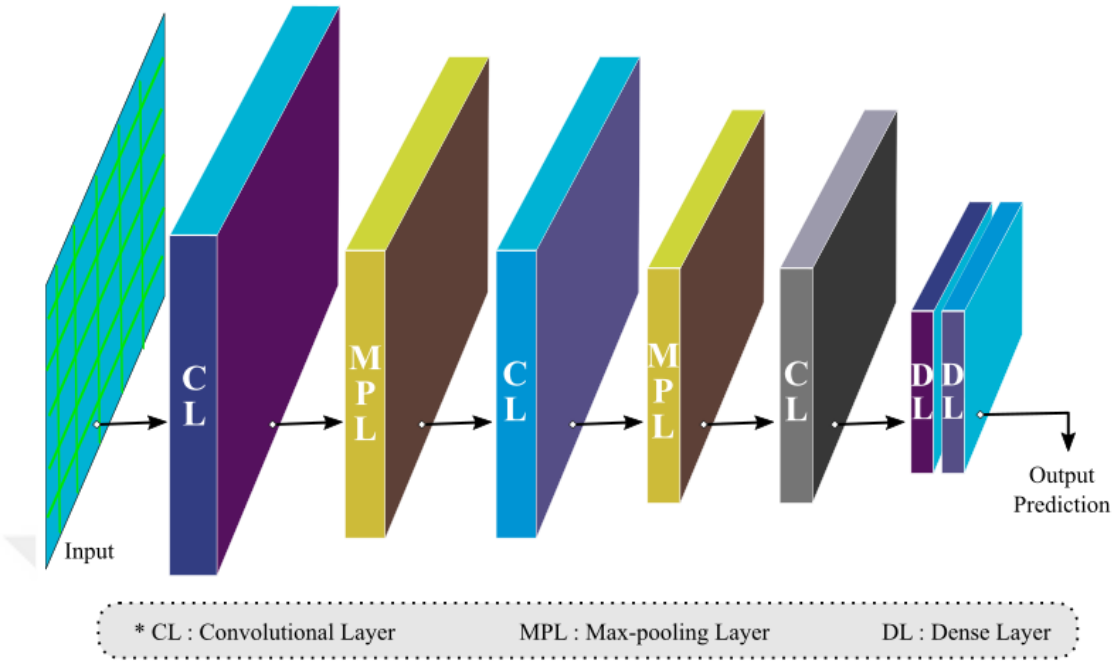
4.3.1. Proposed Models

The main component of proposed framework is deep learning-based NN model that is trained to quickly deduce the existence of text in images. All the models proposed below are based on different combinations of CNN architecture to extract effective features and predict whether the image contains text or not. The details of experimental setup and performance analysis of each model is given below.

4.3.1.1. Model-A

A detail architectural setup of proposed Model-A is depicted in Figure 4.9. In the beginning of the architecture is the input layer that has a size of 32×128×1. The depth is set to 1, because of the grayscale color space conversion, while 32×128 refers to height×width of the input image.

Figure 4.9: Overview of proposed Model-A architecture



The first major layer of proposed model is convolutional layer that has a filter size 3×3 to scan the image for feature extraction. It uses a non-linear activation function called ReLU to squeeze the activation and output a resultant feature map. To ensure that the extracted representation retains the same size as of input map, padding is turned-on. Next is a max-pooling layer with a pool size of 2×2 and a stride of 2 for down-sampling the feature map extracted by convolutional layer. As described in Chapter 3, every convolutional layer has definite number of filters that decides the amount of features maps to be extracted. Therefore, in first convolutional layer 8 filters are used.

The resultant of max-pooling layer is passed to another convolutional layer to extract high-level features. The kernel size of convolutional layer is same as of first convolutional layer, but the number of filters are increased to 16. Again, ReLU is used as activation function for this convolutional layer and padding is turned-on to retain the size of spatial output. This layer is followed by a max-pooling layer that has a stride of 2 with pool-size of 2×2. Then, a last convolutional layer is placed with 32 filters, each having a size of 3×3, and ReLU as activation function.

The training of the network is boosted by employing a batch normalization layer between each ReLU and convolutional layers that normalizes the propagating gradient

and the activations in the network. A fully connected layer is placed at the end of the model to classify the image by gathering all the representation learnt by preceding layers. Being a binary-classification problem, a sigmoid is used as activation in last layer to normalize the output that sums up to one. The network computes the losses based on the probabilities, and accordingly distinguish the image as text or non-text. A detail network topology of proposed model is listed in Table 4.3.

Table 4.3: Network topology of proposed Model-A

CNN Proposed Topology							
Dataset	**Convolution Layers**				**Fully Connected Layers**		
	Kernel Size	ReLU	Max-Pooling	Dropout	No. of neurons	Function	Dropout
Turkish Text	3×3×8	Yes	2×2	No			
	3×3×16	Yes	2×2	No			
	3×3×32	Yes	No	No	128	ReLU	No
					2	Sigmoid	No
Other Parameters							
Batch size	Epochs	Learning rate	Optimizer				
30	25	0.001	SGDM				

The network uses stochastic gradient descent with momentum (SGDM) as optimizer and model is trained for 25 shuffled epochs with an initial learning rate of 0.001 and validation frequency of 30.

The model is first trained on training set and later its performance is analyzed using test set. It achieved an overall accuracy of 99.54% on test set, while it secured a precision, recall and F1-Score of 99.52%, 99.55% and 99.54%, respectively, as listed in Table 4.4. The accuracy and loss curves for training set are depicted in Figure 4.10, whereas Figures 4.11 shows the loss and accuracy curves for testing.

Figure 4.10: Training loss and accuracy curves of proposed Model-A

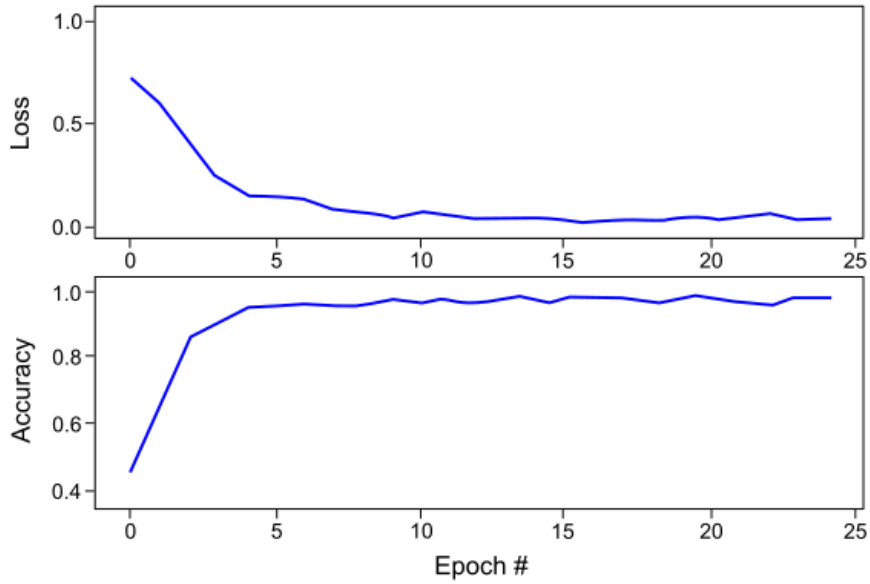
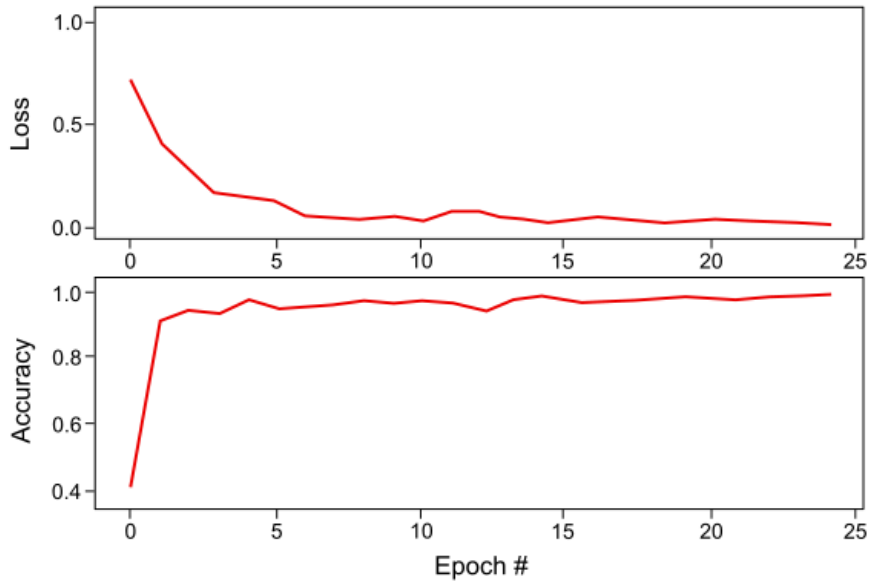


Figure 4.11: Testing loss and accuracy curves of proposed Model-A

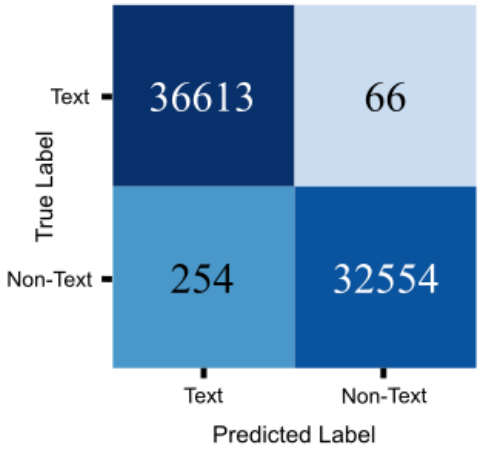


The model misclassified few of text samples and non-text images. As it achieved a precision of 99.82%, recall of 99.31%, and F1-score of 99.56% for textual images, thus it misclassified 66 textual images as non-textual images out of 36679 textual images in test set. On the other hand, model secured a precision of 99.23%, recall of 99.80%, and F1-score of 99.51% for non-textual images, therefore, it wrongly identified 254 non-textual images as textual image, as depicted in Figure 4.12.

Table 4.4: Performance evaluation of proposed Model-A on test set

Test Samples	*Performance Analysis (%)*			
	Precision	***Recall***	***F1-Score***	***Accuracy***
Text	99.82	99.31	99.56	-
Non-text	99.23	99.80	99.51	-
Overall	**99.52**	**99.55**	**99.54**	**99.54**

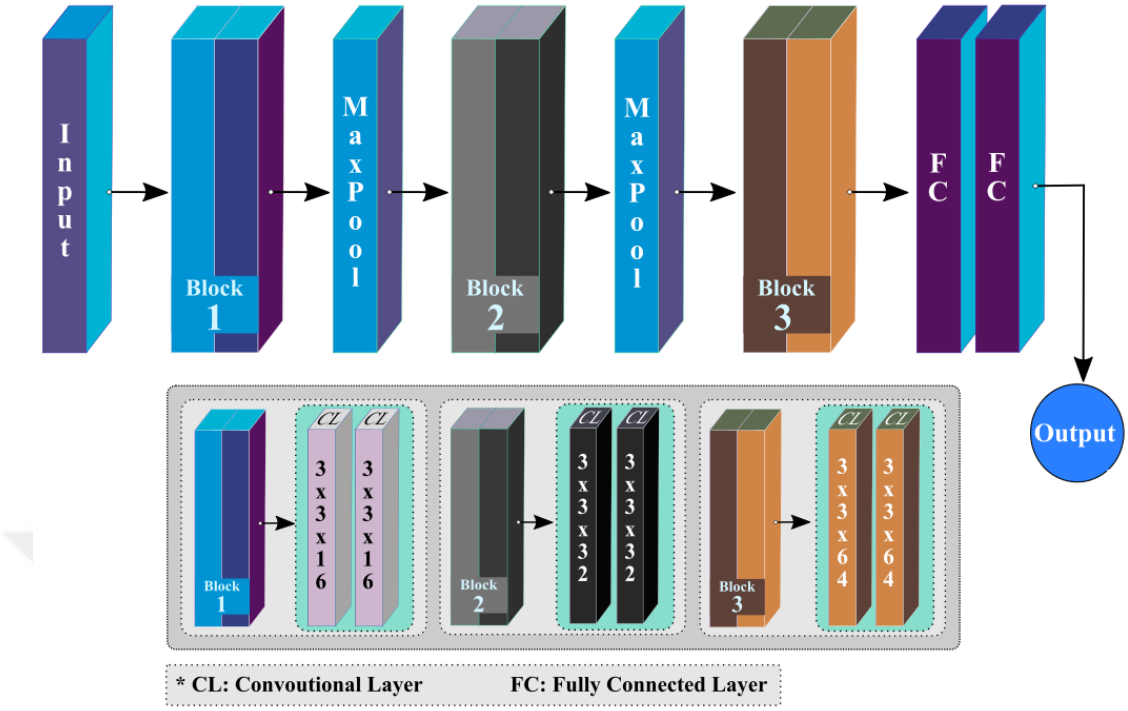
Figure 4.12: Confusion matrix of proposed Model-A



4.3.1.2. Model-B

A different combinational structure of convolutional layers is proposed as CNN-based model, Model-B, depicted in Figure 4.13. This model encompasses several more convolutional layers as compared to Model-A, thus results in a deeper network. Beside this, it also employed a dropout techniques. This model mainly consists of three blocks of convolutional layers; each block has two consecutive convolutional layers followed by a max-pooling layer. A detailed overview of proposed Model-B is given below.

Figure 4.13: Overview of proposed Model-B architecture



Like others, the architecture starts with an input layer that has a size of 32×128×1. The depth is set to 1, because of the grayscale color space conversion, while 32×128 refers to height×width of the input image. Then comes the two consecutive convolutional layers, each has a filter of size 3×3 to scan the image for feature extraction. Moreover, the depth of each convolutional layer is set to 16, whereas in Model-A the depth of first convolutional layer is 8. In Model-B, both convolutional layers use a non-linear activation function called ReLU to squeeze the activation and output a resultant feature map. To ensure that the extracted representation retains the same size as of input map, padding is turned-on. Next is a max-pooling layer with a pool size of 2×2 and a stride of 2 for down-sampling the feature map extracted by convolutional layer. A dropout is performed by setting the value to 10%.

The resultant is then fed to a second block that consists of another two consecutive convolutional layers to extract high-level features. The kernel size of each convolutional layer in this block is same as of first block, but the number of filters or depth of layers are double, raised to 32. Again, ReLU is used as activation function for this convolutional layer and padding is turned-on to retain the size of spatial output. This layer is followed by a max-pooling layer that has a stride of 2 with pool-size of 2×2. At the end of this block, 25% dropout is performed.

