Camera, Audio, Video and Sound (Continued)

CSE 162 – Mobile Computing

Hua Huang
Department of Computer Science and Engineering
University of California, Merced

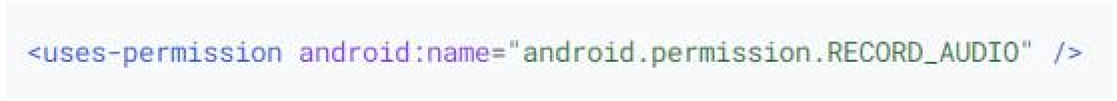
Media (audio and video) Recording in Android

MediaRecorder overview

- The Android multimedia framework includes support for capturing and encoding a variety of common audio and video formats.

Requesting permission to record audio

- To be able to record, your app must tell the user that it will access the device's audio input

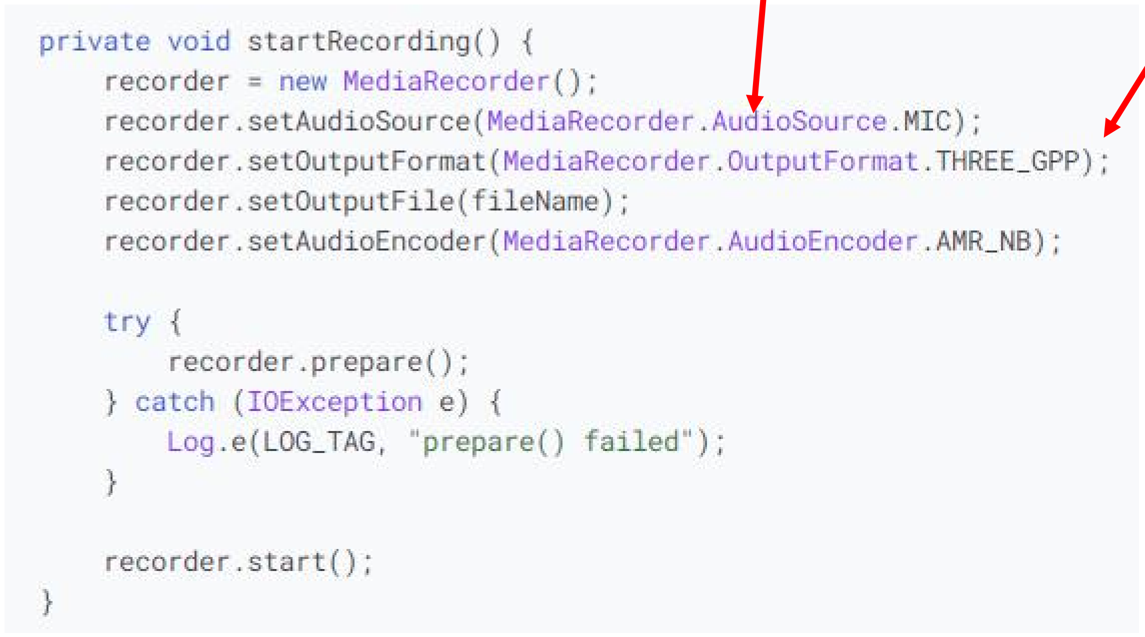


```
<uses-permission android:name="android.permission.RECORD_AUDIO" />
```

- RECORD_AUDIO is considered a "dangerous" permission because it may pose a risk to the user's privacy.
- Starting with Android 6.0 (API level 23) an app that uses a dangerous permission must ask the user for approval at run time.
    - After the user has granted permission, the app should remember and not ask again.

Creating and running a MediaRecorder

- Set the audio source using setAudioSource(). You'll probably use MIC.
- Set the output file format using setOutputFormat()



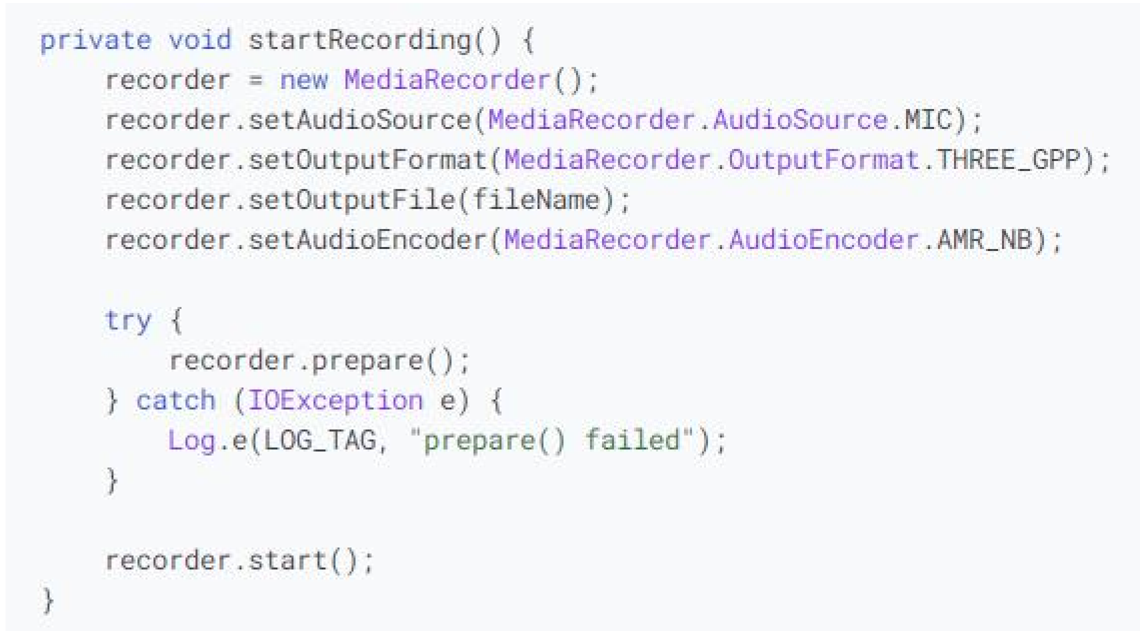
```
private void startRecording() {
    recorder = new MediaRecorder();
    recorder.setAudioSource(MediaRecorder.AudioSource.MIC);
    recorder.setOutputFormat(MediaRecorder.OutputFormat.THREE_GPP);
    recorder.setOutputFile(fileName);
    recorder.setAudioEncoder(MediaRecorder.AudioEncoder.AMR_NB);

    try {
        recorder.prepare();
    } catch (IOException e) {
        Log.e(LOG_TAG, "prepare() failed");
    }

    recorder.start();
}
```

