TREN- TURKISH SPEECH RECOGNITION PLATFORM

Hasan Palaz, Alper Kanak, Yücel Bicil, Mehmet Uğur Doğan, Tuba İslam

National Research Institute of Electronics & Cryptology , The Scientific & Technical Research Council of Turkey
P.O. Box 74, 41470, Gebze, Kocaeli, Turkey
Phone: + (90) 262 6481350, Fax: + (90) 262 6481100, E-Mail: akustiklab@uekae.tubitak.gov.tr

ABSTRACT

TREN (Turkish Recognition ENgine) is a modular, HMM-based (Hidden Markov Model) and speaker-independent speech recognition system whose system software architecture is based on Distributed Component Object Model (DCOM). TREN contains specialized modules that allow a full interoperable platform including a Turkish speech recognizer, feature extractor, end-point detector and a performance monitoring module. TREN has basically two layers: First layer is the central server that distributes the recognition calls to the appropriate remote servers according to their current CPU load of the recognition process after some speech signal preprocessing and the second layer consists of the remote servers which performs the critical recognition task. This component-based architecture enables TREN applicable to distributed environments. TREN is also trained by considering a wide variety of very common words those best represent the Turkish language. In order to obtain a such database a very large word corpus is collected and statistically the widest span of triphones representing Turkish is examined. TREN has been used to assist speech technologies which require a modular and multithreaded recognizer with dynamic load sharing facilities.

1. INTRODUCTION

Since speech processing technologies have a wide variety of application area among Turkic countries with population more than 200 millions of people, developing a robust recognition engine for Turkish language, accepted as the *language of silk road*, is a promising field in the next generation information systems consisting Turkish-specific speech agents. When developing a speech recognition engine for Turkish, the recognition system must be adapted by means of vocabulary, acoustic parameters, language models and the dialog structure of Turkish.

On the other hand, with the rapid growth of distributed systems and speech technologies, distributed speech recognition (DSR) applications have become very attractive. Especially the increasing use of component objects in speech processing makes DSR compatible to Personal Computers (PCs), Personal Digital Assistants (PDAs), wired and wireless network architectures or even World Wide Web (WWW) applications [1, 2].

Recently, component-based software engineering has been very popular for the large-scale speech processing system development. DCOM (Distributed Component Object Model), which is one of the industrial criteria for component-based software development, is accepted as a robust software architecture that allows applications to be built from binary software components [3]. Language-independent, interoperable and location-transparent characteristics makes DCOM a widely used component software model in the world. Thus, this technique was rapidly utilized in designing software systems for various hot research topics and development tools including speech recognition systems. Remote communication established through the Distributed COM (DCOM) makes the design and implementation of a distributed speech processing system easy [3].

Generally there are three alternative strategies in the design of DSR architectures: server-only, client-only and client-server processing. In server-only processing, all speech processing are done at the server side. Most of the telephony speech applications or web-based speech recognition tools are good examples for server-only systems. In client-only strategy, most of the speech processing is done at the client side and the results are transmitted sequentially to the server. Since this approach requires powerful machines at the client side, it is not a practical solution for DSR systems. Client-only model also limits type of clients that are powerful enough to perform the complicated speech recognition. The client-server mechanism concentrates on simple and low power client devices which quantize and packetize the speech data (usually in the form of speech feature vectors) and transmit it over the communication channel to a remote speech recognition server [4, 5].

Due to the emphasized advantages of component objects and distributed systems, a multihtreaded speech recognition engine for Turkish with client-server processing, named TREN (Turkish Recognition ENgine), is proposed as an efficient solution especially for the telephony speech applications. TREN could also be adapted to a wide range of speech technologies including distributed speech recognition, TTS (Text-to-Speech) applications, Interactive Voice Response (IVR) systems, speech translators, etc. The backbone of TREN is designed by an effective and efficient statistical model known as Hidden Markov Models (HMM) [7] which is based on reliable structures to model human hearing system, and thus it is widely used for speech recognition and speaker identification problems. HMM requires a critical training period in order to perform the recognition task. The database used to train HMM-based TREN, a Turkish Telephony speech database -TURTEL (TURkish TELephony)- is collected and the most widely used triphones are examined among this set to obtain better recognition results as well as better utilization scores.

