IMPROVING SPEECH RECOGNITION ERROR PREDICTION FOR MODERN AND OFF-THE-SHELF SPEECH RECOGNIZERS

Prashant Serai *Peidong Wang* *Eric Fosler-Lussier*

The Ohio State University
Department of Computer Science & Engineering

ABSTRACT

Modeling the errors of a speech recognizer can help simulate errorful recognized speech data from plain text, which has proven useful for tasks like discriminative language modeling, improving robustness of NLP systems, where limited or even no audio data is available at train time. Previous work typically considered replicating behavior of GMM-HMM based systems, but the behavior of more modern posterior-based neural network acoustic models is not the same and requires adjustments to the error prediction model. In this work, we extend a prior phonetic confusion based model for predicting speech recognition errors in two ways: first, we introduce a sampling-based paradigm that better simulates the behavior of a posterior-based acoustic model. Second, we investigate replacing the confusion matrix with a sequence-to-sequence model in order to introduce context dependency into the prediction. We evaluate the error predictors in two ways: first by predicting the errors made by a Switchboard ASR system on unseen data (Fisher), and then using that same predictor to estimate the behavior of an unrelated cloud-based ASR system on a novel task. Sampling greatly improves predictive accuracy within a 100-guess paradigm, while the sequence model performs similarly to the confusion matrix.

Index Terms— Speech Recognition, Error Prediction, Low Resource, Sequence to Sequence Neural Networks, Simulated ASR Errors

1. INTRODUCTION

Automatic Speech Recognition (ASR) is proliferating quickly, with a variety of applications having speech as an input modality. Yet, application specific audio data is often a scarce resource, and models trained on text data are widely being paired with cloud based speech recognition services. The nature of speech recognized text can be different from typed text data, notably in the nature of errors i.e. typos vs. speech recognition errors. Prior work has shown that given text data it is possible to simulate the recognition errors that might occur if the text were to be spoken, and the benefit of such simulated data especially in the absence of in application specific ASR data.

Fosler-Lussier et al. described a Weighted Finite State Transducer (WFST) framework that models word errors in speech recognition by measuring kinds of phonetic errors to build a phonetic confusion matrix [1]. Anguita et al. looked inside the HMM-GMM acoustic model of the speech recognizer to directly determine phone distances and model errors [2]. Jyothi and Fosler-Lussier combined the two aforementioned ideas and extended it to predict complete utterances of speech recognized text [3]. Tan et al. looked at use of a phrasal NMT system to simulate ASR, but only evaluating the 1-best word sequence of the final output [4]. Several works have used simulated ASR error data to do discriminative training and improve ASR performance [5, 6, 7]. Simulating ASR errors can also help with downstream tasks: Tsvetkov et al.[8] and Ruiz et al.[9] incorporated knowledge of simulated ASR errors at train time, to improve the performance of Machine Translation systems in the face of real speech recognized data at test time.

There is not prior published work, to the best of our knowledge, that predicts errors for an ASR system using a neural network acoustic model, or for commercial off-the-shelf recognizers. With modern speech recognizers we can no longer determine phone distances by peeking into the acoustic model, so some of the enhanced confusion transducer based models [2, 3] are no longer applicable. Also, off-the-shelf speech recognizers do not afford much access to internals or information about them.

However, there is a key flaw in the Fosler-Lussier et al. WFST model that implies a significant mismatch to posterior based models: the confusion matrix is used *directly* for prediction, and does not exhibit the peaky behavior of frame-level (or CTC-level) posterior estimates. This mismatch creates a poor model when generating errors for neural net based systems. The key insight in this paper is that we use the confusion matrix over all examples to sample *which* phones should appear in the output distribution, but provide a distribution that is much more peaky and ignores the smoother tails of the confusion matrix. When sampling from the confusion matrix instead of composing with its FST form, we find that it better models the stochasticity of errors and confidence of neural network acoustic models.