Creating and running a MediaRecorder

- Set the output file name using setOutputFile(). You must specify a file descriptor that represents an actual file.
- Set the audio encoder using setAudioEncoder().
- Complete the initialization by calling prepare().
- Start the recorder.

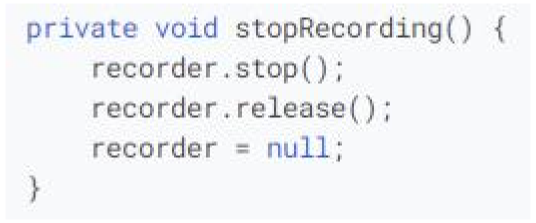


```
private void startRecording() {
    recorder = new MediaRecorder();
    recorder.setAudioSource(MediaRecorder.AudioSource.MIC);
    recorder.setOutputFormat(MediaRecorder.OutputFormat.THREE_GPP);
    recorder.setOutputFile(fileName);
    recorder.setAudioEncoder(MediaRecorder.AudioEncoder.AMR_NB);

    try {
        recorder.prepare();
    } catch (IOException e) {
        Log.e(LOG_TAG, "prepare() failed");
    }

    recorder.start();
}
```

Stop recording



```
private void stopRecording() {
    recorder.stop();
    recorder.release();
    recorder = null;
}
```

Audio Project Ideas

- OpenAudio project, http://www.openaudio.eu/
    - Many tools, dataset available
        - OpenSMILE: Tool for extracting > 1000 audio features
        - Windowing
    - MFCC
    - Pitch
    - Statistical features, etc
- Supports popular file formats (e.g. Weka)
- OpenEAR: Toolkit for automatic speech emotion recognition
- iHeaRu-EAT Database: 30 subjects recorded speaking while eating

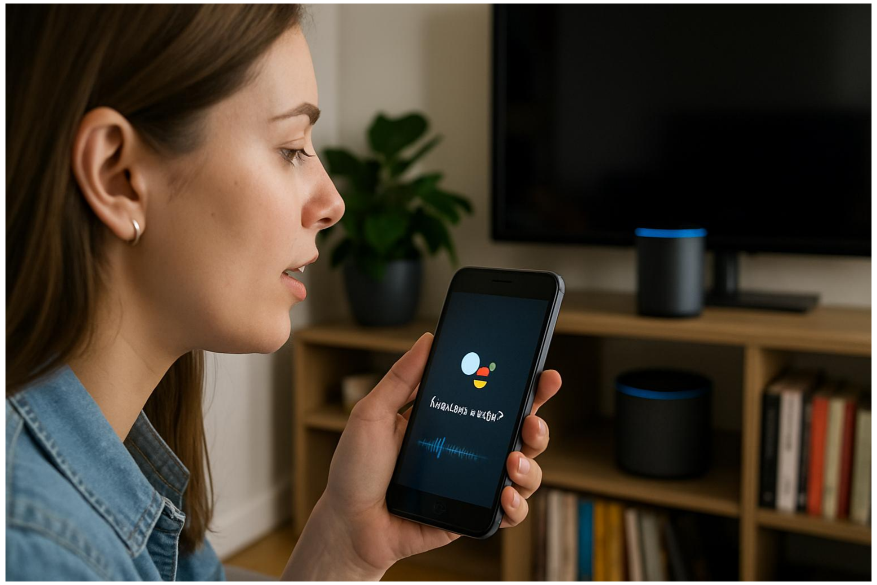
Speech Recognition

- peech recognition, also known as automatic speech recognition (ASR), computer speech recognition or speech-to-text, is a capability that enables a program to process human speech into a written format.

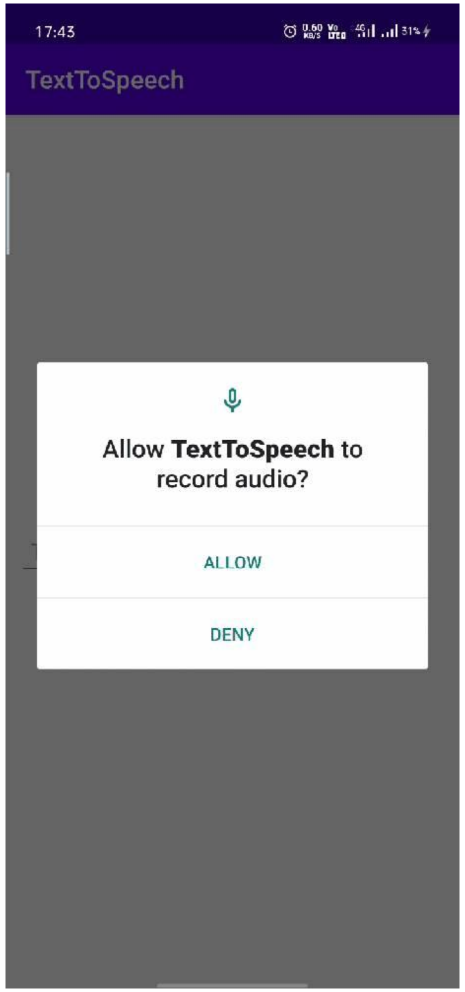
Speech recognition use cases

- Automotive: Speech recognizers improves driver safety by enabling voice-activated navigation systems and search capabilities in car radios.