The organization of the paper is as follows: Section 2 discusses DCOM-based architecture of TREN and the specifications of modules and algorithms used in TREN. The information about TURTEL are given in Section 3. The recognition and load sharing performance of TREN are examined by giving detailed results in Section 4. Finally, Section 5 concludes the paper.

2. TREN MODULAR STRUCTURE

For the complex speech processing systems, a layered architecture which is a natural outgrowth of the client-server model, could be an effective solution concerning the problems such as lack of scalability and portability. Compared with the traditional client-server model, layered architecture of TREN offers a natural way to separate user interface from the background of the hard work performed by the recognizer and provide interoperability. This idea is adopted to design

TREN. The overall system architecture is depicted in Fig 1. Central server constitutes the first layer of the system which is subjected to apply some speech processing routines (feature extraction and end-point detection) to the audio files collected as an input from third party applications or environments. One of the most important roles of central server is authorizing a remote server with the least recognition process load as compared to the other remote servers all of which constitute the second layer of TREN. Once this authorization is accomplished the selected remote server will become ready to serve as a recognizer. This two-layered architecture allows remote servers work in a parallel and distributed manner. Note that this architecture also gives the flexibility to install or uninstall any number of machines according to the application requirements. TREN supports up to 64 simultaneous recognitions resembling a 64-channel system.

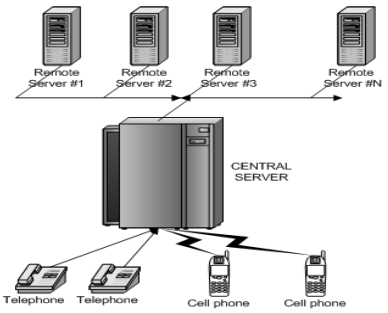

Figure 1: TREN layered structure.

2.1 Modules of TREN

To implement TREN, the facilities of DCOM are heavily considered and an interface-based software architecture is developed. If TREN is examined in a procedural point of view it could be separated into two folds: tasks running on the central server side and one of the remote servers side.

TREN is composed of 3 service-type modules waiting to be invoked by ORPCs (Object Remote Procedure Calls). These awaiting services are the shell of central and remote servers, Dynamic Load Monitoring Service (DLMS) which blinks by reporting the CPU load of recognition process on each remote server simultaneously. Additionally there exists 3 non-service-type modules: Feature Extraction Module (FEM), End-Point Detection Module (EPDM) and Recognition Engine abbreviated as TRENCore on which HMM algorithms are run.

As presented in Fig 2, the procedure begins with attaining of a raw file which will be processed and recognized at the central server side. The first front-end process applied to the pure raw file is cropping the non-speech regions of the waveform by using EPDM. By using FEM, the cropped speech data is then converted to a MFCC-based (Mel-Frequency Cepstral Coefficients) audio feature and stored in a file. Now the feature file to be recognized is ready to be sent to the least-loaded remote server. Meanwhile, each remote server computes its corresponding CPU load of recognition process. Central server collects current loads on remote servers by simple ORPCs and runs a selection algorithm to identify the least-loaded engine. After the least loaded remote server is determined, the feature file is sent to the selected server via an FTP link. This lucky server is invoked by a message and it is ensured to transmit the coming MFCC file. On the other hand, lexicon and word network structure which is antecedently obtained after a training procedure, is already loaded on each recognizer. The transmitted MFCC file is then recognized by TRENCore according to the lexicon and word network file stored as a distinct file. Finally, the recognition result is reported to the central server by an ORPC and the task is accomplished. By changing the lexicon file, the recognition engine could be adapted to a different recognition space.

