INTERSPEECH 2016
September 8–12, 2016, San Francisco, USA

Virtual Machines and Containers as a Platform for Experimentation

Florian Metze[1], *Eric Riebling* [1], *Anne S. Warlaumont*[2], *and Elika Bergelson*[3]

[1]Language Technologies Institute, Carnegie Mellon University; U.S.A.
[2]Cognitive and Information Sciences, University of California, Merced; U.S.A.
[3]Center for Language Sciences, University of Rochester; U.S.A., and
Psychology & Neuroscience Department, Duke University; U.S.A.

`fmetze@cs.cmu.edu`

Abstract

Research on computational speech processing has traditionally relied on the availability of a relatively large and complex infrastructure, which encompasses data (text and audio), tools (feature extraction, model training, scoring, possibly on-line and off-line, etc.), glue code, and computing. Traditionally, it has been very hard to move experiments from one site to another, and to replicate experiments. With the increasing availability of shared platforms such as commercial cloud computing platforms or publicly funded super-computing centers, there is a need and an opportunity to abstract the experimental environment from the hardware, and distribute complete setups as a virtual machine, a container, or some other shareable resource, that can be deployed and worked with anywhere.

In this paper, we discuss our experience with this concept and present some tools that the community might find useful. We outline, as a case study, how such tools can be applied to a naturalistic language acquisition audio corpus.

Index Terms: speech processing, reproducible research, citizen science, shared platforms, cloud computing

1. Introduction

Building and maintaining a state-of-the-art Automatic Speech Recognition (ASR) or Natural Language Processing (NLP) system has become a very complex task: it requires installing large amounts of text and audio data, normalizing them, training acoustic and language models with different tools, combining them to decode test data, perform error analysis, etc. It requires more than one expert to maintain an end-to-end system, and adapt it to new languages, tasks, or conditions. Even rebuilding an existing baseline is prone to errors.

This effort attempts to extend the model of lab-internal knowledge transfer to a community-wide effort through the use of Virtual Machines and/ or Containers. The Speech Recognition Virtual Kitchen (SRVK) [1] presents a way to share not just "recipes", but ready-to run experimental environments – hence the analogy to a kitchen: our setup contains tools, data, logfiles and results from baseline runs, so that classes, evaluations, workshops, or collaborating researchers can easily share setups and compare results meaningfully. In this paper, we present first results from a case study with the HomeBank [2] repository.

The kitchen already provides a number of "dishes", yet we **encourage community feedback and contributions** in order to further develop all the ideas presented here. **Please visit** `http://speechkitchen.org/` or `https://github.com/srvk`.

2. Related Work

The need for shared, reproducible tools in research has been recognized by several other groups and communities, and has already led to many tools being made available on public, shared repositories such as GitHub [3], and under an Open Source license (e.g. Apache 2.0 [4]). In the speech community, Kaldi [5] is certainly the most successful such project, given the large number of "recipes" it contains, some of which rely on Open Source data, e.g. the LIUM corpus [6].

Box [7] is a similar project from the NLP community, which attempts to provide a disk image with several tools already pre-installed. Code and data repositories such as Codalab [8] provide an environment for reproducible research, while repositories such as Covarep [9] strive to make sharing of implementations easier. In the speech community, WebASR [10] and CloudCAST [11] aim to provide a shared infrastructure, and some access to components for researchers, rather than treating ASR as a black box.

The SRVK combines aspects from all of these, in that we aim to provide a ready-to-use tool, that can be run in the cloud or at the client's site. Distributing it as a VM provides complete control over all aspects of processing to the user, allowing for customizations easily, and access to source code and data. As such, the concept is also suitable for education; this concept is discussed in a companion paper [12].

Previously, an SRVK Virtual Machine has been used to distribute a baseline to MediaEval's "QUESST" task [13], which participants found very helpful. An earlier, related effort created tools to facilitate the building of speech recognizers by non-experts [14, 15, 16].

3. The "Kitchen" Concept

The main idea behind the kitchen is to distribute Virtual Machines (VMs) to the community, which contain Open Source software, so that users can freely download, copy, and/ or modify them. A typical kitchen resource is made available as a Git repository containing mostly scripts, in particular a `Vagrantfile` [17]. The user can execute it on his or her own computer or cluster, irrespective of operating system, using Virtualbox [18]. Alternatively, a cloud computing resource can be used; currently, we support Amazon Web Services (AWS) [19]. With this mechanism, it is not necessary to distribute pre-compiled, monolithic VMs, which can easily reach Gigabytes in size, but rather small text files that describe how a VM can be built (and re-built) from scratch, usually in a matter of minutes. Docker [20] containers may also be used.