The extracted feature maps of second block is then supplied to last block. This block includes two convolutional layers and a max-pooling layer. Each convolutional layer has a kernel of size 3×3, and depth is kept to 64. ReLU is used as an activation function in each layer for dimensionality reduction while padding is set. Followed by a max-pooling layer of 2×2 pool-size with a stride of 2. Later is 40% dropout is performed before passing it to flattening layer.

The model ends with dense layers that classifies the input images by gathering all the representation extracted by preceding layers. Being a binary-classification problem, a sigmoid is used as activation in last layer to normalize the output that sums up to one. The network computes the losses based on the probabilities, and accordingly distinguish the image as text or non-text.

The network uses Adam as optimizer with an initial learning rate of 0.001. The network is trained over 25 epochs with a batch size of 10. Table 4.5 enlists the network topology of Model-B.

Table 4.5: Network topology of proposed Model-B

CNN Proposed Topology							
Dataset	**Convolution Layers**				**Fully Connected Layers**		
	Kernel Size	ReLU	Max-Pooling	Dropout	No. of neurons	Function	Dropout
Turkish Text	3×3×16	Yes	No	No			
	3×3×16	Yes	2×2	Yes (10%)			
	3×3×32	Yes	No	No			
	3×3×32	Yes	2×2	Yes (25%)			
	3×3×64	Yes	No	No			
	3×3×64	Yes	2×2	Yes (40%)	128	ReLU	No
					2	Sigmoid	No

Other Parameters			
Batch size	Epochs	Learning rate	Optimizer
10	25	0.001	Adam

This model performed better than previous proposed model (Model-A), as it achieved an overall accuracy of 99.74% on test set, and training accuracy remained nearly same. Furthermore, it secured an overall precision, recall and F1-score of 99.73%, 99.75% and 99.74%, respectively. Table 4.6 outlines the performance analysis of this model for both textual-classes and non-textual-class. The training accuracy and loss curves are depicted in Figure 4.14, whereas Figures 4.15 shows the loss and accuracy curves for testing set.

Table 4.6: Performance evaluation of proposed Model-B on test set

Test Samples	***Performance Analysis***			
	Precision	***Recall***	***F-Score***	***Accuracy***
Text	99.92	99.58	99.75	-
Non-text	99.53	99.91	99.72	-
Overall	**99.73**	**99.75**	**99.74**	**99.74**

Figure 4.14: Training loss and accuracy curves of proposed Model-B

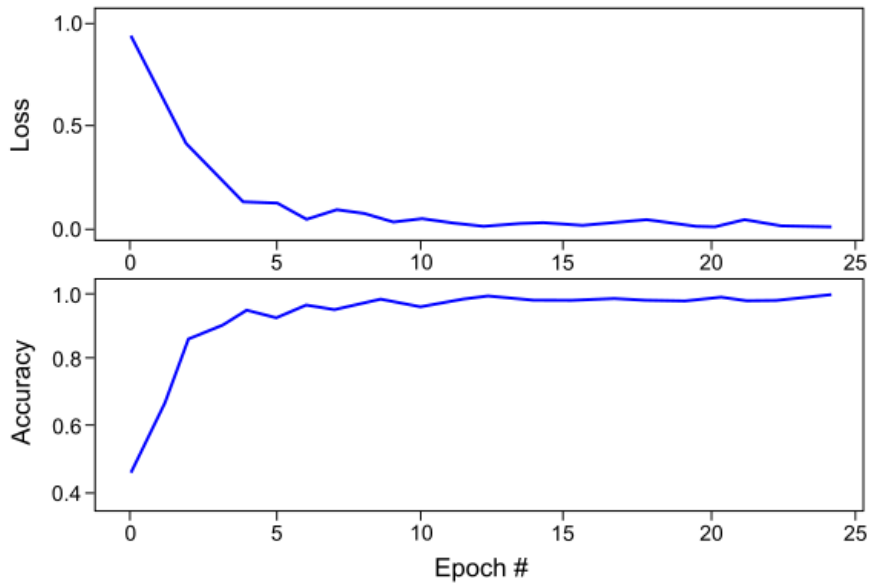
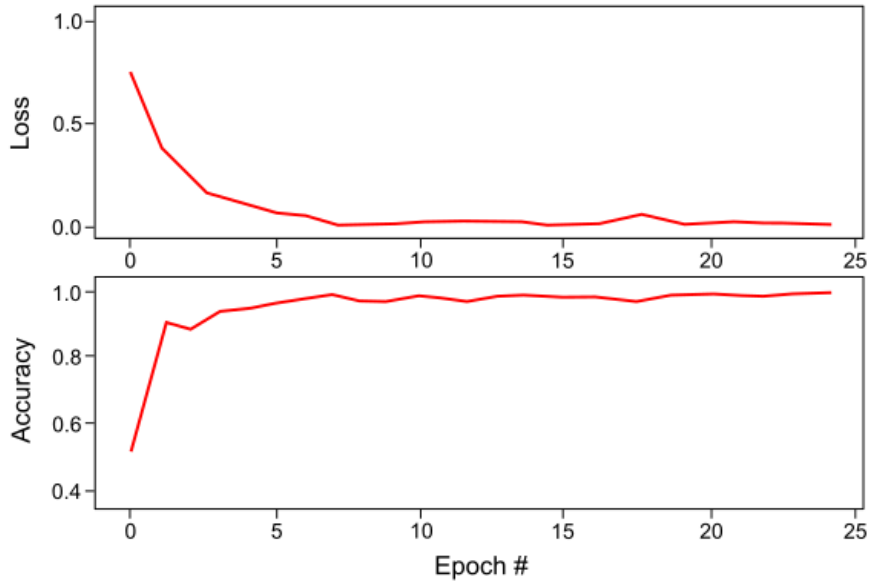
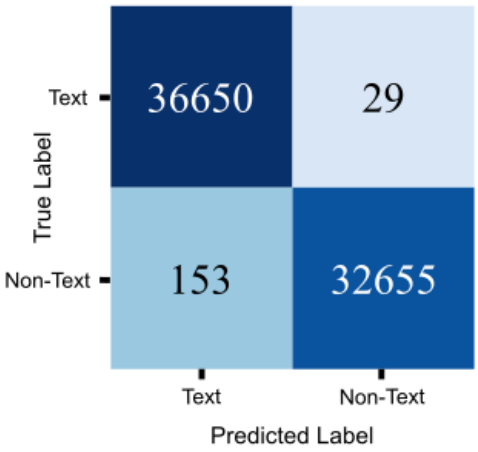


Figure 4.15: Testing loss and accuracy curves of proposed Model-B



The model performed well and generalizes better as it misclassified very little number of text/non-text samples present in test set. Furthermore, it also attained better accuracy/loss values for train set. It just misclassified 29 textual images as non-textual ones; while 153 non-textual images are identified as textual images (see Figure 4.16). The class-wise performance analysis is; for text-class, the model secured a precision, recall and F1-score of 99.92%, 99.58%, and 99.75%, respectively. On other hand, for non-textual class it accomplished a precision, recall and F1-score of 99.53%, 99.75% and 99.74%, respectively.

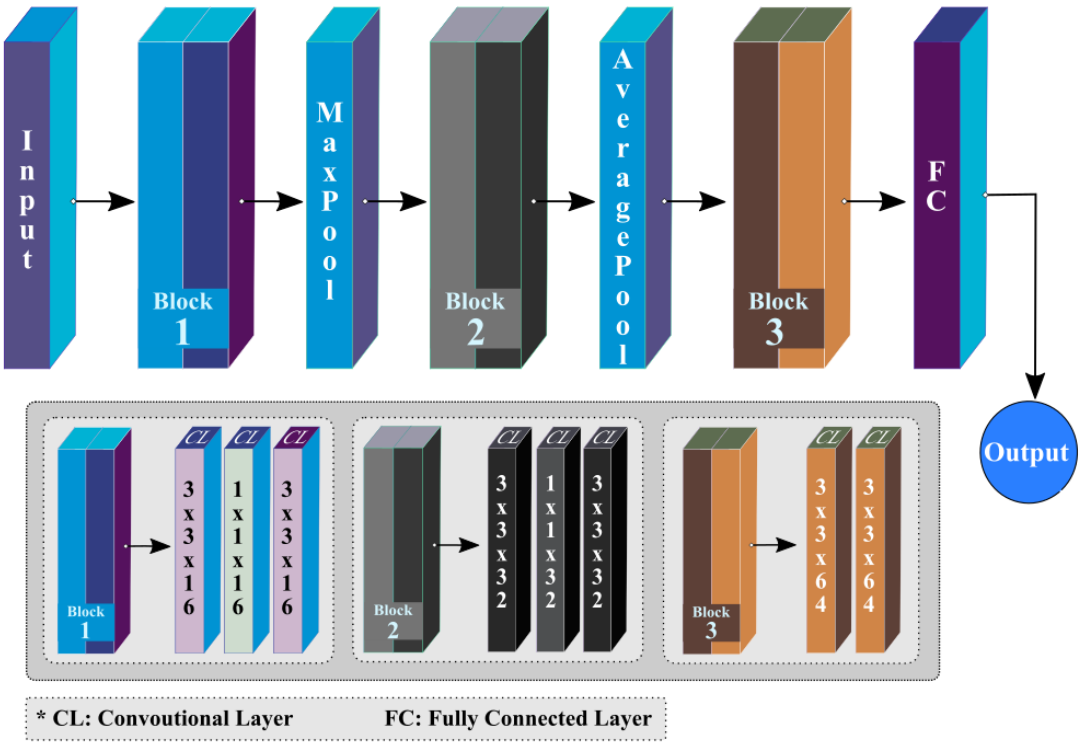
Figure 4.16: Confusion matrix of proposed Model-B



4.3.1.3. Model-C

A third model named as Model-C is implemented with some more variation in Model-B, as depicted in Figure 4.17. This model is similar to Model-B in number of blocks but has different number of convolutional layers. It has three blocks; first two blocks consist of three convolutional layers, whereas third block consists of two convolutional layers. The main difference between Model-B and Model-C is the introduction of 1×1 convolutional layers for non-linear transformation, while having three convolutional layers in first two blocks of Model-C. Thus, it is a more deeper model than previous two models (Model-A and Model-B) Moreover, an average-pooling is performed after second block. A detailed overview of proposed Model-C is described below.

Figure 4.17: Overview of CNN-based proposed Model-C architecture



Like others, the architecture starts with an input layer that has a size of 32×128×1. The depth is set to 1, because of the grayscale color space conversion, while 32×128 refers to height ×width of the input image. Next are the three blocks, each containing different number of convolutional block. The first and second block is followed by pooling layers. The first block consists of three consecutive convolutional layers. The

first and last layers have filters of size 3×3 to scan the image for feature extraction, while a middle layer (second layer) in this block is a convolutional layer of size 1×1 to introduce non-linear transformation. All the convolutional layers uses a non-linear activation function called ReLU to squeeze the activation and output a resultant feature map, while the depth of each layer is set to 16. To ensure that the extracted representation retains the same size as of input map, padding is turned-on. The block is followed by a max-pooling layer with a pool size of 2×2 and a stride of 2 for down-sampling the feature map extracted by convolutional layer. A dropout is performed by setting the value to 10%.

The resultant is of max-pooling layer is fed to a second block that consists of another three consecutive convolutional layers to extract high-level features. The kernel size of first and last convolutional layers in this block is again 3×3, whereas the middle layer is a 1×1 convolutional layer. The number of filters or depth of layers are double than the first block, that is 32. Again, ReLU is used as activation function for this convolutional layer and padding is turned-on to retain the size of spatial output. This layer is followed by an average-pooling layer that has a stride of 2 with pool-size of 2×2. At the end of this block, 15% dropout is performed. A detail network topology of proposed model is listed in Table 4.7.

The extracted feature maps of second block is then supplied to last block. This block includes two convolutional layers and is not followed by any of the pooling layer. Each convolutional layer has a kernel of size 3×3, and depth is kept to 64. ReLU is used as an activation function in each layer for dimensionality reduction while padding is set. Later a 25% dropout is performed before passing it to fully connected layer.

The model ends with fully connected layer that classifies the input images by gathering all the representation extracted by preceding layers. Being a binary-classification problem, a sigmoid is used as activation in last layer to normalize the output that sums up to one. The network computes the losses based on the probabilities, and accordingly distinguish the image as text or non-text. The network uses Adam as optimizer with an initial learning rate of 0.001. The network is trained over 25 epochs with a batch size of 20.

Compared to proposed Model-A and Model-B, The performance of this model (Model-C) degraded as it achieved an overall accuracy of 99.08% on test set, while it

secured an overall precision, recall and F1-Score of 99.05%, 99.11% and 99.08%, respectively. Table 4.8 outlines the performance analysis of this model for textual-class as well as for non-textual-class. The training accuracy and loss curves are depicted in Figure 4.18, whereas Figures 4.19 shows the loss and accuracy curves for testing set.