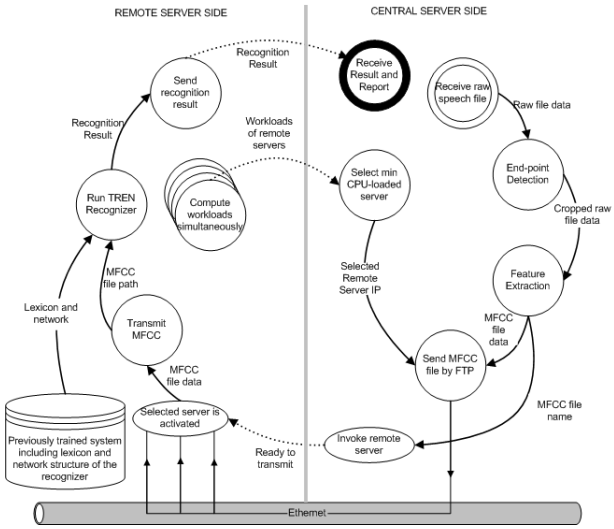

Figure 2: Data Flow Diagram of TREN.

TRENCore is based on a very common method of HMM which have been widely used to model various acoustic phenomena. The temporal characterization of an audio stream can successfully be modelled using an HMM structure where state transitions model temporal correlation and in each state Gaussian classifiers model signal characteristics. The forward-backward algorithm is used to reestimate both output and transition probabilities iteratively, and the Viterbi beam search algorithm is used during recognition to find out the most likely phoneme sequence [7].

In TREN, MFCC is used as the parameterization technique which has a superior performance with respect to other spectral or wave-structural features [7]. FEM which is installed to the central server separately is designed to extract audio features.

The endpoints of an isolated utterance is determined by EPDM which is installed to the central server separately. EPDM algorithm is implemented by applying energy, zero-crossing ratio and duration modelling parameters. In order to diminish the adverse effects of noisy conditions in which energy and zero-crossing approaches are insufficient, a state machine is applied to obtain a hybrid algorithm [8].

One of the most important components of TREN is DLMS, which ensures that no remote server remains idle when the other servers are heavily loaded. As reported in [9] since adaptive load sharing policies or so-called Dynamic Load Sharing (DLS) approaches react to the system state instead of estimating the average behavior of system, they perform better than the static load sharing policies. In TREN, a separate DLS module is designed and this module is installed as a service (DLMS) to each remote server to report the current CPU load of the recognition process simultaneously. DLMS uses the Performance Data Helper (PDH) library which is an enhancement to the performance-monitoring capabilities of Windows NT-based systems.

3. TURTEL SPEECH DATABASE

The triphones (context-dependent phones), are widely used units for the speech recognition tasks because of their capability to best represent the acoustic characteristics of speech. In this view, the TURTEL database is collected by considering statistically the widely used triphones in Turkish

language [6]. The probable number of triphones in Turkish is nearly 27000. Yet, this number of triphones inflates the search space and increases the database size needed to train the system. Thus triphones should be either clustered or reduced in number. In order to collect samples of TURTEL an assumption is made that there is a text group which models the target language extensively and the text group is used to extract the most frequent triphones. This method has two main advantages: First, since the vocabulary is built in control, phonetically balanced vocabularies can be achieved. Second, the used triphones can model the language extensively and no more clustering is needed. Since the most widely used triphones are selected, any tying of triphone states is not applied. Primary drawbacks of this method are based on the fact that the building period of this vocabulary is computationally loaded and requires more effort. Moreover, since the database is formed with read speech it may not model the target environment totally.

The procedure followed to collect TURTEL is as follows: Before the database collection, in order to determine the words that will form the database, a $2,2$ million-word corpus is examined and the most frequent 1000 triphones which span Turkish with 88% are selected. 25 female and 40 male speakers utter phonetically balanced 373 words and 15 sentences with keywords included in this reduced triphone set. These speakers' ages are in between 15 and 55 and they used 3 kinds of telephones: normal, GSM and hands-free.

4. EXPERIMENTAL RESULTS

In order to test the performance of TREN, some adverse conditions are simulated by applying various background noises to the test set of TURTEL and the resulting recognition performances are measured in terms of true recognition rates.