The standard confusion matrix in the WFST framework is

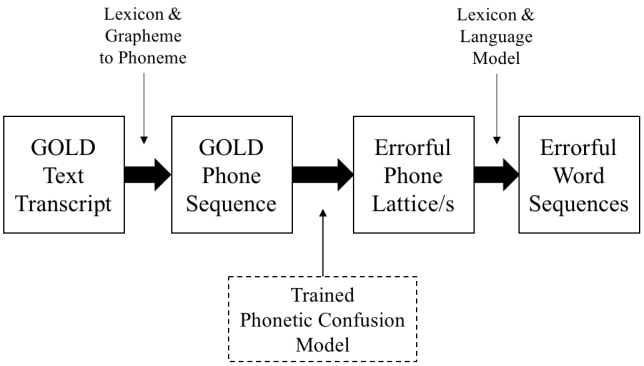

Fig. 1. General pipeline for generating simulated errorful sequences: text transcripts are converted to phone sequences, which are converted to confusable phone lattices through the error prediction model. These are then decoded by a WFST model incorporating a lexicon and language model.

context-independent, which does not reflect well the context-dependence of errors (for example, a canonical vowel next to /r/ will more likely be misidentified as an r-colored vowel than other vowels). However, context-dependent confusion networks require substantial amounts of data to train and will be relatively sparse. In order to model context dependence, we trained a neural Sequence to Sequence model, predicting errorful phone sequences from canonical sequences. We can then either directly use the predictions, or decode using the WFST framework above.

In the next section, we describe the two models used in this experiment. Section 3 details the experimental setup, followed in Section 4 by evaluation both on a same-setting task (train on Switchboard, test on Fisher with the same recognizer) and a cross-setting task (train on Switchboard, test on a dialogue system using a different recognizer).

2. MODEL DESCRIPTIONS

Our general pipeline for going from regular text to text with simulated ASR errors is shown in Figure1. We convert the input word sequence to a phone sequence, simulate phonetic errors, and then decode back to a word sequence. We focus our attention on simulating the errors at the phonetic level.

2.1. Confusion Matrix Based Models

The WFST error prediction approach [1, 3] estimates an N-best list of word confusions W_{conf} from an input word sequence W through the following equation:

$$W_{conf} = W \; o \; P^{-1} \; o \; C \; o \; P \; o \; L$$

The words in the original text sequence are converted to phones using pronunciations from an inverted lexicon P^{-1}, composed with a confusion matrix WFST C, and is then decoded back into a simulated ASR transcript i.e. word sequence by composing a lexicon P and language model L, which are also FSTs. The confusion matrix (ConfMat) transduces every phone at the input to a sequence of phones from length 0 (deletion) to 1 (no error/mutation) or more than 1 (insertion), and can be pre composed with the lexicon and language model for efficiency. For each original text sequence, once we have the final composed W_{conf} graph we calculate the N-best unique strings [10] to obtain N alternative word sequences to the input. We vary the lexicon and grammar in the WFST predictor for each dataset.

Instead of directly using the confusion matrix for prediction, we can also sample the matrix to determine output sequence. We convert the words in the original text sequence to phones, but then instead of composing with the confusion matrix WFST, for each phone in the input we sample without replacement, two options or alternatives for it from the ConfMat based on the probabilities with which they were observed to be confusible with the input phone. The most likely option for each input phone is typically the same phone itself, but the hope here is that over several iterations of sampling, the simulated recognizer will pick errors of various kinds over the input phone. We construct an FST by chaining these sampled alternatives, and weight the first sampled option with the weight of the most likely option, and weight the second sampled option with the weight of the second most likely option in the ConfMat, and then normalize. Finally, we compose with a decoding graph composed from a lexicon and language model as above and calculate the 1-best string for each sampled WFST per input sequence, and combine the various output word sequences obtained, ranking them by frequency of occurrence.

2.2. Neural Sequence to Sequence Based Models

We also experimented with context-dependent prediction of the errors using a sequence-to-sequence model. To model phonetic confusions with information about the context, we use a 2-layer 128-unit recurrent neural Sequence to Sequence Model (Seq2Seq) [11] with attention [12]. We feed the model with the phonetic transcript of the true word sequence at the input, and at each time step of the input we provide an additional one hot vector containing one of five cues representing different kinds of errors (no error, mutation, deletion, insertion of one additional phone, insertion of more than one additional phones). At train time, we align the input and output phone sequence using the same technique as the confusion matrix based system, to determine what kind of error is being made (or not made) for each time step of the input. At test time, we randomly sample to select one of the five aforementioned cues for each timestep of the input, from a collapsed version of the confusion matrix that holds information about the frequencies of these five kinds of errors for each input

phone. At test time, the probability distributions at the output of the Seq2Seq model are then converted into WFSTs, by selecting three phones at each time step, and a softmax function with a temperature $\tau = 10$ is applied as shown below.

