Demystifying Development of Speech Recognizers for Novices

Anuj Kumar, Florian Metze, Eric Riebling
Carnegie Mellon University
407 S. Craig, Pittsburgh, PA, 15213 USA
{anujk1, fmetze, er1k}@cs.cmu.edu

Matthew Kam
American Institutes for Research,
1000 Thomas Jefferson St. NW, Washington D.C., 20007, USA
mkam@air.org

Abstract
Despite recent popularity of interfaces such as Google Now or Siri, speech-enabled systems are not yet developed in abundance to support every type of user group, language, or acoustic scenario. A core issue is the difficulty involved in building a "reasonably accurate" speech recognizer (even though it may not be 100% accurate). In this paper, we discuss two tools to alleviate this problem: first, "The Speech Recognition Virtual Kitchen" that provides virtual machines to provision pre-existing speech experimental setups for the benefit of developers. Second, "SToNE" that lowers the bar of experience needed to make accuracy improvements to automatic speech recognition.

Author Keywords
Novices, Rapid development, Speech recognition

ACM Classification Keywords
H.5.m. Information interfaces and presentation (e.g., HCI): Miscellaneous.

Introduction
Although very recent, the boom in speech-recognition applications is not surprising, given that they offer several potential advantages. First, speech-based interaction offers truly hands-free, eyes-free interaction, a dream that has evaded us for many years. Second, speech is faster than typing on a keyboard, and without the need for an onscreen keyboard, there is much greater flexibility in terms of screen real estate. Finally, speech-driven applications present important opportunities for the 800 million or so illiterate users in developing regions, giving them a feasible way to access computing.

However, speech interfaces are not yet developed in abundance. From a development perspective, there are several reasons, although in our experience, two challenges stand out. First, a multitude of tools need to be setup and properly configured. Very often, this is a leading source of frustration and time sink for novice developers. Second, even with a proper setup, novices (or even developers with 1-2 years of experience) find it extremely hard to build an accurate speech recognizer for a new user group, language, or acoustic scenario [Laput, 2013]. Below we describe two ideas to address these difficulties.

Author's Note: Interest in Workshop

The workshop hits a key point, that speech recognition accuracy need not be 100% to build a useful speech application. Yet, there are not many examples of HCI designers actively incorporating speech in their applications, when in fact, a lot of users can benefit from speech-based functionalities. Our belief is that though 100% accuracy is not needed, a somewhat reasonably accurate recognizer needs to be developed. While the core speech recognition technology has reached a point where, in principle, speech recognizers can be developed and fine-tuned for every possible user group, language or acoustic scenario, it is still very difficult for non-speech developers to engage in the development and fine-tuning process of speech recognizers. This, in our view, is the core problem – and by bringing together HCI and speech technologists, the workshop hits the right note. In this paper, we briefly describe two projects that aim at simplifying the process of developing a "reasonably accurate" speech recognizer. Note the benchmark of what classifies as "reasonable accuracy" is debatable, but our vision is to simplify the process involved in adapting and building a speech recognizer for a new user group so that every application designer can as easily integrate speech modality in their work, as they do with other modalities like touch or keyboard input. Also note, this paper discusses the design of our projects, and we have left the discussion of early results for the workshop itself.

The Speech Recognition Virtual Kitchen

The first challenge in speech recognizer development is getting the correct setup. The setup not only involves getting the correct toolkits, scripts, data, compatible versions of software, etc. installed correctly but also involves setting up the previously finished experiments (by other, more experienced researchers) exactly as they had intended. The latter step is the most challenging part, but is vital as novice developers can get tremendous information on how to either replicate the process (for a different user group), or improve the work (for the same user group) without spending too much time on properly setting up the baseline.

In the "Speech Recognition Virtual Kitchen" (www.speechkitchen.org) project, we aim to build a community research and education infrastructure in speech recognition. The "kitchen" environment aims to promote community sharing of research techniques, foster innovative experimentation, and provide solid reference systems as a tool for education, research, and evaluation with a focus on, but not restricted to, speech and language research. The core of the research infrastructure is the use of virtual machines (VMs) that provide a consistent environment for experimentation. We liken the virtual machines to a "kitchen" because they provide the infrastructure into which one can install "appliances" (e.g., speech recognition toolkits), "recipes" (scripts for creating state-of-the art systems), and "ingredients" (language data). Below we describe

what a developer- user can achieve with this infrastructure.