Table 4.7: Network topology of proposed Model-C

CNN Proposed Topology							
Dataset	**Convolution Layers**				**Fully Connected Layers**		
	Kernel Size	ReLU	Pooling	Dropout	No. of neurons	Function	Dropout
Turkish Text	3×3×16	Yes	No	No			
	1×1×16	Yes	No	No			
	3×3×16	Yes	Max-pooling (2×2)	Yes (10%)			
	3×3×32	Yes	No	No			
	1×1×32	Yes	No	No			
	3×3×32	Yes	Average pooling (2×2)	Yes (15%)			
	3×3×64	Yes	No	No			
	3×3×64	Yes	No	Yes (25%)	128	ReLU	No
					2	Sigmoid	No
Other Parameters							
Batch size	Epochs	Learning rate	Optimizer				
20	25	0.001	Adam				

Table 4.8: Performance evaluation of proposed Model-C on test set

Test Samples	***Performance Analysis (%)***			
	Precision	***Recall***	***F1-Score***	***Accuracy***
Text	99.57	98.70	99.13	-
Non-text	98.53	99.51	99.02	-
Overall	**99.05**	**99.10**	**99.08**	**99.08**

Figure 4.18: Training loss and accuracy curves of proposed Model-C

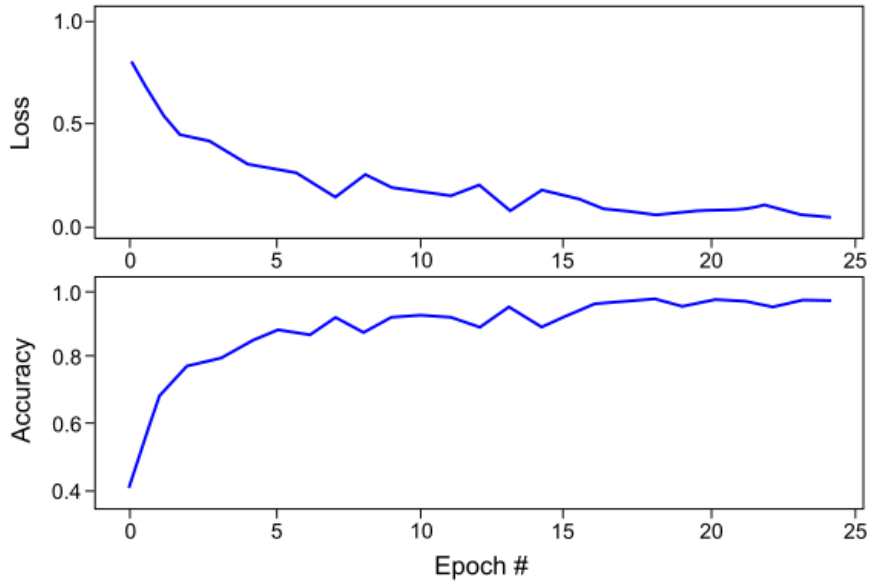
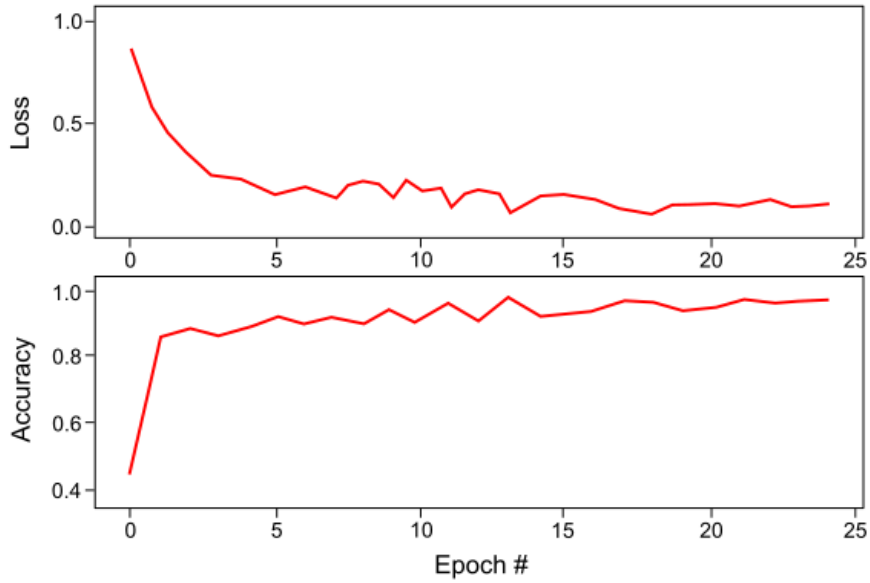


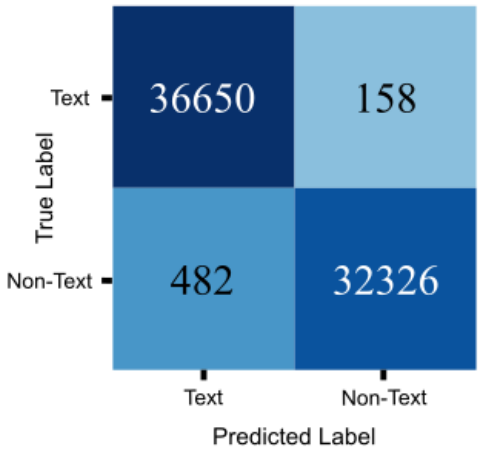
Figure 4.19: Testing loss and accuracy curves of proposed Model-C



For textual-class, network achieved a precision of 99.57%, recall of 98.70%, and F1-score of 99.13%, whereas for non-textual class it secured a precision of 98.53%, recall of 99.51% and F1-score of 99.02%. The network showed less predictability than other proposed models, as 158 textual images and 482 non-textual images were misclassified as non-textual and textual images, respectively, as depicted in confusion matrix shown in Figure 4.16. The study performed extensive experiments with different parameters

for more accurate classification, but the parameters shown in Table 4.7 performs better than any other combination.

Figure 4.20: Confusion matrix of proposed Model-C



4.3.2. Other Deep Learning-based Frameworks

Being the subset of machine learning field, deep learning approaches are considered as machine learning on steroids. Deep learning research community has established several CNN-based advanced models that can understand the representational structure of data and fit it into models so that consumers and experts can utilizes these models later for any related problem at hand. Therefore, this study also investigated the performance of well-established three state-of-the-art CNN-based deep learning frameworks on generated dataset. To compare the proposed models with popular frameworks, VGG16, ResNet50, and DenseNet121 are implemented. These models utilizes advanced techniques, which provides systems an enhanced capacity to discover and magnify the tiny patterns for better image classification. The generated dataset is converted to the desired input size according to theses framework. The following sub-sections describe the performance of each framework.

4.3.2.1. VGG-16

Beside proposed models, VGG16 framework is the first model implemented for this for comparative analysis. The model is trained on 70% of dataset, later analyzed on

test set that achieved an overall accuracy of 98.83%. Table 4.9 presents the detailed performance evaluation metrics of VGG16.

Table 4.9: Performance evaluation of VGG16 on test set

Test Samples	*Performance Analysis*			
	Precision	***Recall***	***F-Score***	***Accuracy***
Text	99.43	98.37	98.90	-
Non-text	98.16	99.36	98.75	-
Overall	**98.79**	**98.86**	**98.82**	**98.83**

The model secured a precision of 99.43%, recall of 98.37% and F1-score of 98.90% on textual class, while it accomplished a precision of 98.16%, recall of 99.36% and F1-score of 98.75% on non-textual class. Figure 4.21 shows the training accuracy and loss curves of framework. The VGG16 achieved an overall precision, recall and F1-score of 98.79%, 98.86% and 98.82%, respectively, with an overall accuracy of 98.83% on testing set, as shown in Figure 4.22.

Figure 4.21: Training loss and accuracy curves of VGG16

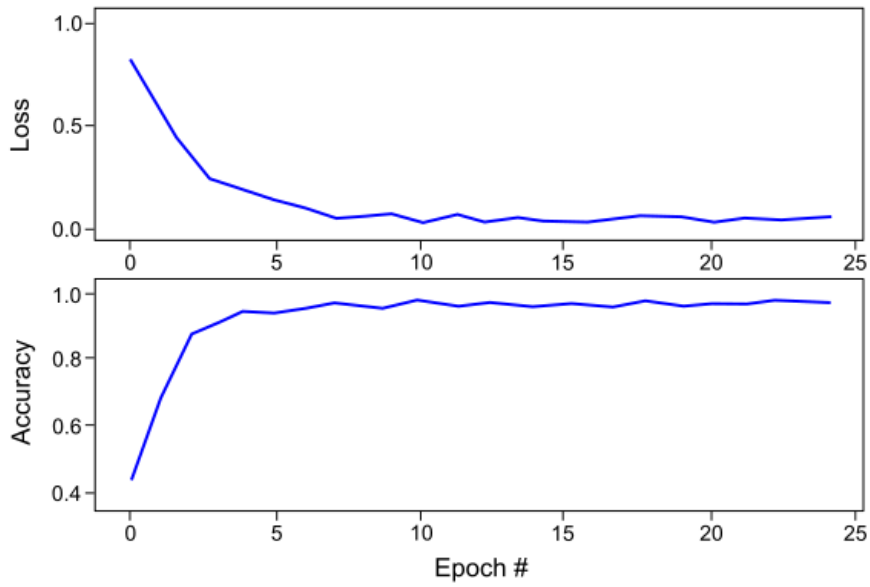
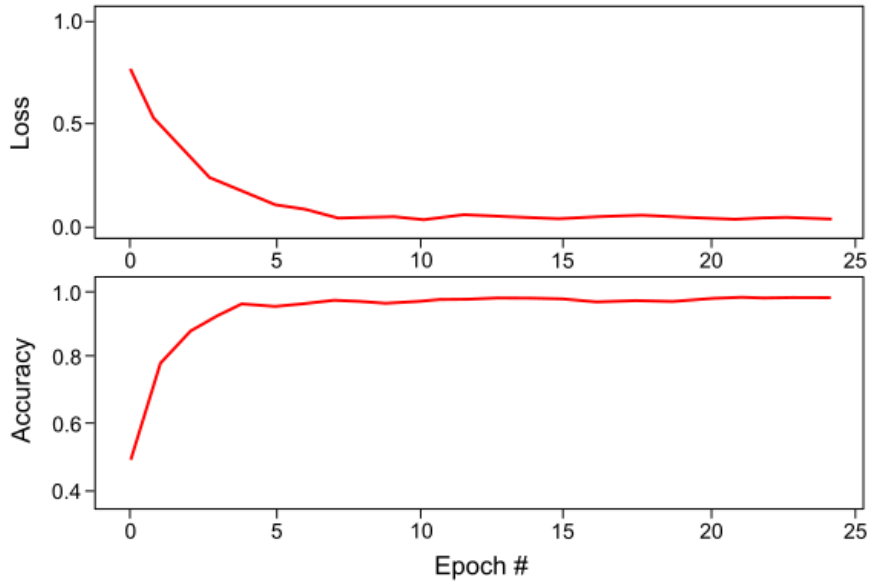
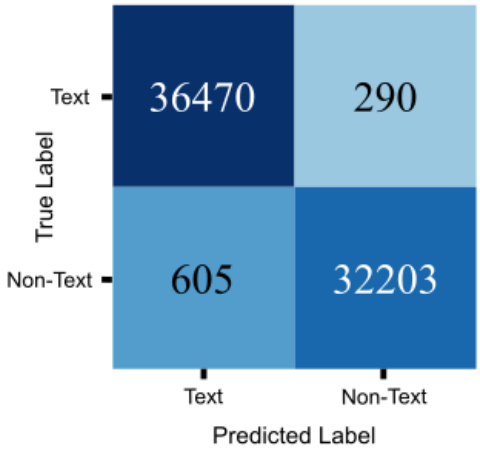


Figure 4.22: Testing loss and accuracy curves of VGG16



As shown in the confusion matrix depicted in Figure 4.23, the implemented framework wrongly identified 290 text images and 605 non-textual images as non-textual and textual images, respectively.

Figure 4.23: Confusion matrix of VGG16



4.3.2.2. ResNet50

The second pre-trained model implemented for text detection in this study is ResNet50 (for framework overview, see Chapter 3). The train set images are adjusted and fed to

ResNet50 for training, later the performance is analyzed on test set. The test set is the same set of images on which the performance of other models are investigated. It secured an overall accuracy of 98.50% as listed in Table 4.10.

Table 4.10: Performance evaluation of ResNet50 on test set

Test Samples	*Performance Analysis*			
	Precision	***Recall***	***F-Score***	***Accuracy***
Text	99.15	98.04	98.59	-
Non-text	97.78	99.04	98.41	-
Overall	**98.46**	**98.54**	**98.50**	**98.50**

ResNet50 model secured a precision of 99.15%, recall of 98.04% and F1-score of 98.59% on textual class, while it accomplished a precision of 97.78%, recall of 99.04% and F1-score of 98.41% on non-textual class. Figure 4.24 depicts the training accuracy and loss curves of framework. Furthermore, the model achieved an overall precision, recall and F1-score of 98.46%, 98.54% and 98.50%, respectively, with an overall accuracy of 98.50% on testing set, as shown in Figure 4.25.

Figure 4.24: Training loss and accuracy curves of ResNet50

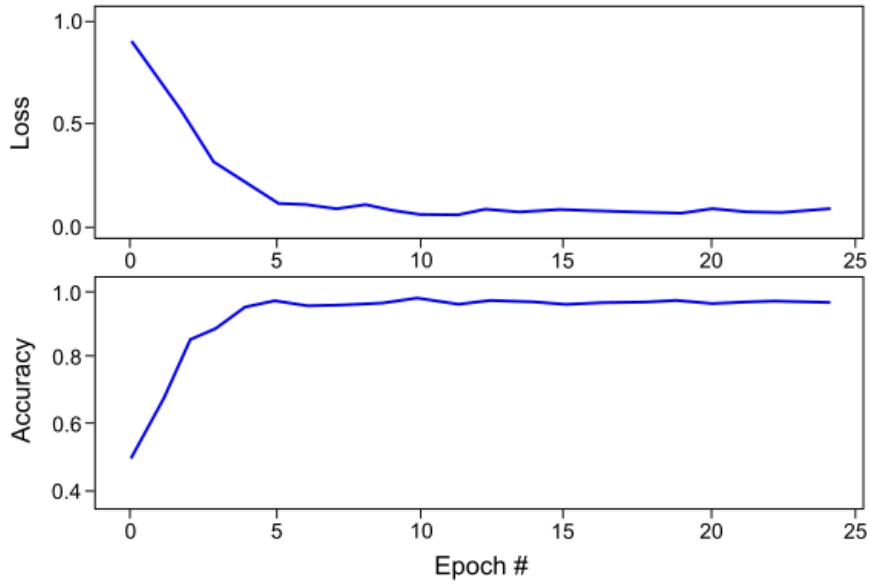
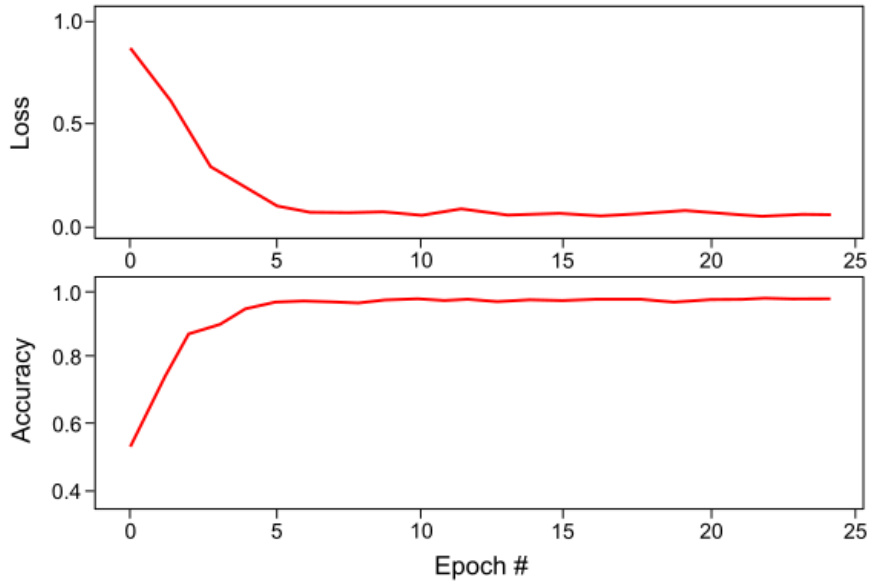


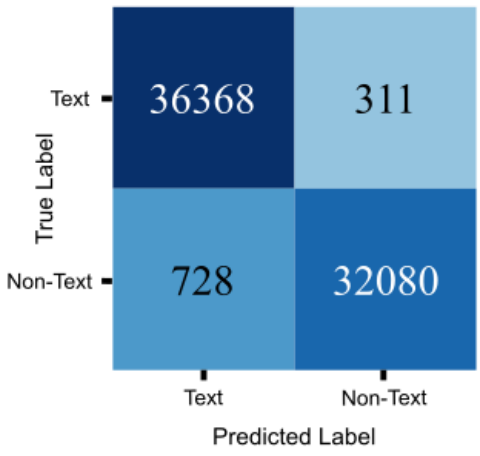
Figure 4.25: Testing loss and accuracy curves of ResNet50



.