$$P_t(a) = \frac{\exp(q_t(a)/\tau)}{\sum_{i=1}^{n} \exp(q_t(i)/\tau)}$$

Once we have the WFSTs, we compose them with a decoding graph as in the case of the confusion matrix model, except the decoding graph is augmented to absorb (with a small cost) any number of "EOS" (End of Sequence) symbols at the end of phone sequences being translated into word sequences. Finally, 5-best word sequences are calculated for each input sample, and combined to produce K alternatives per input text sequence.

Besides sampling for cues at the input of the Seq2Seq model, we also explore the effect of additional sampling from the output probability distribution produced by the Seq2Seq model. Similar to the sampling from the distributions of confusion matrix, instead of directly producing WFSTs from the output of the Neural network, we generate multiple sampled WFSTs and decode them similarly as above.

3. EXPERIMENTAL SETUP

3.1. Data

We use the Kaldi Switchboard recipe to train a Deep Neural Network acoustic model on the Switchboard corpus. A sMBR criterion is used during training and decoding proceeds with a trigram grammar [13]. We use this recognizer to transcribe speech data from the Fisher corpus [14], containing about 1.8 million utterances with a word error rate of roughly 30%. We use the speech recognized text from Fisher paired with gold text as the training set for all our Models, holding out a validation set of 100 utterances for the tuning of hyperparameters. We tested our models on two kinds of data. Firstly, we predicted errorful transcripts for held out data from the Fisher corpus, which was recognized by the aforementioned speech recognizer that we trained. This was a set of 500 utterances containing 504 error chunks across all recognized speech. The WFST prediction module uses the standard Switchboard lexicon and grammar used in the recognizer.

Secondly, we predicted errorful transcripts for data from the Virtual Patient project [15], where volunteers read out text data from doctor trainees querying a patient avatar. The recorded speech was recognized using a cloud based ASR service treated as a black box [16]. This was a set of 756 utterances containing 258 error chunks across all recognized speech, and the word error rate was slightly over 10%. The simulated speech transcripts are used to enable understanding of spoken input when no spoken training data exists (only chatted text) [16]. As there is a vocabulary mismatch between Switchboard and the Virtual Patient, but we did not want to inform the error prediction system of the Virtual Patient vocabulary, we extended the WFST lexicon and language model by leveraging models from the EESEN Offline Transcriber [17], which uses a pruned 3-gram language model provided by Cantab Research [18] trained on TED-LIUM data [19].

3.2. Training Details

To train the confusion matrix models, we first convert the gold and speech recognized transcripts to phone sequences, and align them using a phonetic distance based dynamic programming algorithm [1]. The alignment is done in such a way that each input phone (e.g. /s/) is paired with a sequence of phones of length 0 (deletion, /s/:ϵ), 1 (no error or mutation, /s/:/s/ or /s/:/z/) or more (insertion, /s/:/st/). For each possible input phone, we count frequencies of various "alternative" phone sequences, and normalize them into probabilities. This gives us our confusion matrix for composing or sampling.

For the training the Sequence to Sequence (Seq2Seq) based phonetic confusion model, we start with unaligned pairs of gold and errorful phone sequences, and train it to minimize the cross entropy between the model predictions and the ground truth (i.e., the errorful phone sequence). In our experiments, we found that when we directly used the ground truth sequence, which is a one hot distribution at each time step, the model predictions would be very peaky and not provide much diversity for decoding. To allow the model to learn to produce more diversity at the output, we smoothed the ground truth sequence with alternatives from the confusion matrix at train time:

$$Y_{smooth} = \beta * Y_{original} + (1 - \beta) * C_{11}[y]$$

where y is the original phone label and $Y_{original}$ is the one hot probability distribution corresponding to it, and Y_{smooth} is the smoothed probability distribution. C_{11} is a reduced version of the confusion matrix that only has the alternatives of length 1 (mutation or no error), and β is the smoothing factor (we use value 0.8). Although C_{11} only captures mutation errors, we observed that on smoothing, the neural network was automatically learning to give meaningful weight to output phones at their adjacent timesteps as well.