A typical work-flow looks as follows (cf. Figure 1):

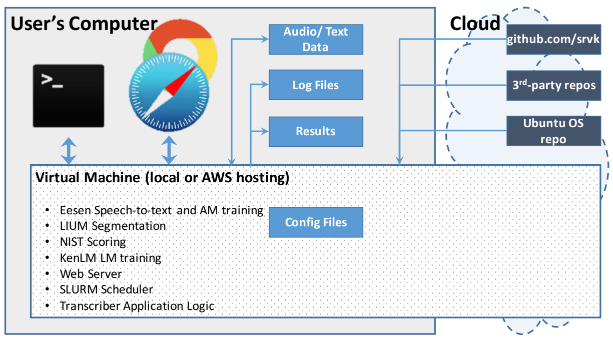


Figure 1: "Eesen Transcriber" architecture: the user controls the VM with a terminal window and/ or a browser on his or her computer, irrespective if the VM has been provisioned on the user's computer, or in the cloud. Data, log files, and results reside in the "shared" folder on the user's computer, while configuration files mostly reside inside the VM, along with the application logic, for portability.

Check out a VM using e.g. `git clone https://github.com/srvk/eesen-transcriber.git`, and `cd` into the newly created directory structure, known as the "shared" folder.

Configure the VM to better match the amount of memory or number of processors used by the VM to the resources available on the host (often, this step may not be needed)

Bring up the VM using `vagrant up` or `sudo docker build`. If required, the VM will pull additional resources from the web. At the end of this "provisioning" step, the VM will print further instructions on how to continue

Work with the VM: This can involve copying data into the shared folder, or accessing a web browser, executing commands in the VM using `vagrant exec`, or logging into the VM to execute commands, debug the application, or further develop the VM

Shut down the VM, leaving it available for re-starting, continuing work from where one left off, or wiping it from the user's computer entirely

In all cases, the repository contains documentation and pointers to shared documentation and community resources that are available online.

4. The "Eesen Transcriber" VM

After discussion with several potential users, easy-to-use speech-to-text that can be treated as a black box, but does permit access to the internals if required, emerged as a central need of the community. We thus combined our recently released "Eesen" ASR toolkit [21] with an Open Source "transcriber" [22] and segmentation [23] framework, and added several additional functionality, releasing the system with English acoustic models trained on TEDLIUM data [24], and a large general purpose language model [25].

Once the system has been started as described in the previous section, the user is given a URL, which points to a control and help page for the current VM. This page is being hosted on the VM, and allows the user to control most aspects of the VM during "normal" operation. The command line interface (via `vagrant ssh` or `vagrant exec`) gives complete control over the VM, including sudo privileges.

4.1. Acoustic Model

The acoustic model is based on a bi-directional Long Short-Term Memory (LSTM) network, which has been trained using the Connectionist Temporal Classification (CTC) criterion [26], which directly optimizes the sequence of output symbols, rather than the frame likelihood. The VM comes with a pre-trained Acoustic Model, so that the user is able to immediately start decoding, and does not need to provide GPU hardware in order to train it. All the scripts are provided however, so that a user can start the VM on a GPU-enabled machine (for example an AWS g2.8xlarge instance), in order to re-train the model, adapt it, or use the provided recipe as a blueprint to train a model for a different language or task.

CTC is a good choice as an optimization criterion, because the model does not depend on the initialization (the alignment is being marginalized out during the optimization of the network), and there is only a single, context independent (CI) model that has to be built. As an example, the "Eesen" training recipe to build a Wall Street Journal (WSJ) system has less then 100 lines of code including data preparation and testing, while the corresponding "Kaldi" recipe (from which the "Eesen" one is derived) has more than 450 lines of code, and reaches about the same Word Error Rate (WER).

CI acoustic models also result in smaller Weighted Finite State Transducer decoding graphs, and decode faster, in particular with the 30ms frame shift that is currently being used.

