Kotlin Framework for Number Retrieval from Phone Call

KOMAL D. PEDNEKAR[1], ROMILA R. HOTEKAR[2], AADITI A. SHIVALKAR[3], MANDAR S. JOSHI[4]

[1, 2, 3] *Finolex Academy of Management and Technology, Mumbai University.*

[4] *Assistant Professor, Finolex Academy of Management and Technology, Mumbai University.*

Abstract- ***Speech is the natural and most important form of communication for human beings. Speech-To-Text (STT) system takes a human speech utterance as an input and requires string of words as output. The objective of this system is to extract, characterized and recognize information about speech. Past research in mathematics, acoustic and speech technology have provided many methods for converting data that can be considered as information if interpreted correctly. In order to find some statistical relevant information from incoming data, it is important to have mechanism for reducing the information in each segment in audio signal into a relatively small no of parameters, or features. Audio features extraction serves as the basic for a wide range of applications in the areas of speech processing, multimedia data management and distribution, security, biometric and bioacoustics. The features can be extracted either directly from the time domain signal or from the transformation domain depending upon choice of the signal analysis approach.***

Indexed Terms- Kotlin, Recognizer Intent, Speech-To-Text, Audio

I. INTRODUCTION

This review paper gives the details about the implementation of an Android application that records the phone call on-demand using RecognizerIntent Class of Google's Speech-to-Text API using Kotlin programming language; converts the recorded speech into text; extracts the number from it. "Automatic Number Retrieval from phone call " is an Android application which records the on-going phone call on-demand and convert the recorded speech into textual format; extract the numbers from the text available after conversion; provides a list of all such recorded numbers when the phone call ends along with the options. The objective to select this as our final year project topic is to have an automated facility of adding the numbers to the contacts without the need of pen-paper or any other thing or device. One more objective is to understand the concept of Speech-To-Text (STT) conversion and feature extraction that are very popular now-a-days to have a direct interaction with your device/gadget.

II. LITERATURE SURVEY

A. Basic principle

ASR system operates in two phases. FIRST, a training phase, during which the system learns the reference patterns representing the different speech sounds (e.g., phases, words phones) that constitute the vocabulary of the application. Each reference is learned from spoken examples and stored either in the form of templates obtained by some averaging method or models that characterize the statistical properties of pattern [3]. Second, a recognizing phase, during which an unknown input pattern, is identified by considering the set of references.

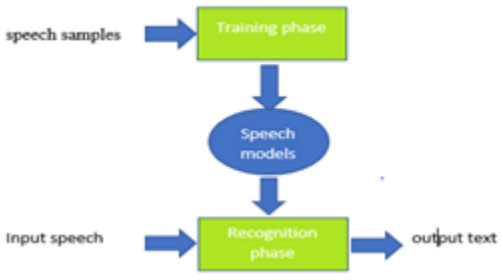


fig.1 Basic principle of ASR

B. Speech-To-Text conversion algorithms overview:

1. Analysis of noise contents:

Additive noise is additive to the speech signal in both, the power spectrum domain and the autocorrelation domain. Channel disturbance on the other hand is

multiplicative in nature in the autocorrelation domain. Thus, the technique to remove additive noise is to subtract in either the power spectrum domain or the autocorrelation domain. the technique to remove channel distortion however, involves a more complicated set of steps

2. End-point Detection Algorithms:
There are two popular algorithms for the same, viz., ZCR-En and VFR.

ZCR-EN (Zero Crossing Rate-short time Energy) algorithm uses two-time domain parameters to decide the boundary between silence voice components. ZCR is zero crossing rate which is defined as the number of times that a signal changes signs in a particular frame.

Variable Frame Rate (VFR)is a technique used for discarding frames that are too much alike. The method emphasizes the transient regions, which are more relevant for speech recognition.

3. Features Extraction techniques:
LPC (Linear predictive Coding Analysis) is a technique used for speech recognition to evaluate basic speech stipulations like pitch, formant and spectral envelop of speech signal in compressed form. it is a technique used for encrypting better quality speech at low bit rate.

MFCC (Mel-frequency cestrum coefficient) is in audio and speech processing. They are derived from a type of spectral (result of taking the IFT of logarithm of spectrum of signal) representation of audio clip. The difference between the Mel-frequency spectrum and spectrum is the frequency bands are equally spaced on the Mel-scale, which approximates the human auditory range more closely than the linear-spaced frequency bands.