4. EVALUATION AND RESULTS

Following prior work[3], we use two metrics to evaluate the effectiveness of our models in simulating ASR errors. The first metric measures the percentage of real test set Error Chunks recalled in a set of "K best" simulated speech recognized utterances for each gold word sequence. The error chunks are again determined by aligning the gold word sequence with the errorful word sequence and removing the longest common subsequence. For example, if the gold sequence is "do you take any other medications except for the tylenol for pain" and the errorful se-

Model	Error Chunks Predicted	Complete Utterances Predicted
ConfMat Direct decoding	14.9%	39.2%
ConfMat Sampled decoding	**25.6**%	38.8%
Seq2Seq Direct decoding	23.8%	38.2%
Seq2Seq Sampled decoding	23.0%	37.8%
Seq2Seq Direct (K=50) + ConfMat Sampled (K=50)	23.8%	**43.4**%

Table 1. Evaluation on unseen Fisher corpus recognition data from the same recognizer

Model	Error Chunks Predicted	Complete Utterances Predicted
ConfMat Direct Decoding	8.5%	66.9%
ConfMat Sampled Decoding	**36.4**%	**72.4**%
Seq2Seq Direct Decoding	20.2%	68.3%
Seq2Seq Sampled Decoding	16.7%	67.7%
Seq2Seq Direct (K=50) + ConfMat Sampled (K=50)	34.9%	71.8%

Table 2. Evaluation on Virtual Patient data from a cloud-based ASR service

quence is "you take any other medicine cations except for the tylenol for pain," the error chunks would be the pairs $\{medications : medicine\ cations\}$ and $\{do : \}$. Our detection of error chunks is strict — for an error chunk to qualify as predicted, the words adjacent to the predicted error chunk should be error-free.

The second metric measures the percentage of times the complete test set utterance is recalled in a set of "K best" simulated utterances for each gold text sequence (including error-free test sequences). We aimed to produce 100 unique simulated speech recognized utterances for each gold word sequence, so for both of these metrics, we evaluate the performance at K=100.

Note that these are both hard metrics since the possibilities of various kinds of errors is quite endless, and no matter the plausibility of the output, the metrics only give credit when the utterance/error chunk exactly matches what was produced.

Table 1 shows the results on the held out set from the Fisher corpus for all of the models tried. The sampled decoding on the confusion matrix greatly improves over the direct baseline model from [1] in terms of real error chunks predicted, maintaining comparable performance on the complete utterance prediction metric. The Seq2Seq models also improve over the baseline in terms of error chunks predicted, although the sampling on the confusion matrix still performed better on that metric. Since the Seq2Seq model has the capacity to model errors in a context dependent manner, we wanted to see if it was learning something different from the confusion matrix models. We combined half the number of utterances from the best ConfMat approach and the best Seq2Seq approach each, and looked at the metrics for that. The number of complete utterances recalled was higher than either of the approaches being combined, suggesting that the Seq2Seq model might be learning things that the ConfMat model is not able to capture.

Table 2 shows the results on predicting recognition errors made by the cloud based ASR service. We see that the baseline ConfMat with direct decoding fares even more poorly here. Our proposed sampling approach yields major gains in terms of both metrics, notably recalling almost four times the number of error chunks as the the baseline. The Seq2Seq model does better than the baseline, but does not do as well as the confusion matrix based system with sampled decoding on either metric.

The confusion matrix based system with sampled decoding has also proven to help with a downstream task: Stiff et al. [16] adapted a chatbot answer prediction system to work better with speech input. The error predictor was used to simulate speech errors which were sampled by the classification system during training. The addition of the sampled errorful data to the training regime improved the spoken interpretation accuracy modestly but consistently across several training conditions (e.g. from 67.6% to 68.3% accuracy in the best performing system).

5. CONCLUSIONS AND FUTURE WORK

We show that the sampling based paradigm greatly improves the error prediction performance of the confusion matrix system. We observe that while the Seq2Seq confusion model might be learning to predict errors in a context dependent manner, the method does not generalize well across corpora, and needs further work. We think the Seq2Seq model may benefit from enhancements such as a more robust generator for the error types, multi-head attention, scheduled sampling, etc. and could be done end-to-end. We would also like to use our work to improve Spoken Machine Translation through input perturbation [20].