4.2. Language Model

Similarly, a pre-compiled tri-gram Language Model is provided, but the VM comes with instructions on how to rebuild the language model, or how to adapt it to different domains. As we expect that more users will want to adapt the language model (and training does not require a GPU node), these instructions are quite detailed, and all required software (currently, we use KenLM [27]) is already installed and set up.

Future developments will also allow the integration of neural network language models, resulting in an all-neural, "end-to-end" experimental work-bench in a VM.

4.3. Running on Own Data

"Eesen Transcriber" implements a complete end-to-end pipeline for processing, indexing and searching audio. Processing can be triggered by either "dropping" an audio file in a "watched folder" on the user's host, in which case a daemon on the VM will pick up the file and trigger processing, or by manually triggering processing from the host using a command like `vagrant ssh -c "vids2web.sh /videos/demo.mp4"` which will directly execute a script on the VM with a file that can be found on the host. The VM uses shared folders or `sshfs` to make file systems accessible in a transparent way.

The following steps are currently supported out of the box:

Normalization of arbitrary audio files to 16kHz sampling frequency and PCM file format

Diarization of the audio file: mono and stereo files can be segmented automatically into speakers, or a provided STM file can be read in order to use a given segmentation. The VM contains documentation on how to alter some of the parameters inherent to the segmentation, if the segmentation is found to be performing badly.

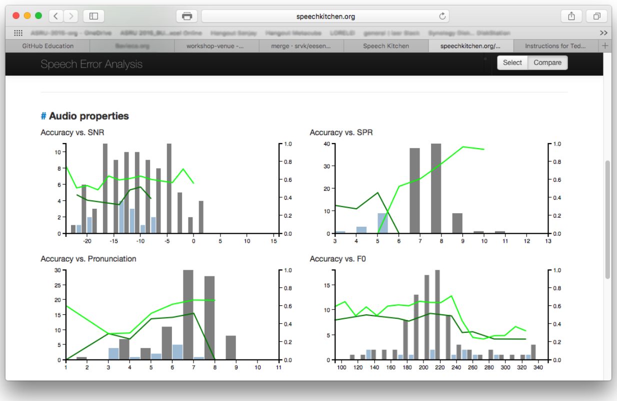
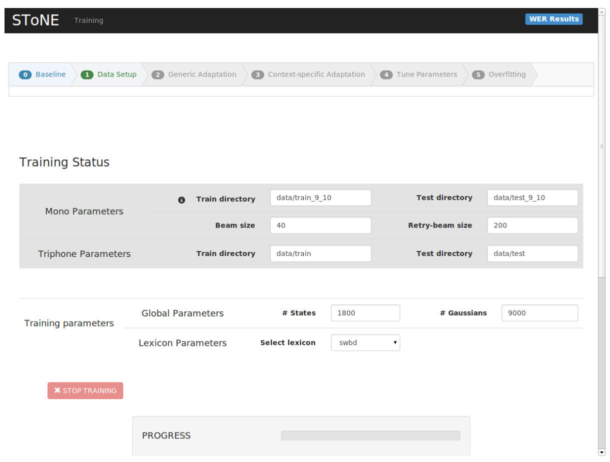


Figure 2: The data analysis tool originally developed in [30]. In this example, the user selected two parts of the data: young children (blue histogram bars), and older children (grey bars). The accuracy for young children (dark green line) is lower than for older children (light green line). The main difference (see the "Accuracy vs. SPR" plot in the top right) is that young children speak slower (lower "speaking rate" value).



SToNE Training | WER Results

0 Baseline | 1 Data Setup | 2 Generic Adaptation | 3 Context-specific Adaptation | 4 Tune Parameters | 5 Overfitting

Training Status

Mono Parameters
- Train directory: data/train_9_10
- Test directory: data/test_9_10
- Beam size: 40
- Retry-beam size: 200

Triphone Parameters
- Train directory: data/train
- Test directory: data/test

Training parameters
- Global Parameters: # States: 1800; # Gaussians: 9000
- Lexicon Parameters: Select lexicon: swbd

✖ STOP TRAINING

PROGRESS

Figure 3: The "SToNE" training module (from [30]), which is being adapted to the "Eesen" framework, and its unique set of parameters. Such a framework allows a user to train a system by selecting fundamental parameters in a GUI, and follow along a typical development process.

