IMPROVING HUMAN-COMPUTER INTERACTION IN LOW-RESOURCE SETTINGS WITH TEXT-TO-PHONETIC DATA AUGMENTATION

Adam Stiff *Prashant Serai* *Eric Fosler-Lussier*

The Ohio State University
Department of Computer Science & Engineering

ABSTRACT

Off-the-shelf speech recognition systems can yield useful results and accelerate application development, but general-purpose systems applied to specialized domains can introduce acoustically small—but semantically catastrophic—errors. Furthermore, sufficient audio data may not be available to develop custom acoustic models for niche tasks. To address these problems, we propose a concept to improve performance in text classification tasks that use speech transcripts as input, without any in-domain audio data. Our method augments available typewritten text training data with inferred phonetic information so that the classifier will learn semantically important acoustic regularities, making it more robust to transcription errors from the general purpose ASR. We successfully pilot our method in a speech-based virtual patient used for medical training, recovering up to 62% of errors incurred by feeding a small test set of speech transcripts to a classification model trained on typescript.

Index Terms— Low-resource, spoken dialog systems, chatbot

1. INTRODUCTION

Speech recognition is error prone, whether done by machines or humans. Contextual awareness, including speaker identity, situational context, conversation history, etc., allows human listeners to outperform automatic systems by placing strong priors on the phonetic and semantic content of a message, which are generally not available to general-purpose automatic speech recognition (ASR) systems. Nonetheless, there is strong interest in using ASR for human interaction with computers, and, of course, for these systems to perform well for a variety of specialized tasks.

An obvious approach to improving the performance of ASR systems deployed in custom domains is to train custom acoustic and/or language models for the application. However, the extensive annotated speech training data required to develop high quality models may not be available in new or especially unique domains. In such low- or no-resource cases, it may make sense to deploy a broad-purpose ASR system, and to make the downstream task tolerant of ASR errors. However, without speech transcripts from the general purpose system in the target domain, training a downstream task to be tolerant of those errors is not straightforward.

In order to increase the robustness of downstream tasks to ASR errors, we propose a simple method that allows us to leverage existing text data within the domain of interest. In brief, we infer phonetic representations of the in-domain data from the text modality using a grapheme-to-phoneme converter, and train the downstream task using both the original text input, as well as the inferred phonetic form of the same data. The speech recognizer, lacking domain-specific models, may sometimes produce transcripts of generally likely words that have no semantic connection to what was said—e.g., "Oreo" instead of "how are you"—but which share some acoustic similarities. The downstream model should know what the important semantic distinctions "sound like," so that if the ASR produces the wrong words, the downstream model still has a chance to reinterpret the sounds correctly.

We find that this method recovers a substantial portion of the errors resulting from naively using speech transcripts as input to a model trained only on the text. We are able to further boost performance by generating alternative versions of the text input that a speech recognizer is likely to produce in error, and randomly sampling these as alternatives of the original text during training. Our error generation method does rely on speech data to determine specific error likelihoods, but importantly, we are able to show a benefit by using an out-of-domain, general purpose speech corpus.

The system that we use to test this concept, and what motivates the work, is a virtual patient dialog system used to train medical students at Ohio State to take patient histories. Traditionally, this skill is trained and evaluated by hiring actors to play patients for students to interview; the virtualization of this interaction reduces costs, increases consistency, and accelerates feedback to the students. Previous iterations of the virtual patient have utilized a keyboard interface to allow students to ask the patient questions. The task of the system is to correctly classify the typed natural language questions as one of over 350 known questions, to elicit the correct prepared response; the current deployed system combines rule-based and machine learning-based systems using the text modality [1].

While speech is a more natural modality for doctor-patient

interactions, data collection and availability is challenging for a number of reasons. Students receive instruction about which questions are important to ask, and why; thus, crowdsourcing collection with untrained laypeople is not useful. Students are also engaged in a rigorous curriculum, so their time is limited, and annotators are likewise busy medical experts. In this study, the only data available for tuning the speech recognition of the virtual patient actually come from the typed conversations of previous versions of the patient.

