BOOTSTRAPPING A SPEECH RECOGNITION SYSTEM BY USING SLIDING VIDEO TEXT RECOGNITION

by
Temuçin Som
B.S., in Electrical and Electronics Engineering, Dokuz Eylül University, 2007

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Electrical and Electronics Engineering
Boğaziçi University
2009

BOOTSTRAPPING A SPEECH RECOGNITION SYSTEM BY USING SLIDING VIDEO TEXT RECOGNITION

APPROVED BY:

Assist. Prof. Murat Saraçlar ....
(Thesis Supervisor)

Assoc. Prof. Burak Acar ....

Prof. Lale Akarun ....

DATE OF APPROVAL: 13.11.2009

ACKNOWLEDGEMENTS

I would like to thank my advisor, Assist. Prof. Murat Saraçlar, for his guidance, understanding and support to this study. He was always eager to discuss and guide at the challenging moments with his great ideas. I always admire his intelligence and being hard-working. I also would like to thank Prof. Lale Akarun and Assoc. Prof. Burak Acar for participating in my thesis committee.

I owe special thanks to Doğan Can, Ebru Arısoy, Erinç Dikici, Haşim Sak and İpek Şen for their support to complete this work.

During this work, I shared a great year at BÜSİM with Arman Savran, Hatice Çınar, Sinan Yıldırım and Oya Çeliktutan. Thanks to the friends at BÜSİM who made the painful hours fun.

I would like to thank my mother Işık Sunu for her great support, eternal understanding and endless love in every single moment in my life.

This work is funded by the TÜBİTAK project No. 105E102. I would like to express my gratitude to TÜBİTAK which funded me to complete this work and master study. This study is partially supported by Murat Saraçlar's TÜBA-GEBİP award.

ABSTRACT

BOOTSTRAPPING A SPEECH RECOGNITION SYSTEM BY USING SLIDING VIDEO TEXT RECOGNITION

In the broadcast news for the hearing impaired, the information is conveyed by three modalities: speech, sign language and sliding video text. In this work, we propose an HMM-based sliding video text recognition (SVTR) system to generate automatic transcriptions of the speech in broadcast news for the hearing impaired. Then, we bootstrap an unsupervised acoustic model by using those automatic transcriptions.

The sliding video text recognition system is trained by using minimal amount of video data (7 minutes). Well known speech processing techniques are applied to model and to recognize the sliding video text. Baseline system gives 2.2% word error rate over the `video test set`. Then character error analysis is provided and a character based language model is employed to correct the errors. Finally semi-supervised training method is applied and significant error reduction is achieved (2.2% → 0.9%).

An automatic speech recognition system is bootstrapped by using the output of the sliding video text recognizer as the transcriptions. The speech data is segmented automatically and aligned with the automatic transcriptions. An unsupervised acoustic model (U-AM) is trained with 83 videos (11 hours). 12.7% word error rate is achieved for U-AM with 200K language model. The Out of vocabulary (OOV) rates of the language models are decreased by adding the automatic transcriptions of the audio train set to the large text corpus and the effect of OOV rate on system performance is investigated. Finally, we compared the U-AM performance with the supervised one which is built from the same acoustic training corpus with manual transcriptions. Supervised acoustic model performs only 0.4% better than the U-AM (12.7% → 12.3%).

ÖZET

VIDEO KAYAN YAZI TANIMA KULLANILARAK KONUŞMA TANIMA SİSTEMİ EĞİTİMİ

İşitme engelliler haber bültenlerinde bilgi üç kip ile aktarılır; konuşma, işaret dili ve kayan yazı. Bu çalışmada, otomatik konuşma tanıma sistemi eğitimi için otomatik çeviriyazı verisi üretmek amacıyla, Saklı Markov Model tabanlı çalışan kayan yazı tanıma sistemi tasarlanmıştır. Ardından, otomatik çeviriyazılar kullanılarak otomatik konuşma tanıma sistemi eğitilmiştir.

Kayan yazı tanıma sistemi asgari video verisi kullanılarak eğitilmiştir. Kayan yazıyı modellemek ve tanımak için konuşma tanıma teknikleri kullanılmıştır. Temel sistem, test setini %2.2 kelime hata oranı ile tanımaktadır. Ardından, karakter hata analizi yapılmış ve bu hataları düzeltmek için karakter tabanlı dil modeli uygulanmıştır. Son olarak, yarı-öğreticili eğitim tekniği uygulanmış ve kelime hata oranında belirgin bir azalma gözlemlenmiştir. Kelime hata oranı %1 altına indirilmiştir.

Otomatik çeviriyazılar kullanılarak, otomatik konuşma tanıma sistemi eğitilmiştir. Konuşma verisi otomatik olarak bölütlenmiş ve çeviriyazılar ile hizalanmıştır. 83 video (11 saat) kullanılarak öğreticisiz konuşma tanıma sistemi eğitilmiştir. Öğreticisiz akustik model 200K dil modeli ile beraber kullanılarak yapılan sınamada %12.7 kelime hata oranı elde edilmiştir. Kayan yazı tanıma sistemi çıktıları kullanılarak dağırcık dışı kelime oranı azaltılmış ve dağırcık dışı kelime oranının tanıma performansına etkisi gözlemlenmiştir. Son olarak, aynı eğitim setinin manuel çeviriyazısı kullanılarak öğreticili akustik model eğitilmiştir. Öğreticisiz akustik modelin performansı, öğreticili model ile karşılaştırılmıştır. Öğreticili akustik model ile öğreticisiz akustik model performansları arasında %0.4 lik kelime hata oranı gözlemlenmiştir (%12.7 → %12.3).

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS/ABBREVIATIONS

a_{ij}	State Transition Probability
A^*	Audio Time Mark Pairs
A	Audio Time Mark Vector
A^*	Candidate Audio Time Mark Pairs
b_i	State Output Probability
d	Deletion
I^n_{RGB}	N^{th} RGB Frame of Video
I^n_{gray}	N^{th} Gray Level Video Image
I^n_{text}	N^{th} Non-overlapping Text Images Extracted from the Sliding Text Band
i	Insertion
L	Language
$length_{band}$	Length of Sliding Video Text Band
O	Observation sequence
Q	State Sequence
S_i	Finite Number of States
$speed_{\text{text}}$	Character Sliding Speed (pixel/frame)
s	Substitution
t_p	Appearance Duration of a Character on the Sliding Text Band
$V_{horizontal}$	Horizontal Projection Vector
$V_{vertical}$	Vertical Projection Vector
$\mathbf{V}$	Sliding Video Text Time Mark Vector
V'	Sliding Video Text Segment Duration Vector
$v^i(x)$	Image Feature Vector
w_c	Character Width
W	Word
λ	Model
π_i	Initial Condition Probability

AI	Artificial Intelligence
ASR	Automatic Speech Recognition
CER	Character Error Rate
DA	Description of abbreviation
DCT	Discrete Cosine Transform
DFT	Discrete Fourier Transform
EM	Expectation Maximization
FSA	Finite State Acceptor
FSM	Finite State Machine
FST	Finite State Transducer
GMM	Gaussian Mixture Model
HMM	Hidden Markov Model
LM	Language Model
MFCC	Mel Frequency Cepstrum Coefficients
NIST	National Institute of Standards and Technology
NN	Neural Networks
OCR	Optical Character Recognition
OOV	Out-of-Vocabulary
PLP	Perceptual Linear Predictive
RGB	Red-Green-Blue
RTF	Real Time Factor
S-AM	Supervised Acoustic Model
STT	Speech to Text
SVTR	Sliding Video Text Recognition
TTS	Text to Speech
U-AM	Unsupervised Acoustic Model
WER	Word Error Rate
WFST	Weighted Finite State Transducer

1. INTRODUCTION

The emerging technologies resulted in an increase in the use of multimedia data. Recently, the data coding and processing techniques made the multimedia data more available. Internet became widespread since 1990s and it made the information world wide accessible. The computers are getting more high-powered and cheaper. In this era, the human habits started to change by the opportunity technology offers (Amazon, YouTube, Facebook, ..). Millions of videos are being watched over Youtube every day. Newspapers are read via Internet. The paradigm shift in communication creates application areas for Optical Character Recognition (OCR) and Automatic Speech Recognition (ASR).

OCR is the translation of the handwritten or printed text which is captured by a scanner into machine editable text. In such a demand of digital data, OCR is an old but still useful research topic. Nowadays there exist many free OCR distributions such as the form based handwriting recognition system provided by National Institute of Standards and Technology (NIST) [1]. One of the best Turkish supported OCR products, FineReader, has shareware version and may be purchased for about 160 Euros [2].

Speech is the most frequently used communication channel among non-verbal and verbal communication types. It is a natural tool and easiest way to communicate with others. Thus, it is a strong candidate to be used for human-computer interaction. The goal of automatic speech recognition is converting the spoken speech signals to the word sequences.

There are many application areas of automatic speech recognition, i.e. human computer interaction, multimedia applications, dictation systems and applications for mute, hearing impaired or visually impaired people.

Latest research in ASR encourages producing commercial products. Many companies such as Microsoft, IBM and Google are investing on automatic speech recognition research for this purpose. Another application of ASR is in voice activated systems like, telephone banking and voice dialing.

Youtube is a good example of a digital archive. In such a huge amount of digital data archive, including video and audio, the need of classification and retrieval arises. It is almost impossible to watch or listen to every single video and annotate them. It needs time and labor force. Manual annotation of a huge digital video archive is ineffective. Automatic speech recognition is used for this purpose by integrating information retrieval. Google Audio Indexing (Gaudi) [3] is a new service from Google which uses automatic speech recognition to find the spoken terms inside the videos and lets the user jump to the demanded portion of the video where these words are spoken. Turkish Sign Language Dictionary Tool is another example of speech retrieval application among many speech retrieval applications [4], [5].

Speech processing techniques are used in speech to text (STT) and text to speech (TTS) conversion systems. Free electronic book provider, Project Gutenberg Literary Archive Foundation [6] uses text to speech systems to create computer generated audio books which are quite successful. A high-tech Turkish company regarding speech and communication technologies, Sestek's [7] text to speech product which is used in a news portal [8] is a successful example of Turkish text to speech application. With this application, instead of reading the news text on the internet site, one can also listen to the text. Such applications have importance for generating educational documents especially for visually impaired.

Speech to text applications, in other words dictation systems are very popular and commonly used applications in medicine and law. It is also possible to generate automatic transcriptions from speech. Automatic transcription generation from speech can be used for automatic subtitle generation. Such systems are useful for broadcast news for hearing impaired.

1.1. Problem Statement

Speech recognition systems require large amount of data to estimate the model parameters. Preparation of training data is a tedious work. The data should be segmented into utterances and each utterance should be transcribed at the word level. This is an expensive and time consuming process. The transcriptions should be accurate. Therefore, it requires experienced human labor. For instance, it takes $1.5 - 2.5$ hours to segment and annotate and $4 - 6$ hours to transcribe 1 hour data [9].

On the other hand, some training materials naturally include the transcription information. In case of movies, the subtitles are transcription sources and are used in machine translation studies. Sliding text band in broadcast news for hearing impaired is also a transcription source and it is a candidate to reduce the data preparation effort and cost.

In this work, we concentrate on reducing the data preparation effort by making use of sliding video text. In addition, we bootstrap an acoustic model for automatic speech recognition by using automatically generated transcriptions which are the output of proposed sliding text recognition system.

1.2. Related Work

This work is composed of two research areas: sliding video text recognition which may also be thought as video optical character recognition and automatic speech recognition in broadcast news. The related research on text recognition in videos/images and unsupervised acoustic model training for automatic speech recognition will be discussed in the following subsections.

1.2.1. Sliding Video Text Recognition

Research on text recognition in images and videos began with optical character recognition for scanned text documents and handwriting. In 1929 Tausheck obtained the first patent in Germany [10] and in 1933 Handel [11] did the same in the United States. The principle of first application is based on template/mask matching. Tauschek prepared an optical and mechanical template matching setup. Light passed through mechanical masks is captured by photo detectors and is scanned mechanically. When an exact match occurs, the light cannot pass through the template and so the machine recognizes the characters printed on the paper.

In 1950s with the arrival of computers, the commercial demand forced optical character recognition to become a core research area. These applications aim to convert any human readable document to machine manipulatable representation. In these years OCR applications generally used template matching techniques in which an input image is compared to a library of images similar to the Tauschek's setup. For handwritten text documents, low-level image processing techniques have been used on the binary image to extract feature vectors, which are then fed to statistical classifiers. Successful, but constrained algorithms have been implemented mostly for Latin characters and numerals. Moreover, some studies on Japanese, Chinese, Hebrew, Indian, Cyrillic, Greek, and Arabic characters and numerals in both machine-printed and handwritten cases were also implemented [12], [13], [14].

Studies up until 1980 suffered from the lack of powerful computer hardware and data acquisition devices. With the explosion of information technology, the previously developed methodologies found a very fertile environment for rapid growth in many application areas, as well as character recognition system development. Structural approaches were initiated in many systems in addition to the statistical methods [15], [16].

The real progress on character recognition systems was achieved during 1990s by using new methodologies. In the early 1990s, image processing and pattern recognition techniques were efficiently combined with artificial intelligence (AI) methodologies. Researchers developed complex character recognition algorithms, which receive high-resolution input data and require extensive number crunching in the implementation phase. Nowadays, in addition to the more powerful computers and more accurate electronic equipments such as scanners, cameras, and electronic tablets, we have efficient, modern use of methodologies such as neural networks (NNs), hidden Markov models (HMMs), fuzzy set reasoning, and natural language processing [17].

The introduction of Hidden Markov Models (HMMs) to the area of speech recognition has brought several useful aspects to this technology. Use of HMMs brings language-independent training and recognition methodology.

A number of studies have been made on HMMs for Optical Character Recognition. [18], [19], [20] used HHMs for off-line printed and handwriting recognition. In these works, the recognition of only a single language was attempted. For language independence, [21], [22], [23] extracts features from thin slices of the image that makes the system language independent. [24] draws a box around each word to be recognized, and extracts features from vertical thin slices. [24] makes experiments on single printed Roman fonts. [25] also uses vertical thin slices for feature extraction to recognize hand-written addresses.

The state of art recognition systems combine the information in the image or video with the prior information about the language (Language model) in order to produce the best possible recognition result. Language models work best when the text contains a significant amount of text, so that tri-gram or four-gram language models can be effectively applied.

[26] show how continuous speech recognition methods can be used for character recognition. The only difference between their continuous speech recognition system and the optical character recognition system is the nature of the input feature. They extract language independent statistical features from the images instead of the speech signals. [27] is an application on multifont Arabic text recognition and [28] uses the same technique for both Arabic and English text recognition.

In 1990s, OCR problem became a text recognition problem in images and videos. The exponential increase in the amount of digital data, text recognition has large application areas in document management, for instance digital video and image indexing and retrieval applications, [29], [30]. A retrieval application, Turkish Sign Language Dictionary, is a suitable application to use video OCR. Another text recognition problem in videos and images is license plate recognition. There exist several research studies [31], [32], [33], [34] and commercial products are developed for this purpose.

The video text recognition can include a number of different post-processing steps. For example, the recognition results for the different instances of a text can be integrated to produce a single best hypothesis [35]. [36] uses the NIST ROVER algorithm developed for speech recognition to combine 1–best hypothesis from different frames of the same text region.

1.2.2. Unsupervised Acoustic Model Training

Traditional classifiers use only labeled data (feature / label pairs) for training. However, labeled data is often difficult, expensive and time consuming to obtain since it requires efforts of experienced human annotators. Meanwhile unlabeled data is relatively easy to obtain. Semi-supervised training addresses this problem by using large amount of unlabeled data with small amount of labeled data to build better classifiers (models).

A good acoustic model requires a large amount of speech data as training material. These materials can be obtained via Internet or television channels. Excluding the broadcast news programs, the training materials reflects less formal, more spontaneous, a natural form of speech data. The broadcast news programs contain generally fluent spoken training material.

A great amount of work has been reported on unsupervised training of acoustic models. [37], [38], [39] explore the techniques to make use of untranscribed data to increase the available training data amount for automatic speech recognition systems.

The dominant factor on performance of speech recognition systems, especially

Large Vocabulary Conversational Speech Recognition Systems is the amount of available training data [40]. [37] addresses the need to increase the amount of available training data by presenting the ways to make use of untranscribed acoustic data. [39] applies the technique for Mandarin Broadcast News and Broadcast Conversations. [41] reduces the word error rate of a speech recognition system by bootstrapping with 30 minutes of initially labeled training data, and make use of unlabeled data. They show that comparable performance increase can be obtained by using twice as much as untranscribed data as transcribed data. [38] trains the acoustic model with less training data (10 minutes) and analysis performance of the system with varying automatically transcribed acoustic training. They obtain significant word error rate reduction (65.3% → 37.4%) by using the automatically transcribed data.

The transcriptions such as closed-captions are also available for some of these audio materials, especially in movies and broadcast news. The benefit of working with these materials is obtaining the labeled audio data which is very valuable in terms of speech recognizer training. [42] uses the aligned captions to help the acoustic model, by doing partially-supervised adaptation.

1.3. Main Contribution

The main contribution of our work is bootstrapping an unsupervised acoustic model by using the overlaid sliding video text in broadcast news for the hearing impaired.

