B. Raghavendhar Reddy, E. Mahender / International Journal of Engineering Research and Applications (IJERA) ISSN: 2248-9622 www.ijera.com
Vol. 3, Issue 1, January -February 2013, pp.253-258

Speech to Text Conversion using Android Platform

B. Raghavendhar Reddy[#1], E. Mahender[*2]
[#1]Department of Electronics Communication and Engineering
Aurora's Technological and Research Institute
Parvathapur, Uppal, Hyderabad, India.
2Asst. Professor
Aurora's Technological and Research Institute
Parvathapur, Uppal, Hyderabad, India.

Abstract
For the past several decades, designers have processed speech for a wide variety of applications ranging from mobile communications to automatic reading machines. Speech recognition reduces the overhead caused by alternate communication methods. Speech has not been used much in the field of electronics and computers due to the complexity and variety of speech signals and sounds. However, with modern processes, algorithms, and methods we can process speech signals easily and recognize the text. In this project, we are going to develop an on-line speech-to-text engine. The system acquires speech at run time through a microphone and processes the sampled speech to recognize the uttered text. The recognized text can be stored in a file. We are developing this on android platform using eclipse workbench. Our speech-to-text system directly acquires and converts speech to text. It can supplement other larger systems, giving users a different choice for data entry. A speech-to-text system can also improve system accessibility by providing data entry options for blind, deaf, or physically handicapped users. Voice SMS is an application developed in this work that allows a user to record and convert spoken messages into SMS text message. User can send messages to the entered phone number. Speech recognition is done via the Internet, connecting to Google's server. The application is adapted to input messages in English. Speech recognition for Voice uses a technique based on hidden Markov models (HMM - Hidden Markov Model). It is currently the most successful and most flexible approach to speech recognition.

Keywords: HMM, Android Os, DVM, Speech Recognition, Intents

I. INTRODUCTION

Mobile phones have become an integral part of our Everyday life, causing higher demands for content that can be used on them. Smart phones offer customer enhanced methods to interact with their phones but the most natural way of interaction remains speech. Market for smart mobile phones provides a number of applications with speech recognition implementation. Google's Voice Actions and recently iphone's Siri are applications that enable control of a mobile phone using voice, such as calling businesses and contacts, sending texts and email, listening to music, browsing the web, and completing common tasks. Both Siri and Voice Actions require an active connection to a network in order to process requests and most of Android phones can run on a 4G network which is faster than the 3G network that the iPhone runs on. There is also an issue of availability, Voice Actions are available on all Android devices above Android 2.2, but Siri is available only for owners of the iPhone 4S. The Siri's advantage is that it can act on a wide variety of phrases and requests and can understand and learn from natural language, whereas Google's Voice Actions can be operated only by using very specific voice commands. In this work we have developed an application for sending SMS messages which uses Google's speech recognition engine. The main goal of application Voice SMS is to allow user to input spoken information and send voice message as desired text message. The user is able to manipulate text message fast and easy without using keyboard, reducing spent time and effort. In this case speech recognition provides alternative to standard use of key board for text input, creating another dimension in modern communications.

II. ANDROID

Android is a software environment for mobile devices that includes an operating system, middleware and key applications [1]. In 2005 Google took over company Android Inc., and two years later, in collaboration with the group the Open Handset Alliance, presented Android operating system (OS).

Main features of Android operating system are:

- Enables free download of development environment for application development.
- Free use and adaptation of operating system to manufacturers of mobile devices.

- Equality of basic core applications and additional applications in access to resources.
- Optimized use of memory and automatic control of applications which are being executed.
- Quick and easy development of applications using development tools and rich database of software libraries.
- High quality of audiovisual content, it is possible to use vector graphics, and most audio and video formats.
- Ability to test applications on most computing platforms, including Windows, Linux…

The Android operating system (OS) architecture is divided into 5 layers (fig. 1.). The application layer of Android OS is visible to end user, and consists of user applications. The application layer includes basic applications which come with the operating system and applications which user subsequently takes. All applications are written in the Java programming language. Framework is extensible set of software components used by all applications in the operating system. The next layer represents the libraries, written in the C and C + + programming languages, and OS accesses them via framework.