Throughout this text we use *typescript* to refer to the typewritten conversations with the virtual patient, and *speakscript* to refer to speech recognition transcripts.

2. RELATED WORK

Dialog systems have recently attracted a fair amount of research attention in terms of both interpreation and generation of natural dialog, e.g. [2]. This work focuses solely on interpretation: for language generation we rely on precise predefined answers required by the medical teaching staff for educational training and assessment.

Dialogue system development in the face of resource constraints has been a challenge for several groups. Plauché et al. describe methods for language adaptation for speech dialog systems in a target language with little recorded speech data, by adapting recognition models as new input is collected [3]. In perhaps the closest work to ours, Sarikaya et al. deploy spoken dialog systems in new domains with little or no resources [4], mining static text resources to develop in-domain language models to improve the speech recognition performance directly. We are unaware of other work utilizing in-domain, cross-modal data to improve the compatibility of a downstream model and a general-purpose speech recognizer.

Of course, we build directly upon the previous work in the target domain [1], and deploy text CNNs [5] as our downstream classification model.

3. MODEL

The virtual patient is a graphical application developed using Unity3D, and deployed on tablet computers. The general-purpose speech recognition system that we use is a commercially available cloud-based ASR system. All dialog management is handled on a central server through an HTTP interface with the graphical client application.

The downstream classification model used to identify questions is an ensemble of text CNNs [5], following [1] (Figure 1). We train two sub-ensembles and combine their output with a stacking network [6]. The stacking network outputs a weighted sum of its inputs, with weights learned to minimize the error. Each sub-ensemble is trained on one of the two forms of the input, i.e. inferred phonetic representation, and original typescript. The output of each sub-ensemble is determined by majority voting, which empirically performs

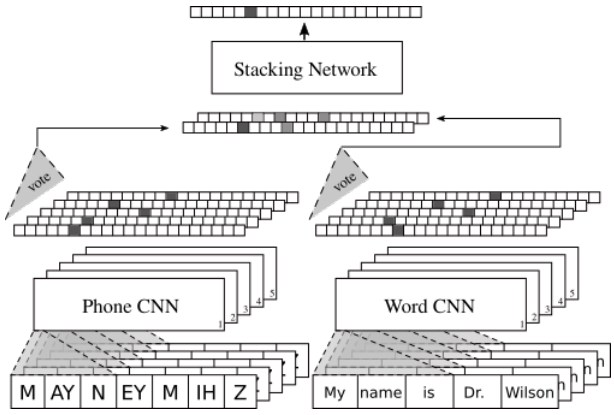

Fig. 1. Overview of the classification model. Phoneme- and word-based representations are input to ensembles of text CNNs. Output of ensembles is determined by majority voting, and combined in a stacking network to produce final classification output.

better than a product of experts [7] or averaging. Vote tallies of each sub-ensemble serve as input to the stacking network.

Both sub-ensembles consist of five convolutional networks, each trained on a different subset of the data. Each of these has a single convolutional layer followed by ReLU activations and max pooling, using dropout of 0.5. This is fed into a single fully-connected linear layer to produce the 359-dimensional softmax output of the network, which is trained using a cross-entropy criterion.

Phoneme-based CNNs take input of one channel of 16-dimensional embeddings (except for the 2-channel condition, see section 4.1), initialized randomly and tuned for the task. Convolutional layers consist of 400 kernels each of widths 2 through 6 phonemes. Input to word-based CNNs are 300-dimensional pretrained word2vec embeddings [8] held static during training. Word-based networks use 300 kernels each of widths 3, 4, and 5 words.

Under sampling conditions, training examples have alternate versions which may be presented for training instead of the original input. These alternate versions simulate generic ASR errors (see Section 4.1). We first randomly determine whether to choose an alternate; if an alternative is desired it is sampled according to the likelihood of its generation.

