Android Application for Language ID*

Pedro Torres-Carrasquillo
Robert A. Ford
Joel C. Acevedo-Aviles

* This work is sponsored by the Department of the Air Force under Air Force Contract #FA8721-05-C-0002. Opinions, interpretations, conclusions and recommendations are those of the author and are not necessarily endorsed by the United States Government.
Outline

• Motivation
• Automatic Language Identification (LID)
• SmartPhone Implementation
• System Performance
• Conclusions and Future Work
• Demo
Motivation

• Any unplanned interaction with someone who does not speak our language requires an interpreter

• Language identification is needed first

• Recent example in San Diego Harbor (March 27, 2011)
  – 26-foot sailboat capsized
  – First responders did not recognize language

“…investigators had to bring in interpreters to speak to them, San Diego Fire-Rescue spokesman Maurice Luque said. He did not know what language they spoke.”
Applications

• Language-based data filtering

• Pre-processing for automated speech applications such as machine translation and speech recognition

• Requesting human interpreters for emergency situations

• Our focus is to develop a Smartphone application that addresses the latter scenario (i.e. routing language service requests in emergency situations)
Project Goals

• Implement a LID system on an Android based Smartphone

• Evaluate tradeoffs between computational complexity and performance across several phones

• Evaluate performance of in-phone LID system with field testing

• Develop a prototype application that integrates existing Smartphone capabilities with LID to quickly and efficiently route language service requests
Automatic Language Identification (LID)

- Automatic language identification is the process of determining the language being spoken in a speech utterance without human intervention.

- Closed-set identification
  - Prior knowledge of all classes

- Open set identification
  - Out of set class

Focus: Closed-set ID
LID System Architecture

- Language identification task: Find messages spoken in a target language

Training speech utterances in known languages:
- Spanish
- English
- German

Set of models: one model per language

Recognition:
- Speech utterance in unknown language
- Feature Extraction
- Recognition Algorithm
- Recognition Output
  - It’s German!
LID System
Feature Processing

- **Spectral Analysis**
  Generate frequency component information through Short-Time Fourier Transform

- **Cepstral Analysis**
  Separates frequency information that is characteristic of a language from what is common across all languages considered (7 features)

- **Channel Compensation**
  Reduces the effects of differences across channels (landline vs cell phone)

- **Deltas**
  Encode temporal variation of Cepstral features by computing the differences among neighboring frames
LID System
Gaussian Mixture Modeling

- Gaussian Mixture Model (GMM)
  - Almost any continuous probability distribution can be approximated by a linear combination of Gaussians
- Each language is modeled as a probability distribution over the feature variables
Android LID System Architecture

- Standard wave file (PCM)
- 16 bits/sample
- 8kHz sample rate

- As the user speaks, streaming audio is sent to the LID component for processing
- The LID system consists of previously developed technology by the HLT group at Lincoln Laboratory (C++)
- Android’s Native Development Kit (NDK) allow us to make use of C++ native code
Main screen

User presses button

User speaks for up to 30s

Detected language is displayed

- App starts capturing user’s speech and does not stop until final decision is displayed
- When a minimum of audio is captured, it is processed
  - Minimum audio is a system parameter; currently 7-seconds
- A score is generated for each language
- If language score > preset threshold
  - Decision is displayed, otherwise
  - Score all audio captured until this point
System performance versus model complexity

Computer Simulation

• Five-Language task: Arabic, Cantonese, English, Mandarin, Spanish

• Test sample nominal length: 30s

• Task: closed-set ID
## SmartPhone configurations

- Multiple phones were evaluated
  - Older platform (HTC Magic)
  - Newer (Atrix)