4. Nature of speech recognition:
Synchronous recognition: sends audio data to the Speech to test API perform recognition on that data and return result after all audio has been processed. Synchronous recognition requests are limited to audio data of 1min or less in duration. Asynchronous reorganization-sends audio data to the STT API and initiates a Long Running Operation. Using this operation, you can periodically poll for reorganization results. Use asynchronous requests for audio data of any duration up to 180 minutes. Streaming Recognition-performs recognition on audio data provided within a bi-directional stream, Streaming requests are designed for real-time reorganization purposes, such as capturing live audio from a microphone. streaming recognition provides interim results while audio is being captured, allowing result to appear, for example, while a user is still speaking.

- API in detail:

Synchronous Speech Recognition Requests A Synchronous Speech-to-Text API request consists of a speech recognition configuration and audio data. A sample request is shown below:

```
{
"config": {
"encoding": "LINEAR16",
"sampleRateHertz": 16000,
"language Code": "end
US",
},
"audio": {
"Uri": "gas://bucket
/path_to_audio_file"
}
}
```

All Speech-to-Text API synchronous recognition requests must include a speech recognition config field (of type RecognitionConfig).

- Embedding audio content:

Embedded audio is included in the speech recognition request when passing a content parameter within the request's audio field. For embedded audio provided as content within a gRPC request, that audio must be compatible for Proto3 serialization, and provided as binary data. For embedded audio provided as content within a REST request, that audio must be compatible with JSON serialization and first be Base64-encoded. See Base64 Encoding Your Audio for more information)

- Speech-to-Text API responses:

As indicated previously, a synchronous Speech-to-Text API response may take some time to return results, proportional to the length of the supplied audio. Once processed, the API will return a response as shown below:

```
{
```

"results": [
{
"alternatives": [
{
"confidence": 0.98267895,
"transcript": "how old is the Brooklyn Bridge"
}
] }]}

These fields are explained below: results contains the list of results (of type SpeechRecognitionResult) where each result corresponds to a segment of audio (segments of audio are separated by pauses). Each result will consist of one or more of the following fields: alternatives contains a list of possible transcriptions, of type SpeechRecognitionAlternatives. Whether more than one alternative appears depends both on whether you requested more than one alternative (by setting max Alternatives to a value greater than 1) and on whether Speech-to-Text produced alternatives of high enough quality.

C. Number Extraction Methods-

After speech to text conversion, we stored result of conversion in text file, so for retrieving number from that text file we have used Regex class and its methods

- The RegEx class-

To work with regular expressions in kotlin, you need to use the Regex (pattern: String) class and invoke functions like find (…) or replace on that regex object. An example on how to use the Regex class that returns true if string contains c or d:

Val regex = Regex (pattern = c|d")
Val matched = regex.containsMatchIn(input = "abc")

- Regex Functions –

containsMatch()

The essential thing to understand with all the Regex functions is that the result is based on matching the regex pattern and the input string. Some of the functions requires a full match , while the rest requires only a partial match. For partial match containsMatch(…) function is used. This function returns a Boolean indicating whether there exists any match of our pattern in the input.

Fun containsMatchIn(input: CharSequence): Boolean
findAll()-

This function returns all the matchings of the specified pattern in the input, searching from the given start index.

Fun findAll(
Input: CharSequence,
startIndex: Int
): Sequence

III. IMPLEMENTATION DETAILS

We have studied different speech to text conversion methods and techniques and tried speech-to-text conversion using platform like MATLAB and Python. In MATLAB to convert input speech to text output we perform four steps - speech database, pre-processing, feature extraction and recognition. In python we used Speech Recognition library for conversion. Also, we experiment basic simple speech to text converter app in android studio using Google API and Kotlin programming language. Based on these experimentations we analysed that Google speech to text API provides better accuracy for voice recognition with less word error rate and multiple language support.

A. Architecture:

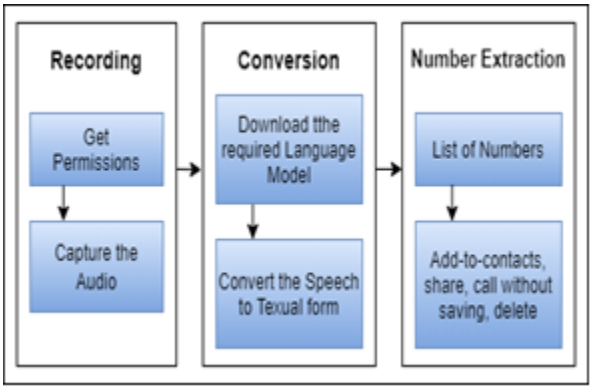


fig.2 Flow of System

B. Kotlin's Framework:

We used Kotlin in Android Studio to develop this project. To develop the application with desired features, we have used the following facilities of Kotlin.