We propose an HMM-based sliding video text recognition system to recognize the overlaid sliding video text in broadcast news for the hearing impaired. The sliding video text is recognized by using the well known speech processing techniques.

The sliding video text is almost synchronized with the speech. By making use of this modality, we bootstrap an unsupervised acoustic model with the automatically generated transcriptions.

1.4. Thesis Outline

This thesis is organized as follows: Chapter 2 gives the background information on automatic speech recognition. In Chapter 3, we give brief information about our database which consists of three parts: video, audio and text corpus. In Chapter 4, we explain our sliding video text recognition system in detail and give the experimental results. The automatic speech recognition system trained with these automatic transcriptions is explained in Chapter 5 and Chapter 6 concludes our work.

Appendix A includes the synthetic data set. The sliding video text recognition system parameter optimization test results are given in Appendix B and Appendix C explains the minimum edit distance (Levenshtein distance) and a dynamic programming algorithm is given in order to calculate minimum edit distance.

2. BACKGROUND

In this chapter, we give background information on speech recognition and the tools that we used in this study. First, we explain the automatic speech recognition background: feature extraction, HMM modeling and application in speech recognition, language modeling and the evaluation metrics in speech recognition (word error rate and real time factor). Then, we explain the usage of semi-supervised training in recognition systems and describe the algorithm applied in this study. Finally, we make a short introduction to finite state automata and explain how to model a speech recognition system by using FSA.

2.1. Automatic Speech Recognition Background

The basic idea of speech recognition is to find the most likely word sequence $\hat{W}$ in a language L given the observation O;

$$\hat{W} = \underset{W \in L}{\arg\max}\, P(W|O) \tag{2.1}$$

When we apply the Bayes rule, Equation 2.1 becomes;

$$\hat{W} = \underset{W \in L}{\arg\max}\, \frac{P(O|W)P(W)}{P(O)} \tag{2.2}$$

The probability of given observation $P(O)$ is constant for a single word sequence. Since we are applying a maximization operation, we only need to find;

$$\hat{W} = \underset{W \in L}{\arg\max}\; \underbrace{P(O|W)}_{\text{Acoustic Model}}\; \underbrace{P(W)}_{\text{Language Model}} \tag{2.3}$$

where $P(O|W)$ is called *likelihood* calculated by the acoustic model and $P(W)$ is the *prior* probability of the word sequence calculated by the language model.

A generic automatic speech recognition system consists of feature extraction, acoustic modeling, language modeling and decoding components.

The acoustic waveforms are captured via microphones and are converted to digital signals by sampling and quantizing. After digitizing the input acoustic waveform, we have a vector $x(n)$ representing the input signal. In order to model the waveforms, the vector specific information called features should be extracted from the digital representation $x(n)$.

The human perception scale is almost linear in the low frequency ($f <$ 1kHz) and logarithmic above 1kHz. Cepstrum coefficients such as Mel frequency cepstrum coefficients (MFCC) or perceptual linear predictive (PLP) coefficients are the most popular features in speech processing. Because these feature sets are designed concerning the human audio perception scale. MFCC coefficients $c(k)$ are calculated via;

$$c(k) = \mathrm{DCT}\left\{ \log|\mathrm{DFT}\{w_k(n) * x(n)\}| \right\} \tag{2.4}$$

where $w_k(n)$ is the filter bank designed based on the Mel frequency scale, DFT is the discrete Fourier transform and DCT is the discrete cosine transform. ΔMFCCs and other additional features can be concatenated to the feature vector. For detailed information see [43].

Hidden Markov Models (HMMs) are powerful statistical method to model speech signals. An HMM represents a language unit, for instance a word or a phoneme. It has a finite number of states S_i and the state transitions a_{ij}. Each state has probabilistic output function b_i which is represented by a random variable. Gaussian distributions are commonly used for representing these state output functions. The initial condition probabilities of an HMM is described by π_i. Figure 2.1 represents a 3-state HMM.

For a given observation sequence, i.e. the feature vectors for speech recognition, an HMM determines the probability of having generated the observations. The process is hidden because the observations do not uniquely define a particular state as in an observable Markov model but the probabilistic functions of being in that state.

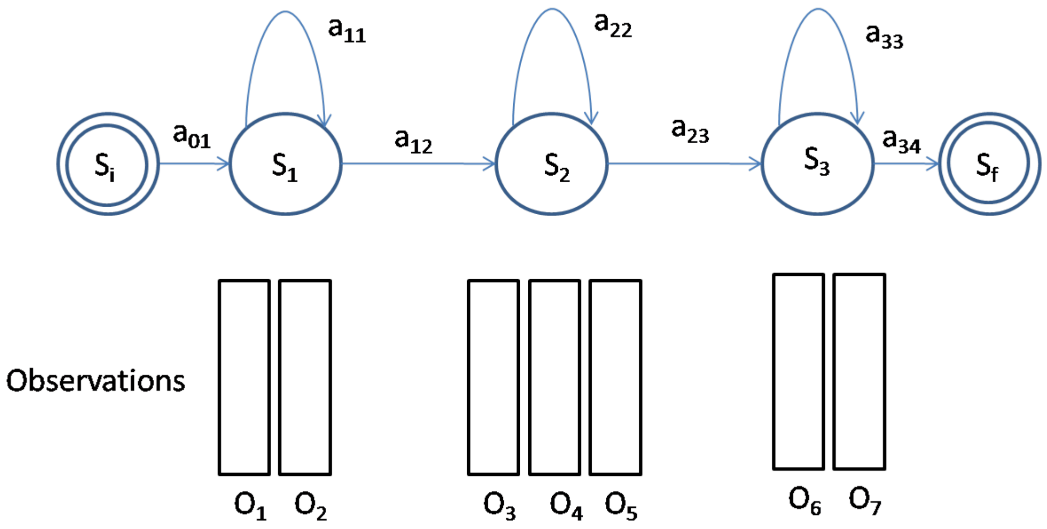


Figure 2.1. Example: a 3-state HMM

In [44], Rabiner states three basic problems associated with HMM for real world applications. These problems are;

- How to compute the probability $P(O|\lambda)$ of the observation sequence $O = O_1O_2...O_T$ given a model $\lambda(A, B, \pi)$.
- How to choose the optimal state sequence which best explains the observation sequence.
- How to adjust, in other words *train*, the model parameters to maximize the $P(O|\lambda)$.

For the solution of the first problem $P(O|\lambda)$, consider a fixed state sequence;

$$Q = q_1q_2...q_T \tag{2.5}$$

where q_1 is the initial state and q_T is the last state of the model. Assume that the observations are statistically independent, the probability of the observation sequence

$O = O_1 O_2 ... O_T$ for the state sequence Q is defined as;

$$P(O|Q, \lambda) = b_{q_1}(O_1) \cdot b_{q_2}(O_2) \cdot ... b_{q_T}(O_T) \tag{2.6}$$

The probability of a state sequence Q given model λ is calculated as;

$$P(Q|\lambda) = \pi_{q_1} a_{q_1 q_2} a_{q_2 q_3} ... a_{q_{T-1} q_T} \tag{2.7}$$

The probability of O and Q occurring at the same time is the joint probability $P(O, Q)$ and calculated by product of $P(O|Q, \lambda)$ and $P(Q|\lambda)$;

$$P(O, Q) = P(O|Q, \lambda) P(Q|\lambda) \tag{2.8}$$

The probability of an observation sequence $O = O_1 O_2 ... O_T$ given a model λ, i.e. $P(O|\lambda)$ is the sum of $P(O, Q)$ over all possible state sequences;

$$\begin{aligned} P(O|\lambda) &= \sum_{\forall Q} P(O, Q) \\ &= \sum_{\forall Q} P(O|Q, \lambda) P(Q|\lambda) \\ &= \sum_{\forall Q} \pi_{q_1} b_{q_1}(O_1) a_{q_1 q_2} b_{q_2}(O_2) a_{q_2 q_3} ... b_{q_T}(O_T) a_{q_{T-1} q_T} \end{aligned} \tag{2.9}$$

The computational complexity of Equation 2.9 is $O(N^T)$ where N is the number of states and T is the length of the observation sequence. The calculation of N^T operations is unfeasible and $P(O|\lambda)$ is calculated by using Forward-Backward Algorithm in a more efficient manner.

The second problem is to find the optimal solution for $P(Q|O, \lambda)$ which becomes a maximization problem for a best Q sequence. It is solved by Viterbi algorithm [45].

Let $\delta_t(i)$ define the best score at time t;

$$\delta_t(i) = \max P(q_1 q_2 ... q_t = i, O_1 O_2 ... O_t | \lambda) \tag{2.10}$$

The best path $\delta_{t+1}(j)$ at time $t + 1$ becomes;

$$\delta_{t+1}(j) = [\max \delta_t(i) a_{ij}] \cdot b_i(O_{t+1}) \tag{2.11}$$

Equation 2.11 states that, we need to keep only the maximized track of each state j at time t. Viterbi algorithm is similar to Forward-Backward algorithm where the only difference is maximization operation over previous states instead of summation of all possible state probabilities.

The last problem is to estimate the model parameters (A, B, π). There are some techniques such as the Baum-Welch Algorithm [46] to estimate the model parameters. These techniques adjust the model parameters to maximize the probability of a given observation sequence O called training data.

The speech sound is produced by articulating the phones. Since the articulation cannot chance shape instantly, each phoneme is effected from previous and following phoneme. Therefore, generally, context dependent models are used to describe the phones in a context. Usually, tri-phone models are used to improve the model efficiency and recognition performance.

In speech recognition, statistical language models are used to estimate the probable word sequences in a language in terms of probability distributions. In Equation 2.3 the prior probability $P(W)$ expresses the language information. $P(W)$ reflects how frequent W word occurs in the language. The probability of a N word sequence is described in Equation 2.12.

$$\begin{aligned} P(W) &= P(w_1 w_2 ... w_N) \\ &= P(w_1)P(w_2|w_1)...P(w_N|w_1 w_2 ... w_{N-1}) \\ &= \prod_{n=1}^{N} P(w_n|w_1 w_2 ... w_{n-1}) \end{aligned} \tag{2.12}$$

where $P(w_n|w_1 w_2 ... w_{n-1})$ is the probability of the word w_n after the word sequence $w_1 w_2 ... w_{n-1}$ called history. A language is modeled in a more accurate manner by considering a longer history. However the longer word sequence tends to appear rare in the corpus which reduces the reliability of the estimate.

The probability of the w_n given the whole history is approximated by the probability given only last $N - 1$ words (see 2.13). This approximation is called n-gram language modeling.

$$P(w_n|w_1 w_2 ... w_{n-1}) \approx P(w_n|w_{n-N+1} ... w_{n-1}) \tag{2.13}$$

The probability of a given word is computed via counting the relative frequencies in the corpus. The probability of a word is calculated as;

$$P(w_n|w_{n-2} w_{n-1}) = \frac{C(w_{n-2} w_{n-1} w_n)}{C(w_{n-2} w_{n-1})} \tag{2.14}$$

where $C(w_{n-2} w_{n-1})$ is the count of the word sequence $w_{n-2} w_{n-1}$ for $N = 3$.

Data sparseness is a problem in language modeling. If the training corpus is not sufficiently enough to estimate the language model parameters, there may be very small or zero probabilities assigned to some sequences. Smoothing gives away to combine less specific, more accurate information with more specific data. This operation increases the small probabilities and decreases the high probabilities to make the probability distribution flatter. Some smoothing methods such Add-One, Witten-Bell, Katz and Kneser-Ney smoothing are used to solve the sparsity problem [43].

The primary evaluation metric to determine the automatic speech recognition system performance is the word error rate (WER) derived from the minimum edit distance [47] (see Appendix C). WER is a common metric of automatic speech recognition and machine translation systems. WER is calculated as in the Equation 2.15 as sum of the number of insertions i, deletions d and substations s divided by the total number of words in the truth text N.

$$WER = \frac{(i + d + s)}{N} \tag{2.15}$$

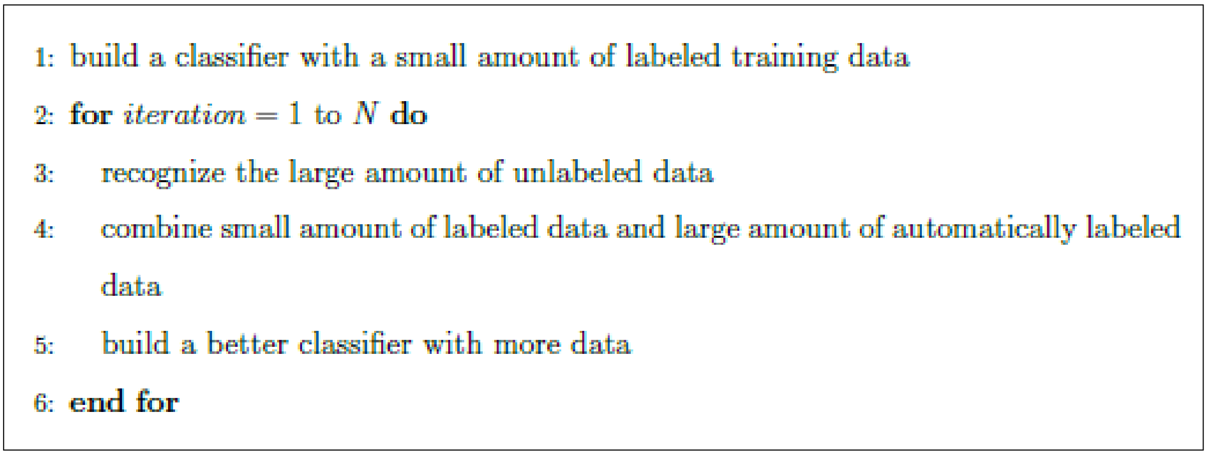
Another issue in measuring automatic speech recognizer performance is the real time factor (RTF). RTF measures the speed of an automatic speech recognition system. It is a hardware dependent value. Equation 2.16 defines the RTF which is the ratio of processing time duration P to the input data duration I. If a system is designed to operate in real time, the RTF should be lower than 1.

$$RTF = \frac{P}{I} \tag{2.16}$$

2.2. Semi-supervised Training

There are two main methods for classification: supervised classification and unsupervised classification. In supervised classification, there are a set of data samples each consisting of measurements of set of variables with associated labels, the class types. In this case, the priori information about the training data set is known. They are used as exemplars for the classifier design. However variable set measurements with associated labels are not always available. There is another pattern recognition task for the training data in absence of the class labels. In unsupervised classification the data are not labeled and the goal is to find groups in the data and the features that distinguish one group from another.

Good models generally requires more training with data. It is easy to find unlabeled data. In order to make use of unlabeled data for training purpose, we can apply semi-supervised training. First, train a initial model from small amount of manually labeled data. Then, we classify the unlabeled data with the initial model. Finally, in order to build a better model, we use both manually labeled and automatic classified data for training. Figure 2.2 shows a basic semi-supervised training algorithm used in this study.



```
build a classifier with a small amount of labeled training data
for $iteration = 1$ to $N$ do
    recognize the large amount of unlabeled data
    combine small amount of labeled data and large amount of automatically labeled data
    build a better classifier with more data
end for
```

Figure 2.2. Semi-supervised training algorithm

2.3. Finite State Machines

A finite state machine (FSM) is a model of behavior of a system by a finite number of states, the transitions between these states and actions. FSMs are commonly used tools in speech recognition for language modeling. Figure 2.3 illustrates a simple finite state machine which models a sheep talk [43].

An FSM which recognizes certain inputs are called an acceptor. The FSM in Figure 2.3 is an acceptor. Figure 2.3 only accepts the words in a sheep language i.e.,

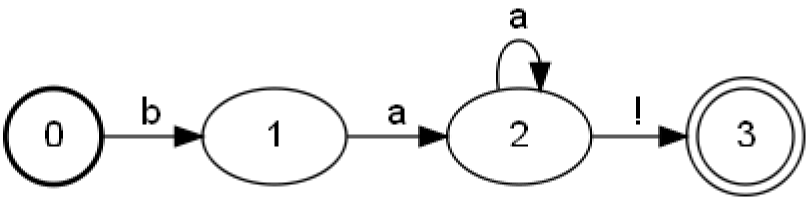


Figure 2.3. Example: a finite state acceptor

"ba!", "baa!", ..., "ba...a!" and any other words are rejected. This specific acceptor has four states; the initial state (state 0), two nodes, a final state (state 4).

Formally, the elements of an FSM are as follows;

- Q: finite set of N states $q_0, q_1, ..., q_{N-1}$
- Σ: an alphabet of finite number of input symbols
- q_0: beginning state
- F: set of final states
- $\delta_{q,i}$: the transition matrix between the states

A FSM which maps a set of input symbols to another symbol set is called a *finite state transducer* (FST). Relating to FSM, the alphabet Σ of an FST contains a finite set of input/output symbol pairs instead of input symbol set. The Figure 2.4 illustrates a simple FST. The transducer maps the input string "baa!" to the string "mee!".