Training of TREN is composed of 2 stages. At the first stage the recognition engine is trained by considering the whole database and this whole set is used again as the test set to find out the outlier speakers. After the speakers with an average personal word error rate are specified, they have been used to find out the system's "real" performance. Although this is a laborious task, it gives the most realistic results with the prepared database. The speakers with the best and the worst rate are eliminated from the test set. At the second stage 5 male and 5 female speakers whose recognition rates are at the nearest neighborhood of average recognition rate are discarded from the whole database and used as the test set. The utterances acquired from the rest of the speakers (35 male and 20 female) is then used for the second training stage. Note that each speaker utters 373 words which are widely used in Turkish telephony speech. All these utterances are collected at the acoustics laboratory of TÜBİTAK-UEKAE (National Research Institute of Electronics & Cryptology, The Scientific & Technical Research Council of Turkey).

In TREN, the temporal characterization of the audio is performed by HMM structures where each triphone is represented by a 3-state left-to-right HMM structure. The number of mixtures used in tied mixture pool is 512. The audio stream has 8 kHz sampling rate. The acoustic noises, which are added to the speech signal to observe the identification performance under adverse conditions, are picked to be office, in-car, white and pink noise. The audio stream is processed over 10 msec frames centered on 20 msec Hamming window. The MFCC feature vector is formed from 13 cepstral coefficients including the 0th gain coefficient using 26 mel frequency bins. The resulting audio feature vector of size 26, includes the MFCC vector along with the first delta MFCC vectors.

4.1 Recognition Performance

In order to examine the recognition performance of TREN, 4 noisy conditions (office, in-car, white and pink noise) and 2 triphone statistical models (continuous or untied and tied-mixture) are considered. In experiments the acoustic noise is applied in three different levels (6dB SNR, 12dB SNR and Quiet). The quiet conditions may also include minor telephony speech environmental adverse effects. The true recognition rates for each test are presented in Table 1.

		Noise		
Statistical Model	Noise Type	Quiet	12 dB SNR	6 dB SNR
Continuous	Office	96.76	68.46	35.66
	In-car		85.04	78.01
	White		54.19	33.71
	Pink		60.13	30.04
Tied-mixture	Office	93.49	65.32	27.44
	In-car		81.70	70.35
	White		60.90	36.57
	Pink		61.65	27.03

Table 1: Recognition performance under adverse conditions.

The results have shown that as the noise level increases the recognition performance significantly decreases. The best recognition rates are obtained under quiet conditions as expected. However, in-car conditions does not significantly affect the overall recognition as compared with the other noisy conditions. Thus, it could be said that TREN performs well for both standard telephony speech and mobile communication purposes. On the other hand, the continuous model performs better than the tied-mixture model in all cases except the case in which white noise is applied.

4.2 Load Sharing Performance

In order to test the load sharing performance of TREN, three remote servers are used (RS1 and RS3 are two identical machines with Intel Pentium 4-1.4 Ghz CPU and 512 MB RAM but RS2 is a weaker one with Pentium 3-0.8 GHz CPU and 256 MB RAM). As mentioned before, since recognition process CPU (abbreviated as PCPU) load is the dominating load on an idle processor, it is measured as the percentage of elapsed time that all of the threads of recognition process used the processor to execute instructions. The main purpose of simulations is to present the gain of load sharing strategy of TREN by comparing the cases with 3 remote servers versus the cases with a single server. In experiments, a fixed number of audio files (300) each of which takes approximately 1 second in length is used. The tests consist of four cases considering the combination of two statistical models, tied-mixture and continuous, and two different network sizes with $N = 373$ and $N = 40$ words.

The experiments have shown that using multiple remote servers significantly decreases the total recognition time of 300 audio files in all cases. As depicted in Fig. 3, the total recognition time of multi-server case is less than half of the single case. On the other hand, the network size significantly influences the total recognition time as expected. As the network size increases the search space increases, and this makes the recognition process more complex. For larger network sizes, the tied-mixture model performs better than the continuous model. However, due to its preprocessing stage where the tied-mixture triphone pool is constructed the tied-mixture model is not efficient for small network sizes in contrast with the continuous model (See Fig. 3).