6. 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 [21]. We thank Adam Stiff for sharing the paired text and speech recognized data from the Virtual Patient project for our experiments.

7. REFERENCES

[1] Eric Fosler-Lussier, Ingunn Amdal, and Hong-Kwang Jeff Kuo, "A framework for predicting speech recognition errors," *Speech Communication*, vol. 46, no. 2, pp. 153–170, 2005.

[2] Jan Anguita, Javier Hernando, Stéphane Peillon, and Alexandre Bramoullé, "Detection of confusable words in automatic speech recognition," *IEEE Signal Processing Letters*, vol. 12, no. 8, pp. 585–588, 2005.

[3] Preethi Jyothi and Eric Fosler-Lussier, "A comparison of audio-free speech recognition error prediction methods," in *Tenth Annual Conference of the International Speech Communication Association*, 2009.

[4] Qun Feng Tan, Kartik Audhkhasi, Panayiotis G Georgiou, Emil Ettelaie, and Shrikanth S Narayanan, "Automatic speech recognition system channel modeling," in *Eleventh Annual Conference of the International Speech Communication Association*, 2010.

[5] Kenji Sagae, Maider Lehr, E Prud'hommeaux, Puyang Xu, Nathan Glenn, Damianos Karakos, Sanjeev Khudanpur, Brian Roark, Murat Saraclar, Izhak Shafran, et al., "Hallucinated n-best lists for discriminative language modeling," in *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*. IEEE, 2012, pp. 5001–5004.

[6] Preethi Jyothi and Eric Fosler-Lussier, "Discriminative language modeling using simulated asr errors," in *Eleventh Annual Conference of the International Speech Communication Association*, 2010.

[7] Gakuto Kurata, Nobuyasu Itoh, and Masafumi Nishimura, "Training of error-corrective model for asr without using audio data," in *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*. IEEE, 2011, pp. 5576–5579.

[8] Yulia Tsvetkov, Florian Metze, and Chris Dyer, "Augmenting translation models with simulated acoustic confusions for improved spoken language translation," in *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, 2014, pp. 616–625.

[9] Nicholas Ruiz, Qin Gao, William Lewis, and Marcello Federico, "Adapting machine translation models toward misrecognized speech with text-to-speech pronunciation rules and acoustic confusability," in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[10] Mehryar Mohri and Michael Riley, "An efficient algorithm for the n-best-strings problem," in *Seventh International Conference on Spoken Language Processing*, 2002.

[11] Ilya Sutskever, Oriol Vinyals, and Quoc V Le, "Sequence to sequence learning with neural networks," in *Advances in neural information processing systems*, 2014, pp. 3104–3112.

[12] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[13] Karel Veselỳ, Arnab Ghoshal, Lukás Burget, and Daniel Povey, "Sequence-discriminative training of deep neural networks.," in *Interspeech*, 2013, pp. 2345–2349.

[14] Christopher Cieri, David Miller, and Kevin Walker, "The fisher corpus: a resource for the next generations of speech-to-text.," in *LREC*, 2004, vol. 4, pp. 69–71.

[15] 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.

[16] Adam Stiff, Prashant Serai, and Eric Fosler-Lussier, "Improving human-computer interaction in low-resource settings with text-to-phonetic data augmentation," *Submitted to 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

[17] Yajie Miao, Mohammad Gowayyed, and Florian Metze, "Eesen: End-to-end speech recognition using deep rnn models and wfst-based decoding," in *Automatic Speech Recognition and Understanding (ASRU), 2015 IEEE Workshop on*. IEEE, 2015, pp. 167–174.

[18] Cantab Research, "Cantab-tedlium language model and lexicon release 1.1," http://cantabResearch.com/cantab-TEDLIUM.tar.bz2, 2015, [Online; accessed 28-Oct-2018].

[19] Anthony Rousseau, Paul Deléglise, and Yannick Esteve, "Enhancing the ted-lium corpus with selected data for language modeling and more ted talks.," in *LREC*, 2014, pp. 3935–3939.

[20] Xiang Li, Haiyang Xue, Wei Chen, Yang Liu, Yang Feng, and Qun Liu, "Improving the robustness of speech translation," *arXiv preprint arXiv:1811.00728*, 2018.

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