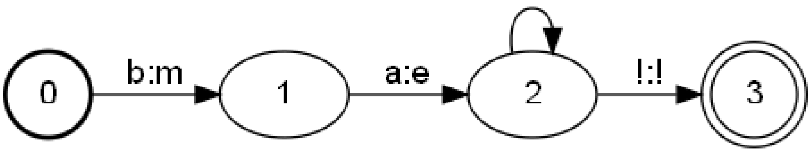


Figure 2.4. Example: a finite state transducer

In FSM and FST, all transitions have the same probability. However, due to the uncertainty of hypothesis in speech recognition, a weight (cost) must be assigned to each transition label. A finite state transducer in which each transition is labeled with a weight is called *weighted finite state transducer* (WFST). Figure 2.5 illustrates the weighted version of the previous finite state transducer. In order to make certain operations on WFSTs, the weights have to form a semiring. Generally log semiring and tropical semiring are used in speech processing.

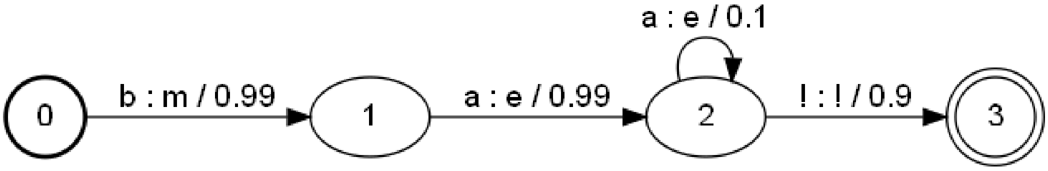


Figure 2.5. Example: a weighted finite state transducer

2.3.1. WFST Applications in Speech Recognition

In the previous section, we explained the parts of an automatic speech recognition system which are: HMMs, Context-dependent phone models, lexicon (pronunciation dictionary) and language model. Each parts of an ASR system can be represented by WFSTs and a speech recognition system can be represented by a composition of the finite state transducers output weights and the output strings [48]. The following equation shows the general speech recognition system representation in terms of finite state transducers from bottom to top as;

$$G \circ L \circ C \circ A \circ O \tag{2.17}$$

where O represents the acoustic observations, A is the acoustic model which maps the acoustic observations to the context dependent phones, C is the context-dependency model which maps the context dependent phones to context independent phones, L is the lexicon mapping the phone sequences to words, and G is the grammar model (Language Model) maps the words sequences to sentences. The system converts the acoustic observation sequences to the sentences.

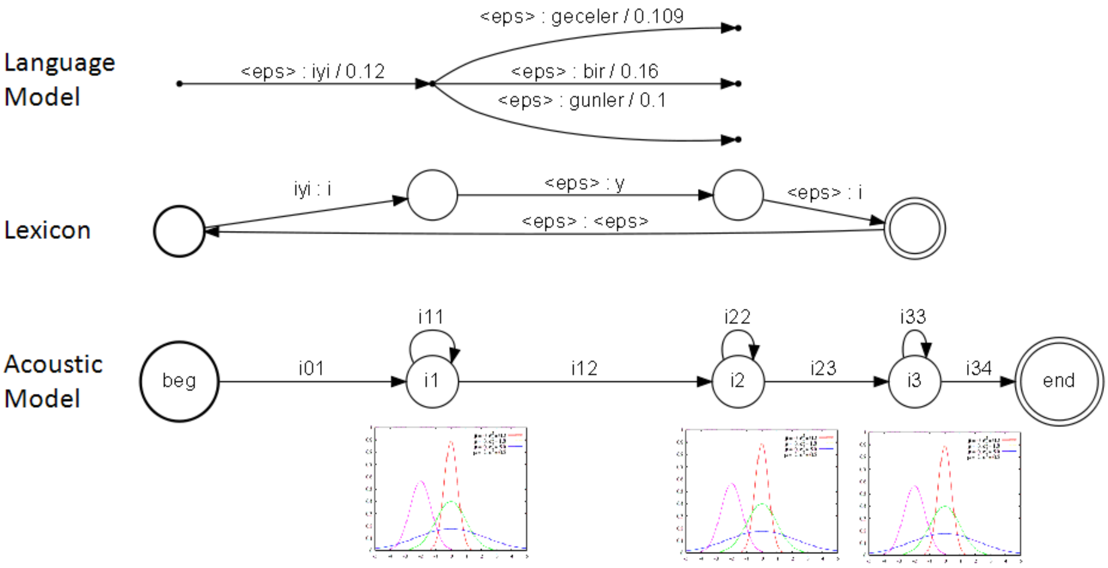


Figure 2.6. Usage of FSMs in automatic speech recognition systems

The composition of acoustic model, context dependency model, lexicon and grammar model cannot be solved explicitly because of the large size of the composition. Instead of searching the whole lattice, a beam pruning is used to find the optimum path in a faster way. Since the best path is of interest, in beam pruning, we select a beam which is calculated by the difference of the weights from the minimum weight less than a predefined threshold and we keep only the paths within the beam during the composition. We can find the most probable path with higher beam width, however, the increase in beam yields an increase in RTF.

3. DATA

A large Turkish Broadcast News Database has been collected at Boğaziçi University. The Turkish Broadcast News Database includes news programs of four television channels, NTV, CNN-Turk, TRT1 and TRT2. In this work, we used the videos of Turkish Radio Television 2 (TRT2) broadcast news for the hearing impaired.

In the news for the hearing impaired videos, the speaker (narrator) signs simultaneously as she reads the news. Furthermore, the news transcriptions are superimposed on the bottom portion of the videos as a sliding text band. This sliding text is synchronized with the sign and the speech. Thus, in these videos, the news is narrated via three different modalities; speech, sign language and sliding video text. These modalities is shown in the Figure 3.1.

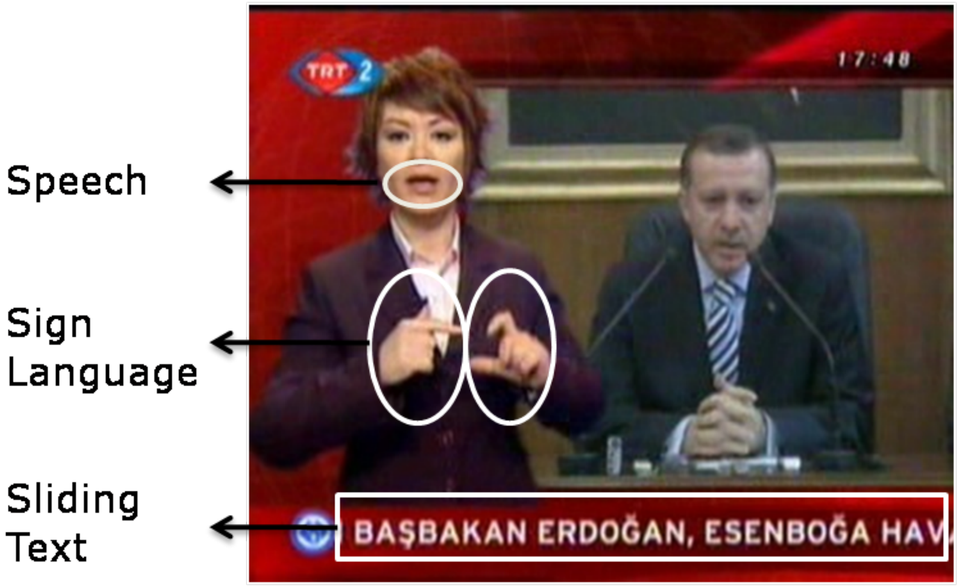


Figure 3.1. Three modalities in the broadcast news for the hearing impaired are: speech, sign language and sliding video text

The video, audio and the text corpus used in this work is explained in detail in the following sections.

3.1. Video

The video corpus (for sliding video text recognition) consists of 127 short video clips. These videos are recorded from Turkish Radio Television 2 (TRT2) broadcast news for the hearing impaired at a resolution of 352x288 at 25fps in DivX® format. The duration of each video is approximately 7 minutes.

The sliding text band is overlaid on the bottom portion of the video. The position of the sliding text region does not change within a video. However, it may differ among the videos. The overlaid text is monochromatic (white). The background color is almost monochromatic (red) and it gets darker towards the right end of the text region. The characters are rigid; their shape, size, color or orientation does not change from frame to frame or video to video. The text moves left with an almost constant speed of 4.3pixel/frame.

Due to the nature of the broadcast news data, character heights on the overlaid text varies between 13 – 16 pixels in the dynamic range of 18 pixel height (see Figure 3.2). Therefore, sliding text band height is defined as 20 pixels. Each video hosts the logo of news for the hearing impaired at the left side of the sliding text band. The logo width is 47 pixels, so the sliding text band width is found as 305 pixels by subtracting the logo width from the video width.

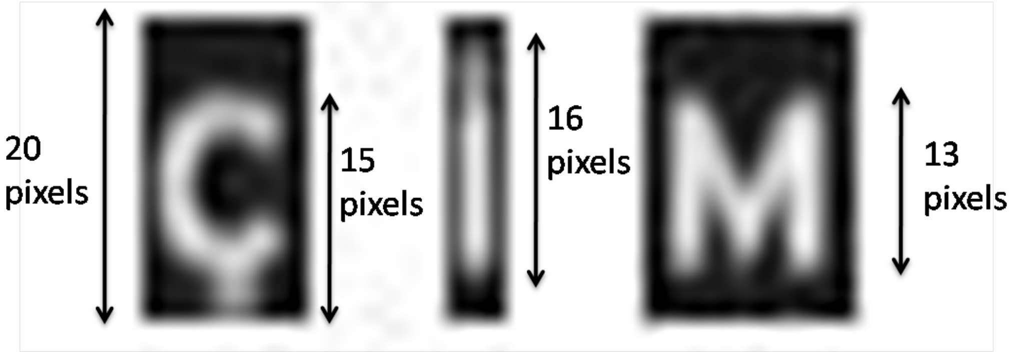


Figure 3.2. Character heights

The text is composed of 29 capital letters (Turkish alphabet), 10 numerals, 8 punctuation marks and 3 different spaces (intra-word space, inter-word space and inter-paragraph space) - a total of 50 characters. See Appendix A for detailed information about the character list.

The space characters are defined based on their durations. Intra-word space cs defines the 2-3 pixel wide space between the characters in a word. Inter-word space ws defines the space between the words. It has approximate duration of 7 pixels. The widest space model, inter-paragraph space ps, is the space between each news segment and its duration is wider than 150 pixels. Figure 3.3 shows an example of each intra-word and inter-word spaces.

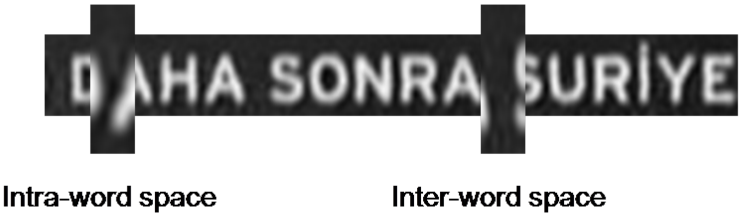


Figure 3.3. Example: Intra-word space and inter-word space character

Table 3.1. Sliding Video Text Recognition Video Corpora

Name	**Video**	**Duration**(min)	**Word**	**Character**
`Video Train Set`	1	7	602	4568
`Video Development Set`	9	60	5090	44591
`Video Test Set`	9	78	6397	57176

19 videos are selected from the video corpus and divided into three non-overlapping sets for sliding text recognition. `Training set` includes only one video and the other 18 videos are divided into two equal parts as `development set` and `test set`. The specifications of these sets are shown in Table 3.1.

3.2. Audio

Audio corpus is obtained by extracting the audio of the video corpus. Audio files are recorded at 32kHz 16-bit PCM WVA format. These audio files are segmented automatically based on the algorithm proposed by [49]. Each news segment, *utterance*, duration is approximately 30 seconds. Currently, the audio corpus includes approximately 17 hours of clean speech considering Classical Hub4 classes.

These audio files are transcribed automatically by using the superimposed sliding text band at the video with the proposed sliding video text recognition system.

Table 3.2. Automatic Speech Recognition Audio Corpora

Name	**Video**	**Duration**(hours)
`Audio Train Set`	83	11
`Audio Test Set`	26	3.5

We divided the audio corpus into two non-overlapping sets. We selected the first 83 audio clips for training set and last 26 audio clips for test set. The audio clips between training set and test set are not used. The audio duration statistics are stated in Table 3.2.

3.3. Text Corpus

A large text corpora is needed for the applications and statistical methods used in language processing. For this purpose, [50] prepared a Turkish large text corpus which is composed of four sub corpora. We used the $NewsCor$ which is collected by crawling three newspaper web pages, to built a language model for automatic speech recognition. A part of $NewsCor$ called $MilliyetCor$ is used to built a character based language model for sliding text recognition task. Table 3.3 gives the statistics of the text corpora used in this work.

Table 3.3. Text corpora statistics

Corpus	**Words**	**Characters**
Milliyet	67M	499M
NewsCor	182M	1486M

4. SLIDING VIDEO TEXT RECOGNITION

The sliding video text in broadcast news for the hearing impaired is the transcription of the speech. An HMM-based sliding video text recognition system is designed to recognize the overlaid sliding video text in broadcast news for the hearing impaired to generate automatic transcriptions from the sliding video text.

In the proposed recognition system, we used well knows speech processing techniques to model and recognize the sliding video text. The front-end process includes the sliding text band detection, text image extraction and feature extraction from those text images. The only difference between an automatic speech recognition system and proposed sliding video text recognition system is the nature of the input, the features are extracted from images instead of speech. The characters are modeled by HMMs and the glyph models are trained by using minimal amount of labeled data. Character based language models are imposed to the system and finally semi-supervised training is investigated to make use of unlabeled data.

4.1. Preprocessing

The goal of preprocessing stage is to obtain the sliding text images from the videos and to extract the image features from these images. Preprocessing stage covers three steps; sliding text band detection and localization, text image extraction and feature extraction as shown in Figure 4.1.

First the sliding text band location is detected. The text images from the sliding text band is extracted within the rules defined in Section 4.1.2. Finally text image features are extracted.

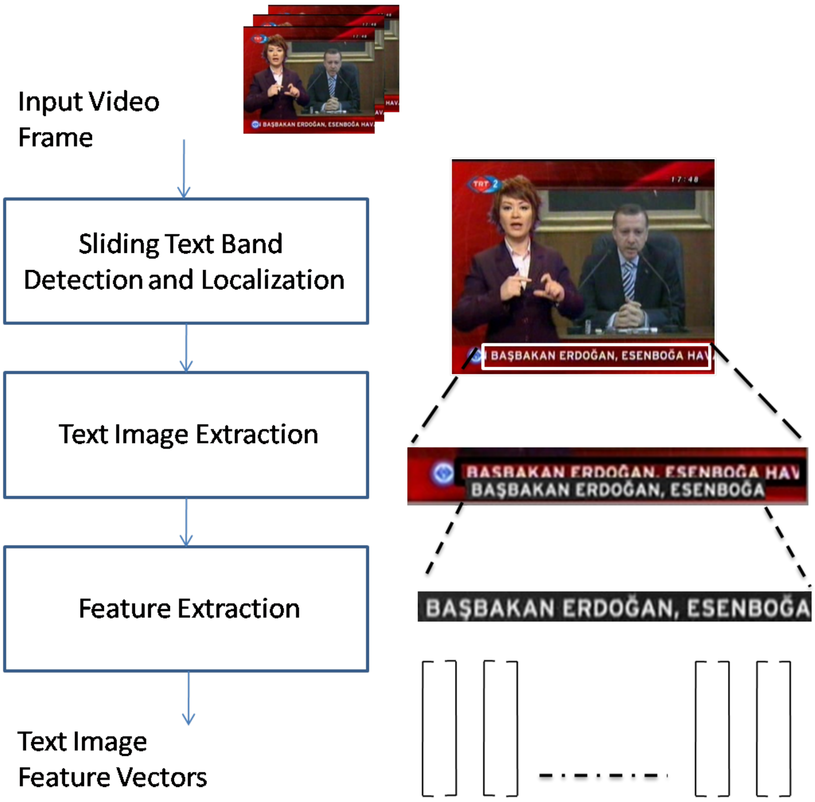

Input Video Frame
Sliding Text Band Detection and Localization
Text Image Extraction
Feature Extraction
Text Image Feature Vectors
BAŞBAKAN ERDOĞAN, ESENBOĞA HAV
BAŞBAKAN ERDOĞAN, ESENBOĞA

Figure 4.1. SVTR preprocessing stage flow diagram

4.1.1. Sliding Text Band Detection and Localization

Neglecting breaking news, the position of the sliding text band does not change in a video. In case of the breaking news, the sliding text band may change position for a $15 - 20$ seconds interval. Apart from that, it is safe to assume the sliding text band position is stationary. However, the position may differ among the videos. Therefore, the need for sliding text band localization arises for each video.

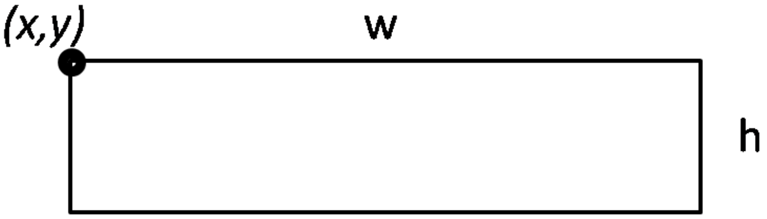


Figure 4.2. A rectangle defined by an offset point $c(x, y)$ and rectangle $r(w, h)$

The sliding text band on an image is defined by an offset point $c(x, y)$ and a rectangle $r(w, h)$ where,

- x defines x-coordinate of the upper-left rectangle corner,
- y defines y-coordinate of the upper-left rectangle corner,
- w defines the width of the rectangle,
- h defines the height of the rectangle as shown in Figure 4.2.