Dalvik Virtual Machine (DVM), forms the main part of the executive system environment. Virtual machine is used to start the core libraries written in the Java

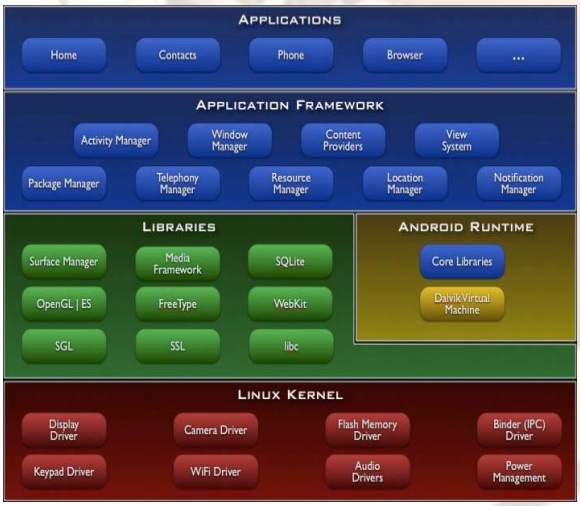

Fig1: Android Architecture

programming language. Unlike Java's virtual machine, which is based on the stack, DVM bases on registry structure and it is intended for mobile devices. The last architecture layer of Android operating system is kernel based on Linux OS, which serves as a hardware abstraction layer. The main reasons for its use are memory management and processes, security model, network system and the constant development of systems. There are four basic components used in construction of applications: activity, intent, service and the content provider. An activity is the main element of every application and simplified description defines it as a window that users see on their mobile device. The application can have one or more activities. Main activity is the one that is used as startup. The transition between the activities is carried out in a way that launched activity calls a new activity. Each activity as a separate component is implemented with inheritance of Activity class. During the execution of applications, activities are added to the stack, currently running activity is on the top of the stack.

An intent is a message used to run the activities, services, or recipient's multicast. An intent can contain the name of the components you need to run, the action which is necessary to execute, the address of stored data needed to run the component, and component type. A service is a component that runs in the background to perform long running operations or to perform work for remote processes. One service can link multiple applications and service is executed until a connection with all applications is done. A content provider manages a shared set of application data. Data can be stored in the file system, a SQLite database, on the web, or any other persistent storage location which application can access [1].

Through the content provider, other applications can query or even modify the data (if the content provider allows it).

III. SPEECH RECOGNITION

Speech recognition for application Voice SMS is done on Google server, using the HMM algorithm. HMM algorithm is briefly described in this part. Process involves the conversion of acoustic speech into a set of words and is performed by software component. Accuracy of speech recognition systems differ in vocabulary size and confusability, speaker dependence vs. independence, modality of speech (isolated, discontinuous, or continuous speech, read or spontaneous speech), task and language constraints [2].

Speech recognition system can be divided into several blocks: feature extraction, acoustic models database which is built based on the training data, dictionary, language model and the speech recognition algorithm. Analog speech signal must first be sampled on time and amplitude axes, or digitized. Samples of speech signal are analyzed in even intervals. This period is usually 20 ms because signal in this interval is considered stationary. Speech feature extraction involves the formation of equally spaced discrete vectors of speech characteristics. Feature vectors from training database are used to estimate the parameters of acoustic models. Acoustic model describes properties of the basic elements that can be recognized. The basic element can be a phoneme for

continuous speech or word for isolated words recognition.

Dictionary is used to connect acoustic models with vocabulary words. Language model reduces the number of acceptable word combinations based on the rules of language and statistical information from different texts. Speech recognition systems, based on hidden Markov models are today most widely applied in modern technologies. They use the word or phoneme as a unit for modeling. The model output is hidden probabilistic functions of state and can't be deterministically specified. State sequence through model is not exactly known. Speech recognition systems generally assume that the speech signal is a realization of some message encoded as a sequence of one or more symbols [3]. To effect the reverse operation of recognizing the underlying symbol sequence given a spoken utterance, the continuous speech waveform is first converted to a sequence of equally spaced discrete parameter vectors. Vectors of speech characteristics consist mostly of MFC (Mel Frequency Cepstral) coefficients [4], standardized by the European Telecommunications Standards Institute for speech recognition. The European Telecommunications Standards Institute in the early 2000s defined a standardized MFCC algorithm to be used in mobile phones [5]. Standard MFC coefficients are constructed in a few simple steps. A short-time Fourier analysis of the speech signal using a finite-duration window (typically 20ms) is performed and the power spectrum is computed. Then, variable bandwidth triangular filters are placed along the perceptually motivated mel frequency scale and filter bank energies are calculated from the power spectrum.