The confusion matrix is also constructed for this model, as shown in Figure 4.26, to analyze the distribution of correct prediction in accordance with actual label. The model correctly predicted 36368 text images out of 36679 images, and 32080 non-textual images out of 32808 images.

Figure 4.26: Confusion matrix of ResNet50



4.3.2.3. DenseNet121

For this research, DenseNet121 network model is also implemented to investigate its performance for text detection. A brief overview of DenseNet121 architecture is given in Chapter 3 of this study. The model is trained with adjusted images in training set, later its performance is investigated on test set. A detailed performance evaluation is provided in in Table 4.11.

Table 4.11: Performance evaluation of DenseNet121 on test set

Test Samples	*Performance Analysis*			
	Precision	***Recall***	***F-Score***	***Accuracy***
Text	99.18	97.82	98.50	-
Non-text	97.53	99.07	98.29	-
Overall	**98.35**	**98.45**	**98.40**	**98.40**

The model performed approximately same as ResNet50 model by securing a precision of 99.18%, recall of 97.82% and F1-score of 98.50% on textual class, while it achieving a precision of 97.53%, recall of 99.07% and F1-score of 98.29% on non-textual class. Figure 4.27 depicts the training accuracy and loss curves of network. Moreover, the network achieved an overall precision, recall and F1-score of 98.35%, 98.45% and 98.40%, respectively, with an overall accuracy of 98.40% on testing set, as shown in Figure 4.28.

Figure 4.27: Training loss and accuracy curves of DenseNet121

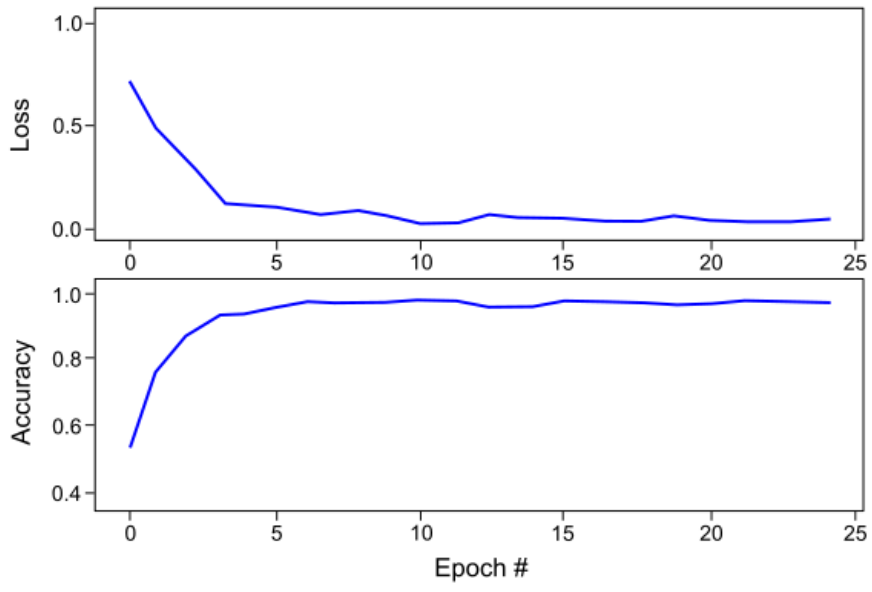
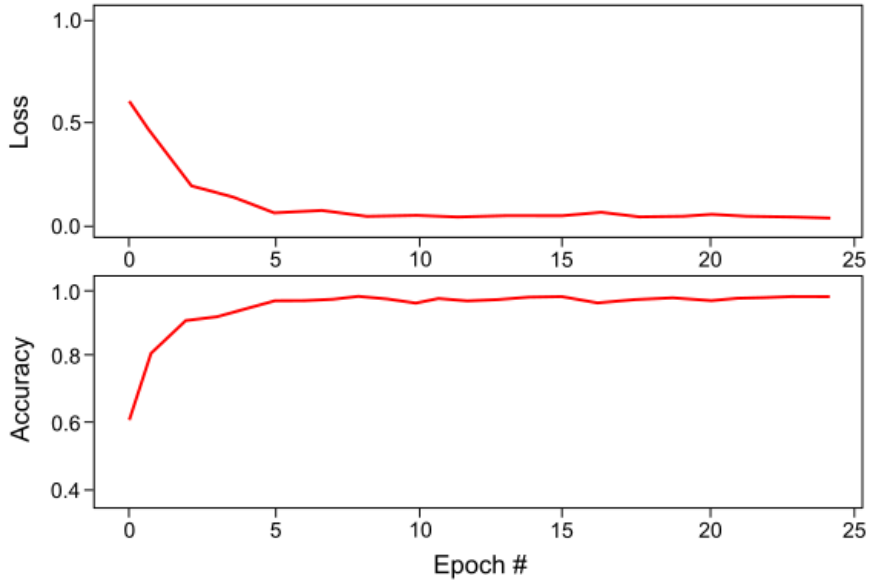
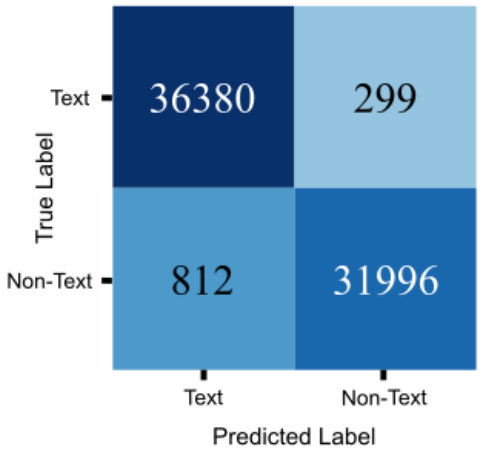


Figure 4.28: Testing loss and accuracy curves of DenseNet121



The further analysis shows that DenseNet121 network also misclassified few of the images. Out of 36679 textual images, it correctly identified 36380, whereas it misclassified the non-textual images little more as represented in confusion matrix shown in Figure 4.29.

Figure 4.29: Confusion matrix of DenseNet121



4.3.3. Other Machine Learning-based Methods

Besides deep-learning approaches, the machine-learning algorithms are also widely adopted across computer vision problems. Thu, this study not only examined the deep learning models or algorithms, but also focuses on investigating few machine-learning algorithms that are responsible for identification and classification task at hand. In this research, two machine-learning algorithms namely, SVM and LR are used to examine the image to for detecting whether it contains text or not. These algorithms practices statistical techniques to explore patterns in massive amount of generated dataset, later based on these patterns the models make appropriate predictions (see Chapter 3 for detailed overview of such methods). This section presents the performance of SVM and LR models on generated dataset for this study.

4.3.3.1. Support Vector Machine

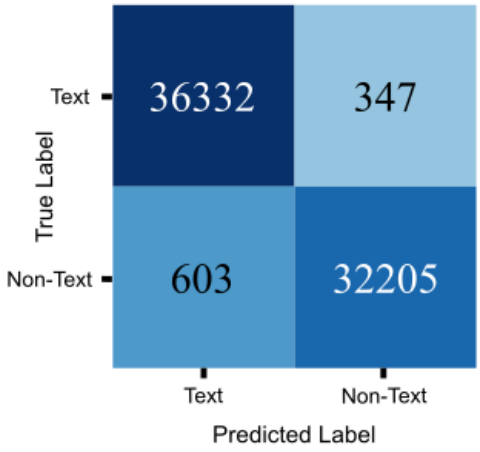
Finding a relationship between independent and dependent variables, classification parameters plays a pivotal role in image classification task. Especially for non-linear problem, the selection of kernel and other parameters are important, therefore, for this study RBF kernel is preferred because of its adoption at large scale in computer vision and deep learning task. The major reason behind the selection of RBF as kernel is its effectiveness in classification tasks. As discussed in Chapter 3, the selection of gamma and C parameters' values in RBF kernel are crucial for better generalization of network. For this study, gamma and C parameters are set to 1.0E4 and 12, respectively. Both the parameters are calculated empirically and fine-tuned. The network is first trained with above specified parameters and kernel, which is then evaluated with test set. The network showed some promising results in terms of accuracy, precision, recall and f1-score as listed in Table 4.12.

Table 4.12: Performance evaluation of support vector machine on test set

Test Samples	***Performance Analysis (%)***			
	Precision	***Recall***	***F1-Score***	***Accuracy***
Text	99.05	98.37	98.71	-
Non-text	98.16	98.93	98.55	-
Overall	**98.61**	**98.65**	**98.63**	**98.63**

Being a traditional machine-learning algorithm, SVM model classifier performed better than many deep learning frameworks for this particular problem. It secured an overall accuracy of 98.63%, precision of 98.61%, recall of 98.65%, and f1-score of 98.63%. The SVM model generalizes well with few wrong predictions. It performed better on test set, as for textual-class, network achieved a precision of 99.05%, recall of 98.37%, and f1-score of 98.71%, whereas for non-textual class it secured a precision of 98.16%, recall of 98.93% and f1-score of 98.55%. The predicted versus true value is also computed for this classifier, as depicted in Figure 4.30. The network predicted 36322 textual images and 32205 non-textual images correctly out of 36679 textual and 32808 non-textual images, respectively.

Figure 4.30: Confusion matrix of support vector machine



4.3.3.2. Logistic Regression

The LR model is the most simplest and easier network to implement for classification in machine learning task. Like other deep learning and machine learning models, the dataset is also used to train and test the LR network. In this study, the LR network is tained with a default optimizer called ‘lbfgs’. After training the model, the network is test and performance is recorded against test set. Table 4.13 outlines the performance analysis of this model for textual-class as well as for non-textual-class.

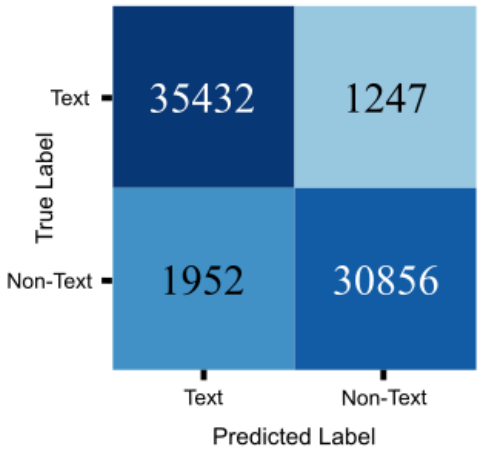
Table 4.13: Performance evaluation of logistic regression model on test set

Test Samples	*Performance Analysis (%)*			
	Precision	***Recall***	***F1-Score***	***Accuracy***
Text	99.60	94.79	95.68	-
Non-text	94.05	96.12	95.07	-
Overall	**95.33**	**95.45**	**95.38**	**95.40**

The performance of LR classifier is not promising as compared to other machine learning and deep learning models and frameworks. It achieved an overall accuracy of 95.40% on test set, while it secured a precision, recall and F1-Score of 95.33%, 95.45% and 95.38%, respectively.

The LR model failed to generalize well as it showed better performance on training set, but on testing set, it misclassified more. It accomplished an accuracy of 97.9% on training set, whereas on test set it achieved less (95.4%). Similarly, the training loss also equates to less as compared to testing, thus results in overfitting problem. For textual-class, network achieved a precision of 99.60%, recall of 94.79%, and F1-score of 95.68%, whereas for non-textual class it secured a precision of 94.05%, recall of 96.12% and F1-score of 95.07. The network showed worst predictability than other proposed models, as 1247 textual images and 1952 non-textual images were misclassified as non-textual and textual images, depicted in confusion matrix shown in Figure 4.31.

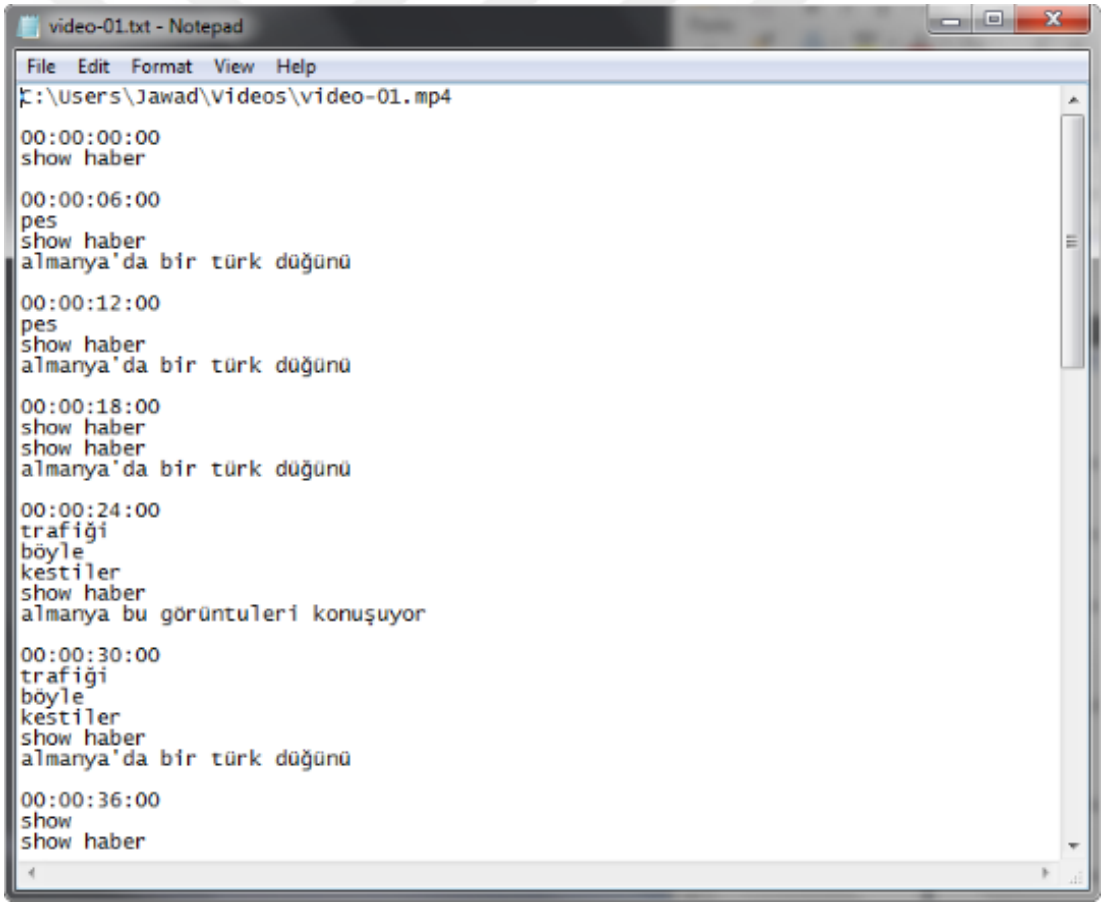
Figure 4.31: Confusion matrix of logistic regression



4.4. Text Recognition using Tesseract OCR Engine

Acquiring the exact regions of an image containing text, achieved in above described experimental work, is the primary step to have the best out of OCR experience. The textual regions obtained by proposed CNN model are later fed to Tesseract OCR engine to read the text embedded in those images. The concerned images are fed in form of PNG format and language is set to *Turkish* to accurately get the Turkish characters. Iterating through all the extracted images of each video, the Tesseract OCR engine recognizes text. The recognized Turkish text are simultaneously being stored in plain text file. A sample plain text file is shown in Figure 4.32.