- Technology: Virtual agents are increasingly becoming integrated within our daily lives, particularly on our mobile devices.
  - We use voice commands to access them through our smartphones, such as through Google Assistant or Apple's Siri, for tasks, such as voice search, or through our speakers, via Amazon's Alexa or Microsoft's Cortana, to play music.
- Healthcare: Doctors and nurses leverage dictation applications to capture and log patient diagnoses and treatment notes.



Build a speech recognizer in Android


17:43

TextToSpeech

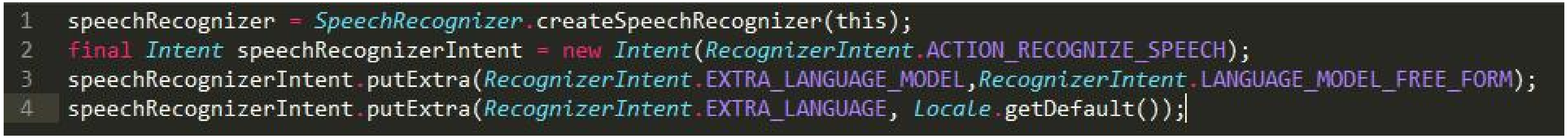
Allow **TextToSpeech** to record audio?

The Android SpeechRecognizer Class

- This class provides access to the speech recognition service.
- The implementation of this API is to stream audio to remote servers to perform speech recognition.
    - Not suitable for continuous recognition
    - consume a significant amount of battery and bandwidth.

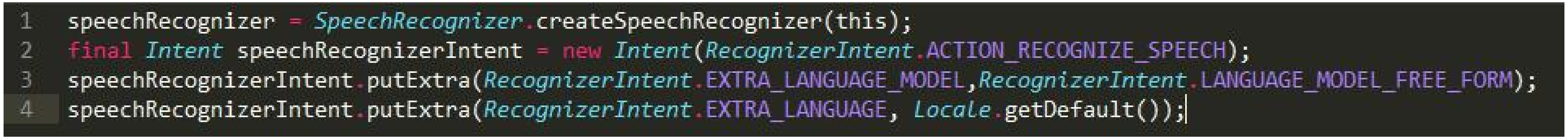
Usage of the SpeechRecognizer Class

- Create the class using *createSpeechRecognizer*()



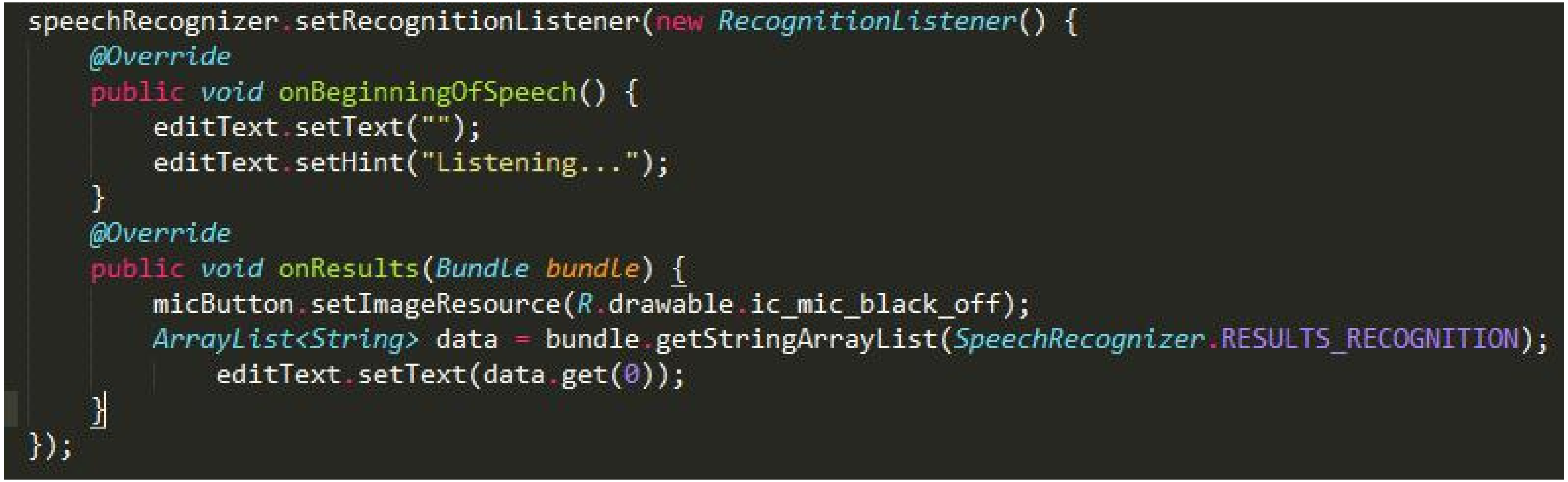
```java
speechRecognizer = SpeechRecognizer.createSpeechRecognizer(this);
final Intent speechRecognizerIntent = new Intent(RecognizerIntent.ACTION_RECOGNIZE_SPEECH);
speechRecognizerIntent.putExtra(RecognizerIntent.EXTRA_LANGUAGE_MODEL,RecognizerIntent.LANGUAGE_MODEL_FREE_FORM);
speechRecognizerIntent.putExtra(RecognizerIntent.EXTRA_LANGUAGE, Locale.getDefault());
```

- Create a speechRecognizerIntent.
  - The constant **ACTION_RECOGNIZE_SPEECH** starts an activity
    - The activity will prompt the user to speak and send it through a speech recognizer.
  - ***EXTRA_LANGUAGE_MODEL***: Informs the recognizer which speech model to use
  - ***EXTRA_LANGUAGE_MODEL***: Informs the recognizer which speech model to prefer when performing ACTION_RECOGNIZE_SPEECH.
  - ***EXTRA_LANGUAGE****:* Optional IETF language tag (as defined by BCP 47), for example, “en-US”.