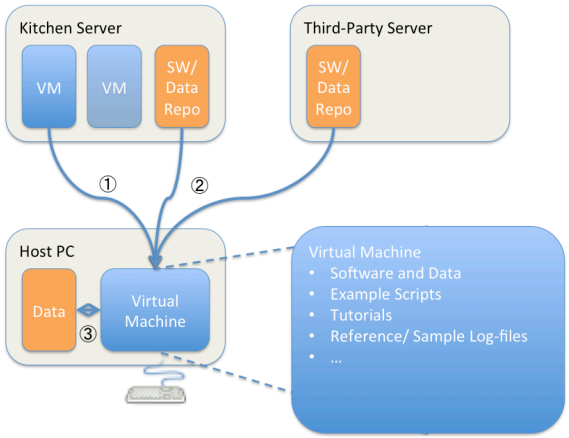

Figure 1: Speech Kitchen provides Virtual Machines that can self-configure to suit developer-user's requirements, e.g. they can self-configure to install relevant speech toolkits (Kaldi, Sphinx, etc.), scripts (for speaker adaptation, feature extraction, scoring, etc.), and data (for training and testing). They can also download "published" experiments as exemplars along with all the log files and documentation for guiding the development process.

Download Self-Configuring Virtual Machines

The development of speech recognizers is, in most circumstances, best suited for Linux based systems. With this in mind, Virtual Machines (VMs) provide two primary advantages: first, they make the development of speech recognizers independent of the host operating system, e.g. developers who have other OS's on their machines can participate without having to go through a lengthy disk partitioning to install Ubuntu or other Linux-based OS's. Second, VMs provide a consistent way to package an entire experiment consistently, and distribute one's work widely.

In addition to those advantages, in Speech Kitchen, we provide self-configuring Virtual Machines. This means that users can get setup with a Virtual Machine that is configured to their requirement, i.e. with a toolkit (Kaldi, Sphinx, etc.) of their choice, scripts they need, and the data they require for their work. Moreover, it provides a way to ensure that latest, stable, and compatible versions of all software is installed, which is often a big source of developer pain in complex systems such as speech recognizers.

Platform to Share Published Experiments

Speech Kitchen provides a way for the speech recognition experts to share their experiments (results, acoustic and language models, logs, etc.) with the broader community in a consistent way, i.e. by packaging their entire setup in form of standard .deb files. Typically, published results are the de-facto medium of sharing one's work, but in large, complex systems, every small nuance is hard to document in a paper. A standardized method can greatly help improve community and knowledge sharing.

SToNE: A Speech Toolkit for Non-Experts

While the Speech Recognition Virtual Kitchen helps get developers' setup the development environment correctly, the developers may next struggle with improving recognition accuracy beyond setting up a baseline. Our second project, SToNE – A Speech Toolkit for Non-Experts addresses exactly that problem. Based on what the developer is working on, it provides technical guidance on understanding "why might the

recognizer be failing?" or "what to do to fix it?" Below we describe the two essential modules that make up SToNE.

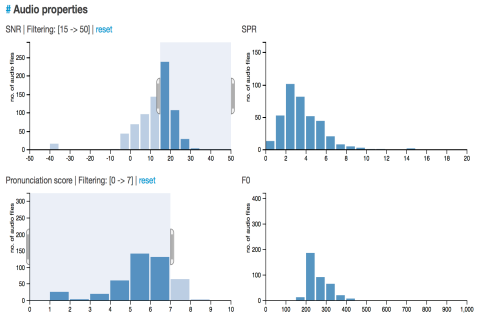

Figure 2: The visualizations module displays the distribution of test data along several metrics. The user can also select smaller subsets of this data to compare and contrast recognition accuracy, and identify reasons of error.

Feature Extractor & Error Analysis Visualizer
This module assists in answering the first question: why is the recognizer failing? A speech recognizer can fail because of many reasons, ranging from acoustic issues (pronunciation, speaking rate, noise, speaking pitch, etc.) to language modeling issues (out-of-vocabulary, high perplexities, etc.), or more generally, a mismatch between training and testing dataset.

For a given test dataset, this module does two things: first, it automatically extracts relevant features that correspond well to known reasons of failure, per-audio-file. Second, it plots these features in a visualizations module along with accuracy correlations to help understand reasons of error. The visualizations module also helps in comparing two subsets of the test dataset (for instance, a high performing subset and a low performing subset) to contrast and compare reasons of respective performance.

Knowledge Base & Optimization Advisor
This module helps in identifying relevant speech techniques that can improve the recognition accuracy beyond a baseline. It does so by utilizing the quantified values of several metrics (from the Feature Extractor module) and performing a univariate and multivariate regression analysis with accuracy (as the dependent variable) to identify the most significant factors impacting recognition.

Next, it looks up a rule-based knowledge base to recommend the corresponding optimization technique(s). This rule-based knowledge base was developed as a result of contextual interviews with speech recognition experts at Carnegie Mellon University, and results of this work were published in InterSpeech [Kumar, 2013]. At the workshop, we will discuss these results and also showcase working implementation for both the above projects.

Acknowledgements

This work is supported by NSF Grant Nos. CNS-1205589 & IIS-1247368.

References

[1] Laput G.P, et al. PixelTone: A Multimodal Interface for Image Editing. In Proc. ACM *CHI*, 2013, pp. 2185–2194.

[2] Kumar, A., Metze, F., Wang, W., Kam, M. Formalizing Expert Knowledge for Developing Accurate Speech Recognizers. In Proc. *InterSpeech*, ISCA; Lyon, France, 2013.