Magnitude compression is employed using the logarithmic function. Finally, auditory spectrum thus obtained is decorrelated using the DCT and first (typically 13) coefficients represent the MFCCs.

These vectors of speech characteristics are called observations and used in further calculations. To develop an acoustic model, it is necessary to define states. Continuous speech recognition, each state represents one phoneme. Under the concept of training we mean the determination of probabilities of transition from one state to another and probabilities of observations. Iterative Baum-Welch procedure is used for training [6]. The process is repeated until a certain convergence criterion is reached, for example good accuracy in terms of small changes of estimated parameters, in two successive iterations. In continuous speech the procedure is performed for each word in complex HM model. Once states, observations and transition matrix for HMM are defined, the decoding (or recognition) can be performed. Decoding represents finding of most likely sequence of hidden states using Viterbi algorithm, according to the observed output sequence. It is defined by recursive relation. During the search, n-best word sequences are generated using acoustic models and a language model.

IV. MAIN PARTS OF THE PROJECT

A. Voice Recognition Activity class

Voice Recognition Activity is startup activity defined as launcher in AndroidManifest.xml file. REQUEST_CODE is static integer variable, declared on the beginning of activity and used to confirm response when engine for speech recognition is started. REQUEST_CODE has positive value. Results of recognition are saved in variable declared as *ListView* type.

Method onCreate is called when activity is initiated.

This is where the most initialization goes: setContentView (R.layout.voice_recognition) is used to inflate the user interface defined in res > layout > voice_recognition.xml, and findViewById(int) to programmatically interact with widgets in the user interface. In this method there is also a check whether mobile phone, on which application is installed, has speech recognition possibility. Package Manager is class for retrieving various kinds of information related to the application packages that are currently installed on the device. FunctiongetPackageManager() returns PackageManager instance to find global package information. Using this class, we can detect if the phone has a possibility for speech recognition. If a mobile device doesn't have one of many Google's applications which integrate speech recognition, further work of this application Voice SMS will be disabled and message on the screen will be "Recognizer not present". Recognition process is done trough one of Google's speech recognition applications. If recognition activity is present user can start the speech recognition by pressing on the button and thus launching startActivityForResult (Intent intent, int requestCode). The application uses startActivityForResult() to broadcast an intent that requests voice recognition, including an extra parameter that specifies one of two language models. Intent is defined with intent.putExtra (RecognizerIntent.

EXTRA_LANGUAGE_MODEL,
RecognizerIntent.LANGUAGE_MODEL_FREE_FORM).

The voice recognition application that handles the intent processes the voice input, then passes the recognized string back to Voice SMS application by calling the onActivityResult() callback. In this method we manage result of activity startActivityForResult. Received result *Data* is stored in ListView variable and shown on the screen (Fig. 3.).