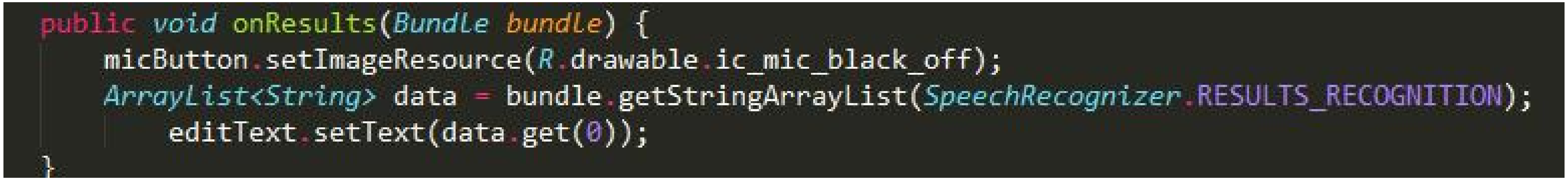
```
speechRecognizer = SpeechRecognizer.createSpeechRecognizer(this);
final Intent speechRecognizerIntent = new Intent(RecognizerIntent.ACTION_RECOGNIZE_SPEECH);
speechRecognizerIntent.putExtra(RecognizerIntent.EXTRA_LANGUAGE_MODEL,RecognizerIntent.LANGUAGE_MODEL_FREE_FORM);
speechRecognizerIntent.putExtra(RecognizerIntent.EXTRA_LANGUAGE, Locale.getDefault());
```

Create a speechRecognitionListener class



```
speechRecognizer.setRecognitionListener(new RecognitionListener() {
    @Override
    public void onBeginningOfSpeech() {
        editText.setText("");
        editText.setHint("Listening...");
    }
    @Override
    public void onResults(Bundle bundle) {
        micButton.setImageResource(R.drawable.ic_mic_black_off);
        ArrayList<String> data = bundle.getStringArrayList(SpeechRecognizer.RESULTS_RECOGNITION);
            editText.setText(data.get(0));
    }
});
```

- We get an ArrayList of String as a result
- We can use it to parse a command, or input texts.



```java
public void onResults(Bundle bundle) {
    micButton.setImageResource(R.drawable.ic_mic_black_off);
    ArrayList<String> data = bundle.getStringArrayList(SpeechRecognizer.RESULTS_RECOGNITION);
        editText.setText(data.get(0));
}
```

Speech Analytics

Voice Based Analytics

- Voice can be analyzed, lots of useful information extracted
  - Who is talking? (Speaker identification)
  - How many social interactions a person has a day
  - Emotion of person while speaking
  - Anxiety, depression, intoxication, of person, etc.
- For speech recognition, voice analytics used to:
  - Discard useless information (background noise, etc)
  - Identify and extract linguistic content

Mel Frequency Cepstral Coefficients (MFCCs)

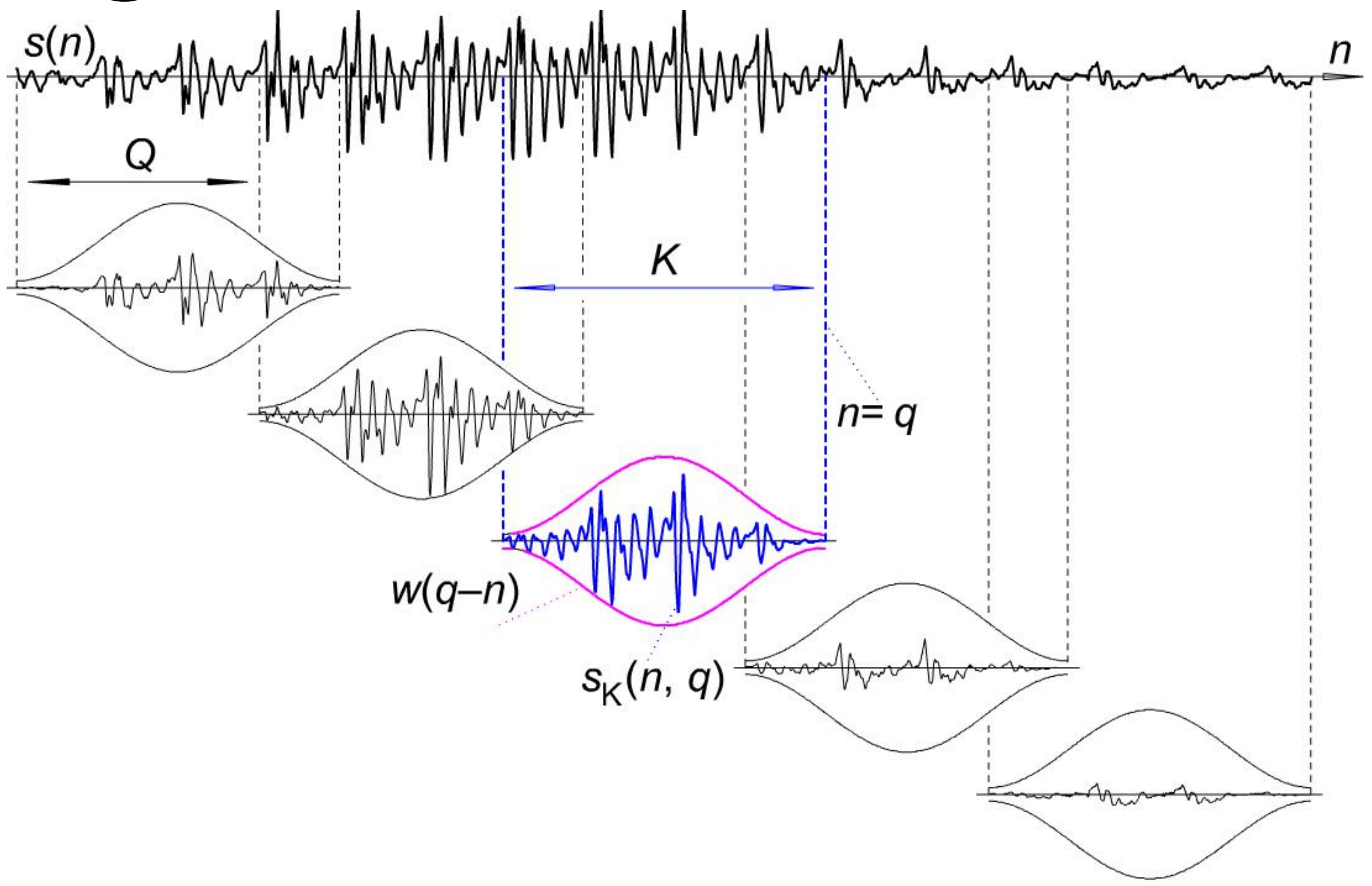
- MFCCs widely used in speech and speaker recognition for representing envelope of **power spectrum of voice**
- **Power spectrum?** Amount of power at various frequencies
- Roughly
  - Male voice: low frequency (bass)
  - Female voice: high frequency (treble)
- Popular approach in Speech recognition
  - MFCC features + Hidden Markov Model (HMM) classifiers