Feature extraction of lMEL and pitch features (by default); other approaches can be implemented easily if required

Speech-to-text to produce hypotheses, lattices, word confidences, and n-best lists with time alignments; a number of output file formats are supported, e.g. CTM, VTT, SBV, etc.

Indexing of the ASR output in a simple database, implemented within a standard web server

Browsing and Searching in a web browser on the user's computer, permitting play-back of audio and video with subtitles, similar to a video library [28] or a lecture browser [29]

Processing controlled by the SLURM workload manager, so that any amount of data can be "dumped" on the VM with a single action. Command shortcuts are being provided for the most important tasks such as rebuilding the index, cleaning cached or intermediate files, processing files, etc. While most of the configuration files are kept within the VM, most of the data files and indices are kept outside of the VM, so that the user has easy access to all of them, and can process them further or store them as required. Figure 1 shows the most important components.

4.4. Analyzing Results

Error analysis is an important part of system development, which is often tedious for experts and sometimes impossible for non-experts, because they do not know what to look for. Figure 2 (generated on `http://speechkitchen.org/home/error-analysis-tools/`) shows a selection of online tools, which we are currently porting into the VM. It will compute a number of features such as signal-to-noise ratio, speaking rate, pronunciation score, fundamental frequency, and power for training and test data, and allow the user to correlate these with known demographics of the data (age, gender, dialect, etc.). The VM further integrates the NIST SCTK scoring tools [31], and keeps track of out-of-vocabulary words.

The VM thus provides tools to identify mis-matched data and the reasons behind poor performance, and enables the user to further develop the system accordingly.

4.5. Re-training the Models

Leveraging earlier work [30], we are currently porting the Kaldi-based "SToNE" (Speech Toolkit for Non-Experts) "guided" training module to Eesen and the Transcriber VM, which will allow a non-expert user to develop a usable speech recognition system. It uses an iterative process, in which a rule-based expert system analyzes the system's errors, and gives recommendations as to which parts of the system should be modified, in order to improve performance.

The focus is less on exploring the best choice of neural network architecture, but on getting the fundamentals in place, e.g.: *What is the out-of-vocabulary rate? Is the test data too conversational? Are errors correlated with noise? Etc.*

4.6. Performance and Characteristics

The "Eesen Transcriber" is currently distributed with a TEDLIUM version 2 [24] system, which delivers a WER of less than 15% on the development set, using the pruned Cantab LM [25]. It decodes about four times faster than real-time on a single processor, and can be retrained from scratch within 6 days on a single GPU.

Any "Eesen" or Kaldi recipe can be integrated into the "Transcriber" VM, if the user has access to the required data, either as an additional download, or as a build experiment.

4.7. Prospects

The "Eesen Transcriber" VM thus presents a complete speech processing toolbox which can be used to process and analyze arbitrary audio. It will ingest any audio or video file in a format known to SoX and/ or FFmpeg, and allows the user to index, browse, and search his collection. If a reference is provided, it will compute various statistics and visualizations that can be helpful in tuning or developing a system.

The "Transcriber" VM separates data (outside of the VM) from tools (inside of the VM), and keeps the "non-user-serviceable" parts hidden from the novice user. It gives a semi-expert easy access to all the components of the system and allows him to make modifications or improvements as required. Because all the code is assembled using a provisioning solution such as Vagrant or Docker, a system can always be re-created from scratch, and variants can be created and shared easily using a version control system (Git in our case).

5. Case Study: the HomeBank Repository

Many child language researchers are now using LENA recorders to collect highly naturalistic, long-form audio samples. The recorders store up to 16h of continuous audio on small recording devices worn by children [32]. The audio files can then be processed using LENA's proprietary segmentation and diarization. Recordings of this kind have been successfully used in semi- or fully automated applications to, e.g. help diagnose children with language delays and deficits in a reliable way [33], to help understand the role of parental responses in speech development [34], and to help understand the differences in language input to children in low-income households [35]. Such recordings have also been used to examine how the acoustic properties of real-world mother-child exchanges evolve over time [36].

The behavioral scientists involved in these studies often have very little computer science training. The closed, black-box nature of available commercial signal processing solutions has been found to be a roadblock when trying to process the above naturalistic recordings: available segmentation and diarization methods have shown reasonable reliability compared to human listeners, and have been demonstrated to be sufficient for some scientific and applied purposes. Yet, the algorithm has not been substantially updated since the most popular software was released for purchase a decade ago, and quality is unsatisfactory on large parts of the data that is available today. Clearly, easy access to open and state-of-the-art speech technology should enable collaboration and speed up analysis.