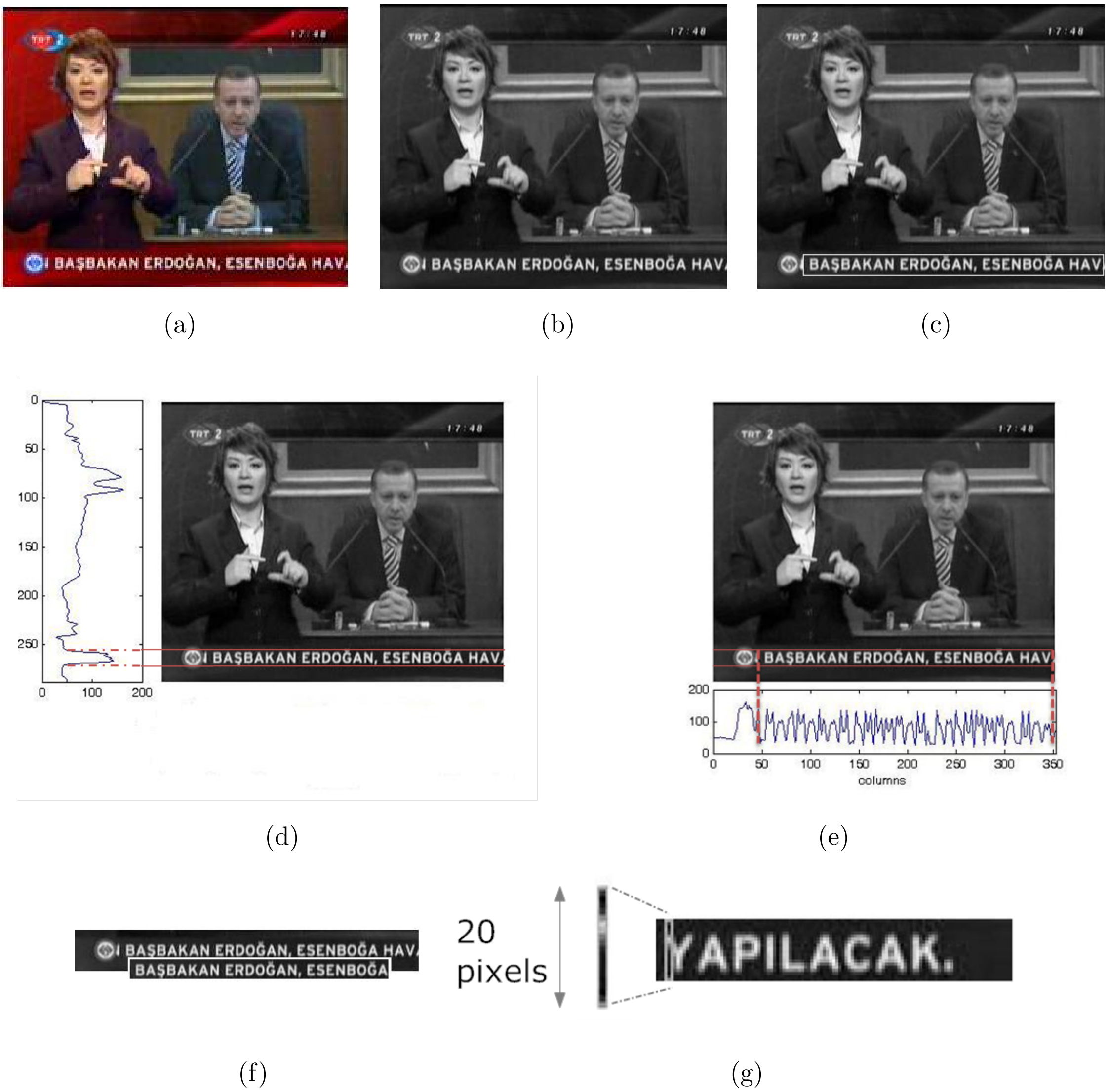


Figure 4.3. SVTR preprocessing stage figures in detail; a)Snapshot, b)Gray-level snapshot, c)Band detection, d)Horizontal projection, e)Vertical projection, f)Text image extraction, g)Feature extraction

The sliding text band detection and localization step is applied only on the first frame of the each video. The first frame $I_{\mathrm{RGB}}^{n=0}$ of each video is grabbed (see Figure 4.3a). Due to the recording setup I_{RGB}^{0} contains 288 rows and 352 columns. Then I_{RGB}^{0} (the grabbed frame) is converted to gray-level image $I_{\mathrm{gray}}^{0}(x, y)$ (see Figure 4.3b), where x is the horizontal direction and y is the vertical direction of I_{gray}. The gray level image pixel values $I_{\mathrm{gray}}^{0}(x, y)$ varies between $[0, 255]$ $\forall x \in [0, 351], \quad \forall y \in [0, 287]$.

The sliding text band detection and localization algorithm [51] is applied to the gray-level image I_{gray} and the sliding text band position of the corresponding video is obtained on $I_{\mathrm{gray}}^{0}(x, y)$. Figures 4.3a, 4.3b, 4.3c are obtained at detection and localization step.

Let P represent a projection operation. Equation 4.1 defines the horizontal projection of $I_{\mathrm{gray}}^{0}(x, y)$ which results the horizontal projection vector $V_{horizontal}(y)$. The y-coordinate y' of the offset c occurs at the hills of $V_{horizontal}(y)$ as shown in Figure 4.3d. The band I_h defined by $c(0, y')$ and $r(352, 20)$ is cropped. I_h hosts the logo of news for hearing impaired at the left side.

$$\begin{aligned} P_{horizontal}(I_{\mathrm{gray}}(x, y)) &= V_{horizontal}(y) \quad \forall y \in [0, 287] \\ &= \sum_{x=0}^{351} I_{\mathrm{gray}}(x, y) \end{aligned} \tag{4.1}$$

Vertical projection of the I_h is used to eliminate the logo and to find the beginning of the sliding text band. Eq 4.2 results the vertical projection vector $V_{vertical}(x)$ where the second zero value of $V_{vertical}(x)$ is the x-coordinate x' of the offset c. Figure 4.3e shows that the second zero value of the vertical projection is the beginning coordinate of the sliding text band.

$$\begin{aligned} P_{vertical}(I_{\mathrm{gray}}(x,y)) &= V_{vertical}(x) \\ &= \sum_{y=0}^{20-1} I_{\mathrm{h}}(x,y) \end{aligned} \tag{4.2}$$

The logo position on x-axis does not vary from video to video, the width of sliding text band is a constant for all videos and is found as 305 by subtracting the logo width w from the band width.

After elimination of the logo, the sliding text band region defined by $c(x', y')$ and $r(305, 20)$ is used for the text image extraction process. Once sliding text region is located in the first frame, it is safe to assume that it appears at the same location for the rest of the video.

4.1.2. Text Image Extraction

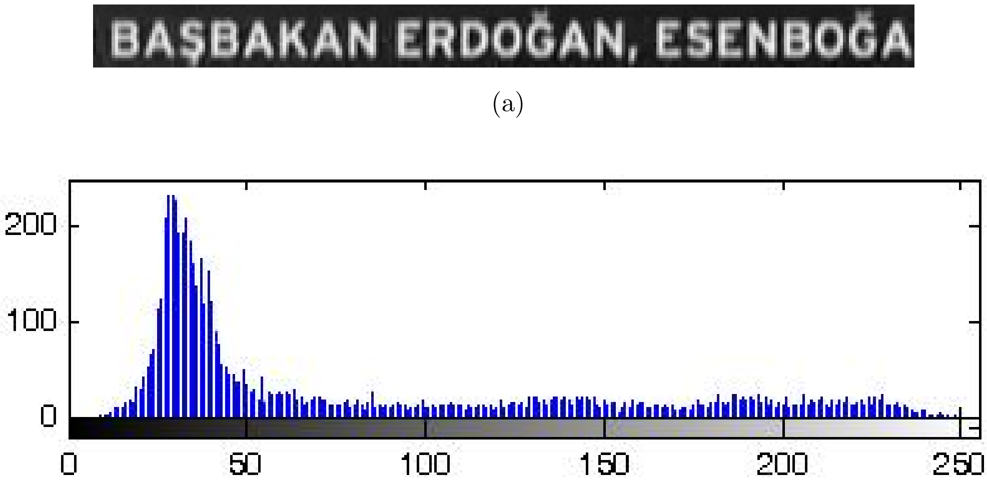
Let $I^i_{text}(x, y)$ defines the i^{th} text image obtained from the text band for $\forall x \in [0, M^i]$, $\forall y \in [0, 19]$, $\exists I^i_{text}(x, y) \in [0, 255]$. The following requirements are forced for the extracted text images.

- The text images should be non-overlapping.
- Each text image should include maximal amount of words.
- The words in text images should be complete words. No half word or character is allowed at either end of the text image.

Assume that the inter-word spaces are wider than 5 pixels.

$$P_{vertical}(I^i_{text}(x + k, y)) < \text{threshold} \tag{4.3}$$

defines an inter-word space $\forall k \in [0,4]$, $\forall x \in [0, M^i - 4]$ and $\forall y \in [0,19]$. threshold is found by observing the histogram of $P_{vertical}(I^i_{text}(x,y))$ shown in Figure 4.4b.



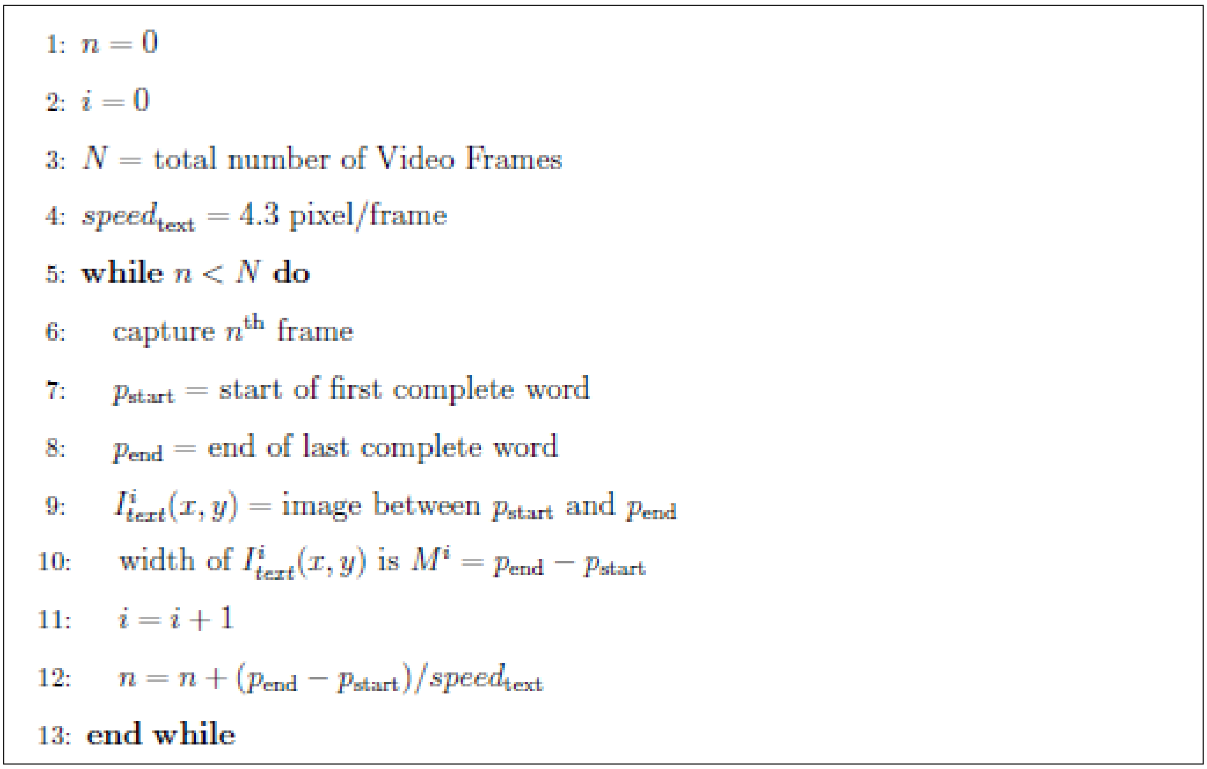
(a)

(b)

Figure 4.4. a)Text Image $I^i_{text}(x,y)$, b)Histogram of $I^i_{text}(x,y)$

These requirements (or rules) are fulfilled by the algorithm stated in Figure 4.5 and this algorithm yielded a total number of 11487 words in 3456 text images which are extracted from development set and test set. All $I_{text}(x,y)$ have a constant height of 20 pixels. The width of $I^i_{text}(x,y)$,M^i, depends on the number of complete words and word lengths. Therefore, it may vary depending on the text images. Figure 4.6 shows the non-overlapping text images extracted from the video text.

$$\begin{aligned} t_p &= \frac{length_{band}}{speed_{text} \times frameRate} \\ &= \underbrace{305pixels}_{\text{length}_{band}} \times \underbrace{\frac{frames}{4.3pixels}}_{\text{speed}_{text}} \times \underbrace{\frac{sec}{25frames}}_{\text{frameRate}} \\ &= 2.83(sec) \end{aligned} \tag{4.4}$$



```
$n = 0$
$i = 0$
$N$ = total number of Video Frames
$speed_{\text{text}} = 4.3$ pixel/frame
**while** $n < N$ **do**
    capture $n^{\text{th}}$ frame
    $p_{\text{start}}$ = start of first complete word
    $p_{\text{end}}$ = end of last complete word
    $I^i_{text}(x, y)$ = image between $p_{\text{start}}$ and $p_{\text{end}}$
    width of $I^i_{text}(x, y)$ is $M^i = p_{\text{end}} - p_{\text{start}}$
    $i = i + 1$
    $n = n + (p_{\text{end}} - p_{\text{start}})/speed_{\text{text}}$
**end while**
```

Figure 4.5. SVTR text image extraction algorithm

On the other hand, as far as data collection and recognition tasks are concerned, sliding video text provides a unique opportunity that does not exist in the case of speech. In case of sliding video text, each character stays on the screen for a certain period of time while moving from right to left (Equation 4.4). During this period, each frame provides a new image of that character since the constituting pixels does not remain as the same as the text moves. To this end the training set may be sampled at different rates to produce more training data and development and test sets may be sampled at different rates to make a more precise decision by multi frame integration [52].

4.1.3. Feature Extraction

Our aim is a language and character independent sliding text recognition system. So the selected features should be language and character independent. Therefore, any shape or punctuation related features are not used in this work. [21], [22], [23] define

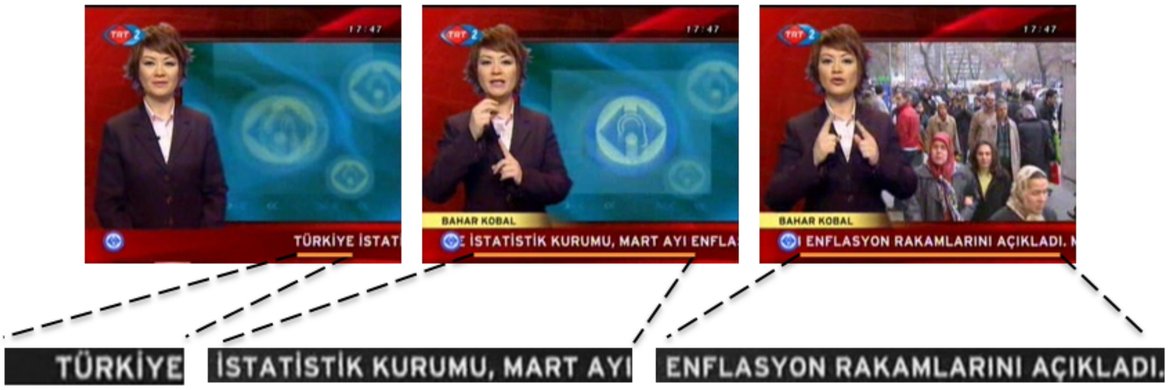

TÜRKİYE İSTATİ
İSTATİSTİK KURUMU, MART AYI ENFLA
BAHAR KOBAL
ENFLASYON RAKAMLARINI AÇIKLADI.
TÜRKİYE
İSTATİSTİK KURUMU, MART AYI
ENFLASYON RAKAMLARINI AÇIKLADI.