MFCC Steps: Overview

1. Frame the signal into short frames.
2. For each frame calculate the periodogram estimate of the power spectrum.
3. Apply the me lfilter bank to the power spectra, sum the energy in each filter.
4. Take the logarithm of all filter bank energies.
5. Take the DCT of the log filter bank energies.
6. Keep DCT coefficients 2-13, discard the rest.

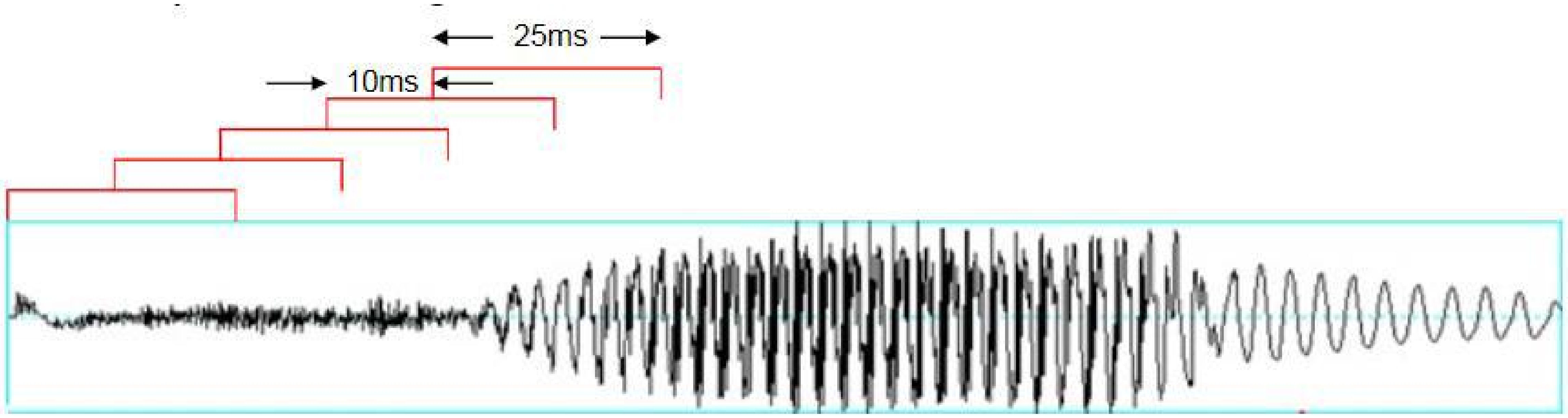
Step 1: Windowing

- Audio is continuously changing.
- Break into short, overlapping segments (20-40 milliseconds)
- Can assume audio does not change in short window



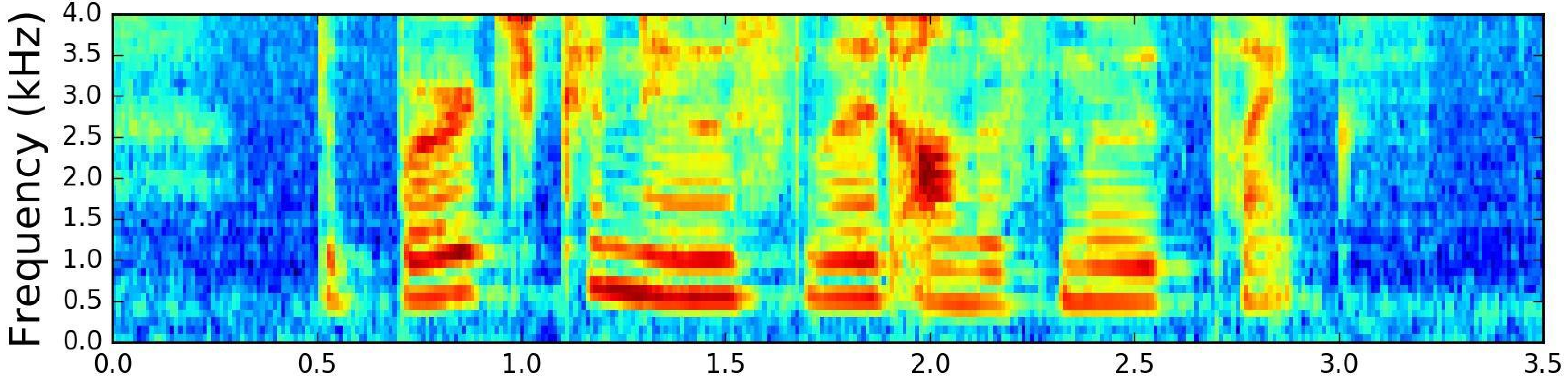
Step 1: Windowing

- Essentially, break into smaller overlapping frames
- Need to select frame length (e.g. 25 ms), shift (e.g. 10 ms)