<table>
<thead>
<tr>
<th></th>
<th>HTC Magic</th>
<th>Samsung Nexus S</th>
<th>HTC Sensation</th>
<th>LG G2x, Motorola Atrix</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>Qualcomm MSM7200A</td>
<td>Hummingbird</td>
<td>Qualcomm MSM 8x60</td>
<td>Tegra 2</td>
</tr>
<tr>
<td>Processor design</td>
<td>ARM 11</td>
<td>Cortex A8</td>
<td>Scorpion</td>
<td>Cortex A9</td>
</tr>
<tr>
<td>Clock</td>
<td>528 MHz</td>
<td>1 GHz</td>
<td>1.2 GHz</td>
<td>1 GHz</td>
</tr>
<tr>
<td>Process</td>
<td>90 nm</td>
<td>45 nm</td>
<td>45 nm</td>
<td>40 nm</td>
</tr>
<tr>
<td>Out of Order Execution</td>
<td>no</td>
<td>no</td>
<td>partial</td>
<td>yes</td>
</tr>
<tr>
<td>Year introduced</td>
<td>2009</td>
<td>2011</td>
<td>2011</td>
<td>2011</td>
</tr>
<tr>
<td>Relative performance</td>
<td>1</td>
<td>6</td>
<td>16</td>
<td>18</td>
</tr>
</tbody>
</table>
Average execution time versus model order
In-SmartPhone Evaluation

Benchmark Performance

- Task: 5-language closed-set Id
- Test sample average duration: 30s
- Averaged over 4 test samples
Average execution time versus model order
In-SmartPhone Evaluation

Benchmark Performance

- Task: 5-language closed-set Id
- Test sample average duration: 30s
- Averaged over 4 test samples
Average execution time for different tasks
In-SmartPhone Evaluation

- Comparison in execution time between 5 and 50-language task
- Fixed model complexity 2048
- 30-sec samples
- Small overhead in computation since in both cases the language independent model is scored and takes most of the processing time

<table>
<thead>
<tr>
<th>Phone</th>
<th>5 Languages</th>
<th>50 Languages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nexus S (Hummingbird)</td>
<td>16.9s</td>
<td>21s</td>
</tr>
<tr>
<td>Motorola Atrix</td>
<td>7.9s</td>
<td>9s</td>
</tr>
<tr>
<td>HTC</td>
<td>10.7s</td>
<td>11s</td>
</tr>
<tr>
<td>Droid X2</td>
<td>5.2s</td>
<td>8.6s</td>
</tr>
<tr>
<td>LGE G2X</td>
<td>7.9s</td>
<td>9s</td>
</tr>
</tbody>
</table>
System performance

• Benchmark Classification Error Rate: 9.1%
  – Computer LID system
  – Development Set

• In-phone evaluation
  – Classification Error Rate: ~20%
  – 7 languages
    Arabic, Mandarin, Vietnamese, Hindi, Russian, Spanish, Turkish
  – Half of captured segments < 10s

• Potential issue with mismatch between 30s system training and amount of speech provided by users
### System performance
#### Impact of test segment duration

- **Match system**
  - Train and test samples of same duration

<table>
<thead>
<tr>
<th>Speech (s)</th>
<th>Classification error rate (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>28.1</td>
</tr>
<tr>
<td>12</td>
<td>16.8</td>
</tr>
<tr>
<td>18</td>
<td>10.2</td>
</tr>
<tr>
<td>24</td>
<td>9.4</td>
</tr>
<tr>
<td>30</td>
<td>9.2</td>
</tr>
<tr>
<td>Full sample</td>
<td>9.1</td>
</tr>
</tbody>
</table>

- **Mismatched system**
  - Trained on full sample length

<table>
<thead>
<tr>
<th>Speech (s)</th>
<th>Classification error rate (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>29.8</td>
</tr>
<tr>
<td>12</td>
<td>17.6</td>
</tr>
<tr>
<td>18</td>
<td>12.6</td>
</tr>
<tr>
<td>24</td>
<td>10.3</td>
</tr>
<tr>
<td>30</td>
<td>9.3</td>
</tr>
<tr>
<td>Full sample</td>
<td>9.1</td>
</tr>
</tbody>
</table>
Conclusions and Summary

• State of the art LID technology has been implemented in Android platform

• Successful evaluation conducted over multiple handsets
  – Newer handset perform in real time

• Additional data is needed to support more in-phone testing
Future Work

• Implementation of robustness techniques to enhance mismatch between training data and telephone

• Can performance be improved by using multiple systems in combination?

• Is current speech activity detection aggressive enough?

• Use speech time instead of audio time

• Compensate for shorter durations
  – Likely main current source of mismatch

• Evaluate power consumption/battery life for field use

• Study the open-set problem

• Leverage current implementation to extend to speaker identification