Figure 4.32: Sample text file generated after text recognition operation through Tesseract optical character recognizer



The first line of each text file contains the location of video being processed. For example in Figure 4.32, the first line stating '*C:\users\Jawad\videos\video-01.mp4*' corresponds to the location of '*video-01*' stored in disk. Next is the blank line followed

by the time at which the frame (being processed) appeared in video such as ‘00:00:00:00’. For example, ‘00:00:06:00’ refers to the frame appeared at sixth second of the ‘*video-01’*. Each preceding line until next blank line corresponds to text of each individual regions identified by proposed CNN models (Model-B).

For each video, a new text file is generated and stored for later usage. These text files are processed to fill the required tables in database to properly index the video, described in later section of this chapter.

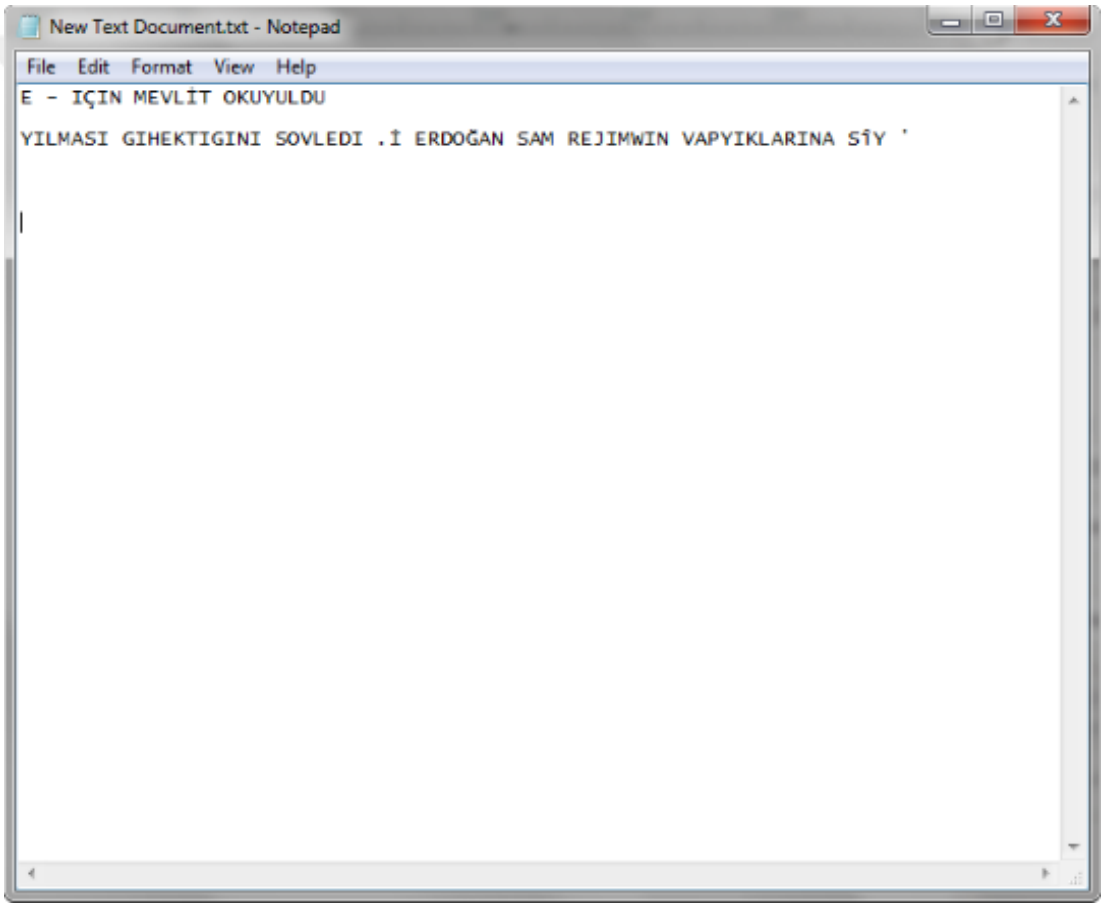
4.4.1. Why Separate Text Detection Method?

To address the issue of using a separate text detection and extraction method when Tesseract or any other OCR engine itself a text detector beside recognizer, the original full frame images are fed to OCR directly without identifying the text regions through methods proposed earlier in this study. This results in very slow processing with high classification errors. For example, an original image frame shown in Figure 4.33 having multiple textual lines is fed directly to Tesseract OCR package without passing through the proposed deep learning based detection system. The Tesseract highly misclassified and recognized the text present in this image.

Figure 4.33: Sample of original image having multiple textual lines, each line has different font and sizes

The resultant text recognized by Tesseract OCR is shown in Figure 4.34. It is evident from the figure that it did not recognized most of the textual lines and just detected the last two textual lines of the frame. Even these two lines are not recognized properly. This misclassification is due to several issues mentioned in the Chapter 3 of this study. It is mainly because of multiple font faces. For example, the text '*Oğlum şu habeleri düzgün yaz lan...*' is in italic with different font, while the rest of the text are not italic.

Figure 4.34: Partially recognized text after passing the image directly to Tesseract optical character recognizer without passing through proposed method



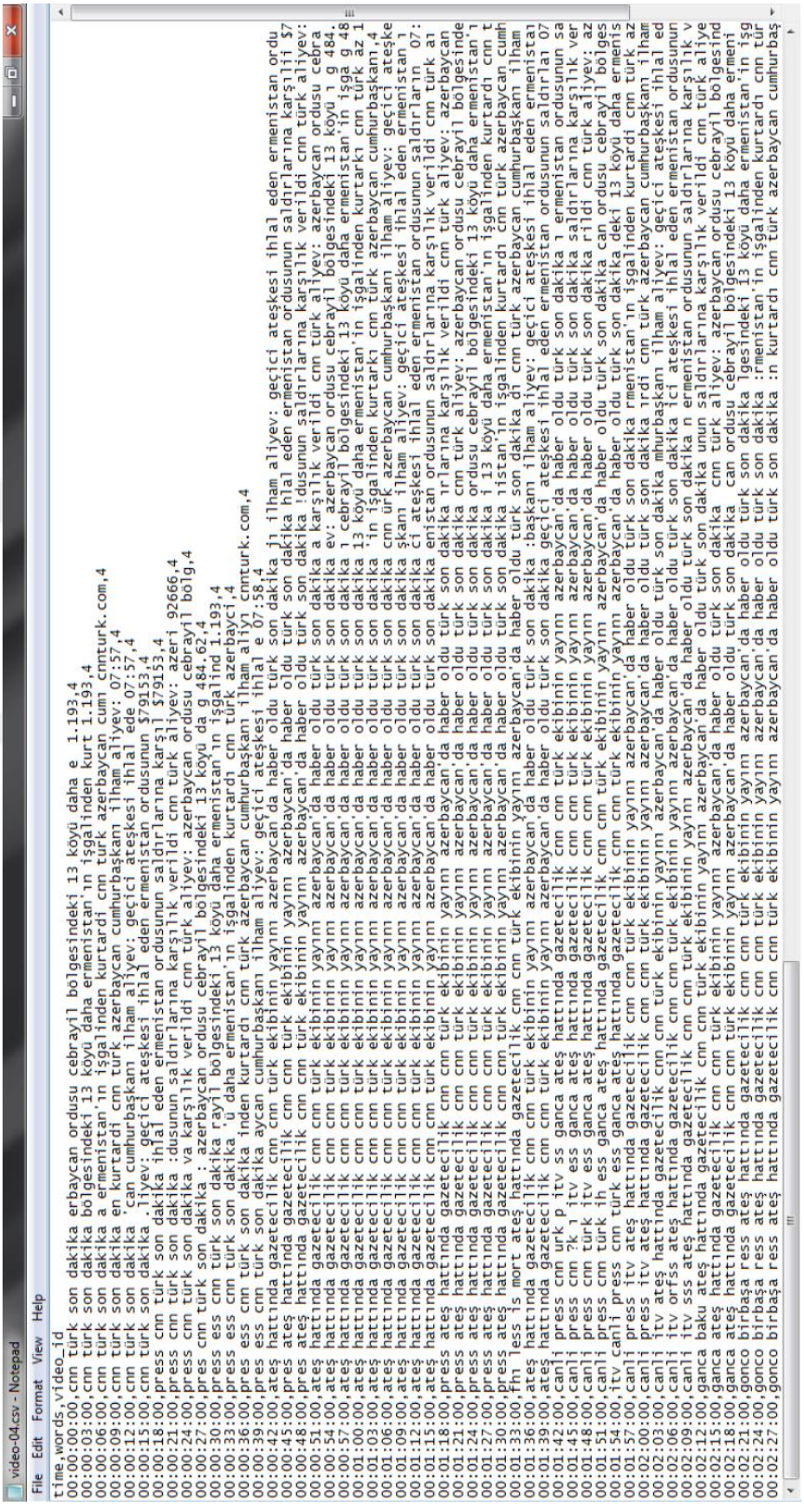
The other issue is the font size. The frame incorporates text with a huge variation of font sizes; few text-lines are small while others are large comparative to the rest. These multiple variations compromise the performance of OCR if original image is directly passed to OCR. Therefore, to smooth the process and extend the performance of OCR, a separate text localization and detection method is a necessity. Thus, identifying the

textual ROI and cropping the desired part using proposed system, significantly improves the overall performance in terms of accuracy and computational cost, as cropping the line mainly contains similar font faces and sizes.

4.5. Video Indexing and Retrieval

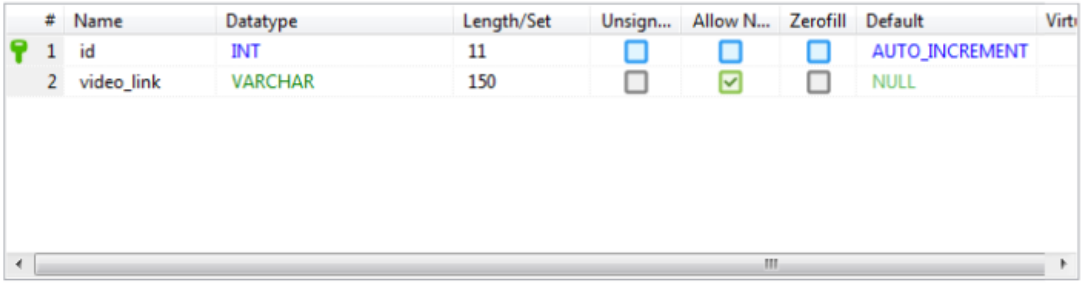
The recognized text are then saved into CSV format for easy insertion in database. A screenshot of sample CSV file is shown in Figure 4.35. Each line corresponds to a frame that depicts the occurrence time of the frame in the video, its text and the video information to which it belongs.

Figure 4.35: Sample CSV file containing recognized text and other video information



A database is created in two tables are created in the database: *video* and *word_data* tables. First table named as *video*, contains the ID of the video and its stored location on the hard disk. As represented in Figure 4.36, the *id* refers to the identification number given to the processed video, while *video_link* corresponds to its location at which it is stored in the system.

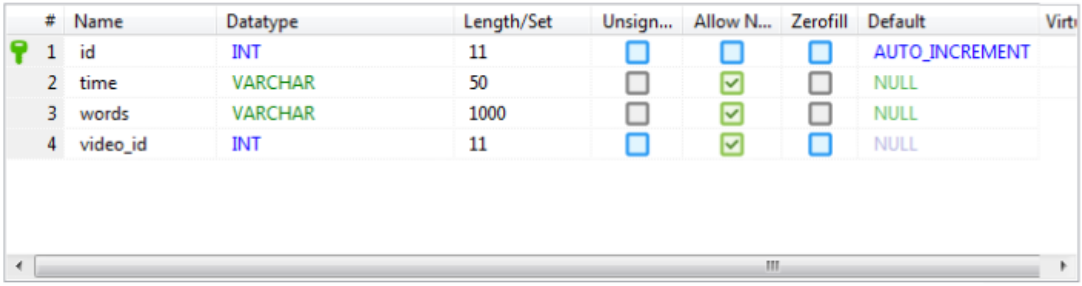
Figure 4.36: Database table 'video' for storing video links



#	Name	Datatype	Length/Set	Unsign...	Allow N...	Zerofill	Default	Virt
1	id	INT	11	☐	☐	☐	AUTO_INCREMENT	
2	video_link	VARCHAR	150	☐	☑	☐	NULL	

Another table named as *word_data* contains the extracted words, the time of its occurrence in the video, and the video ID, as shown in Figure 4.37. Later the system processes the CSV file to insert the required data into appropriate tables for indexing.