4. EXPERIMENTS

The main experimental variables that we manipulate are the representation of phonemes used, and the rate at which we randomly sample erroneous alternative forms of the input. In this section we describe in detail the data used in the experiments, as well as the experimental conditions.

4.1. Data

The base training data consists of 94 typescript conversations with a virtual patient experiencing back pain. Students col-

lect information about the patient's present illness, past medical history, family medical history, and social history. The 94 conversations comprise 4,330 individual queries and responses, with all questions annotated as belonging to one of 359 known classes. Class frequencies exhibit a long tail, with the top quintile of data consisting of only eight classes, and the bottom quintile consisting of 265 classes. The least frequent classes have only one example in the dataset. Spelling errors occur regularly, and are generally left as-is (although some unpronounceable special characters were stripped for the purpose of phonetic inference).

Phonetic data is derived from the typescript by looking up the pronunciation in CMUdict (omitting stress), or using a Phonetisaurus [9] grapheme-to-phoneme model trained on CMUdict for unknown words. We experiment with three variations on the phonetic representation. The *plain phones* condition simply concatenates the phoneme sequences of the constituent words in the sentence in order. The *boundary tokens* condition adds a single "boundary phoneme" to the alphabet, which is inserted between words, to explore the value of word segmentation information in the semantic classification task. Finally, the *2-channel* condition adds word boundary information in a second channel to the text CNNs comprising the phonetic sub-ensemble. In other words, each phoneme is represented as both the identity of the phoneme, as well as whether or not that sound is the start of a word (both encoded as a 16-dimensional embedding vector).

Simulated ASR error alternatives are generated by a method due to [10]. Briefly, the method aims to simulate a neural acoustic model making incorrect predictions, and to decode the resulting lattice as usual, generating text that is likely to be erroneously produced by a non-domain specific acoustic model. The technique samples both when to produce an erroneous phoneme, and which one to produce, if so. The distribution of phoneme choice is learned from the confusions produced by a trained model, but the posterior probabilities of erroneous phonemes are determined using the confusibility of the original phoneme, to simulate an over-confident system. This method is used to generate up to 100 alternatives for a given typescript input sentence, along with the frequency with which each alternative is produced.

To evaluate the effects of our data augmentation, we collected a small test set of speakscript. Six adult, native English-speaking volunteers, three each male and female, read dialogs from a typescript dataset distinct from the set used for training. These read-speech utterances were fed into the target ASR system to collect the corresponding speakscript. The dataset consists of 756 transcribed utterances. After spelling correction of the typescript input, word error rate of the speakscript output was calculated at approximately 10%. Classification accuracy of the speakscript test set in the combined typescript-trained model was 65.7% (cf. 69.9% for typescript input). We also generated phonetic forms of the speakscript test data.

The test set does have several shortcomings: its size does not admit very many of the types of errors we would be able to correct with our method; read speech is better-behaved than spontaneous speech; and it includes some unseen labels. Nonetheless, it allows an evaluation of our approach.

4.2. Experimental details

All models in each sub-ensemble are trained individually, using distinct 90/10 train/dev splits, using the Adadelta learning rule [11], with initial learning rate 1.0. We train each sub-model for 25 epochs, and keep the model from the epoch with the best dev set performance. We report accuracy for each sub-ensemble (Phonemes and Words), as well as the accuracy of the combined system. With the exception of the "all alternatives" condition (see below), all of the different inferred phonetic representations used the same training and development splits during training. The training data for every model in the ensemble was supplemented with a list of the "canonical" sentences for each of the 359 classes. Thus, the development set for every model was guaranteed to have no unseen classes.

We include the results of an early experiment in which we simply trained the whole system using all of the available alternative error forms, randomly shuffled, and split 90/10 for each sub-model. This experiment was unsuccessful (see sections 5 and 6), but was motivation for implementing the sampling paradigm, so we report its results for comparison.