The average and standard deviation of PCPU usage for each remote server is depicted in Fig. 4. According to the mean and standard deviation values, RS1 and RS3 often

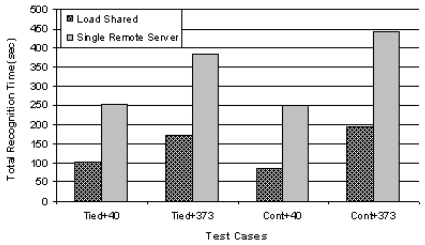

Figure 3: Total recognition time in seconds for the single and multiple usage of remote servers.

share a large portion of recognition load while RS2 only answers the urgent recognition calls. RS2 is recoursed only when RS1 and RS3 are heavily loaded. Since the last case where continuous model with $N = 373$ network size is used is very complex and requires much computation, the load on RS2 significantly increases. On the other hand, the differences between the overall PCPU load of two identical remote servers could be explained by the least-loaded server selection algorithm of the central server. According to the selection algorithm, when PCPU load of two or more remote servers are same, last component in the remote server list is selected greedily.

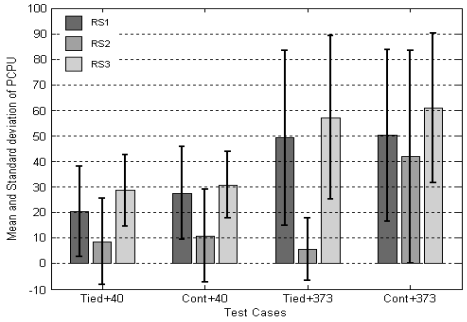

Figure 4: Mean and standard deviation of PCPU usage for each remote server.

5. CONCLUSION

TREN, presented in this paper, is a powerful recognition system in three folds. First, TREN is a reliable recognition engine with its high accuracy results. The experimental studies have shown that TREN performs very well for both tied-mixture and continuous models in various acoustical conditions under different noisy circumstances. Second, the database used to train the HMM-based recognizer best models the Turkish speech by considering the widely triphones. Third, DCOM-based modular architecture of TREN is an object oriented and component based approach which enables the system work on distributed environments. The two-layered architecture of TREN including a central server and remote servers forms a flexible structure which guarantees to serve whenever a problem occurs in any of the remote servers. TREN is designed in a DCOM-based manner, any update or development in components could easily be integrated to the system with a little effort.

In conclusion, TREN is a robust and reliable Turkish speech recognition engine which could be integrated in or with most of the speech applications such as IVR, TTS, authentication systems, etc. As a further study, TREN might be revisioned to respond faster as the time-critic real life applications are becoming more popular. The recognition engine could also be improved to get lower error rates for larger databases.

REFERENCES

[1] D. Goddeau, "Deploying Speech Application over the Web," *Proceedings of Eurospeech*, pp. 685-688, Rhodes, Greece, September 1997.

[2] M. Sokolov, "Speaker Verification over the World Wide Web," *Proceedings of Eurospeech*, pp. 847-850, Rhodes, Greece, September 1997.

[3] Microsoft Corp., "DCOM technical Overview," 1996.

[4] "The Aurora Project," announced at Telecom 95, http://gold.ity.int/TELECOM/wt95, Geneva, Oct.95.

[5] D. Stallard, "The BBN SPIN System," *Voice on the Net Conference*, Boston MA, September 1997.

[6] Ü. Yapanel, M.U. Doğan, L.M. Arslan, "Türkçe Anahtar Sözcük Yakalama Sistemi için farklı Atık Modellerinin Karşılaştırılması," *SIU proc.*, pp. 122-127, Gazimagusa, April 2001.

[7] L. Rabiner, B. Juang, *Fundamentals of Speech Recognition* P. Hall Signal Processing Series, N.Jersey, 1993.

[8] L.R. Rabiner, M.R. Sambur, "An algorithm for determining the endpoints of isolated utterances," *Bell Syst. Tech. J.*, vol. 54, pp. 297-315,Feb. 1975.

[9] H.D. Karatza, "A Comparison of Load Sharing and Job Scheduling in a Network Of Workstations," *Int. Jour. of Simulation: Systems, Science, Technology*, Vol. 4, No 3-4, pp. 4-11, 2003.