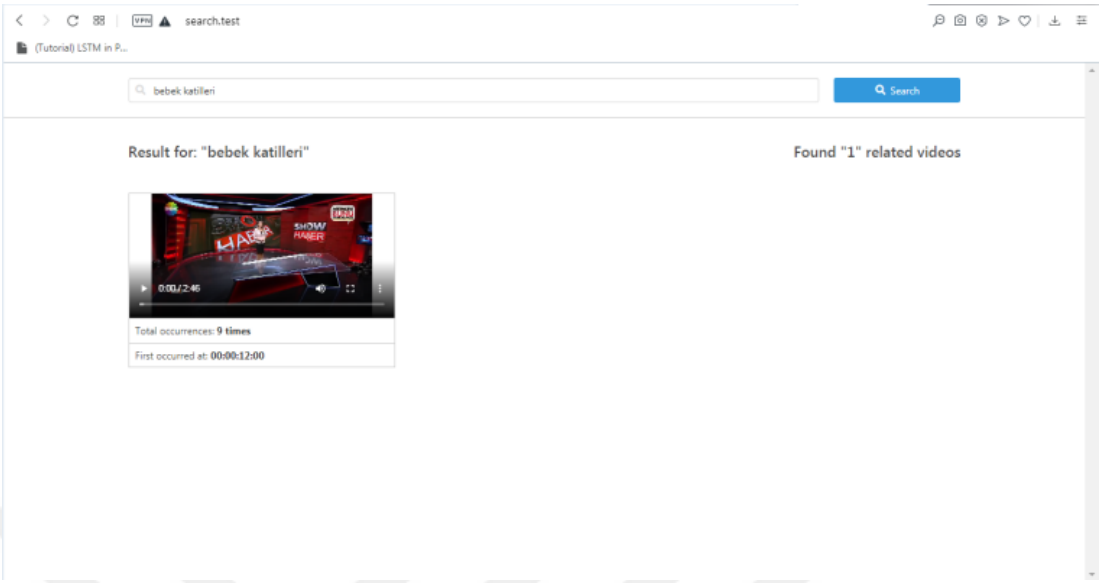
Figure 4.37: Database table 'word_data' for storing extracted words and video information



#	Name	Datatype	Length/Set	Unsign...	Allow N...	Zerofill	Default	Virt
1	id	INT	11	☐	☐	☐	AUTO_INCREMENT	
2	time	VARCHAR	50	☐	☑	☐	NULL	
3	words	VARCHAR	1000	☐	☑	☐	NULL	
4	video_id	INT	11	☐	☑	☐	NULL	

In the end, a web-based user interface is built so that a user can interact with the system for retrieving the desired videos. PHP and HTML languages are used to design and develop user interface. Figure 4.38 depicts the user interface that provides a text box where user can enter the keywords as query. The system gets the entered keywords and formulate a database query. The provided interface retrieves the information of most relevant videos from the database based on the queried words.

Figure 4.38: User interface for querrying and retrieving the relevant videos



The based on the links of most relevant videos returned by database, the proposed system then loads those relevant videos from their stored folders and display it in the corresponding thumbnail of user interface. For example; as illustrated in Figure 4.38, the user entered '*bebek katilleri*' as keywords and the user-interface returned the most relevant video which is only one video in this case. Beneath the thumbnail, it also displays how many time the word appeared in this particular video and the time of its first occurrence. The time of first occurrence is provided so that user can directly sleek the bar of the video to the point of interest while watching. If a user clicks on the video thumbnail, the system will play the video. All the basic video playing/control functionalities such as play, pause, minimize, maximize, volume controls, and sleek bar are provided for more convenience, thus user does not need any special player to watch the videos. Furthermore, download facility is also given in case a user needs to download the particular video. A unique *picture in picture* feature is also incorporated so that if a user can simultaneously utilize the screen in another task while watching the video as well, as shown in Figure 4.39.

Figure 4.39: Special feature ‘picture in picture’ for user convenience

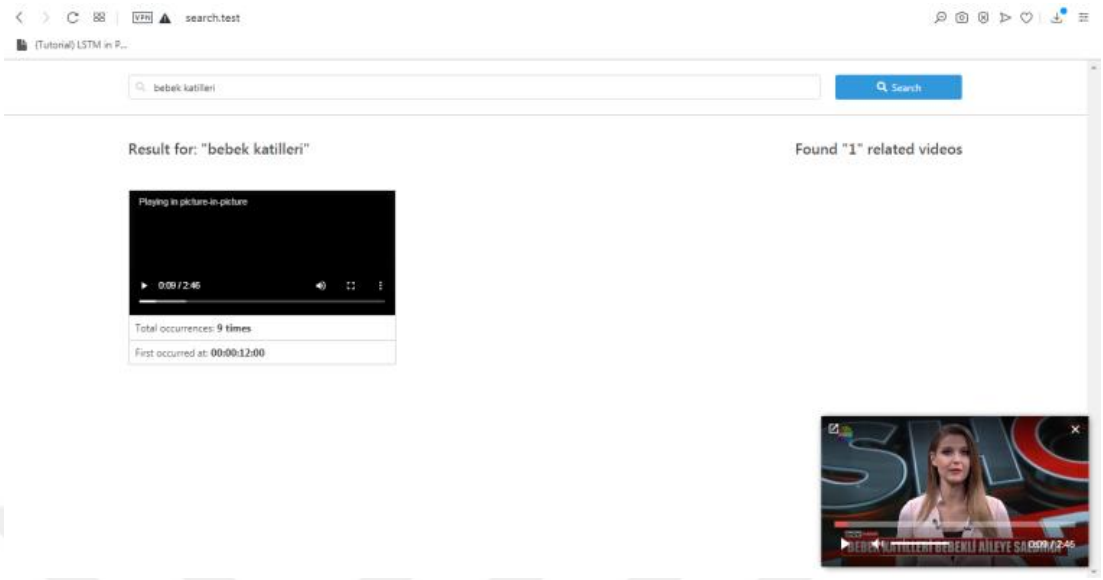
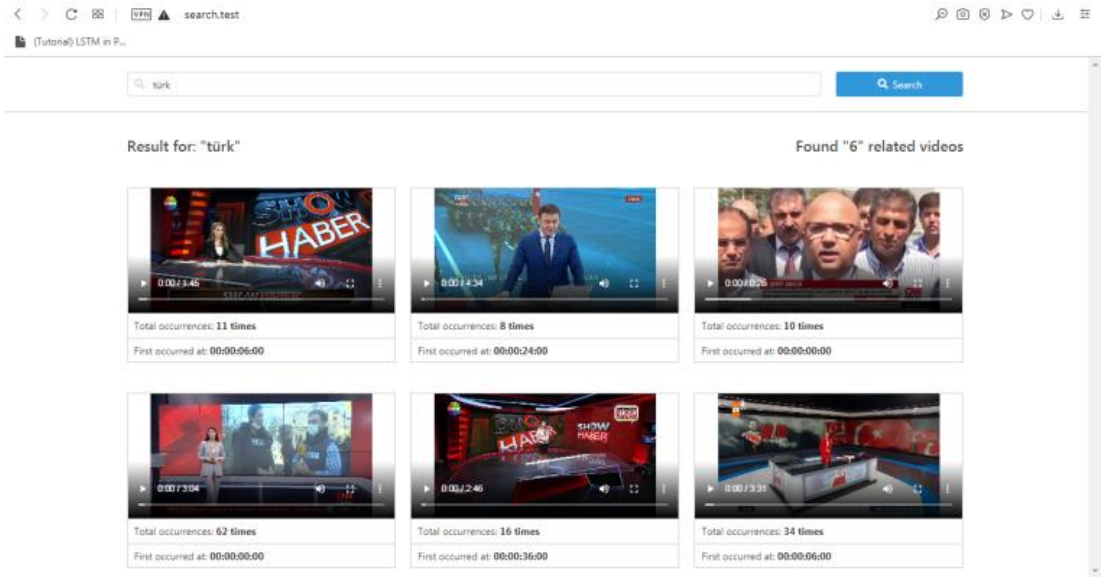


Figure 4.40 depicts the results of another query, where user entered the word ‘türk’, and six most relevant videos are returned and displayed.

Figure 4.40: Most relevant videos obtained for ‘türk’ as keyword



CHAPTER 5
CONCLUSION

5.1. Summary and Discussion

The dissertation aims to build a deep learning-based Turkish video indexing and retrieval system. The goal is to extract the Turkish text appearing inside videos, which is then used for content-based video indexing and retrieval system. As the literature review in Chapter 2 suggested, the previously published work based on advanced deep learning approaches are mostly related to languages other than Turkish. Contrarily, a very little work, mostly based on image processing techniques has been done to detect Turkish language in videos. As Turkish text detection is in fancy state, there is no dataset available publicly to train the deep learning models, whereas, deep learning models requires huge amount of data for training in order to effectively operate for desired task. Thus, this study generated its own dataset by downloading Turkish videos related to News, business, cartoon, and sports channels from various online video hosting platforms.

Some image pre-processing techniques have been employed to generate the dataset that is composed of 231622 samples out of which 122262 images contain textual content while remaining 109360 instances are non-textual images. The study proposed a deep CNN-based network to detect the textual regions in an image. Extensive experiments are carried with different combinations of convolutional and pooling layers to form the best possible network that can accurately perform the task at hand. In this regard, three different CNN-based models (Model-A, Model-B, Model-C) have been proposed and trained over a huge dataset. Later, each proposed model is evaluation on test dataset, to examine the performance.

Besides the proposed models, three different deep learning-based state-of-the-art image classification models are also implemented for comparative analysis. Furthermore, traditional machine learning algorithms such as SVM and LR are also taken into consideration. All these models and classifiers are trained on the same dataset as the proposed models are trained on. Later, identical test set is used to evaluate the classification performance of all the networks.

Table 5.1 summarizes the performance of proposed text detection models along with other deep learning frameworks and machine learning algorithms. The generated dataset is divided into train and test set with a ratio of 70-30%. Thus, 162135 samples are used to train the models while 69487 samples are utilized for testing purposes.

Table 5.1: Overall performance comparison of implemented frameworks and proposed models

Model	NoL	Pooling	Performance Metrics (%)			
			P	R	F1	Acc
Proposed Model-A	5	Yes (MP)	99.52	99.55	99.54	99.54
Proposed Model-B	**8**	**Yes (MP)**	**99.73**	**99.75**	**99.74**	**99.74**
Proposed Model-C	9	Yes (MP & AP)	99.05	99.10	99.08	99.08
VGG16	16	Yes (MP)	98.79	98.86	98.82	98.83
ResNet50	50	Yes (MP & AP)	98.46	98.54	98.50	98.50
DenseNet121	121	Yes (MP, AP, & GAP)	98.35	98.45	98.40	98.40
SVM	-	-	98.61	98.65	98.63	98.63
LR	-	-	95.33	95.45	95.38	95.40

* Acc = Accuracy, AP = Average Pooling, F1 = F1-score, GAP = Global Average Pooling, MP = Max-pooling, P = Precision, R = Recall,

It is evident from the experimental results and performances shown in Table 5.1 that Model-B outperformed other proposed and implemented models by achieving an overall accuracy of 97.74% with a precision, recall and f1-score of 99.73%, 99.75% and 99.74%, respectively. Other proposed models have slightly performed less. However, the start-of-the-art frameworks (DenseNet121, ResNet50 and VGG16) also attained promising results but misclassified little more than the proposed. Among state-of-the-art models, VGG16 stands out with more accurate predictions thus accomplished and accuracy of 98.83%. One reason for degraded performance of state-of-the-art models can be due higher number of convolutional layers. Overall, LR performed worst than all other implemented models by hardly securing an accuracy of 95.40%.

Later, the detected and extracted textual regions are passed to publicly available Tesseract OCR engine to recognize the Turkish words, which are then written in plain

text file. The text generated in text files are then inserted and stored into the database for indexing purposes along with the video file location.

In the end, an easy to use responsive web-based interface is provided, where a user can write the desired keywords, and system will retrieve the most relevant videos from the database. The retrieved videos are the ones in which the queried keywords appeared. The designed and developed webpage will show the retrieved videos in a responsive thumbnail design. Along these thumbnails, the system also displays the number of time the words appeared in that particular video and also the time of its first appearance inside the video. If a user clicks on any thumbnail, the interface will launch the command to play the selected video.

5.2. Future Work

Even though the proposed model attained good accuracy on various genres of Turkish videos, but its generality can be verified with other video genres. Generally, the change in network parameters affects its performance including recall and precision. Thus, the parameters are refined and optimized empirically and iteratively to attain better outcome, therefore, it needs further investigation to evaluate selection of parameters for inclusion and testation of new genres.

To date, the proposed model is good at video indexing and retrieval based on textual content appearing inside videos, but it can be enhanced to automate the semantic understanding by incorporating natural language processing techniques. This will enhance the retrieval process, as system can utilize the features, which are beyond the queried words, such as synonyms, so get the most relevant features. Moreover, NLP techniques like NER can be exploited to get more meaningful understanding to match the annotation and queried keywords. The addition of dictionary that contains particular semantic concepts would be helpful in optimizing the sharing of common entities.

Although, the developed web-interface is responsive, which can fit in smaller screen of smart devices, a more effective web-interface can be built to sort the retrieved videos by ascending or descending order or by time intervals.

REFERNCES

Ahmed, S. Bin, Naz, S., Razzak, M. I., & Yousaf, R. (2017). Deep learning based isolated Arabic scene character recognition. *2017 1st International Workshop on Arabic Script Analysis and Recognition (ASAR)*, 46–51. IEEE. https://doi.org/10.1109/ASAR.2017.8067758

Ansari, G. J., Shah, J. H., Yasmin, M., Sharif, M., & Fernandes, S. L. (2018). A novel machine learning approach for scene text extraction. *Future Generation Computer Systems*, *87*, 328–340. https://doi.org/10.1016/j.future.2018.04.074

Bhunia, A. K., Roy, P. P., Mohta, A., & Pal, U. (2018). Cross-language framework for word recognition and spotting of Indic scripts. *Pattern Recognition*, *79*, 12–31. https://doi.org/10.1016/j.patcog.2018.01.034

Bouaziz, B., Amara, J., & Mahdi, W. (2017). Content Video Browsing Based on Text Regions Extraction and Classification Using Convolutional Neural Network. *2017 IEEE/ACS 14th International Conference on Computer Systems and Applications (AICCSA)*, 730–737. IEEE. https://doi.org/10.1109/AICCSA.2017.195

Cao, M., Zou, Y., Yang, D., & Liu, C. (2019). GISCA: Gradient-Inductive Segmentation Network with Contextual Attention for Scene Text Detection. *IEEE Access*, *7*, 62805–62816. https://doi.org/10.1109/ACCESS.2019.2915513

Chaithanya, C. P., Manohar, N., & Bazil Issac, A. (2019). Automatic text detection and classification in natural images. *International Journal of Recent Technology and Engineering*, *7*(5), 176–180.