From a speech technology standpoint, the first challenge has been developing automatically derived measures that perform well on audio files that contain noisy recordings from a single microphone source and where speaker demographics, speech registers, and background noise vary widely. This makes speaker diarization and segmentation challenging for even state-of-the-art algorithms.

With this experience in mind, the "Eesen Transcriber" has been enhanced to accept available manual or automatic segmentations of the test data. New and existing segmentation and diarization solutions from the few labs that have both data and technical expertize are currently being tested and prepared for integration into a VM. Users of the system will thus benefit from application of alternative segmentation and diarization methods, which have the potential to be more accurate as well as to provide more detailed information.

Additionally, the LENA system does not attempt any speech transcription. Here we explore the possibility of applying the latest ASR methods to HomeBank data. HomeBank [37] is a repository for long-form child audio recordings, their metadata, human transcriptions, and code used to process and analyze this data [2]. Presently, the modal use of HomeBank is for LENA recorded files. ASR would be a hugely beneficial addition to LENAs automatic segmentation and diarization, and other long-form audio recordings. For example, an ability to detect a specified set of keywords could be very useful for scientists interested in how natural audio environments supports word learning, which should be an achievable goal [30, 38].

Pilot testing using audio recorded by the LENA system with the Speech Kitchen VMs provides grounds for both caution and optimism. We conducted two preliminary analyses. First, we used "out of the box" ASR settings, and gold-standard human segmentation on HomeBank's public VanDam dataset (157 5-minute segments), focusing on word recognition accuracy. Initial results indicated accuracies just below $<50\%$ using the general language model. Experiments with better data normalization and adaptation of the language model (also supported by the Transcriber) are ongoing. Subsets of the data however achieved $>90\%$ word accuracy.

We also conducted a second analysis on a small sample of human-selected "clean" (i.e. relatively low background noise) conversational blocks from the Bergelson SEEDLingS corpus [39], including 629 words from 9 exchanges. (We defined conversational blocks using LENAs definition, as regions with at least 5s of silence on each end). Here too results were modest but promising. While overall accuracy was low, 1/3 of conversations attained $>70\%$ word-level accuracy. Eesen's word confidence ratings were significantly correlated with their accuracy, as compared to human transcription (Spearman's $\rho = 0.4$; $p < .0001$). Moreover, the VM-generated segmentation was far better aligned to human intuition than the LENA alternative, which over-segments vis-à-vis human listeners.

This pilot work highlights two directions for improvement. First, language models trained on spontaneous adult- and child-directed speech, with iterative manual corrections and appropriate "dictionaries" should improve performance significantly. Second, a pipeline that selects segments that are "clean enough" may prove to be a critical step for making the Eesen VM useful for naturalistic data analysis. More specifically, errors could be reduced by eliminating or separately tagging child speech, background noise, overlap, and signal distortions.

Despite the large room for future improvements, our preliminary results indicate that Speech Kitchen ASR will be a useful addition to the toolkits of researchers collecting highly naturalistic, long-form child audio recordings. In our experience, VMs have shown to be stable, reproducible processing pipelines both for child language researchers applying the methods and for those developing new algorithms for application to this type of data.

6. Outlook and Future Work

SRVK is currently preparing several dishes, to be served, amongst others, at INTERSPEECH tutorials. In addition to the improvements discussed above, the "Eesen Transcriber" VM might be expanded to include not just speech retrieval, but a complete multi-media summarization and retrieval system, which can also explore visual features.

Other menu items are designed to help with teaching courses, or exploring speech synthesis. Information can be found at `http://speechkitchen.org/`, code and downloads at `https://github.com/srvk`. A demo of some of these VM will be shown at the INTERSPEECH 2016 special session [40], and copies will be made available to attendants.

HomeBank and DARCLE are especially potent diners for the dishes served by the Speech Kitchen because one main goal of the group is to develop a fully open-source pipeline for data analysis. The "Eesen Transcriber" VM is a perfectly suited basis for this task. Relatedly, a subset of HomeBank data is currently being prepared for a Computational Paralinguistics Challenge.