Figure 4.6. Non-overlapping text image extraction

a feature vector as a function of horizontal position within a vertical line on the text image. [26] defines a frame as a narrow vertical strip, with a width that is a small fraction of the height of the text image. Each frame is divided into 20 overlapping cells and the features they compute are;

- Intensity as a function of vertical position
- Vertical derivative of intensity
- Horizontal derivative of intensity
- Local slope and correlation in a window of 2 cells square

We simply select the pixel bitmap values as features [53]. Each column of $I^i_{text}(x,y)$ is selected as a feature vector as shown in Figure 4.3f. Equation 4.5 defines M^i 20 dimensional feature vectors of $I^i_{text}(x,y)$. Equation 4.6 shows $I^i_{text}(x,y)$ in matrix form.

$$\mathbf{v^i}(\mathbf{x}) = I^i_{text}(x,y) \quad \forall x \in [0, M^i - 1] \tag{4.5}$$

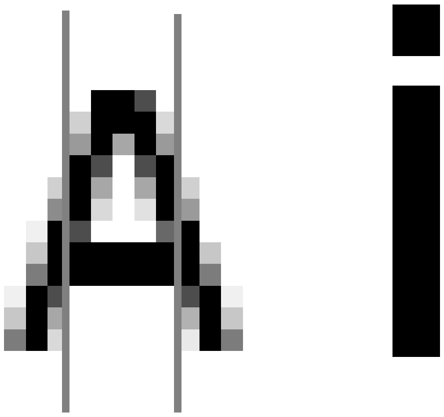
$$I^i_{text}(x,y) = \left[\overbrace{\begin{pmatrix} I_{text}(0,0) \\ I_{text}(0,1) \\ \vdots \\ I_{text}(0,19) \end{pmatrix}}^{\mathbf{v^i(0)}} \overbrace{\begin{pmatrix} I_{text}(1,0) \\ I_{text}(1,1) \\ \vdots \\ I_{text}(1,19) \end{pmatrix}}^{\mathbf{v^i(1)}} \overbrace{\begin{pmatrix} \cdots \\ \cdots \\ \ddots \\ \cdots \end{pmatrix}}^{\dots} \overbrace{\begin{pmatrix} I_{text}(M^i-1,0)) \\ I_{text}(M^i-1,1) \\ \vdots \\ I_{text}(M^i-1,19) \end{pmatrix}}^{\mathbf{v^i(M^i-1)}} \right] \tag{4.6}$$

4.2. Modeling

In automatic speech recognition systems, context dependent models are used to account for the co-articulation effects between adjacent phones. [26] states that in case of optical character recognition, Arabic script and hand writing can be considered as context dependent. However, in our case character shapes are independent from previous and following characters, that are we are dealing with context independent characters. Therefore, context independent HMMs are used to model characters in this work.

There are three different spacing in the sliding video text: inter-character space which has a short duration of 2–3 pixels, inter-word space with a medium duration of 6–7 pixels and inter-paragraph space which lasts for more than 10 pixels. There are a total of 50 different character models in the sliding text recognition system including 3 different spacing. Most characters like "A" have a start, middle and end portion, so we model them as 3-state HMMs. Others like "İ" are modeled as single state HMMs. See Figure 4.7 for examples of both character types.

Each state of the HMM has an output probability distribution over the features modeled as a single Gaussian. The maximum likelihood estimate of the HMM parameters are obtained by iteratively aligning the sequence of feature vectors with the sequence of character models using Expectation Maximization (EM) algorithm.



(a) (b)

Figure 4.7. Examples of a)3-state character “A”, b)1-state character “İ”

We use a self-mapping transducer as a lexicon. The lexicon can accept any single character and returns back with ϵ. The following figure illustrates graphical representation of the lexicon.

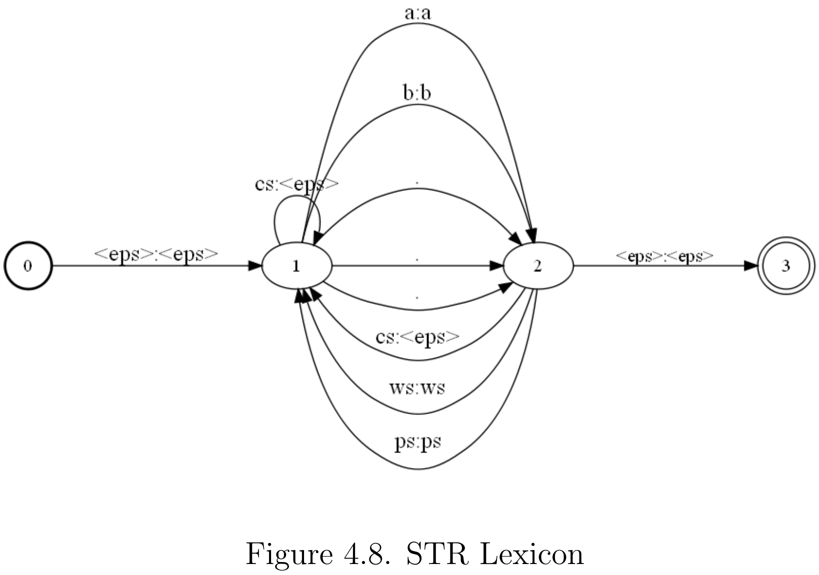


Figure 4.8. STR Lexicon

4.3. Experimental Results of Different Training Setups

Before training, a bootstrap glyph model is trained from the synthetic data set which includes a single example of each character. See Appendix A for more detail about the synthetic data set. Synthetic data is automatically segmented by dividing the pixel width of each character image with the number of states for the corresponding character. Gaussian models are initialized with the state means and unit variance for all dimensions. The bootstrap model is used to initialize the training system by segmenting the train set.

As far as data collection and recognition tasks are concerned, sliding video text provides a unique opportunity that does not exist in the case of speech as mentioned in Section 4.1.2. Each text character stays on the screen for a certain period t_p. During this period, each frame provides a new image of that character. This fact can be exploited to boost the amount of available training data. To this end, training set, whose transcription includes 602 words and 4568 characters, is sampled at two different rates (5fps and 1.25fps). Table 4.1 gives the statistics of the sampled train set at two sampling rates. However, the same fact is not exploited during recognition with development and test sets. The development and test set text images are extracted based on the algorithm shown in Figure 4.5 without overlapping.

Table 4.1. Training video data statistics

Sampling Rate	**#frame**	**#word**	**#character**
5.00fps	2084	9726	63850
1.25fps	524	2444	16060

Only the training data is transcribed at word level with time marks that we call "labeled" data. Each $I^i_{text}(x, y)$ in train set has features and time marked transcriptions corresponding to these feature. The development and test sets are remained unlabeled.

Word error rate is used to measure the sliding text system performance (see Equation 2.15). The baseline glyph model experiments give the performance of the sliding text recognition system using the glyph models obtained from the train set under two different sampling rates. The effect of labeled data size is investigated over recognition performance. We then provide an analysis of character errors and suggest character-based language modeling as a remedy. Finally, we investigate the semi-supervised training possibility and obtain results comparable to those from baseline training.

4.3.1. Baseline Experiments

Two baseline glyph models are trained from the train set. As shown in Table 4.1 train set is sampled at two different rates. Baseline glyph model-I is trained from 5.00fps sampled data and baseline glyph model-II is trained from 1.25fps sampled data. The baseline glyph model training and test flow diagram is shown in Figure 4.9.

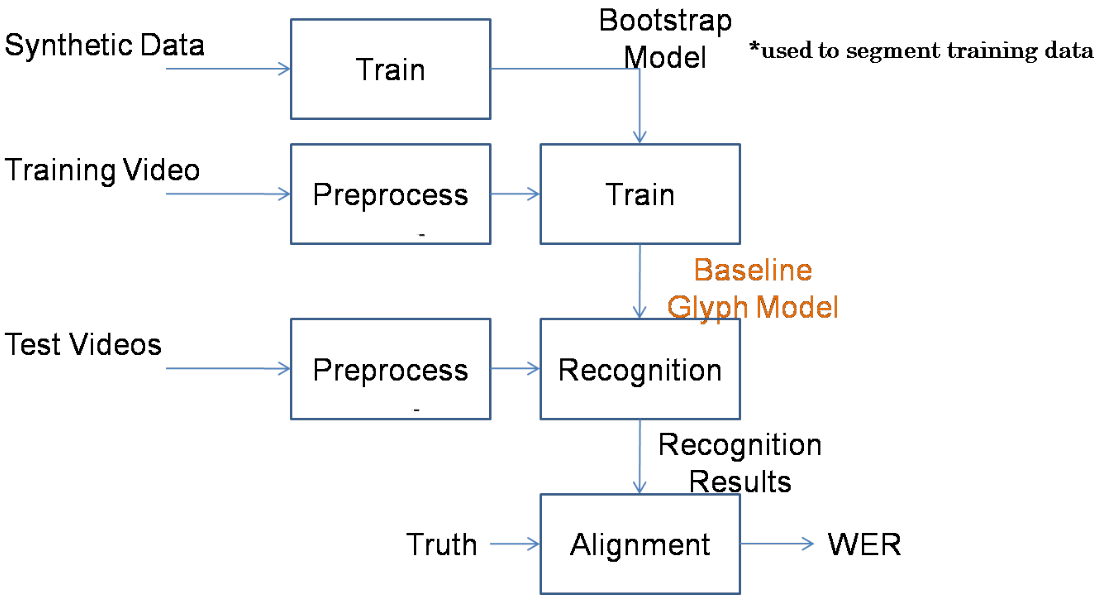


Figure 4.9. SVTR baseline experiment flow diagram

Table 4.2 utilizes these baseline glyph models' WER performances on development and test sets. We see that a very small amount of training data is enough for a good glyph model. The baseline glyph model-I which is trained with 9726 words recognition performance is higher than the baseline glyph model-II which is trained with 2444 words recognition performance over both development and test sets. The

Table 4.2. SVTR baseline glyph model WER performance on development and test sets

Model Type	Test Data	
	`dev`	`test`
Baseline Glyph Model-I (5.00 fps)	2.4	1.9
Baseline Glyph Model-II (1.25 fps)	2.6	2.2

increase in training data yields a better model as expected.

Table 4.3. SVTR baseline glyph model-I character confusion table

Error Types	Truth	Rec	Count
Substitution	0	O	138
	O	0	26
	I	İ	10
	8	B	6
	L	İ	5
	Other		42
	All		227
Deletion	All		331
Insertion	All		94

Then, character confusion analysis is provided on baseline glyph model-I recognition results. Recognition results are compared with the truth. Table 4.3 gives the character confusions made by the baseline glyph model-I. We see that character errors are mostly due to the confusion between visually similar characters like the letter ”O” and the numeral ”0”. The letter ”O” and the numeral ”0” are confused 164 times in 11K words.

4.3.2. Baseline with Language Modeling Experiments

Error analysis suggests that the confusions between visually similar characters can be resolved by incorporating contextual information into the system. To this end, we employed character-based Kneser-Ney smoothed n-gram language models during recognition (see Figure 4.10). The language models are built from *Milliyet* news corpus with SRILM tools [54].

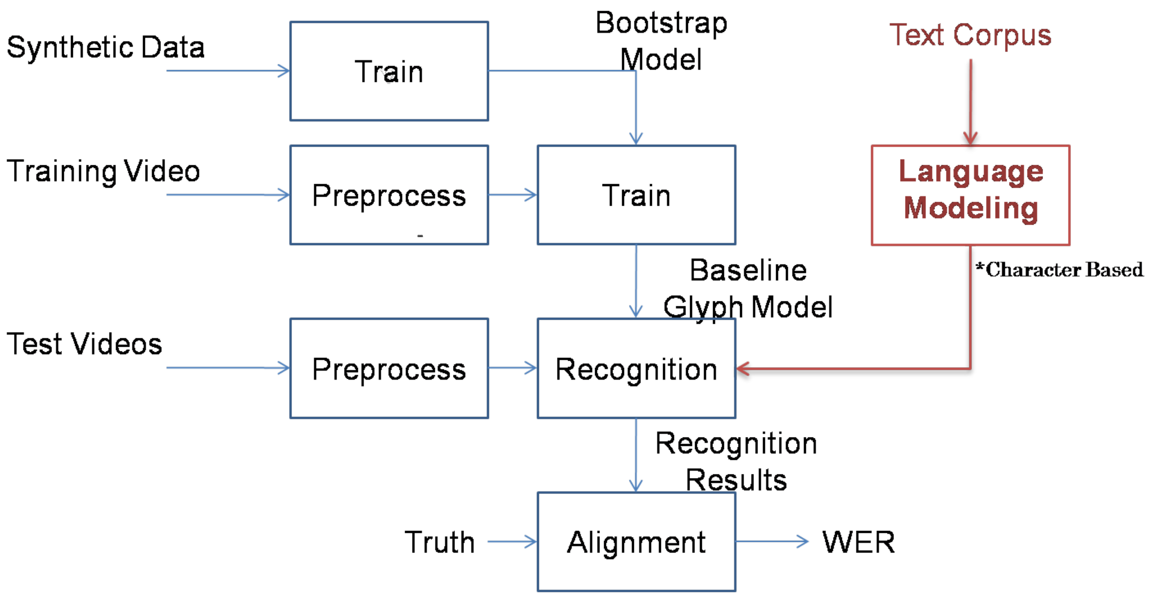


Figure 4.10. SVTR baseline with language model experiment flow diagram

System parameters, duration multiplier *dur* and grammar multiplier *grm* were optimized over the development set for each n-gram language model. We employed a grid test $\forall dur, grm \in [3, 13]$ $\forall n \in [3, 7]$. The grid test results for all n-gram language models are shown in Appendix B. The optimum values of duration and grammar multipliers are found as 9 and 11 due to the model parameter optimization test results. The optimized system performance for each language model are given in Table 4.4.

Optimization tests in Table 4.4 show that any test with language model gives better performance than the test results excluding language models shown in Table

Table 4.4. SVTR baseline glyph model-I with N-gram language model tests over development set

LM Type	**None**	**3-gram**	**4-gram**	**5-gram**	**6-gram**	**7-gram**
WER	2.6	1.4	1.2	1.1	1.0	1.1

4.2. 6-gram language model gives the best performance and 3-gram language model has the poorest performance. The n-gram statistics of 3-gram and 6-gram language models are stated in Table 4.5.

Table 4.5. The # of n-grams of 3-gram and 6-gram language models

LM Type	**unigram**	**bigram**	**trigram**	**4-gram**	**5-gram**	**6-gram**
3-gram	57	2588	41093	–	–	–
6-gram	57	2650	46427	344821	1432498	4143765

Table 4.6 shows the character confusions of baseline glyph model-I without language model and with 3-gram and 6-gram language models. Incorporating language model to the system significantly reduces character substitutions and deletions as shown at the seventh and ninth rows of Table 4.6. The confusion of visually similar characters problem is almost solved. However, the same effect is not observed for deletions as shown at eighth row of Table 4.6. The reason for that is, incorporating a character based language model results in word deletions. This fact may increase the character error rate.

The second, third, fifth and sixth rows of Table 4.7 gives the system performance when 3-gram and 6-gram language models are used along with the glyph models. We see a significant decrease in WERs as suggested by the character confusion analysis of baseline glyph model tests. Since the system parameters are optimized on development set, we observe a higher relative reduction in WER over development set (2.4 → 1.0) compared to test set (1.9 → 1.5) results.

Table 4.6. SVTR character confusions with different language models

Error Types	**Truth**	**Rec**	**Count**		
			-	**3-gram**	**6-gram**
Substitution	0	O	138	7	3
	O	0	26	2	2
	I	İ	10	15	4
	8	B	6	0	0
	L	İ	5	0	0
	Other		42	24	5
	All		227	48	13
Deletion	All		331	672	244
Insertion	All		94	18	12

4.3.3. Semi-Supervised Glyph Model Experiments

In this experiment, we investigate the effects of semi-supervised training by training the glyph models at two steps. At first step, the models are trained only with the labeled training data at two different sampling rates as trained in Section 4.3.1. The baseline glyph model-I and baseline glyph model-II is used to generate more training samples from previously unlabeled data. The development set is recognized and labeled by those models and more training data is automatically obtained for second step.

At second step, baseline glyph models are retrained with the expanded training set including both `test set` and `development set` and the final models, semi-supervised glyph models, are obtained as retraining resultant models. The expanded training set statistics are shown in Table 4.8 and the flow diagram of semi-supervised training is shown in Figure 4.11.

Table 4.9 shows the performance of all glyph models with different language models. Fourth, fifth, ninth and tenth row of Table 4.9 gives the performance of semi-supervised glyph model performances. The system performance is significantly

Table 4.7. Baseline glyph models with 3-gram and 6-gram language models WER performance on development and test sets

Model Type	**Language Model**	**Test Data**	
		`dev`	`test`
Baseline Glyph Model-I	–	2.4	1.9
	3-gram	1.4	1.7
	6-gram	1.0	1.5
Baseline Glyph Model-II	–	2.6	2.2
	3-gram	1.7	1.9
	6-gram	1.2	1.5

Table 4.8. Expanded video training data statistics

Data Set	**#word**	**#character**
`Train set`(@5.00fps) + `Dev Set`	14786	108441
`Train set`(@1.25fps) + `Dev Set`	7534	60651

increased with semi-supervised training. A WER under 1% is achieved over `test set`.