- For recognition of the speech:

We used Google speech to text API for recording voice

- Class used – Recognizer intent
- Method used – ACTION_RECOGNIZE_SPEECH LANGUAGE_MODEL_FREE_FORM EXTERNAL_LANGUAGE_MODEL
- Code –

```
Val intent = Intent (RecognizerIntent.ACTION_RECOGNIZE_SPEECH)

Intent.putExtra(

RecognizerIntent.EXTRA_LANGUAGE_MODEL,
RecognizerIntent.LANGUAGE_MODEL_FREE_FORM
 )
Intent.putExtra(RecognizerIntent.EXTRA_LANGUAGE, Locale.getDefault())
Intent.putExtra(RecognizerIntent.EXTRA_PROMPT,"")
startActivityForResult(intent,100)
```

C. Number Extraction

For number extraction purpose we used Regular Expression.

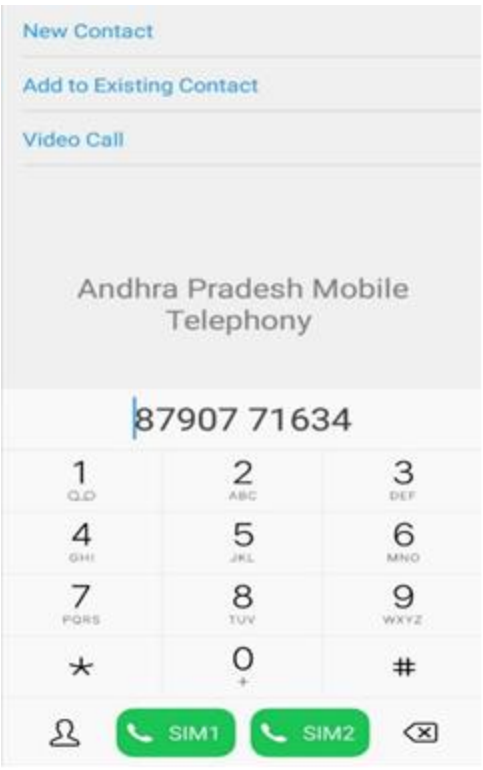
- Class used – Regex class
- Method used – findAll()
- Code -

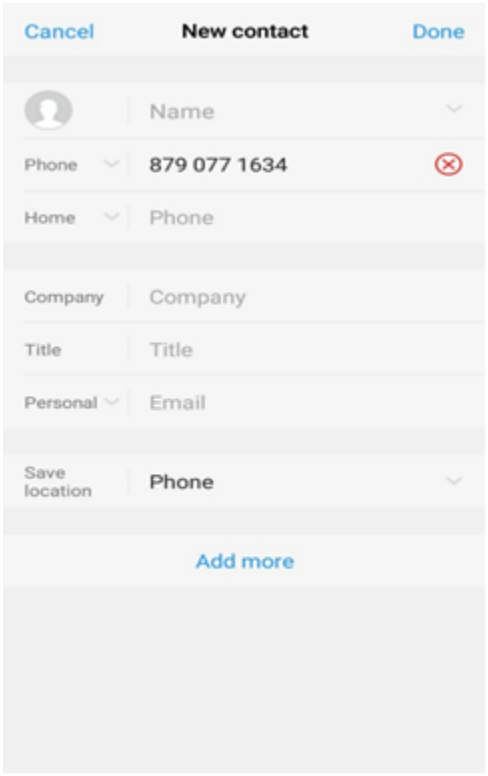
```
try {
if (reader.readLine().also { string = it } == null) break
                } catch (e: IOException) {
                    e.printStackTrace()
                }
                stringBuilder.append(string)
                 Val
P. =stringBuilder.replace("\\s".toRegex(),
"")

                val sp = Regex("""\d{10}""").findAll(p)
                val r = java.lang.StringBuilder()
                for(findtext in sp){
                    r.append(findtext.value+" ")
                }
```