Cheng, P., Cai, Y., & Wang, W. (2019). A Direct Regression Scene Text Detector with Position-Sensitive Segmentation. *IEEE Transactions on Circuits and Systems for Video Technology*, *8215*(c), 1–1. https://doi.org/10.1109/tcsvt.2019.2947475

Chollet, F. (2018). *Deep Learning with Python*. Manning Publications Co.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning*, *20*(3),

273–297. https://doi.org/10.1007/BF00994018

Dikici, E., & Saraclar, M. (2008). Sliding text recognition in broadcast news. *2008 IEEE 16th Signal Processing, Communication and Applications Conference*, 1–4. IEEE. https://doi.org/10.1109/SIU.2008.4632711

Fisher, R., Perkins, S., Walker, A., & Wolfart, E. (1997). *Hypermedia image processing reference*. John Wiley and Sons Ltd. https://doi.org/0471962430

Francis, L. M., & Sreenath, N. (2019a). Live detection of text in the natural environment using Convolutional Neural Network. *Future Generation Computer Systems*, *98*, 444–455. https://doi.org/10.1016/j.future.2019.03.054

Francis, L. M., & Sreenath, N. (2019b). Robust scene text recognition: Using manifold regularized Twin-Support Vector Machine. *Journal of King Saud University - Computer and Information Sciences*, (xxxx). https://doi.org/10.1016/j.jksuci.2019.01.013

Gao, X., Han, S., & Luo, C. (2019). A Detection and Verification Model Based on SSD and Encoder-Decoder Network for Scene Text Detection. *IEEE Access*, *7*, 71299–71310. https://doi.org/10.1109/ACCESS.2019.2919994

Gao, Y., Chen, Y., Wang, J., Tang, M., & Lu, H. (2019). Reading scene text with fully convolutional sequence modeling. *Neurocomputing*, *339*, 161–170. https://doi.org/10.1016/j.neucom.2019.01.094

González, Á., & Bergasa, L. M. (2013). A text reading algorithm for natural images. *Image and Vision Computing*, *31*(3), 255–274. https://doi.org/10.1016/j.imavis.2013.01.003

Gonzalez, A., Bergasa, L. M., & Yebes, J. J. (2014). Text detection and recognition on traffic panels from street-level imagery using visual appearance. *IEEE Transactions on Intelligent Transportation Systems*, *15*(1), 228–238. https://doi.org/10.1109/TITS.2013.2277662

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep Residual Learning for Image Recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778. IEEE. https://doi.org/10.1109/CVPR.2016.90

He, W., Zhang, X.-Y., Yin, F., & Liu, C.-L. (2017). Deep Direct Regression for Multi-oriented Scene Text Detection. *2017 IEEE International Conference on*

Computer Vision (ICCV), 745–753. IEEE. https://doi.org/10.1109/ICCV.2017.87

He, W., Zhang, X. Y., Yin, F., Luo, Z., Ogier, J. M., & Liu, C. L. (2020). Realtime multi-scale scene text detection with scale-based region proposal network. *Pattern Recognition*, *98*, 107026. https://doi.org/10.1016/j.patcog.2019.107026

He, W., Zhang, X., Yin, F., & Liu, C. (2018). Multi-Oriented and Multi-Lingual Scene Text Detection With Direct Regression. *IEEE Transactions on Image Processing*, *27*(11), 5406–5419. https://doi.org/10.1109/TIP.2018.2855399

Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. (2017). Densely Connected Convolutional Networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2261–2269. IEEE. https://doi.org/10.1109/CVPR.2017.243

Huang, Y., Sun, Z., Jin, L., & Luo, C. (2020). EPAN: Effective parts attention network for scene text recognition. *Neurocomputing*, *376*, 202–213. https://doi.org/10.1016/j.neucom.2019.10.010

Iwata, S., Ohyama, W., Wakabayashi, T., & Kimura, F. (2016). Recognition and transition frame detection of Arabic news captions for video retrieval. *2016 23rd International Conference on Pattern Recognition (ICPR)*, 4005–4010. IEEE. https://doi.org/10.1109/ICPR.2016.7900260

Jamil, A., Rasheed, J., & Bayram, B. (2019). Local statistical features for multilingual artificial text detection from video images. *Internation Conference on Advance Technologies, Computer Engineering and Science (ICATCES)*, 256–260. Retrieved from http://muh.karabuk.edu.tr/bilgisayar/icatces/proceeding_book_2019.pdf

Jemni, S. K., Kessentini, Y., & Kanoun, S. (2019). Out of vocabulary word detection and recovery in Arabic handwritten text recognition. *Pattern Recognition*, *93*, 507–520. https://doi.org/10.1016/j.patcog.2019.05.003

Jiao, L., Wu, H., Wang, H., & Bie, R. (2018). Text recovery via deep CNN-BiLSTM recognition and Bayesian inference. *IEEE Access*, *6*, 76416–76428. https://doi.org/10.1109/ACCESS.2018.2882592

Kanagarathinam, K., & Sekar, K. (2019). Text detection and recognition in raw image dataset of seven segment digital energy meter display. *Energy Reports*, *5*, 842–852. https://doi.org/10.1016/j.egyr.2019.07.004

Kessentini, Y., Besbes, M. D., Ammar, S., & Chabbouh, A. (2019). A two-stage deep neural network for multi-norm license plate detection and recognition. *Expert Systems with Applications*, *136*, 159–170. https://doi.org/10.1016/j.eswa.2019.06.036

Koo, H. Il, & Kim, D. H. (2013). Scene text detection via connected component clustering and nontext filtering. *IEEE Transactions on Image Processing*, *22*(6), 2296–2305. https://doi.org/10.1109/TIP.2013.2249082

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017). ImageNet classification with deep convolutional neural networks. *Communications of the ACM*, *60*(6), 84–90. https://doi.org/10.1145/3065386

Kuccuk, D., & Yazici, A. (2010). A text-based fully automated architecture for the semantic annotation and retrieval of Turkish news videos. *International Conference on Fuzzy Systems*, 1–8. IEEE. https://doi.org/10.1109/FUZZY.2010.5584676

Küçük, D., & Yazici, A. (2011). A Hybrid Named Entity Recognizer for Turkish with Applications to Different Text Genres. In *Computer and Information Sciences. Lecture Notes in Electrical Engineering* (pp. 113–116). https://doi.org/10.1007/978-90-481-9794-1_23

Küçük, D., & Yazıcı, A. (2009). *Named Entity Recognition Experiments on Turkish Texts*. https://doi.org/10.1007/978-3-642-04957-6_45

Küçük, D., & Yazıcı, A. (2011a). Exploiting information extraction techniques for automatic semantic video indexing with an application to Turkish news videos. *Knowledge-Based Systems*, *24*(6), 844–857. https://doi.org/10.1016/j.knosys.2011.03.006

Küçük, D., & Yazıcı, A. (2011b). Multilingual Video Indexing and Retrieval Employing an Information Extraction Tool for Turkish News Texts: A Case Study. *Flexible Query Answering Systems. FQAS 2011. Lecture Notes in Computer Science*, 128–136. https://doi.org/10.1007/978-3-642-24764-4_12

Lecun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278–2324. https://doi.org/10.1109/5.726791

Lei, M., Zhou, Y., Zhou, L., Zheng, J., Li, M., & Zou, L. (2019). Noise-robust wagon text extraction based on defect-restore generative adversarial network. *IEEE Access*, *7*, 168236–168246. https://doi.org/10.1109/ACCESS.2019.2954475

Li, H., Wang, P., & Shen, C. (2019). Toward End-to-End Car License Plate Detection and Recognition with Deep Neural Networks. *IEEE Transactions on Intelligent Transportation Systems*, *20*(3), 1126–1136. https://doi.org/10.1109/TITS.2018.2847291

Liao, M., Shi, B., & Bai, X. (2018). TextBoxes++: A Single-Shot Oriented Scene Text Detector. *IEEE Transactions on Image Processing*, *27*(8), 3676–3690. https://doi.org/10.1109/TIP.2018.2825107

Liu, F., Chen, C., Gu, D., & Zheng, J. (2019). FTPN: Scene text detection with feature pyramid based text proposal network. *IEEE Access*, *7*, 44219–44228. https://doi.org/10.1109/ACCESS.2019.2908933

Liu, Y., Jin, L., & Fang, C. (2020). Arbitrarily Shaped Scene Text Detection With a Mask Tightness Text Detector. *IEEE Transactions on Image Processing*, *29*, 2918–2930. https://doi.org/10.1109/TIP.2019.2954218

Liu, Y., Jin, L., Zhang, S., Luo, C., & Zhang, S. (2019). Curved scene text detection via transverse and longitudinal sequence connection. *Pattern Recognition*, *90*, 337–345. https://doi.org/10.1016/j.patcog.2019.02.002

Lu, W., Sun, H., Chu, J., Huang, X., & Yu, J. (2018). A Novel Approach for Video Text Detection and Recognition Based on a Corner Response Feature Map and Transferred Deep Convolutional Neural Network. *IEEE Access*, *6*, 40198–40211. https://doi.org/10.1109/ACCESS.2018.2851942

Luo, C., Jin, L., & Sun, Z. (2019). MORAN: A Multi-Object Rectified Attention Network for scene text recognition. *Pattern Recognition*, *90*, 109–118. https://doi.org/10.1016/j.patcog.2019.01.020

Mosleh, A., Bouguila, N., & Hamza, A. Ben. (2013). Automatic inpainting scheme for video text detection and removal. *IEEE Transactions on Image Processing*,

22(11), 4460–4472. https://doi.org/10.1109/TIP.2013.2273672

Pan, Y. F., Hou, X., & Liu, C. L. (2011). A hybrid approach to detect and localize texts in natural scene images. *IEEE Transactions on Image Processing*, *20*(3), 800–813. https://doi.org/10.1109/TIP.2010.2070803

Park, J., Lee, G., Kim, E., Lim, J., Kim, S., Yang, H., … Hwang, S. (2010). Automatic detection and recognition of Korean text in outdoor signboard images. *Pattern Recognition Letters*, *31*(12), 1728–1739. https://doi.org/10.1016/j.patrec.2010.05.024

Pei, W. Y., Yang, C., Meng, L. Y., Hou, J. B., Tian, S., & Yin, X. C. (2018). Scene video text tracking with graph matching. *IEEE Access*, *6*, 19419–19426. https://doi.org/10.1109/ACCESS.2018.2797181

Peng, C.-Y. J., Lee, K. L., & Ingersoll, G. M. (2002). An Introduction to Logistic Regression Analysis and Reporting. *The Journal of Educational Research*, *96*(1), 3–14. https://doi.org/10.1080/00220670209598786

Phan, T. Q., Shivakumara, P., & Tan, C. L. (2009). A Laplacian Method for Video Text Detection. *2009 10th International Conference on Document Analysis and Recognition*, 66–70. IEEE. https://doi.org/10.1109/ICDAR.2009.153

Raghunandan, K. S., Shivakumara, P., Roy, S., Kumar, G. H., Pal, U., & Lu, T. (2019). Multi-script-oriented text detection and recognition in video/scene/born digital images. *IEEE Transactions on Circuits and Systems for Video Technology*, *29*(4), 1145–1162. https://doi.org/10.1109/TCSVT.2018.2817642

Rasheed, J., Dogru, H. B., & Jamil, A. (2020). Turkish Text Detection System from Videos Using Machine Learning and Deep Learning Techniques. *2020 IEEE Third International Conference on Data Stream Mining & Processing (DSMP)*, 116–120. IEEE. https://doi.org/10.1109/DSMP47368.2020.9204036

Rasheed, J., Jamil, A., Dogru, H. B., Tilki, S., & Yesiltepe, M. (2019). A Deep Learning-based Method for Turkish Text Detection from Videos. *2019 11th International Conference on Electrical and Electronics Engineering (ELECO)*, 935–939. IEEE.

https://doi.org/10.23919/ELECO47770.2019.8990633

Rasheed, J., Jamil, A., Yahyaoui, A., & Madey, A. S. A. (2020). Automatic Video Indexing and Retrieval System for Turkish Videos. *2020 28th IEEE Signal Processing and Communications Applications Conference (SIU)*, Automatic Video Indexing and Retrieval System for. Gaziantep, Turkey: IEEE.

Ren, X., Zhou, Y., He, J., Chen, K., Yang, X., & Sun, J. (2017). A Convolutional Neural Network-Based Chinese Text Detection Algorithm via Text Structure Modeling. *IEEE Transactions on Multimedia*, *19*(3), 506–518. https://doi.org/10.1109/TMM.2016.2625259

Rice, S. V, Jenkins, F. R., & Nartker, T. (1995). The Fourth Annual Test of OCR Accuracy. *Technical Report*, 95–03. Las Vegas: Information Science Research Institute, University of Nevada.

Rong, X., Yi, C., & Tian, Y. (2020). Unambiguous Scene Text Segmentation with Referring Expression Comprehension. *IEEE Transactions on Image Processing*, *29*, 591–601. https://doi.org/10.1109/TIP.2019.2930176

Rosenblatt, F. (1957). The Perceptron - A Perceiving and Recognizing Automaton. In *Report 85, Cornell Aeronautical Laboratory*. New York, NY, USA. Retrieved from https://www.import.io/wp-content/uploads/2017/06/rosenblatt-1957.pdf

Roy, S., Shivakumara, P., Jain, N., Khare, V., Dutta, A., Pal, U., & Lu, T. (2018). Rough-fuzzy based scene categorization for text detection and recognition in video. *Pattern Recognition*, *80*, 64–82. https://doi.org/10.1016/j.patcog.2018.02.014

Shi, B., Bai, X., & Yao, C. (2017). An End-to-End Trainable Neural Network for Image-Based Sequence Recognition and Its Application to Scene Text Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *39*(11), 2298–2304. https://doi.org/10.1109/TPAMI.2016.2646371

Simonyan, K., & Zisserman, A. (2015). Very Deep Convolutional Networks for Large-Scale Image Recognition. In Y. Bengio & Y. LeCun (Eds.), *3rd International Conference on Learning Representations, {ICLR 2015}, May 7-9,*. San Diego,

CA, USA,. Retrieved from http://arxiv.org/abs/1409.1556

Smith, R. (2007). An Overview of the Tesseract OCR Engine. *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007) Vol 2*, 629–633. IEEE. https://doi.org/10.1109/ICDAR.2007.4376991

Smith, R. W. (1987). *The Extraction and Recognition of Text from Multimedia Document Images*. University of Bristol.