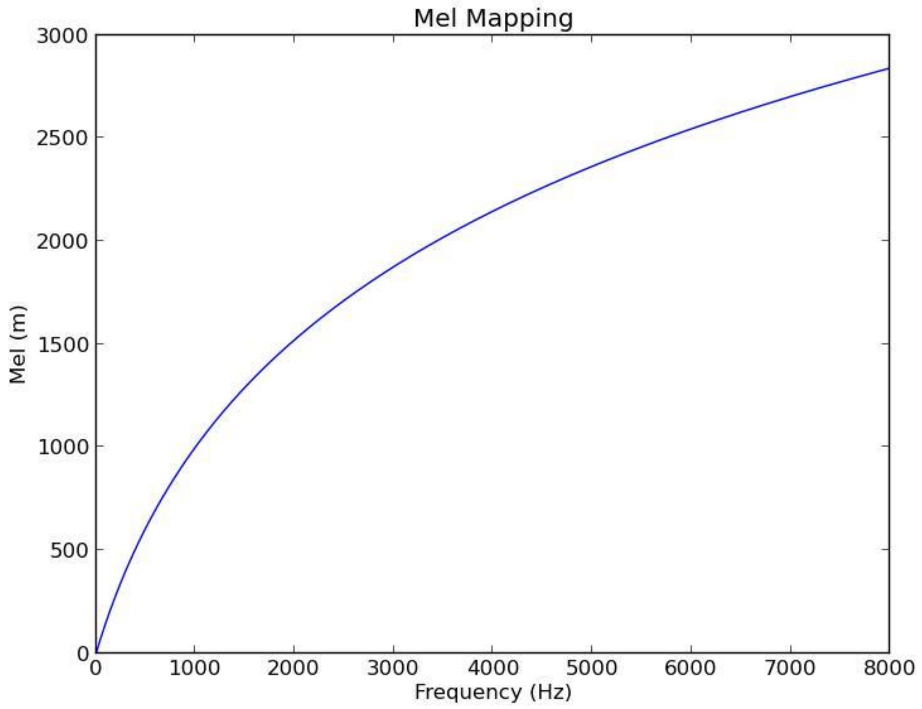
Step 2: Calculate Power Spectrum of each Frame

- Cochlea (**kaa**·klee·uh), part of human ear, vibrates at different parts depending on sound frequency
- Power spectrum Periodogram similarly identifies frequencies present in each frame



Background: Mel Scale

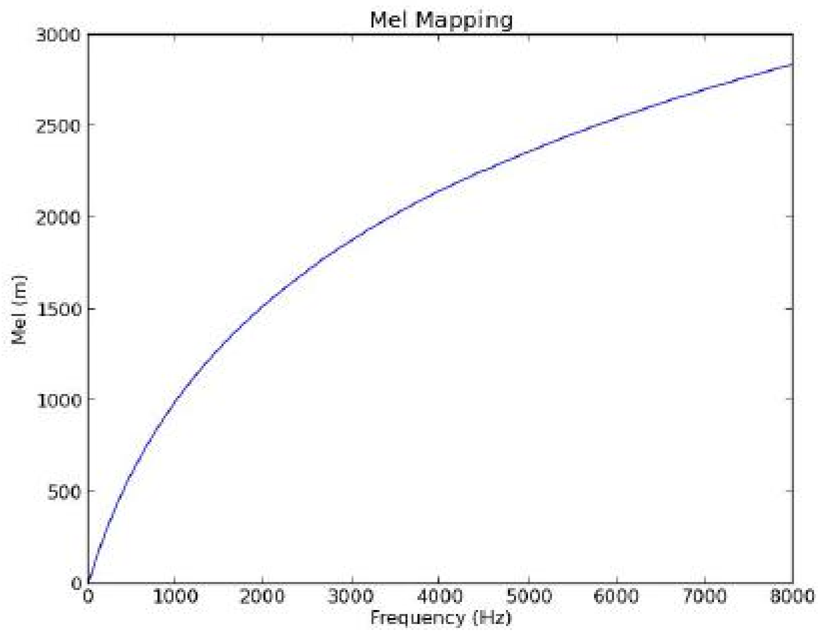
- Transforms speech attributes (frequency, tone, pitch) on non-linear scale based on human perception of voice
- Result: non-linear amplification, MFCC features that mirror human perception
  - E.g. humans good at perceiving small change at low frequency than at high frequency



Step 3: Apply Mel FilterBank

Non-linear conversion from frequency to Mel Space

$$M(f) = 1125 \ln(1 + f/700) \qquad (1)$$



Step 4: Apply Logarithm of Mel Filterbank

- Take log of filterbank energies at each frequency
- This step makes output mimic human hearing better
- We don't hear loudness on a linear scale
- Changes in loud noises may not sound different