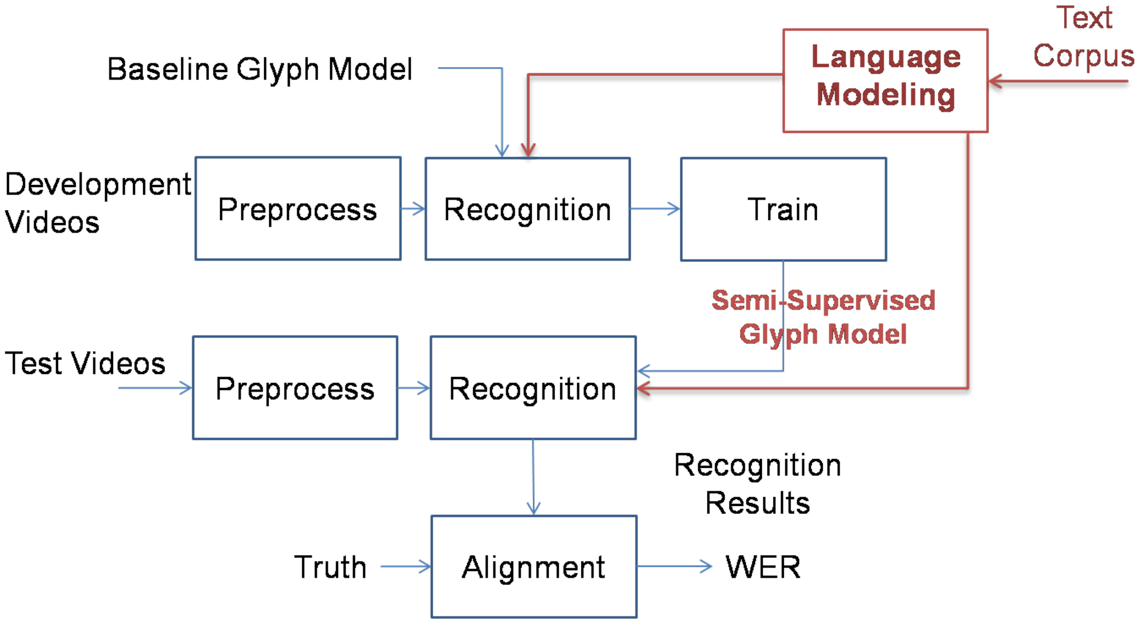


Figure 4.11. SVTR semi-supervised glyph model experiments flow diagram

Table 4.9. Semi-supervised glyph models with 3-gram and 6-gram language models WER performance on `development` and `test` sets

Model Type	**Language Model**	**Test Data**	
		`dev`	`test`
Baseline Glyph Model-I	–	2.4	1.9
	3-gram	1.4	1.7
	6-gram	1.0	1.5
Semi-Supervised Glyph Model-I	3-gram	1.2	1.4
	6-gram	1.0	1.0
Baseline Glyph Model-II	–	2.6	2.2
	3-gram	1.7	1.9
	6-gram	1.2	1.5
Semi-Supervised Glyph Model-II	3-gram	1.5	1.4
	6-gram	1.0	**0.9**

5. AUTOMATIC SPEECH RECOGNITION

In the broadcast news for the hearing impaired, the information is conveyed by three modalities; speech, sign language and sliding video text. In other words, the transcriptions of the speech are already embedded on the news videos as an overlaid text band. In Chapter 4, we propose an HMM-based sliding video text recognition system and generate the automatic transcriptions of the speech by using the sliding video text. In this chapter, we bootstrap an unsupervised acoustic model by using automatically generated transcriptions.

At the front end of the system, we segment the audio data into utterances, align the utterances with the automatic transcriptions and extract the audio features. A bootstrap model training is started with the artificial segmentations and then three iterations of EM algorithm is applied. The bootstrap model is only used to build the unsupervised acoustic model. Finally, unsupervised acoustic models performance is compared to the supervised model.

5.1. Preprocessing: Audio Segmentation and Reference Text Alignment

In preprocessing stage, the audio files are segmented to utterances and aligned with the reference text. The audio features are extracted from the utterances and audio feature / text pairs are prepared.

The reference text is automatically generated by the proposed sliding video text recognition system. Each broadcast news video clip is segmented into N short news segments, *video utterances*, by using inner paragraph spaces of sliding video text. Each video utterance is recognized and transcriptions with the time marks $\mathbf{V}$ are obtained from the sliding video text. The sliding video text time mark vector $\mathbf{V}$ has N time pairs indicating the beginning and ending time of each paragraph of the sliding video text.

$$\mathbf{V} = (t^v_{b1}, t^v_{e1}), (t^v_{b2}, t^v_{e2})...(t^v_{bN}, t^v_{eN}) \quad , t^v_{b1} < t^v_{e1} < t^v_{b2} < t^v_{e2} < ... < t^v_{bN} < t^v_{eN} \tag{5.1}$$

where t^v_b is the beginning time index and t^v_e is the ending time index of a video segment.

The speech/non-speech intervals are determined by the algorithm in [49]. There exists a non-speech interval between each audio paragraph. As a first step, we label the paragraph beginnings and endings by searching non-speech intervals longer than 1 second. M audio segments are obtained with time mark pairs $\mathbf{A}$.

$$\mathbf{A} = (t^a_{b1}, t^a_{e1}), (t^a_{b2}, t^a_{e2})...(t^a_{bM}, t^a_{eM}) \tag{5.2}$$

where t^a_b is the beginning time index and t^a_e is the ending time index of an audio segment.

However, there also exists non-speech intervals longer than 1 second within a paragraph i.e., between the sentences or between some words as well. Hence, the number of audio segments are always greater than the number of video segments ($M \geq N$). Time mark pairs $\mathbf{A}$ are not reliable time mark pairs and do not exactly indicate the audio paragraph segments.

Since it is not possible to segment the audio paragraphs with the corresponding transcriptions by using only speech/non-speech intervals, we propose to use sliding text transcription time marks $\mathbf{V}$ for audio data segmentation. However sliding video text is not fully synchronized with the speech and there exists an unknown time offset $\triangle t$ between the audio and the video data. Figure 5.1 shows the audio and video segments along a time line.

We need to incorporate additional information to segment the audio data. Since the narrator reads the news as the video text slides, we can assume that the durations of both sliding video text segments and audio segments are equal. Therefore, we decided

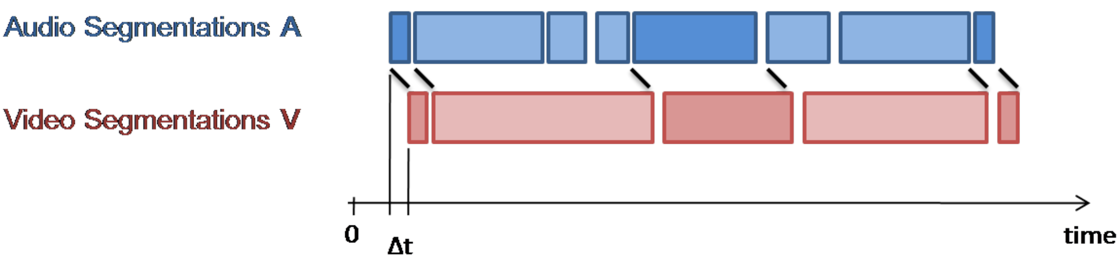


Figure 5.1. Audio and video segmentation

to use the sliding video text segment durations as an additional information source for segmentation and alignment.

Let $\mathbf{V}'$ be the sliding video text segment duration vector;

$$\mathbf{V}' = (t^v_{e1} - t^v_{b1}), (t^v_{e2} - t^v_{b2})...(t^v_{eN} - t^v_{bN}) \tag{5.3}$$

We select all possible N time pairs from the audio time sequence $\mathbf{A}$. Let $\hat{A}$ be a N time pair candidate time mark vector for $t^a_{b1} < t^a_{e1} < t^a_{b2} < t^a_{e2} < ... < t^a_{bN} < t^a_{eN}$.

Define error err as;

$$err = \sum_{n=1}^{N} |\hat{A}_k(n) - \mathbf{V}'(n)| \tag{5.4}$$

The audio time mark pairs A^* is the candidate time mark vector which minimizes the error.

$$\begin{aligned} A^* &= \arg\min_{\hat{A}} err \\ &= \arg\min_{\hat{A}} \sum_{n=1}^{N} |\hat{A}_k(n) - \mathbf{V}'(n)| \end{aligned} \tag{5.5}$$

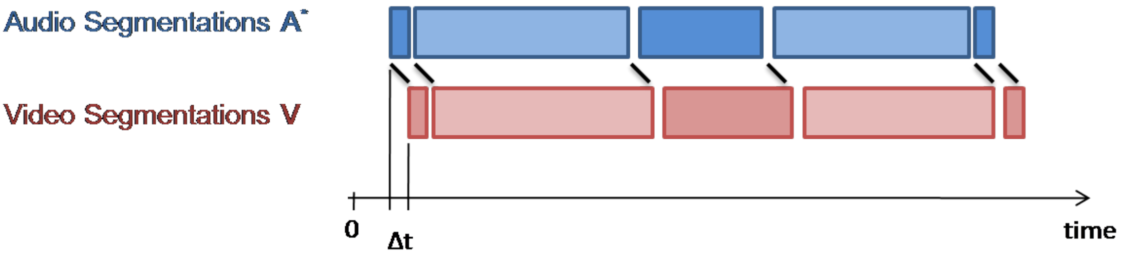


Figure 5.2. Aligned audio and video segmentations

Figure 5.2 shows the aligned audio segmentations with the corresponding video segmentations. By using the sliding video text duration $\mathbf{V}'$ as a supplementary information, the split audio segments $\mathbf{A}$ are merged and the exact audio paragraph segments A^* are aligned with the sliding video text segments automatically.

The audio features between the time pairs A^* are extracted as mentioned in 2.1.

5.2. Modeling

5.2.1. Phone Modeling

The phone set includes 29 Turkish phones and one silence phone - a total of 30 phones and each phone is modeled by 3-state HMMs. The output distribution of each HHM state is defined by Gaussian Mixture Models. The number of Gaussian mixture components are calculated based on the observation count. Figure 5.3 shows the three state representation of the phone "A".

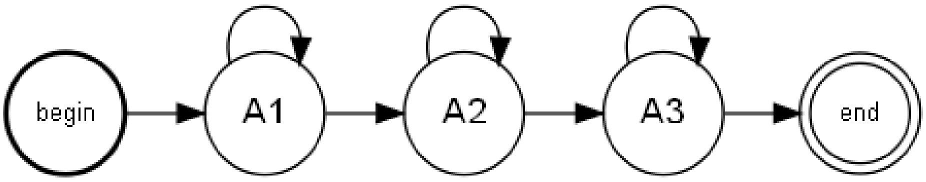


Figure 5.3. 3-state representation of phone "A"

5.2.2. Bootstrap Model Training

So far audio features between the time marks A^* are extracted. Each audio paragraph has a feature vector set and automatically generated transcription. The feature / transcription pairs of each paragraph are prepared for each video.

18 audio clips are used to train the bootstrap acoustic model. The audio features are initially segmented uniformly. We assume each character in a paragraph has the same duration as shown in Figure 5.4 and the number of uniform segments are calculated by dividing the number of feature vectors by the number of characters in a paragraph.

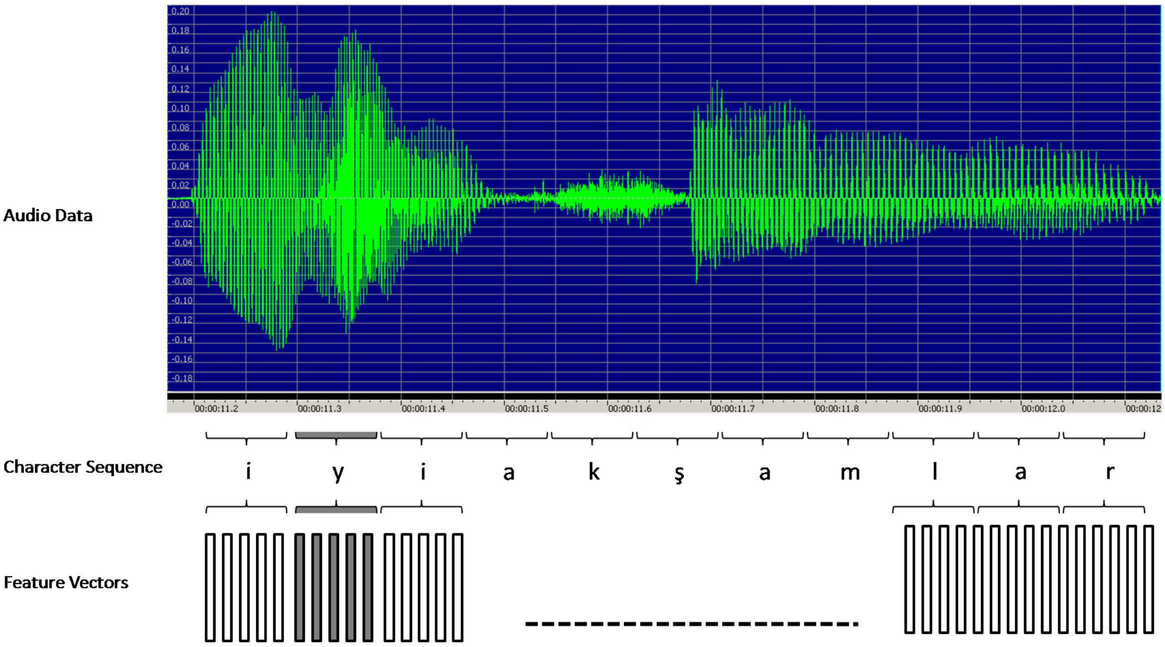


Figure 5.4. Uniform segmentations

The bootstrap model is a context independent model. The following lexicon in Figure 5.5 is used as the training lexicon. Training lexicon accepts all the possible phones in the phone list and reflects the input to the output.

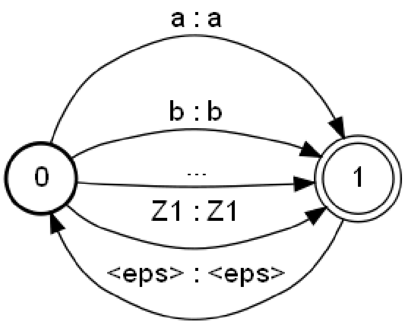


Figure 5.5. Automatic speech recognizer training lexicon

The initial model is trained with those uniformly divided segmentations and we applied three iterations of Maximum Likelihood training to obtain the bootstrap model. The bootstrap model is only used for segmenting the training data automatically and building the acoustic models which are mentioned in the following chapters.

Figure 5.6 shows the bootstrap model segmentations. These segmentations are not uniformly divided and more realistic compared to the initial segmentations.

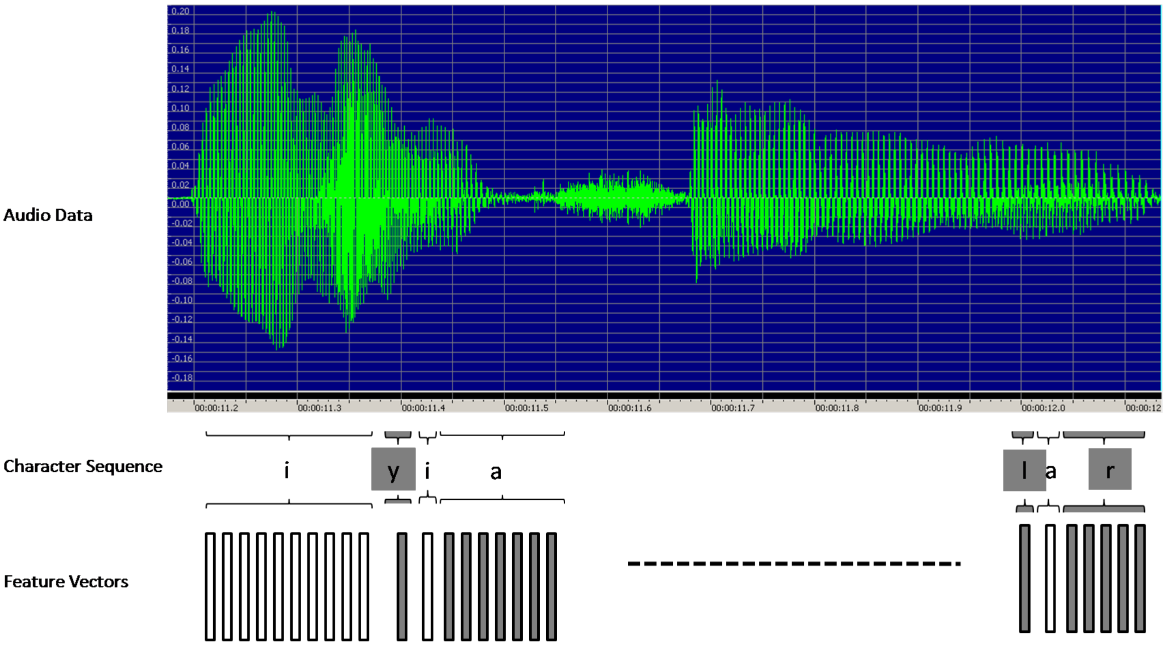


Figure 5.6. Bootstrap model segmentations

5.2.3. Lexicon and Language Modeling

Lexicon model and language model (also called grammar model) are essential parts of speech recognition systems. The composition of lexicon and language model ($L \circ G$) is used in recognition tasks. The lexicon is a dictionary which maps the words to phone sequences and the language model defines the word sequence probabilities calculated from the large text corpus by counting the word sequence frequencies.

Turkish orthography is almost phonetic and the pronunciation of a word can be completely identified by its spelling (phones). Therefore, the lexicon is created by mapping each word in the vocabulary to its phone sequences. Figure 5.7 illustrates the lexicon of some words in the vocabulary.