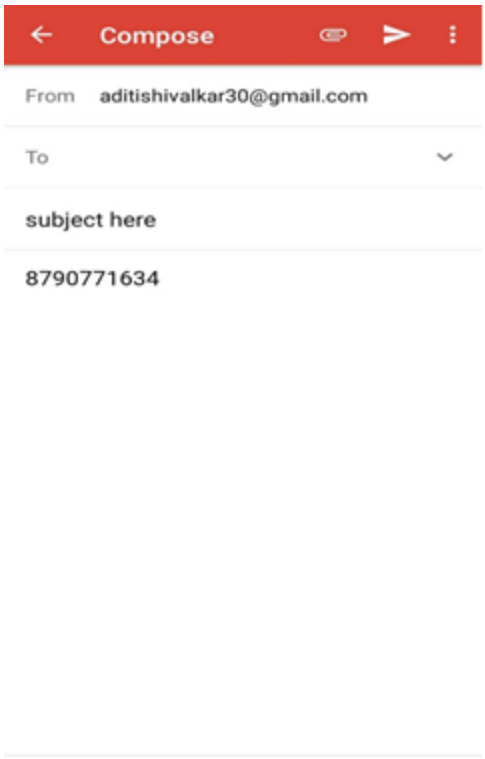
IV. RESULT


ANRFVC
8790771634
1334447690
EXTRACT NO


New Contact
Add to Existing Contact
Video Call
Andhra Pradesh Mobile Telephony
87907 71634
SIM1
SIM2


Cancel
New contact
Done
Name
Phone
879 077 1634
Home
Phone
Company
Company
Title
Title
Personal
Email
Save location
Phone
Add more


ANRFVC
8790771634
1334447690
label
WhatsApp
EasyShare
Messages
Email
Gmail
Save to Drive
Nearby Share
Edmodo



RESULT ANALYSIS

Google Speech-to-Text typically processes audio faster than Realtime, processing 30 seconds of audio in 15 seconds on average. In cases of poor audio quality, your recognition request can take significantly longer. Google has also optimized the service to transcribe noisy audio without requiring additional noise cancellation. It cancels background noise for clear and accurate audio from phone call.

Apart from smart number retrieval following features can also be added:

- Addresses Retrieval address spoken can also be extracted from the speech and the user can be automatically navigated to that address using Google Maps and second one is Simple Text Retrieval this feature can also be incorporated that can be further used to send the text as SMS or as a message in other application or can be used to set reminder or can be saved simply as a memo.

CONCLUSION

Thus, we have successfully developed an android application using Kotlin that can record the on-going voice call on user demand, convert the recorded speech into text, extract the number from it and display number with facilities to “add to contacts”,” share”, “call without saving” and “delete”. As the analysis part, we have studied different speech-to-text (STT) conversion techniques like ZCR-EN, VFR, LPC, MFCC, SVM, HMM etc. these algorithms helped us to understand the methods underlying the conversion.

ACKNOWLEDGMENT

I thank our college Finolex Academy of Management and Technology for allowing us to publish this technical research paper. I thank our project guide Prof. Mandar S. Joshi sir who provided insight and expertise that greatly assisted the research. I am thankful to and fortunate enough to get constant encouragement, support and guidance from Teaching staffs of Department of Information Technology. Also, I thank our librarians for helping us with the material required as the reference. Any errors are our own and should not tarnish the reputations of these esteemed persons.

REFERENCES

[1] Sanjib Das, "Speech Recognition Technique: A Review”, International Journal of Engineering Research and Applications (IJERA) ISSN: 2284-9622 Vol. 2, Issue 3, May-Jun 2012.

[2] Pratik K. kurzekar , Ratnadeep R. Deshmukh, Vishal B. Waghmare, Pukhraj P. Shrishrimal," A comparative Study of Feature Extraction Techniques forSpeech Recognition System", International Journal of Innovative Research in Science, Engineering andTechnology(An ISO 3297: 2007 certified organization) Vol. 3, Issue 12, December 2014.

[3] Santosh K. Gaikwad, Bharti W.Gawali,Pravin Yannawar, "A Review on Speech Recognition Technique", International Journal of Computer Applications (0975-8887)Vol. 10-No.3, November 2010.

[4] "A Review on Speech To Text conversation Methods" , International Journal of dvanced Reaserch in Computer Engineering & Technology(IJARCET) Volume 4 Issue 7, July 2015.

[5] B. Raghavendhar Reddy, E. Mahender, "Speech to Text conversation using Android Platform ", International Journal of Engineering Research and Applications (IJERA) ISSN: 2284-9622 Vol. 3, Issue 1, January-February 2013.

[6] Suman K. Saksamudre, P.P. Shrishrimal, R.R. Deshmukh," A Review on Different Approaches for Speech Recognition System", International Journal of Computer Applications (0975-8887) Vol. 115-