The authors would like to thank Anuj Kumar for his work on the error analysis and system building web application. We would also like to thank Brian MacWhinney and Melanie Soderstrom.

This work used the Extreme Science and Engineering Discovery Environment (XSEDE), which is supported by U.S. National Science Foundation grant number OCI1053575. The Speech Recognition Virtual Kitchen is supported by the NSF under grant numbers IIS-1247368, CNS-1305365, 1305319, and 1305215. HomeBank is supported by NSF SBE-1539129, 1539133, and 1539010.

7. References

[1] F. Metze, E. Fosler-Lussier, and R. Bates, "The speech recognition virtual kitchen," in *Proc. INTERSPEECH*. Lyon; France: ISCA, Aug. 2013, https://github.org/srvk.

[2] M. VanDam, A. S. Warlaumont, E. Bergelson, A. Cristia, M. Soderstrom, P. DePalma, and B. MacWhinney, "Homebank, an online repository of daylong child-centered audio recordings," *Seminars in Speech and Language*, 2016, in press.

[3] https://github.com/.

[4] http://www.apache.org/licenses/.

[5] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlíček, Y. Qian, P. Schwarz, J. Silovský, G. Stemmer, and K. Veselý, "The Kaldi speech recognition toolkit," in *Proc. Automatic Speech Recognition and Understanding (ASRU), 2011 IEEE Workshop on*. Waikoloa, HI; U.S.A.: IEEE, Dec. 2011.

[6] A. Rousseau, P. Deléglise, and Y. Estève, "TED-LIUM: an automatic speech recognition dedicated corpus," in *Proc. LREC*. Istanbul; Turkey: ELRA, May 2012.

[7] A. Axelrod, "Box: Natural language processing research using amazon web services," *The Prague Bulletin of Mathematical Linguistics*, vol. 104, no. 1, pp. 27–38, 2015.

[8] http://codalab.org.

[9] G. Degottex, J. Kane, T. Drugman, T. Raitio, and S. Scherer, "Covarepa collaborative voice analysis repository for speech technologies," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. Florence; Italy: IEEE, May 2014, pp. 960–964.

[10] T. Hain, A. El Hannani, S. N. Wrigley, and V. Wan, "Automatic speech recognition for scientific purposes-webasr." in *Proc. INTERSPEECH*, Brisbane; Australia, Sep. 2008.

[11] P. Green, R. Marxer, S. Cunningham, H. Christensen, F. Rudzicz, M. Yancheva, A. Coy, M. Malavasi, and L. Desideri, "Remote speech technology for speech professionals - the CloudCAST initiative," in *6th Workshop on Speech and Language Processing for Assistive Technologies (SLPAT)*, 2015.

[12] R. Bates, E. Fosler-Lussier, F. Metze, M. Larson, G.-A. Levow, and E. M. Provost, "Experiences with shared resources for research and education in speech and language processing," in *Proc. INTERSPEECH*. San Francisco, CA; U.S.A.: ISCA, Sep. 2016, submitted.

[13] X. Anguera, L. J. Rodríguez-Fuentes, I. Szöke, A. Buzo, and F. Metze, "Query by example search on speech at Mediaeval 2014," in *Proc. MediaEval Workshop*, Barcelona; Spain, Oct. 2014, http://ceur-ws.org/Vol-1263.

[14] A. Kumar, F. Metze, W. Wang, and M. Kam, "Formalizing expert knowledge for developing accurate speech recognizers," in *Proc. INTERSPEECH*. Lyon; France: ISCA, Aug. 2013.

[15] A. Kumar, F. Metze, and M. Kam, "Enabling the rapid development and adoption of speech-user interfaces," *IEEE Computer Magazine*, vol. 46, no. 1, Jan. 2014, http://dx.doi.org/10.1109/mc.2014.11.

[16] A. Kumar, F. Metze, E. Riebling, and M. Kam, "Demystifying development of speech recognizers for novices," in *Proc. Designing Speech and Language Interactions Workshop at CHI*. Toronto; Canada: ACM, Apr. 2014.

[17] https://vagrantup.com/.

[18] https://virtualbox.org/.

[19] https://aws.amazon.com/.

[20] https://docker.com/.