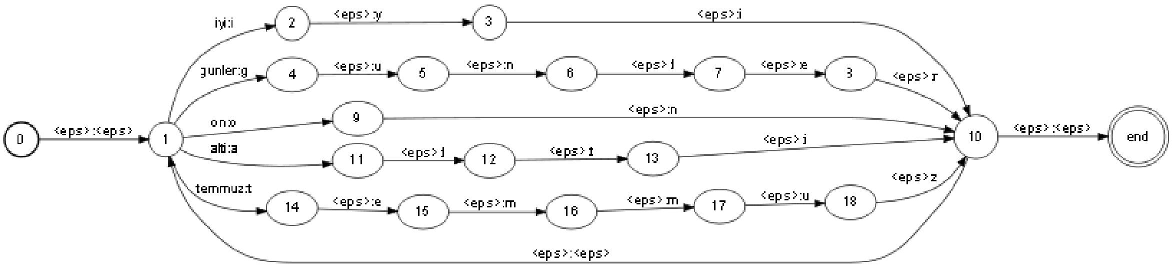


Figure 5.7. Lexicon Example

Table 5.1. Lexicon

Word	Phone
iyi	i-y-i
günler	g-ü-n-l-e-r
on	o-n
altı	a-l-t-ı
temmuz	t-e-m-m-u-z

We built three 3-gram Kneser-Ney smoothed language models from the *NewsCor*, including 50K, 100K and 200K words. The composition of lexicon and language model, $L \circ G$ is taken in account during the recognition. $L \circ G$ accepts phones which are the output of the Acoustic Model and output of $L \circ G$ is the recognition results, words.

The words that are introduced to the recognizer but are not found in the vocabulary are called the out-of-vocabulary (OOV) words. Due to the agglutinative nature of Turkish, the OOV word ratio is very large. If a word is OOV word, the word is not included neither in lexicon nor language model, thus, the recognizer has no chance to recognize the word correctly. Thus, OOV rate defines the performance limit of the automatic speech recognition system. Therefore, the coverage of the language model is a very important metric for recognition performance.

The OOV rates of the language models, 50K - `LM1`, 100K - `LM1` and 200K - `LM1` which are built from the *NewsCor* are shown in the first, second and third rows of Table 5.2. The OOV rate decreases when the coverage of language model increases.

The words in the audio training data set more probably occur in the audio test set than the words in the *NewsCor*. Therefore, we build another three set of language models which contain the training video set sliding text recognition outputs in its vocabulary. The fourth, fifth and sixth rows of Table 5.2 shows the OOV rates of the expanded language models, 50K - `LM2`, 100K - `LM2`, 200K - `LM2`. We obtained 1% reduction in the OOV rate of 50K - `LM2`. However the same effect was not observed in 200K - `LM2`.

5.3. Unsupervised Acoustic Model (U-AM)

5.3.1. U-AM Training

We used the bootstrap model to initiate the unsupervised acoustic model training. The audio training set including 83 files is segmented by the bootstrap model and

Table 5.2. OOV rates of language models

Language Model	OOV Rate
50K - LM1	7.5%
100K - LM1	3.9%
200K - LM1	2.2%
50K - LM2	6.6%
100K - LM2	3.7%
200K - LM2	2.1%

aligned with automatic transcriptions. The lexicon shown in the Figure 5.5 is used as the automatic speech recognizer training lexicon. We applied four iterations of EM algorithm and built a context dependent unsupervised acoustic model.

5.3.2. U-AM Performance

The system performance is examined by investigating the WER change due to the RTF. The word error rate and the real time factor are defined by the Equation 2.15 and Equation 2.16. We provide recognition on the `audio test set` with different pruning beam values. If pruning beam increases, the number of paths in the lattice increases, therefore, WER probably decreases. In contrast, a higher value of pruning beam increases the number of operations, hence the real time factor increases.

The performance of the unsupervised acoustic model with 50K–LM1, 100K–LM1, 200K–LM1 language models is illustrated in Figure 5.8. The recognition performance converges a constant value and after a certain point, pruning beam increase yields a little improvement in recognition performance but a higher increase in RTF.

Recognition with 200K language model gives the best recognition performance (12.7%).

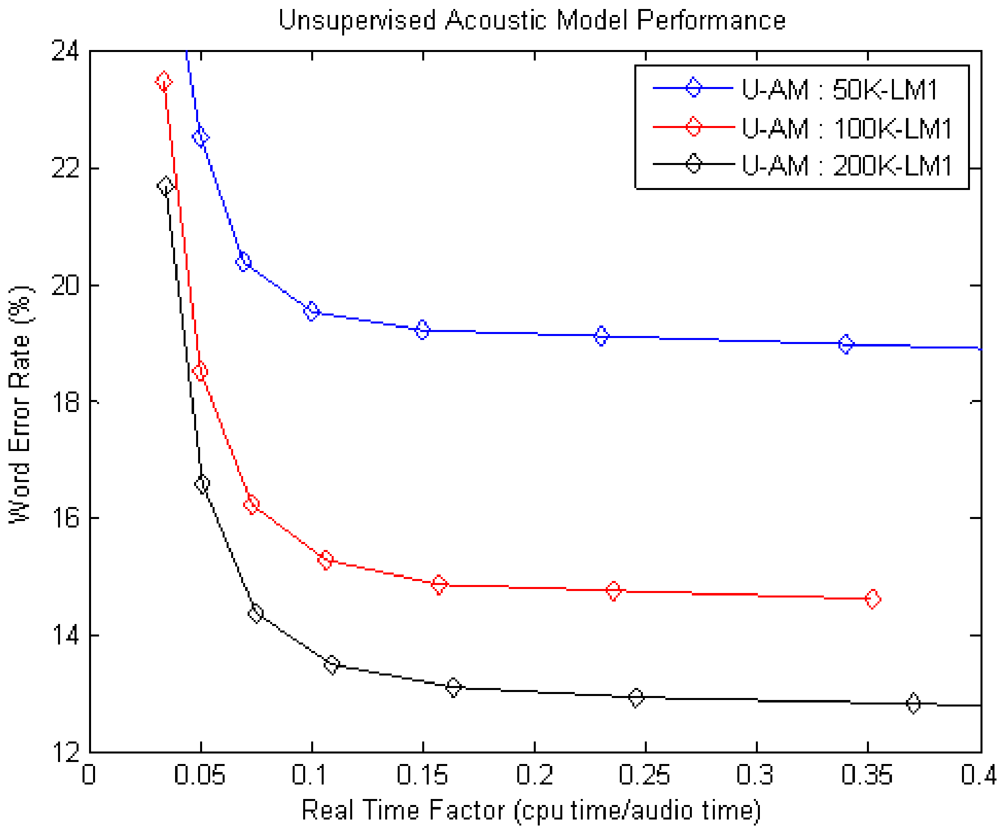


Figure 5.8. Unsupervised acoustic model performance with 50K–LM1, 100K–LM1, 200K–LM1 language models

We then investigate the effect of OOV rate on recognition performance. As mentioned before, we decreased the OOV rates of LM–2 set language models by adding the words in audio train set sliding text recognition outputs to the language model vocabulary. We perform recognition on audio test set with the language models 50K–LM2, 100K–LM2, 200K–LM2. The recognition performance with 50K–LM2, 100K–LM2, 200K–LM2 language models is shown in Figure 5.9. We obtained 1% word error rate performance increase with 50K–LM2 language model. However the same performance increase could not been observed with 100K–LM2 and 200K–LM2 language models. Because 100K–LM2 and 200K–LM2 language models' vocabularies already contain almost all of the words which exist in the automatic transcriptions of the audio training set. The recognition performance of unsupervised acoustic model with the language model sets LM–1 and LM–2 are stated in Table 5.3.

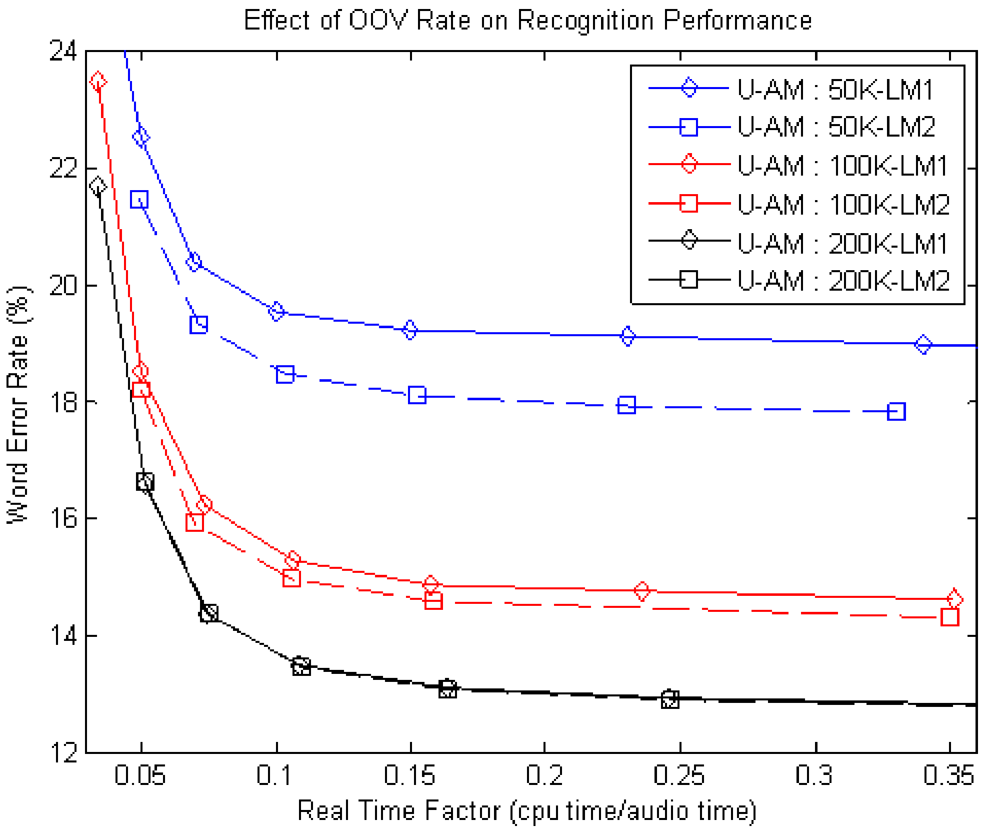


Figure 5.9. Effect of OOV rate on recognition performance

5.4. Performance Comparison of Unsupervised v.s. Supervised AM

We build an unsupervised acoustic model and perform recognition with different language models. In this experiment we compare the performance of the unsupervised acoustic model with the supervised acoustic model.

We segmented and labeled the training audio data manually. The supervised acoustic model is built by those manually prepared training data. The same training and recognition procedure is followed as done in unsupervised acoustic model training and recognition. The bootstrap model is used to segment the audio training set features. We then apply four iterations of EM algorithm and obtain the supervised acoustic model.

We perform recognition in audio test set with 50K–LM1, 100K–LM1, 200K–

Table 5.3. Effect of OOV rate in recognition performance

Language Model		**OOV Rate (%)**	**WER (%)**
50K	LM1	7.5	18.8
	LM2	6.6	17.8
100K	LM1	3.9	14.6
	LM2	3.7	14.3
200K	LM1	2.2	12.7
	LM2	2.1	12.7

LM1 language models. The performance comparison of unsupervised and supervised acoustic models is illustrated in Figure 5.10. The unsupervised acoustic model works a little faster when compared to the unsupervised model. With 50K–LM1, the supervised acoustic model and the unsupervised acoustic model converges the same word error rate of 18.8%. The word error rates of 14.2% and 12.3% are obtained by supervised acoustic model performance test with 100K–LM1, 200K–LM1 language models. As stated in Table 5.4, supervised acoustic model performs only 0.4% better than the unsupervised one for 200K–LM1. The performance comparison graph of U-AM and S-AM is illustrated in Figure 5.10.

Table 5.4. Performance comparison table: unsupervised v.s. supervised acoustic model

Acoustic Model	**WER (%)**		
	50K-LM1	100K-LM1	200K-LM1
U-AM	18.8	14.6	12.7
S-AM	18.8	14.3	12.3

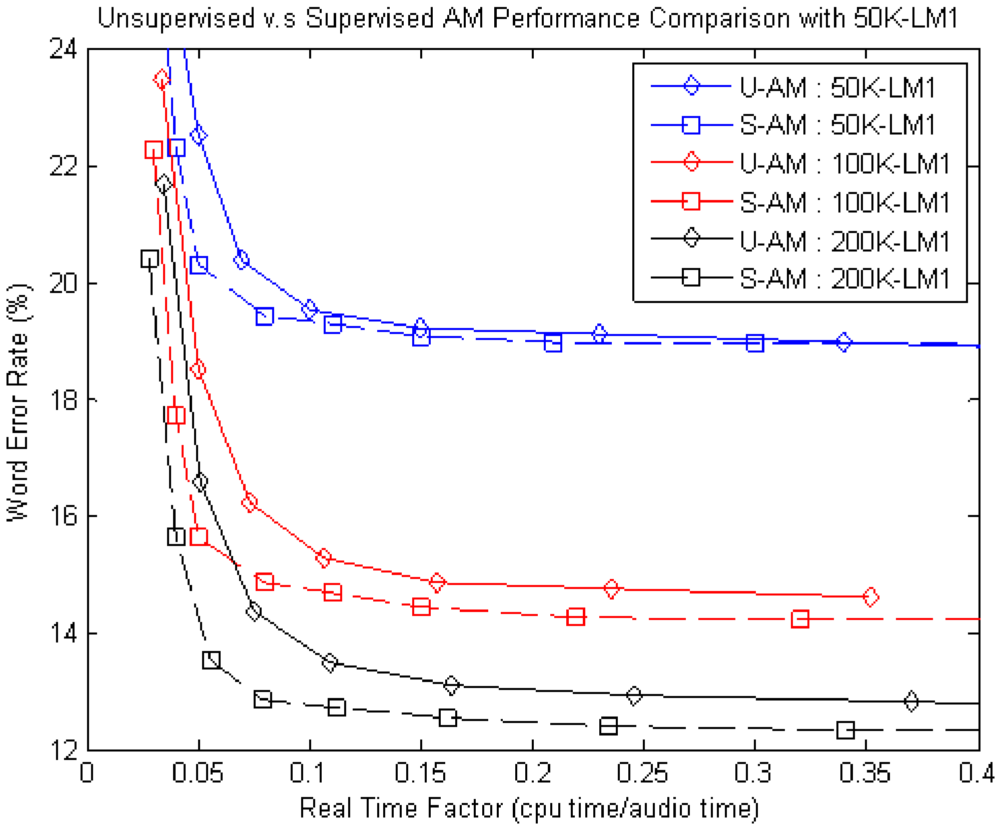


Figure 5.10. Performance comparison: unsupervised v.s. supervised acoustic model

6. CONCLUSION

Automatic speech recognition (ASR) systems require large amount of training data. It is easy to find audio material (CDs, broadcast news or Internet..) However, audio materials should be prepared before training. It is a tedious work to segment and transcribe each utterance at word level. In literature there exist studies to reduce the data preparation effort for ASR systems. This study addresses the speech data preparation problem.

In the broadcast news for the hearing impaired, the information is conveyed in three modalities; speech, sliding video text and sign language. Sliding video text recognition (SVTR) is an easy task and admits automatic transcription generation for ASR and sign language data preparation.

We build a SVTR system to make use of untranscribed audio and sign language video data for training. The front end of the SVTR system consists of sliding text band detection, video image extraction and feature extraction. We applied speech processing techniques to model and recognize the sliding video text. The baseline glyph model performance is increased by imposing a character based language model to the system (2.2% → 1.5%). Semi-supervised training method is applied to make use of untranscribed video data for training and 32% relative word error reduction is achieved. As a result our language independent SVTR system performs with under 1% word error rate.

The main contribution of this thesis is that; we bootstrap an unsupervised acoustic model by using the sliding video text in broadcast news for the hearing impaired. Without any manual data preparation, the transcriptions are generated by the proposed SVTR system at 0.9% word error rate. Audio data is segmented automatically and aligned with the automatic transcriptions. We obtained 12.7% word error rate with unsupervised acoustic model with 200K–LM1. When we compare unsupervised acoustic model to the supervised one, supervised acoustic model yields only 0.4% performance increase in word error rate. The unsupervised acoustic model performs quite successful in comparison to the supervised acoustic model performance.

Future work for the SVTR system includes increasing the variety of training and test data, expanding the feature set and multiple frame integration to obtain more robust SVTR system. Adaptation may be applied to recognize different fonts and character sizes. Instead of using only pixel values of the image, additional features may be added to the feature set. In addition, the multiple frame hypothesis may be taken in account to improve the recognition performance. The automatic transcriptions generated by our SVTR system, also can be used for data preparation of sign language recognition system or for broadcast news retrieval applications. In order to increase the ASR performance, morph-based language model may be used. The unsupervised acoustic model performs quite successfully for clean speech. Adaptation may be applied to improve performance for noisy speech data sets.

APPENDIX A: SYNTHETIC DATA SET

As mentioned in Chapter 4.3, a synthetic character set including only one sample of each character is prepared manually. The font of character images are selected from a similar font to sliding text. The character list is shown in Table A.1. The number of states are assigned according to the character image duration (width)w_c.

The characters are segmented automatically by dividing the image duration by the number of states. The state means and artificial (unity) variances are assigned as initial Gaussian model parameters. This model is used to bootstrap the actual baseline model. The training data is segmented with bootstrap model and baseline model is training is started with these segmentations.