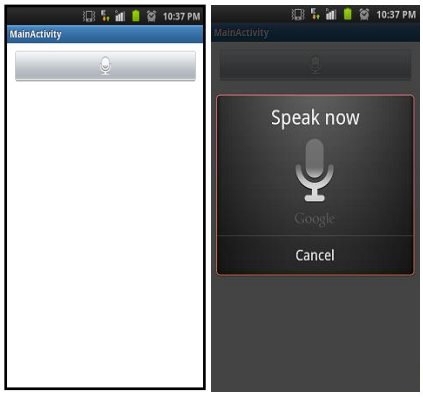

Fig2: Enables search after clicking image button

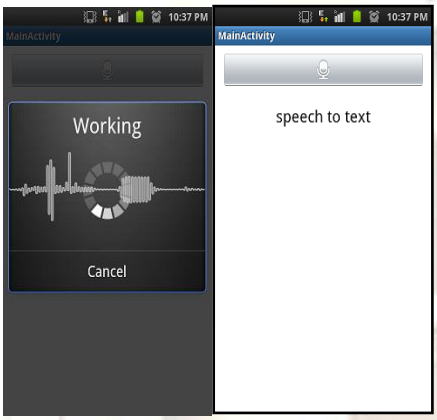

Fig3: Processes and gives text output

Fig4: Interface for sending SMS

B. SMS class

In onCreate method label setContentView (R.layout.sms) opens new user interface. Command final String message1 = getIntent().getExtras().GetString ("Key") retrieves the data associated with the string "key" which in this case is selected text from recognition Fig3. Testing Figure 4. Interface for sending SMS activity. The text is entered in the space for writing messages and displayed on the screen. By clicking the Send SMS button application checks whether the message and the number of recipient are entered to perform sending of message. When cursor is positioned in the space for recipient number from contacts, button attribute visibility is changed from default *gone* to *visible*. Pressing the button the command that allows you to enter the contact numbers. label Intent forContacts new Intent = (Intent.ACTION_PICK, ContactsContract.Contacts.CONTENT_URI) startActivityForResult (forContacts, REQUEST_CODE) is executed. After selecting desired contact, message can be sent.

C. XML files

Application consists of two different interfaces. When the user runs application screen is defined in voice_recognition.xml. The linear arrangement of elements allows adding widget one below another. Width and height are defined with *fill_parent* attribute, which means to be equal as parent (in this case the screen). The second interface, defined within sms.xml file, is displayed when the user chooses one of offered messages. AndroidManifest.xml realizes installing and launching applications on the mobile device. Every application must have an AndroidManifest.xml file (with precisely that name) in its root directory. The manifest presents essential information about the application to the Android system, information the system must have before it can run any of the application's code. It defines the activities and permissions required for the application. Permissions are used to remove restrictions that prevent access to certain pieces of code or data on mobile device. Each permit is defined by a special label. If application needs data with distrait if must seek the necessary authorization. The syntax is: <uses-permission android:name= "Unique name of permission"> </ uses-permission>.

Every activity must be named in manifest. Name is used as a parameter that is passed to the constructor of intent which is used to start desired activity. Activities can have different <category> attribute.

VoiceRecognitionActivity.java is the main class and its onCreate method is executed at application startup. The category tag is specially defined String that describes this feature of activity with keyword *Launcher* while other classes are marked with keyword *Default*. AndroidManifest.xml has to include permission to send and receive messages, to access Internet and

permission to access the list of contacts from the phonebook.
Permissions are:
_ android.permission.INTERNET.
_ android.permission.SEND_SMS.
_ android.permission.RECEIVE_SMS.
_ android.permission.READ_CONTACTS.

V. APPLICATION FUNCTIONALITY PRINCIPLE

Application Voice SMS integrates direct speech input enabling user to record spoken information as text message, and send it as SMS message. After application has been started display on mobile phone shows button which initiate voice recognition process. When speech has been detected application opens connection with Google's server and starts to communicate with it by sending blocks of speech signal. Simultaneously the figure of waveform is generated on the screen. Speech recognition of the received signal is preformed on server. Google has accumulated a very large database of words derived from the daily entries in the Google search engines well as the digitalization of more than 10 million books in Google Book Search project. The database contains more than 230 billion words. If we use this kind of speech recognizer it is very likely that our voice is stored on Google's servers. This fact provides continuous increase of data used for training, thus improving accuracy of the system.

When process of recognition is over, user can see the list of possible statements. Process can be repeated clicking on the button *Image Button*. Pressing the most accurate option, selected result is entered into interface for writing SMS messages.

Interface for writing SMS messages has all standard features. User can correct text and input recipient number in the empty textbox. Button *Contacts* opens interface with contact numbers from mobile phone and enables user to chose phone number on which message will be sent after pressing button *Send.* Customer receives response in a little cloud (toast) if message has been sent.

VI. CONCLUSION

With the development of software and hardware capabilities of mobile devices, there is an increased need for device-specific content, what resulted in market changes. Speech recognition technology is of particular interest due to the direct support of communications between human and computers.