In addition to altering the phonetic input representation, we experiment with sampling rates ranging from 0-50% for error alternatives. In experiments using sampling, dev set examples never use sampled alternatives.

5. RESULTS

Results are reported in Table 1. Likely due to the small sizes of the training and test sets, as well as the randomness introduced by sampling, the results exhibit a fair amount of variation from run to run. Therefore, we report averages over three runs under identical model parameterizations.

The best-performing combined system is plain phonemes with a sampling rate of 20%; this recovers approximately 62% of the increased error rate in using speakscript within a typescript model. Plain phonemes also exhibit the best performance among the non- sampled conditions. 2-channel bounds give the best performing phoneme sub-ensemble, although the difference comes nowhere close to statistical significance. Combined systems are always at least as good as either of the constituent sub-ensembles, and usually much better.

6. DISCUSSION

The aforementioned test set issues make it difficult to make sweeping pronouncements about the results, but we do find

	Sampling	Phonemes	Words	Combo
Baseline (typescript)	N/A	System trained as combination only		69.9
Baseline (speakscript)	N/A	System trained as combination only		65.7
All alternatives	N/A	64.95	65.48	65.74
Plain phonemes	0%	67.15	66.27	67.55
	5%	66.89	66.76	67.68
	10%	66.75	66.40	67.73
	20%	66.75	66.00	**68.30**
	50%	66.36	66.09	67.50
Boundary tokens	0%	66.45	66.09	67.64
	5%	66.67	66.05	67.86
	10%	66.58	66.88	67.90
	20%	65.88	66.76	67.77
	50%	65.96	66.31	67.99
2-channel bounds	0%	**67.37**	**66.89**	67.37
	5%	66.67	66.58	67.59
	10%	66.48	66.40	67.42
	20%	66.62	**66.89**	68.12
	50%	67.11	66.36	67.95

Table 1. Test set question classification accuracy, reported as the average of three runs. Column maxima in **bold font**. All "Combo" results are a significant improvement over the speakscript baseline using Pearson's χ^2 and the Benjamini-Hochberg multiple tests correction [12] with a false discovery rate of 10%.

some encouraging trends. The two clearest such results are 1) that even inferred phonetic representations can improve speech recognition input for downstream tasks, and 2) that sampling generated errors further boosts performance.

The motivation for sampling in the first place derives from the negative result from the "all alternatives" condition. In essence, including all alternatives just allowed for serious overfitting: with only minor variations in the training examples, it overspecialized on the specific sentences underlying the alternate forms, harming performance on unseen sentences. This may have been mitigated with smarter stratification of the development sets, but sampling alternatives also enhances variety in the surface forms for each label without making the development sets easier.

Because sampling does not seem to benefit solely word-based or phoneme-based systems, it would seem that sampling encourages diversification across the two sub-ensembles, as the best combination results usually do not have the best component results. Indeed, the best-performing individual systems that contribute to the averages shown in the table maintain this trend (data not shown).

The benefit of word boundary information is less clear: the best-performing model on average included no word boundary information, and the second best average used the version of word boundaries that is easiest to ignore; however, boundary tokens sometimes outperform other representations under otherwise equivalent conditions. This speaks to the need for further experiments with more statistical power.

7. FUTURE WORK

As this was a pilot study, there are many avenues for improvements to the current work, as well as new questions identified by the experiments presented. First and foremost is the need for more data, both to improve generalizability of the models, as well as to put the results on firmer statistical footing. This study will actually facilitate the collection of that data, as even slight improvements will improve the user experience of new students using the application for their training.