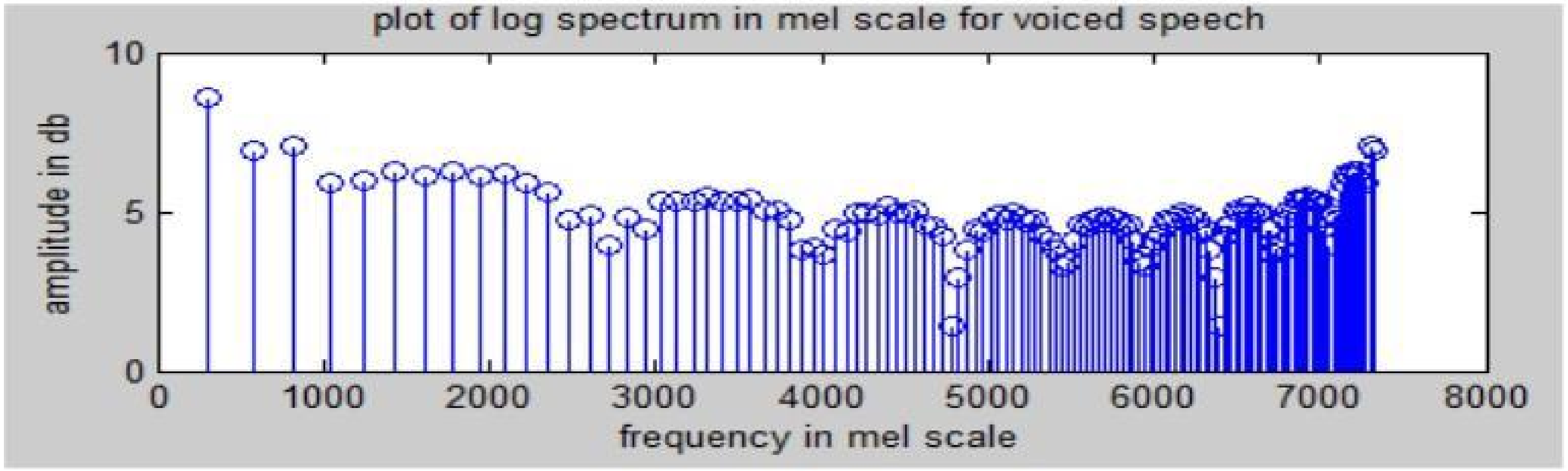


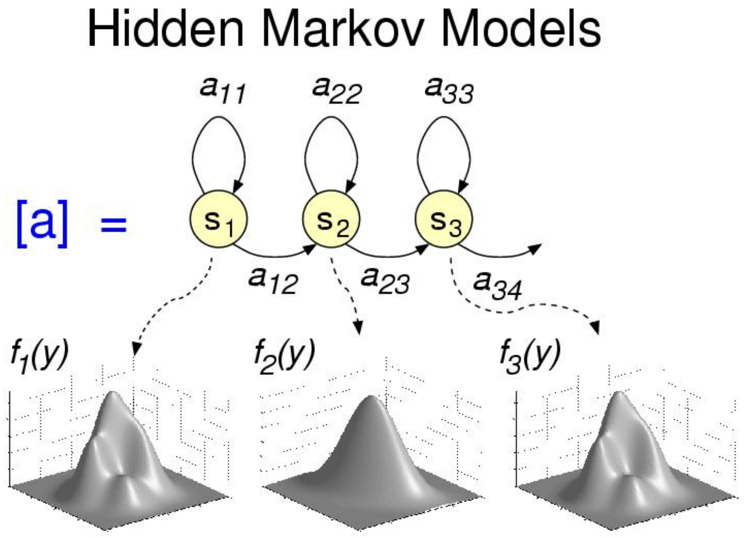
Fig.7. Spectrum of voiced speech

Step 5: DCT of log filterbank:

- There are correlations between signals at different frequencies
- Discrete Cosine Transform (DCT) extracts most useful and independent features
- Final result: 39-element acoustic vector used in speech processing algorithms

Speech Classification

- Human speech can be broken into phonemes
- Example of phoneme is /k/ in the words (**c**at, s**ch**ool, s**k**ill)
- Classic Speech recognition tries to recognize sequence of phonemes in a word
- Typically uses Hidden Markov Model (HMM)
  - Recognizes letters, then words, then sentences
  - Like a state machine that strings together sequence of sounds recognized



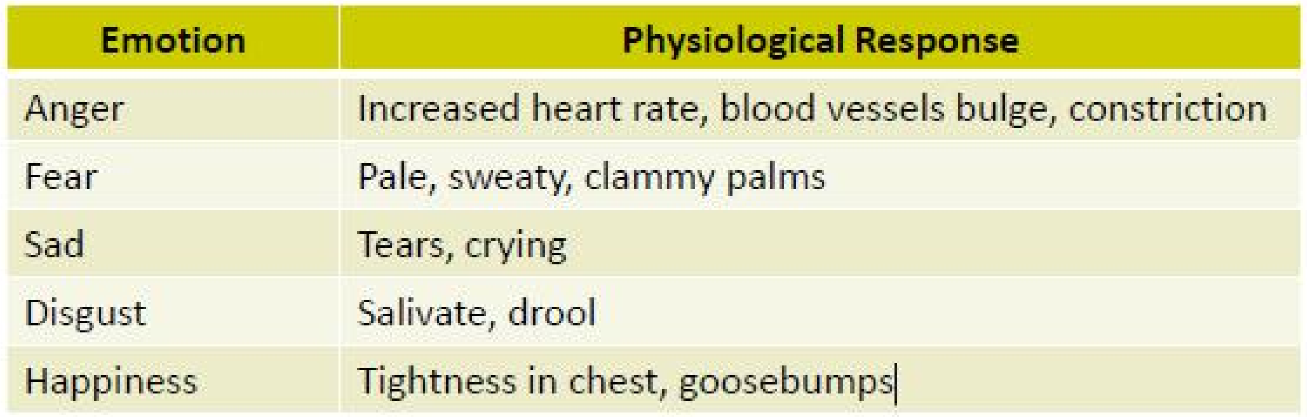
Emotion Detection

Definitions

- Affect
  - Broad range of feelings
  - Can be either emotions or moods
- Emotion
  - Brief, intense feelings (anger, fear, sadness, etc)
  - Directed at someone or something
- Mood
  - Less intense, not directed at a specific stimulus
  - Lasts longer (hours (4?) or days)

Physiological Measurement of Emotion

- **Biological arousal:** heart rate, respiration, perspiration, temperature, muscle tension
- **Expressions:** facial expression, gesture, posture, voice intonation, breathing noise