Using the speech recognizer, which works over the Internet, allows much faster data processing. Another advantage is the much larger databases that are used. Many believe that this mode is a big step for speech recognition technology. The accuracy of the system has significantly increased and become more accessible to everyone. We have been given a possibility to manage mobile devices without installing complex software for speech processing, resulting in memory savings. A lack of application Voice SMS is its adaption only for English language, and the need for permanent Internet connection.

The objective for the future is development of models and databases for multiple languages which could create a foundation for everyday use of this technology worldwide. The main goal of application is to allow sending of text messages based on uttered voice messages. Testing has shown simplicity of use and high accuracy of data processing. Results of recognition give user option to select the most accurate result. Further work is planned to implement the model of speech recognition for different language.

LITERATURE

[1] "Android developers", http://developer.android.com
[2] J. Tebelskis, Speech Recognition using Neural Networks, Pittsburgh: School of Computer Science,Carnegie Mellon University, 1995.
[3] S. J. Young et al., "The HTK Book", Cambridge University Engineering Department, 2006.
[4] K. Davis, and Mermelstein, P., "Comparison of parametric representation for monosyllabic word recognition in continuously spoken sentences", IEEE Trans. Acoust., Speech, Signal Process. vol. 28, no. 4, pp. 357–366,1980.
[5] ETSI, "Speech Processing, Transmission and Quality Aspects (STQ); Distributed speech recognition; Front-end feature extraction algorithm; Compression algorithms.", Technical standard ES 201 108, v1.1.3, 2003.
[6] L. E. Baum, T. Petrie, G. Soules, and N. Weiss, "A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains", Ann. Math. Statist., vol. 41, no. 1, pp. 164–171, 1970.

REFERENCES

[1] C. Allauzen, M. Riley, J. Schalkwyk, W. Skut, and M. Mohri. OpenFst: A general and efficient weighted _nite-state transducer library. Lecture Notes in Computer Science, 4783:11, 2007.
[2] M. Bacchiani, F. Beaufays, J. Schalkwyk, M. Schuster, and B. Strope. Deploying GOOG-411: Early lessons in data, measurement, and testing. In Proceedings of ICASSP, pages 5260{5263, 2008.

[3] MJF Gales. Semi-tied full-covariance matrices for hidden Markov models. 1997.
[4] B. Harb, C. Chelba, J. Dean, and G. Ghemawhat. Back-o_ Language Model Compression. 2009.
[5] H. Hermansky. Perceptual linear predictive (PLP) analysis of speech. Journal of the Acoustical Society of America, 87(4):1738{1752, 1990.
[6] Maryam Kamvar and Shumeet Baluja. A large scale study of wireless search behavior: Google mobile search. In CHI, pages 701{709, 2006.
[7] S. Katz. Estimation of probabilities from sparse data for the language model component of a speech recognizer. In IEEE Transactions on Acoustics, Speech and Signal Processing, volume 35, pages 400{01, March 1987.
[8] D. Povey, D. Kanevsky, B. Kingsbury, B. Ramabhadran, G. Saon, and K. Visweswariah. Boosted MMI for model and feature-space discriminative training. In Proc. of the Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP), 2008.
[9] R. Sproat, C. Shih, W. Gale, and N. Chang. A stochastic _nite-state word-segmentation algorithm for Chinese. Computational linguistics,22(3):377{404, 1996.
[10] C. Van Heerden, J. Schalkwyk, and B. Strope. Language modeling for what-with-where on GOOG-411. 2009.

WEB SITES

Minimalist GNU for Windows, 06/01/2010, http://www.mingw.org/
Code Blocks, The open source, cross platform, IDE, 06/01/2010, http://www.codeblocks.org/
Java JDK SE 1.6 update 18, 02/04/2010, http://java.sun.com/javase/
ECLIPSE GALILEO Release 3.5.2, 02/04/2010, http://www.eclipse.org/galileo/
Java sound API documentation, 11/04/2010, http://java.sun.com/j2se/1.5.0/docs/guide/sound/programmer_guide/contents.html
Android 2.2 SDK release 6, http://developer.android.com/sdk/index.html
Android 2.2 SDK reference page, http://developer.android.com/reference/packages.html