An intriguing question raised by the current study is *why* random alternative sampling affords a benefit, and whether the mechanism of the benefit is the same for phone representations as for words. One possibility is that sampling is just a form of regularization, which may be born out by the slight drops in performance for each of the sub-ensembles — suggesting the need to directly compare to other types of regularization. A further possibility is just that, by luck, the random samples introduce some of the specific errors seen in the test set. If this were the case, we might expect less benefit in a broader domain, as confusible words begin to impinge on important topics. To borrow an example from the present set of alternatives, "toaster" is probably a safe replacement for "mister" in our domain, but only because questions about breakfast are irrelevant to the patient's back pain.

Also of interest are ways in which we might more directly encourage acoustic similarities to be represented in the input, instead of depending on the distant supervision of the correct semantic class and generated errors to encourage similarities to emerge. One straightforward option to try would be to initialize the embedding matrix for phonemes with corresponding average MFCCs or GMM representations.

Finally, we are currently experimenting with a form of knowledge distillation for application to this task, in which we seek to minimize the mean squared error between analogous layers in a high-performing network and a network learning from alternative versions of the same input. In this way we hope to encourage the representations of the alternatives to become similar in semantically coherent ways. Initial results have been promising, but did not surpass the best models presented here.

8. ACKNOWLEDGEMENTS

This material is based upon work supported by the National Science Foundation under Grant No. 1618336. We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Quadro P6000 GPU used for this research. Additional computing resources provided by the Ohio Supercomputer Center [13].

9. REFERENCES

[1] Lifeng Jin, Michael White, Evan Jaffe, Laura Zimmerman, and Douglas Danforth, "Combining cnns and pattern matching for question interpretation in a virtual patient dialogue system," in *Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications*, 2017, pp. 11–21.

[2] Jiwei Li, Will Monroe, Tianlin Shi, Sėbastien Jean, Alan Ritter, and Dan Jurafsky, "Adversarial learning for neural dialogue generation," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2017, pp. 2157–2169.

[3] Madelaine Plauché, Özgür etin, and Udhaykumar Nallasamy, "How to build a spoken dialog system with limited (or no) language resources," in *AI in ICT4D*. ICFAI University Press, India, 2008.

[4] Ruhi Sarikaya, Agustin Gravano, and Yuqing Gao, "Rapid language model development using external resources for new spoken dialog domains," in *Acoustics, Speech, and Signal Processing, 2005. Proceedings.(ICASSP'05). IEEE International Conference on*. IEEE, 2005, vol. 1, pp. I–573.

[5] Yoon Kim, "Convolutional neural networks for sentence classification," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, 2014, pp. 1746–1751.

[6] David H Wolpert, "Stacked generalization," *Neural networks*, vol. 5, no. 2, pp. 241–259, 1992.

[7] Geoffrey E Hinton, "Training products of experts by minimizing contrastive divergence," *Neural computation*, vol. 14, no. 8, pp. 1771–1800, 2002.

[8] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[9] Josef R Novak, Nobuaki Minematsu, and Keikichi Hirose, "Wfst-based grapheme-to-phoneme conversion: Open source tools for alignment, model-building and decoding," in *Proceedings of the 10th International Workshop on Finite State Methods and Natural Language Processing*, 2012, pp. 45–49.

[10] Prashant Serai, Peidong Wang, and Eric Fosler-Lussier, "Improving speech recognition error prediction for modern and off-the-shelf speech recognizers," *Submitted to 2019 IEEE International Conference on Acoustics, Speech, and Signal Processing. (ICASSP).*

[11] Matthew D Zeiler, "Adadelta: an adaptive learning rate method," *arXiv preprint arXiv:1212.5701*, 2012.

[12] Yoav Benjamini and Yosef Hochberg, "Controlling the false discovery rate: a practical and powerful approach to multiple testing," *Journal of the Royal statistical society: series B (Methodological)*, vol. 57, no. 1, pp. 289–300, 1995.

[13] Ohio Supercomputer Center, "Ohio supercomputer center," `http://osc.edu/ark:/19495/f5s1ph73`, 1987.