Som, T., Can, D., & Saraclar, M. (2009). HMM-based sliding video text recognition for Turkish broadcast news. *2009 24th International Symposium on Computer and Information Sciences*, 475–479. IEEE. https://doi.org/10.1109/ISCIS.2009.5291877

Sundararajan, D. (2017). Morphological Image Processing. In *Digital Image Processing* (pp. 217–256). Singapore: Springer Singapore. https://doi.org/10.1007/978-981-10-6113-4_8

Szegedy, C., Wei Liu, Yangqing Jia, Sermanet, P., Reed, S., Anguelov, D., … Rabinovich, A. (2015). Going deeper with convolutions. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1–9. IEEE. https://doi.org/10.1109/CVPR.2015.7298594

Tang, J., Yang, Z., Wang, Y., Zheng, Q., Xu, Y., & Bai, X. (2019). SegLink++: Detecting Dense and Arbitrary-shaped Scene Text by Instance-aware Component Grouping. *Pattern Recognition*, *96*, 106954. https://doi.org/10.1016/j.patcog.2019.06.020

Tang, Y., & Wu, X. (2018). Scene text detection using superpixel-based stroke feature transform and deep learning based region classification. *IEEE Transactions on Multimedia*, *20*(9), 2276–2288. https://doi.org/10.1109/TMM.2018.2802644

Tian, S., Lu, S., & Li, C. (2017). WeText: Scene Text Detection under Weak Supervision. *Proceedings of the IEEE International Conference on Computer Vision*, *2017-Octob*, 1501–1509. https://doi.org/10.1109/ICCV.2017.166

Unar, S., Wang, X., Zhang, C., & Wang, C. (2019). Detected text-based image retrieval approach for textual images. *IET Image Processing*, *13*(3), 515–521. https://doi.org/10.1049/iet-ipr.2018.5277

Wang, X., Feng, X., & Xia, Z. (2019). Scene video text tracking based on hybrid deep text detection and layout constraint. *Neurocomputing*, *363*, 223–235. https://doi.org/10.1016/j.neucom.2019.05.101

Wei, Y., Shen, W., Zeng, D., Ye, L., & Zhang, Z. (2018). Multi-oriented text detection from natural scene images based on a CNN and pruning non-adjacent graph edges. *Signal Processing: Image Communication*, *64*(March), 89–98. https://doi.org/10.1016/j.image.2018.02.016

Wei, Y., Zhang, Z., Shen, W., Zeng, D., Fang, M., & Zhou, S. (2017). Text detection in scene images based on exhaustive segmentation. *Signal Processing: Image Communication*, *50*(June 2016), 1–8. https://doi.org/10.1016/j.image.2016.10.003

Xu, J., Shivakumara, P., Lu, T., Tan, C. L., & Uchida, S. (2016). A new method for multi-oriented graphics-scene-3D text classification in video. *Pattern Recognition*, *49*, 19–42. https://doi.org/10.1016/j.patcog.2015.07.002

Xue, W., Li, Q., & Xue, Q. (2020). Text Detection and Recognition for Images of Medical Laboratory Reports with a Deep Learning Approach. *IEEE Access*, *8*, 407–416. https://doi.org/10.1109/ACCESS.2019.2961964

Yan, C., Xie, H., Liu, S., Yin, J., Zhang, Y., & Dai, Q. (2018). Effective Uyghur Language Text Detection in Complex Background Images for Traffic Prompt Identification. *IEEE Transactions on Intelligent Transportation Systems*, *19*(1), 220–229. https://doi.org/10.1109/TITS.2017.2749977

Yang, H., Jin, L., Huang, W., Yang, Z., Lai, S., & Sun, J. (2018). Dense and tight detection of Chinese characters in historical documents: Datasets and a recognition guided detector. *IEEE Access*, *6*, 30174–30183. https://doi.org/10.1109/ACCESS.2018.2840218

Zayene, O., Seuret, M., Touj, S. M., Hennebert, J., Ingold, R., & Amara, N. E. Ben. (2016). Text Detection in Arabic News Video Based on SWT Operator and Convolutional Auto-Encoders. *2016 12th IAPR Workshop on Document Analysis Systems (DAS)*, 13–18. IEEE. https://doi.org/10.1109/DAS.2016.80

Zeiler, M. D., & Fergus, R. (2014). *Visualizing and Understanding Convolutional Networks*. https://doi.org/10.1007/978-3-319-10590-1_53

Zhang, X., Gao, X., & Tian, C. (2018). Text detection in natural scene images based on color prior guided MSER. *Neurocomputing*, *307*, 61–71. https://doi.org/10.1016/j.neucom.2018.03.070

Zhao, X., Zhao, C., Guo, H., Zhu, Y., Tang, M., & Wang, J. (2019). Elite Loss for scene text detection. *Neurocomputing*, *333*, 284–291. https://doi.org/10.1016/j.neucom.2018.12.009

Zheng, Yuchen, Iwana, B. K., & Uchida, S. (2019). Mining the displacement of max-pooling for text recognition. *Pattern Recognition*, *93*, 558–569. https://doi.org/10.1016/j.patcog.2019.05.014

Zheng, Yuqiang, Xie, Y., Qu, Y., Yang, X., Li, C., & Zhang, Y. (2020). Scale robust deep oriented-text detection network. *Pattern Recognition*, *102*(xxxx), 107180. https://doi.org/10.1016/j.patcog.2019.107180

Zhong, Z., Sun, L., & Huo, Q. (2019). Improved localization accuracy by LocNet for Faster R-CNN based text detection in natural scene images. *Pattern Recognition*, *96*. https://doi.org/10.1016/j.patcog.2019.106986

Zhou, X., Yao, C., Wen, H., Wang, Y., Zhou, S., He, W., & Liang, J. (2017). EAST: An Efficient and Accurate Scene Text Detector. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, *2017-Janua*, 2642–2651. IEEE. https://doi.org/10.1109/CVPR.2017.283

Zuo, L. Q., Sun, H. M., Mao, Q. C., Qi, R., & Jia, R. S. (2019). Natural Scene Text Recognition Based on Encoder-Decoder Framework. *IEEE Access*, *7*, 62616–62623. https://doi.org/10.1109/ACCESS.2019.2916616

. . . . . .

RELATED PUBLICATION DETAILS

Rasheed, J., Jamil, A., Hameed, A.A.: Text detection in last decade. Journal of Computational Science. Impact Factor: 2.644 (Under Review)

Rasheed, J., Jamil, A., Hameed, A.A.: Deep learning based Turkish video indexing and retrieval system. Computer Vision and Image Understanding. Impact Factor: 3.121 (Under Review)

Rasheed, J., Jamil, A., Yahyaoui, A., Madey, A.S.A.: Automatic Video Indexing and Retrieval System for Turkish Videos. In: 2020 28th IEEE Signal Processing and Communications Applications Conference (SIU). p. Automatic Video Indexing and Retrieval System for. IEEE, Gaziantep, Turkey (2020).

Rasheed, J., Jamil, A., Dogru, H.B.: Turkish Text Detection System from Videos Using Machine Learning and Deep Learning Techniques. In: 2020 IEEE Third International Conference on Data Stream Mining & Processing (DSMP). pp. 116–120. IEEE (2020). https://doi.org/10.1109/DSMP47368.2020.9204036.

Rasheed, J., Jamil, A., Dogru, H.B., Tilki, S., Yesiltepe, M.: A Deep Learning-based Method for Turkish Text Detection from Videos. In: 2019 11th International Conference on Electrical and Electronics Engineering (ELECO). pp. 935–939. IEEE (2019). https://doi.org/10.23919/ELECO47770.2019.8990633.

. . . . . .

CURRICULUM VITAE (CV)

Jawad RASHEED

EDUCATION

Ph.D.: Istanbul Sabahattin Zaim University, Computer Engineering, Istanbul, Turkey

MS: Bahçeşeshir University, Electrical and Electronics Engineering, Istanbul, Turkey

BS: National University of Computer and Emerging Sciences, Telecommunication Engineering, Islamabad, Pakistan

ACADEMIC EXPERIENCE

Research Assistant, Istanbul Sabahattin Zaim University, Department of Computer Engineering, October 2018 – January 2021

RESEARCH INTERESTS

Big Data and IoT, Deep Learning, Image and Signal Processing, Machine Learning.

PUBLICATION (SCI-E)

1. Rasheed, J., Hameed, A.A., Djeddi, C., Jamil, A., Al-Turjman, F.: A Machine Learning-based Framework for Diagnosis of COVID-19 from Chest X-ray Images. Interdisciplinary Sciences: Computational Life Sciences. (2021). https://doi.org/10.1007/s12539-020-00403-6. (Impact Factor: 1.512)

2. Rasheed, J., Jamil, A., Hameed, A.A., Aftab, U., Aftab, J., Shah, S.A., Draheim, D.: A survey on artificial intelligence approaches in supporting frontline

workers and decision makers for the COVID-19 pandemic. Chaos, Solitons & Fractals. 141, 110337 (2020). https://doi.org/10.1016/j.chaos.2020.110337. (Impact Factor: 3.764)

3. Rasheed, J., Hameed, A.A., Djeddi, C., Jamil, A.: An artificial intelligence-based framework for identification of COVID-19 disease. Expert Systems. (2021). (Accepted). (Impact Factor: 1.546)

4. Rasheed, J., Jamil, A., Hameed, A.A., Al-Turjman, F.: COVID-19 in the age of artificial intelligence: a comprehensive review. Interdisciplinary Sciences: Computational Life Sciences. (2021). (Under Review). (Impact Factor: 1.512)

5. Hameed, A.A., Jamil, A., Rasheed, J.: An improved novel self-organizing map based on reweighted zero-attracting. IEEE Transactions on Neural Networks and Learning System. (Under Review). (Impact Factor: 8.793)

OTHER PEER-REVIEWED JOURNAL

1. Hameed, A.A., Dehghanian, K., Moghimi, S., Ajlouni, N., Ajlouni, F., Zontul, M., Sonmez, F., Rasheed, J.: A comparative study on signal processing of accelerated time histories. Manchester Journal of Artificial Intelligence and Applied Sciences. 1, 9–14 (2020). ISSN: 2634-1034

2. Hameed, A.A., Ajlouni, N., Dehghanian, K., Moghimi, S., Ajlouni, F., Zontul, M., Sonmez, F., Ozyavas, A., Rasheed, J.: Sonar signals discrimination using BPVAM algorithm. Manchester Journal of Artificial Intelligence and Applied Sciences. 1, 39–42 (2020). ISSN: 2634-1034

3. Kalyoncu, S., Jamil, A., Karatas, E., Rasheed, J., Djeddi, C.: Stock Market Value Prediction using Deep Learning. Data Science and Application. 3, 10–14 (2020). ISSN: 2717-6649 http://www.jdatasci.com/index.php/jdatasci/article/view/42/23

4. Kiraç, F., Jamil, A., Hameed, A.A., Rasheed, J., Yesiltepe, M., Bayram, B.: Machine Learning Methods for Land Cover Classification from Multi-Spectral

Images. Manchester Journal of Artificial Intelligence and Applied Sciences. 2, 1–5 (2021). ISSN: 2634-1034

INTERNATIONAL CONFERENCES

1. Rasheed, J., Jamil, A., Hameed, A.A., Ilyas, M., Özyavaş, A., Ajlouni, N.: Improving Stock Prediction Accuracy Using CNN and LSTM. In: International Conference on Data Analytics for Business and Industry. pp. 1–5. IEEE, Bahrain (2020).

2. Rasheed, J., Hameed, A.A., Ajlouni, N., Jamil, A., Özyavaş, A., Orman, Z.: Application of Adaptive Back-Propagation Neural Networks for Parkinson's Disease Prediction. In: International Conference on Data Analytics for Business and Industry. pp. 1–5. IEEE, Bahrain (2020).

3. Hameed, A.A., Jamil, A., Ajlouni, N., Rasheed, J., Özyavaş, A., Orman, Z.: Classification of Epileptic Seizures using Artificial Neural Network with Adaptive Momentum. In: 2020 International Conference on Data Analytics for Business and Industry. pp. 1–4. IEEE, Bahrain (2020).

4. Rasheed, J., Jamil, A., Yahyaoui, A., Madey, A.S.A.: Automatic Video Indexing and Retrieval System for Turkish Videos. In: 2020 28th IEEE Signal Processing and Communications Applications Conference (SIU). p. Automatic Video Indexing and Retrieval System for. IEEE, Gaziantep, Turkey (2020).

5. Rashid, H., Khalaji, E., Rasheed, J., Batunlu, C.: Fault Prediction of Wind Turbine Gearbox Based on SCADA Data and Machine Learning. In: 2020 10th International Conference on Advanced Computer Information Technologies (ACIT). pp. 391–395. IEEE (2020). https://doi.org/10.1109/ACIT49673.2020.9208884.

6. Rasheed, J., Jamil, A., Dogru, H.B.: Turkish Text Detection System from Videos Using Machine Learning and Deep Learning Techniques. In: 2020 IEEE Third International Conference on Data Stream Mining & Processing (DSMP).

pp. 116–120. IEEE (2020). https://doi.org/10.1109/DSMP47368.2020.9204036.

7. Rasheed, J., Alimovski, E., Rasheed, A., Sirin, Y., Jamil, A., Yesiltepe, M.: Effects of Glow Data Augmentation on Face Recognition System based on Deep Learning. In: 2020 International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA). pp. 1–5. IEEE (2020). https://doi.org/10.1109/HORA49412.2020.9152900.

8. Kocaman, T., Yesiltepe, M., Jamil, A., Rasheed, J.: Columnists Classification from Turkey Writing in English and Turkish Language. In: Gök, M. (ed.) 3rd International Conference on Data Science and Applications (ICONDATA'20). pp. 287–292. , Istanbul, Turkey (2020). http://www.icondata.org/uploads/kcfinder/upload/files/ICONDATA20__FullTextBook.pdf

9. Kalyoncu, S., Jamil, A., Rasheed, J., Yesiltepe, M., Djeddi, C.: Machine Learning Methods for Stock Market Analysis. In: Gök, M. (ed.) 3rd International Conference on Data Science and Applications (ICONDATA'20). pp. 73–77. , Istanbul, Turkey (2020). http://www.icondata.org/uploads/kcfinder/upload/files/ICONDATA20__FullTextBook.pdf

10. Rasheed, J., Jamil, A., Dogru, H.B., Tilki, S., Yesiltepe, M.: A Deep Learning-based Method for Turkish Text Detection from Videos. In: 2019 11th International Conference on Electrical and Electronics Engineering (ELECO). pp. 935–939. IEEE (2019). https://doi.org/10.23919/ELECO47770.2019.8990633.

11. Yesiltepe, M., Jamil, A., Rasheed, J., Kurulay, M.: Hexadecimal Hash Value Hiding in a Message in Web Service against Deep Learning. In: 2019 11th International Conference on Electrical and Electronics Engineering (ELECO). pp. 693–697. IEEE (2019). https://doi.org/10.23919/ELECO47770.2019.8990576.

12. Rasheed, J., Alimovski, E., Rasheed, A.: mManagement: Wi-Fi Hotspot based Attendance Application using Android Smartphone. In: 2019 1st International Informatics and Software Engineering Conference (UBMYK). pp. 1–5. IEEE (2019). https://doi.org/10.1109/UBMYK48245.2019.8965588.

13. Yahyaoui, A., Jamil, A., Rasheed, J., Yesiltepe, M.: A Decision Support System for Diabetes Prediction Using Machine Learning and Deep Learning Techniques. In: 2019 1st International Informatics and Software Engineering Conference (UBMYK). pp. 1–4. IEEE (2019). https://doi.org/10.1109/UBMYK48245.2019.8965556.

14. Jamil, A., Rasheed, J., Bayram, B.: Local Statistical Features for Multilingual Artificial Text Detection from Video Images. In: Internatıonal Conference on Advanced Technologies, Computer Engineering and Science (ICATCES2019). , Alanya, Turkey (2019). https://muh.karabuk.edu.tr/bilgisayar/icatces/proceeding_book_2019.pdf

15. Fahim, M., Cebe, H.I., Rasheed, J., Kiani, F.: mHealth: Blood donation application using android smartphone. In: 2016 Sixth International Conference on Digital Information and Communication Technology and its Applications (DICTAP). pp. 35–38. IEEE (2016). https://doi.org/10.1109/DICTAP.2016.7543997.

……..