Table A.1. Synthetic data states and durations

Characters	**States**	**Duration**	**Characters**	**States**	**Duration**
A	3	13	U	3	12
B	3	12	Ü	3	12
C	3	12	V	3	13
Ç	3	12	V	3	21
D	3	12	Y	3	14
E	3	11	Z	3	11
F	3	10	1	1	7
G	3	13	2	3	10
Ğ	3	13	3	3	10
I	1	4	4	3	10
İ	1	4	5	3	10
J	3	10	6	3	10
K	3	13	7	3	10
L	3	10	8	3	10
M	3	13	9	3	10
N	3	12	0	3	10
O	3	13	.	1	4
Ö	3	13	,	1	4
P	3	11	’	1	4
R	3	13	:	1	4
S	3	11	-	1	6
Ş	3	11	;	1	4
T	3	12	?	3	6
			cs	1	2
			ws	1	7
			ps	1	120

APPENDIX B: SVTR SYSTEM PARAMETER OPTIMIZATION

In this set of experiments, we optimize the system parameters ,duration multiplier *dur* and grammar multiplier *grm*, over `SVTR development set`. We perform a grid test $\forall dur, grm = [3, 5, 7, 9, 11, 13]$. Recognition tests $\forall dur, grm = [3, 5, 7, 9, 11, 13]$ are performed for each n-gram language models $\forall n = [3, 4, 5, 6, 7]$. The performance results are shown in the following Tables B.1,B.2,B.3,B.4,B.5

Table B.1. Duration and grammar multiplier parameter optimization of SVTR for 3-gram language model

3-gram Tests						
dur/grm	3	5	7	9	11	13
3	2.141	1.925	1.886	2.083	2.593	3.477
5	2.161	1.866	1.807	1.650	1.768	2.181
7	2.299	1.650	1.690	1.513	**1.493**	1.788
9	3.733	1.788	1.532	1.591	1.552	1.611
11	7.682	2.299	1.749	1.552	1.572	1.690
13	10.059	2.809	1.945	1.611	1.572	1.729

The duration and grammar multiplier values which minimize the WER are found as 9 and 11 respectively. These values are kept constant and are used for all SVTR performance tests.

Table B.2. Duration and grammar multiplier parameter optimization of SVTR for 4-gram language model

4-gram Tests						
dur/grm	3	5	7	9	11	13
3	2.043	1.807	1.631	1.552	1.729	1.984
5	2.122	1.709	1.552	1.415	1.454	1.591
7	2.141	1.572	1.454	1.336	1.356	1.454
9	3.399	1.493	1.395	1.336	1.277	1.316
11	7.289	1.729	1.415	**1.198**	1.198	1.316
13	9.528	1.965	1.454	1.277	1.277	1.336

Table B.3. Duration and grammar multiplier parameter optimization of SVTR for 5-gram language model

5-gram Tests						
dur/grm	3	5	7	9	11	13
3	1.827	1.552	1.434	1.395	1.611	1.886
5	1.945	1.493	1.415	1.198	1.277	1.454
7	2.063	1.434	1.336	1.218	1.238	1.257
9	3.084	1.415	1.238	1.159	1.159	1.297
11	6.994	1.473	1.316	1.179	**1.081**	1.198
13	9.371	1.650	1.356	1.139	1.139	1.257

Table B.4. Duration and grammar multiplier parameter optimization of SVTR for 6-gram language model

6-gram Tests						
dur/grm	3	5	7	9	11	13
3	1.749	1.473	1.316	1.395	1.434	1.552
5	1.788	1.513	1.336	1.081	1.159	1.297
7	2.004	1.434	1.316	1.159	1.100	1.139
9	3.084	1.493	1.297	1.139	**1.041**	1.100
11	7.014	1.454	1.297	1.159	1.081	1.139
13	9.450	1.572	1.336	1.159	1.159	1.179

Table B.5. Duration and grammar multiplier parameter optimization of SVTR for 7-gram language model

7-gram Tests						
dur/grm	3	5	7	9	11	13
3	1.729	1.513	1.375	1.356	1.434	1.631
5	1.749	1.552	1.356	1.179	1.238	1.434
7	1.925	1.473	1.356	1.218	**1.100**	1.218
9	2.986	1.454	1.277	1.179	1.120	1.179
11	6.935	1.513	1.356	1.238	1.139	1.198
13	9.411	1.552	1.316	1.198	1.120	1.159

APPENDIX C: MINIMUM EDIT DISTANCE

Minimum edit distance (Levenshtein distance) is a metric for measuring the amount of difference between two sequences. It calculates the cheapest way to transform one sequence to another. It often used in applications that need to determine how similar, or different the two sequence are like in case of spell checkers or DNA sequence alignments.

The cost is the number of operations to transform a sequence to another, where an operation is,

- An insertion (i), to supplement an element to the sequence
- A deletion (d), to remove an element from the sequence
- A substitution (s), to replace an element with another element.

The algorithm for computing minimum edit distance is as follows,

```
$m$ = size of the first string $s1$ , $n$ = is the size of the second string $s2$
for $i$ from 0 to $m$ do
    $d[i, 0] = i$ deletions
end for
for $j$ from 0 to $n$ do
    $d[0, j] = j$ insertions
end for
for $j$ from 0 to $n$ do
    for $i$ from 0 to $m$ do
        if $s1[i] = s2[j]$ then
            $d[i, j] = d[i - 1, j - 1]$
        else
            $d[i, j] = min\{d[i - 1, j] + 1, d[i, j - 1] + 1, d[i - 1, j - 1] + 1\}$
        end if
    end for
end for
return d[m,n]
```

The distance table for the two example strings $s1$ = "BU AKŞAM" and $s2$ = "8U AŞANN" is shown in Table C.1. The minimum path is highlighted. There are 2 substitutions ($B \rightarrow 8$, $M \rightarrow N$), 1 deletion (K) and 1 insertion (N) in the example. The $d[m,n]^{th}$ value is total number of operations which is 4 for this example. So the CER is calculated as 50% by Equation C.1.

$$e = \frac{(i + d + s)}{N} \tag{C.1}$$

Table C.1. Distance table

		8	U		A	Ş	A	N	N
	0	1	2	3	4	5	6	7	8
B	1	**1**	2	3	4	5	6	7	8
U	2	2	**1**	2	3	4	5	6	7
	3	3	2	**1**	2	3	4	5	6
A	4	4	3	2	**1**	2	3	4	5
K	5	5	4	3	**2**	2	3	4	5
Ş	6	6	5	4	3	**2**	3	4	5
A	7	7	6	5	4	3	**2**	**3**	4
M	8	8	7	6	5	4	3	3	**4**

REFERENCES

1. Garris M. D. , J. L. Blue, G. T. Candela, P. J. Grother, S. A. Janet, and C. L. Wilson, “Nist form-based handprint recognition system (release 2.0),” 1997.

2. Abbyy Inc. “FineReader 10”, available: “http://www.abbyy.com/.” [accessed: November 2009]

3. Google Inc., “Google Audio Indexing (Gaudi)”, available: “http://labs.google.com/gaudi.”, [accessed: November 2009]

4. Aran O., I. Ari, L. Akarun, E. Dikici, S. Parlak, M. Saraclar, P. Campr, and M. Hruz, “Speech and sliding text aided sign retrieval from hearing impaired sign news videos,” *Journal on Multimodal User Interfaces*, vol. 2, no. 2, pp. 117–131, 2008.

5. Arisoy E., D. Can, S. Parlak, H. Sak, and M. Saraclar, “Turkish broadcast news transcription and retrieval,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 17, pp. 874–883, July 2009.

6. Gutenberg Project, “Free e-book Archive”, avaliable: “www.gutenberg.org”, [accessed: November 2009]

7. Sestek Ses ve İletişim Teknolojileri A.Ş., “Konuşma Sentezi Uygulaması”, available: “www.sestek.com.tr.”, [accessed: November 2009]

8. Turkuvaz Gazete Dergi Basım A.Ş., “Sesli Haber Sistemi” available: “www.sabah.com.tr.”, [accessed: November 2009]

9. Parlak S., “Speech retreival for turkish broadcast news,” Master’s thesis, Boğaziçi University, 2009.

10. Tauschek G., “Reading machine u.s. patent 2026329. dec.,” 1935.

11. Handel P., “Statistical machine u.s. patent 1915993. june.,” 1933.

12. El-Sheikh T. S. and R. M. Guindi, “Computer recognition of arabic cursive scripts,” *Pattern Recogn.*, vol. 21, no. 4, pp. 293–302, 1988.

13. Mori S., K. Yamamoto, and M. Yasuda, “Research on machine recognition of handprinted characters,” vol. 6, pp. 386–405, July 1984.

14. Suen C., M. Berthod, and S. Mori, “Automatic recognition of handprinted characters: The state of the art,” vol. 68, pp. 469–487, April 1980.

15. Belaid A.and J. Haton, “A syntactic approach for handwritten mathematical formula recognition,” vol. 6, pp. 105–111, January 1984.

16. Shridhar M. and A. Badreldin, “A high-accuracy syntactic recognition algorithm for handwritten numerals,” vol. 15, pp. 152–158, 1985.

17. Mori S., C. Y. Suen, and K. Yamamoto, “Historical review of ocr research and development,” pp. 244–273, 1995.

18. Al-Badr B. and S. A. Mahmoud, “Survey and bibliography of arabic optical text recognition,” *Signal Process.*, vol. 41, no. 1, pp. 49–77, 1995.

19. Allam M., “Segmentation versus segmentation-free for recognizing Arabic text,” in *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series* (L. M. Vincent & H. S. Baird, ed.), vol. 2422 of *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, pp. 228–235, Mar. 1995.

20. Bose C. B. and S.-S. Kuo, “Connected and degraded text recognition using hidden markov model,” *Pattern Recognition*, vol. 27, no. 10, pp. 1345 – 1363, 1994.

21. Cho W., S.-W. Lee, and J. H. Kim, “Modeling and recognition of cursive words with hidden markov models,” *Pattern Recognition*, vol. 28, no. 12, pp. 1941 – 1953, 1995.

22. Bunke H., M. Roth, and E. G. Schukat-Talamazzini, "Off-line cursive handwriting recognition using hidden markov models," *Pattern Recognition*, vol. 28, no. 9, pp. 1399 – 1413, 1995.

23. Mohamed M. and P. Gader, "Handwritten word recognition using segmentation-free hidden markov modeling and segmentation-based dynamic programming techniques," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 18, pp. 548–554, May 1996.

24. Aas K. and L. Eikvil, "Text page recognition using grey-level features and hidden markov models," *Pattern Recognition*, vol. 29, no. 6, pp. 977 – 985, 1996.

25. Kornai A., "An experimental hmm-based postal ocr system," in *Acoustics, Speech, and Signal Processing, 1997. ICASSP-97., 1997 IEEE International Conference on*, vol. 4, pp. 3177–3180, Apr 1997.

26. Schwartz R., C. LaPre, J. Makhoul, C. Raphael, and Y. Zhao, "Language-independent ocr using a continuous speech recognition system," in *Pattern Recognition, 1996., Proceedings of the 13th International Conference on*, vol. 3, pp. 99–103 vol.3, Aug 1996.

27. LaPre C., Y. Zhao, C. Raphael, R. Schwartz, and J. Makhoul, "Multi-font recognition of printed arabic using the bbn byblos speech recognition system," in *ICASSP '96: Proceedings of the Acoustics, Speech, and Signal Processing, 1996. on Conference Proceedings., 1996 IEEE International Conference*, (Washington, DC, USA), pp. 2136–2139, IEEE Computer Society, 1996.

28. Bazzi I., C. LaPre, J. Makhoul, C. Raphael, and R. M. Schwartz, "Omnifont and unlimited-vocabulary ocr for english and arabic," in *ICDAR '97: Proceedings of the 4th International Conference on Document Analysis and Recognition*, (Washington, DC, USA), pp. 842–846, IEEE Computer Society, 1997.

29. Sato T., T. Kanade, E. K. Hughes, M. A. Smith, and S. Satoh, "Video ocr: indexing

digital new libraries by recognition of superimposed captions," *Multimedia Syst.*, vol. 7, no. 5, pp. 385–395, 1999.

30. Lienhart R. and W. Effelsberg, "Automatic text segmentation and text recognition for video indexing," *Multimedia Syst.*, vol. 8, no. 1, pp. 69–81, 2000.

31. Wu C., L. C. On, C. H. Weng, T. S. Kuan, and K. Ng, "A macao license plate recognition system," in *Machine Learning and Cybernetics, 2005. Proceedings of 2005 International Conference on*, vol. 7, pp. 4506–4510, Aug. 2005.

32. Ahmed M., M. Sarfraz, A. Zidouri, and W. Al-Khatib, "License plate recognition system," in *Electronics, Circuits and Systems, 2003. ICECS 2003. Proceedings of the 2003 10th IEEE International Conference on*, vol. 2, pp. 898–901, Dec. 2003.

33. Zidouri A. and M. Deriche, "Recognition of arabic license plates using nn," in *Image Processing Theory, Tools and Applications, 2008. IPTA 2008. First Workshops on*, pp. 1–4, Nov. 2008.

34. Huang Y.-P., S.-Y. Lai, and W.-P. Chuang, "A template-based model for license plate recognition," in *Networking, Sensing and Control, 2004 IEEE International Conference on*, vol. 2, pp. 737–742, 2004.

35. Mohri M., "Edit-distance of weighted automata," in *In Jean-Marc Champarnaud and Denis Maurel, editor, Seventh International Conference, CIAA 2002*, pp. 1–23, Springer Verlag, 2002.

36. Prasad R., S. Saleem, E. MacRostie, P. Natarajan, and M. Decerbo, "Multi-frame combination for robust videotext recognition," in *Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on*, pp. 1357–1360, 31 2008-April 4 2008.

37. Zavaliagkos G. and T. Colthurst, "Utilizing untranscribed training data to improve performance," in *DARPA Broadcast News Transcription and Understanding*

Workshop, Landsdowne, pp. 301–305, 1998.

38. Lamel L., J.-L. Gauvain, and G. Adda, "Unsupervised acoustic model training," in *Acoustics, Speech, and Signal Processing, 2002. Proceedings. (ICASSP '02). IEEE International Conference on*, vol. 1, pp. 877–880, 2002.

39. Wang L., M. Gales, and P. Woodland, "Unsupervised training for mandarin broadcast news and conversation transcription," in *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, vol. 4, pp. IV–353–IV–356, April 2007.

40. Zavaliagkos G., J. McDonough, D. Miller, A. El-Jaroudi, J. Billa, F. Richardson, K. Ma, M. Siu, and H. Gish, "The bbn byblos 1997 large vocabulary conversational speech recognition system," in *Acoustics, Speech and Signal Processing, 1998. Proceedings of the 1998 IEEE International Conference on*, vol. 2, pp. 905–908 vol.2, May 1998.

41. Kemp T. and A. Waibel, "Unsupervised training of a speech recognizer: recent experiments," in *Proc. Europeech 99*, pp. 2725–2728, September 1999.

42. Placeway P. and J. Lafferty, "Cheating with imperfect transcripts," in *Spoken Language, 1996. ICSLP 96. Proceedings., Fourth International Conference on*, vol. 4, pp. 2115–2118, Oct 1996.

43. Jurafsky D. and J. H. Martin, *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics and Speech Recognition (Prentice Hall Series in Artificial Intelligence)*. Prentice Hall, 2 ed., February 2000.

44. Rabiner L., "A tutorial on hmm and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, pp. 257–286, February 1989.

45. Forney J., G.D., "The viterbi algorithm," *Proceedings of the IEEE*, vol. 61, pp. 268–278, March 1973.

46. Baum L. E. , T. Petrie, G. Soules, and N. Weiss, “A maximization technique occurring in the statistical analysis of probabilistic functions of markov chains,” *The Annals of Mathematical Statistics*, vol. 41, no. 1, pp. 164–171, 1970.

47. Wagner R. A. and M. J. Fischer, “The string-to-string correction problem,” *J. ACM*, vol. 21, no. 1, pp. 168–173, 1974.

48. Mohri M., “Finite-state transducers in language and speech processing,” *Comput. Linguist.*, vol. 23, no. 2, pp. 269–311, 1997.

49. Rabiner L. R. and M. R. Sambur, “An algorithm for determining the endpoints of isolated utterances,” *Bell System Technical Journal*, vol. 54, no. 2, pp. 297–315, 1975.

50. Sak H., T. Güngör, and M. Saraçlar, “Turkish language resources: Morphological parser, morphological disambiguator and web corpus,” in *Proc. 6th International Conference on Natural Language Processing, GoTAL*, 2008.

51. Dikici E. and M. Saraclar, “Sliding text recognition in broadcast news,” in *Procedings of the 2008 IEEE Signal Processing, Communication and Applications Conference, SIU*, pp. 1–4, April 2008.

52. Prasad R., S. Saleem, E. MacRostie, P. Natarajan, and M. Decerbo, “Multi-frame combination for robust videotext recognition,” in *Proceedings of the 2008 IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP*, pp. 1357–1360.

53. Theodoridis S. and K. Koutroumbas, *Pattern Recognition*, pp. 270–272. Academic Press, 2nd ed., 2003.

54. Stolcke A., “SRILM – An extensible language modeling toolkit,” in *Proc. ICSLP*, vol. 2, (Denver), pp. 901–904, 2002.