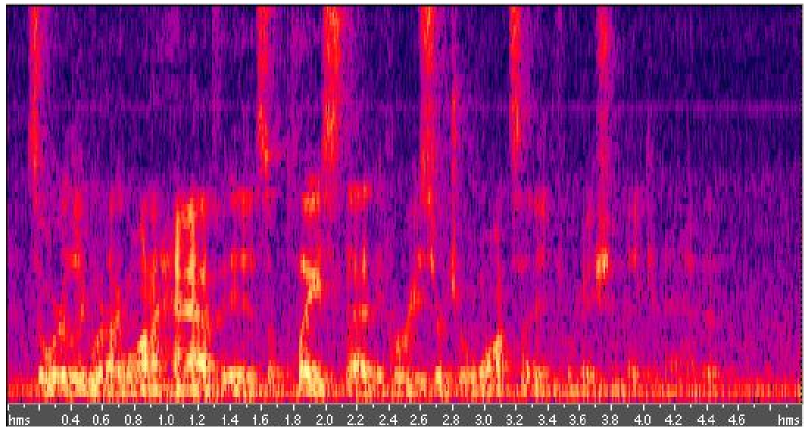
Emotion	Physiological Response
Anger	Increased heart rate, blood vessels bulge, constriction
Fear	Pale, sweaty, clammy palms
Sad	Tears, crying
Disgust	Salivate, drool
Happiness	Tightness in chest, goosebumps

Audio Features for Emotion Detection

- MFCC widely used for analysis of speech content, Automatic Speaker Recognition (ASR)
  - Who is speaking?
- Other audio features exist to capture sound characteristics/dynamics (prosody)
  - Useful in detecting emotion in speech
- **Pitch:** the frequency of a sound wave. E.g.
  - Sudden increase in pitch => Anger
  - Low variance of pitch => Sadness

Audio Features for Emotion Detection

- **Intensity:** Energy of speech, intensity. E.g.
  - Angry speech: sharp rise in energy
  - Sad speech: low intensity
- **Temporal features:**
  - Speech rate, voice activity (e.g. pauses)
  - E.g. Sad speech: slower, more pauses
- **Other emotion features:** Voice quality, spectrogram, statistical measures



Uses of Affect Detection E.g. Using Voice on Smartphone

- Audio processing (especiallyto detect affect, mental health) can revolutionize healthcare
  - Detection of mental health issues automatically from patientsvoice
  - Population-level (e.g campus wide) mental health screening
  - Continuous, passive stress monitoring
  - Suggest interventions: breathing exercises, play relaxing music
  - Monitoring social interactions, recognize conversations (number and duration per day/week, etc)

Voice Analytics Example: SpeakerSense (Lu et al)

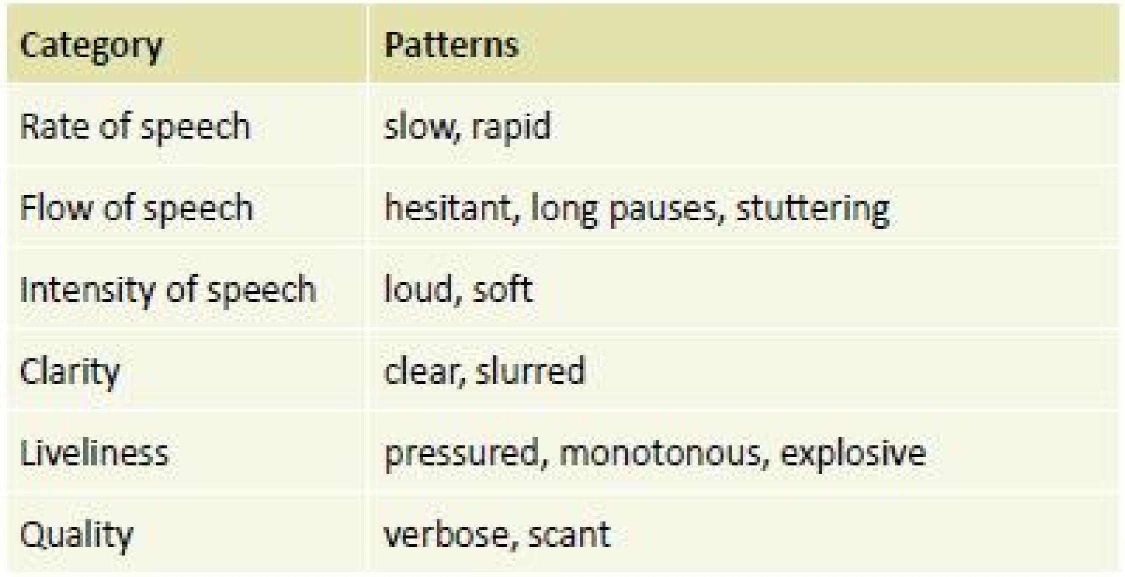
- Identifies speaker, who conversation is with

Voice Analytics Example: StressSense (Lu et al)

- Detected stress in speaker’s voice
- Features: MFCC, pitch, speaking rate
- Classification using GMM
- Accuracy: indoors (81%), outdoors (76%)

Voice Analytics Example: Mental Illness Diagnosis

- What if depressed patient lies to psychiatrist, says “I’m doing great”
- Mental health (e.g. depression) detectable from voice, can be used to detect lying patient
- Doctors pay attention to speech aspects when examining patients
- E.g. depressed people have slower responses, more pauses, monotonic responses and poor articulation



Category	Patterns
Rate of speech	slow, rapid
Flow of speech	hesitant, long pauses, stuttering
Intensity of speech	loud, soft
Clarity	clear, slurred
Liveliness	pressured, monotonous, explosive
Quality	verbose, scant

Detection of COVID from Respiratory sounds

- large-scale crowdsourced dataset of respiratory sounds collected to aid diagnosis of COVID-19.
- Coughs and breathing to understand how discernible COVID-19 sounds are from those in asthma or healthy controls.
- Simple binary machine learning classifier is able toclassify correctly healthy and COVID-19 sounds.
- Were able to distinguish
  - User who had COVID-19 + cough vs healthy user with a cough
  - Users who had COVID-19 + cough vs. Users with asthma and a cough.