[21] Y. Miao, M. Gowayyed, and F. Metze, "EESEN: End-to-End Speech Recognition using Deep RNN Models and WFST-based Decoding," in *Proc. Automatic Speech Recognition and Understanding Workshop (ASRU)*. Scottsdale, AZ; U.S.A.: IEEE, Dec. 2015, https://github.com/srvk/eesen.

[22] https://github.com/alumae/kaldi-offline-transcriber.

[23] M. Rouvier, G. Dupuy, P. Gay, E. Khoury, T. Merlin, and S. Meignier, "An open-source state-of-the-art toolbox for broadcast news diarization," in *Proc. INTERSPEECH*. Lyon; France: ISCA, Sep. 2013.

[24] A. Rousseau, P. Deléglise, and Y. Estève, "Enhancing the ted-lium corpus with selected data for language modeling and more ted talks," in *Proc. LREC*. Reykjavik; Iceland: ELRA, May 2014.

[25] W. Williams, N. Prasad, D. Mrva, T. Ash, and T. Robinson, "Scaling recurrent neural network language models," in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*. Brisbane; Australia: IEEE, May 2015.

[26] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *Proceedings of the 23rd international conference on Machine Learning*. ACM, 2006, pp. 369–376.

[27] K. Heafield, "KenLM: faster and smaller language model queries," in *Proceedings of the EMNLP 2011 Sixth Workshop on Statistical Machine Translation*, Edinburgh, Scotland, United Kingdom, July 2011, pp. 187–197. [Online]. Available: http://kheafield.com/professional/avenue/kenlm.pdf

[28] A. G. Hauptmann and M. J. Witbrock, "Informedia: News-on-demand multimedia information acquisition and retrieval," in *Intelligent Multimedia Information Retrieval*, M. T. Maybury, Ed. Cambridge, MA, USA: MIT Press, 1997, pp. 215–239. [Online]. Available: http://dl.acm.org/citation.cfm?id=266260.266279

[29] J. R. Glass, T. J. Hazen, D. S. Cyphers, K. Schutte, and A. Park, "The mit spoken lecture processing project," in *Proc. NAACL*, 2005.

[30] A. Kumar, "Enabling non-speech experts to develop usable speech-user interfaces," Ph.D. dissertation, School of Computer Science, Carnegie Mellon University, Aug. 2014, cMU-HCII-14-105.

[31] http://www.nist.gov/itl/iad/mig/tools.cfm.

[32] https://www.lenafoundation.org.

[33] D. Oller, P. Niyogi, S. Gray, J. Richards, J. Gilkerson, D. Xu, U. Yapanel, and S. Warren, "Automated vocal analysis of naturalistic recordings from children with autism, language delay, and typical development," *Proceedings of the National Academy of Sciences*, vol. 107, no. 30, pp. 13 354–13 359, 2010.

[34] A. S. Warlaumont, J. A. Richards, J. Gilkerson, and D. K. Oller, "A social feedback loop for speech development and its reduction in autism," *Psychological science*, p. 0956797614531023, 2014.

[35] A. Weisleder and A. Fernald, "Talking to children matters early language experience strengthens processing and builds vocabulary," *Psychological science*, vol. 24, no. 11, pp. 2143–2152, 2013.

[36] E.-S. Ko, A. Seidl, A. Cristia, M. Reimchen, and M. Soderstrom, "Entrainment of prosody in the interaction of mothers with their young children," *Journal of child language*, pp. 1–26, 2015.

[37] http://homebank.talkbank.org/.

[38] J. Trmal, G. Chen, D. Povey, S. Khudanpur, P. Ghahremani, X. Zhang, V. Manohar, C. Liu, A. Jansen, D. Klakow, D. Yarowsky, and F. Metze, "A keyword search system using open source software," in *Proc. IEEE Workshop on Spoken Language Technology*. South Lake Tahoe, NV; USA: IEEE, Dec. 2014.

[39] S. Skoorathota, S. Morton, A. Amatuni, and E. Bergelson, "6 & 7-month-olds noun input: Human and automated corpus analyses," in *Proc. International Congress on Infant Studies*. New Orleans; U.S.A.: ICIS, May 2016.

[40] R. Bates, E. Fosler-Lussier, F. Metze, M. Larson, G.-A. Levow, and E. M. Provost, "Experiences with shared resources for research and education in speech and language processing," in *Proc. INTERSPEECH*. San Francisco, CA; U.S.A.: ISCA, Sep. 2016, accepted.