HALLUCINATED N-BEST LISTS FOR DISCRIMINATIVE LANGUAGE MODELING

K. Sagae[a], M. Lehr[b], E. Prud'hommeaux[b], P. Xu[c], N. Glenn[d], D. Karakos[c], S. Khudanpur[c], B. Roark[b], M. Saraçlar[e], I. Shafran[b], D. Bikel[f], C. Callison-Burch[c], Y. Cao[c], K. Hall[f], E. Hasler[g], P. Koehn[g], A. Lopez[c], M. Post[c], D. Riley[h]

[a]USC, [b]OHSU, [c]JHU, [d]BYU, [e]Boğaziçi U., [f]Google, [g]Edinburgh, [h]Rochester

ABSTRACT

This paper investigates semi-supervised methods for discriminative language modeling, whereby n-best lists are "hallucinated" for given reference text and are then used for training n-gram language models using the perceptron algorithm. We perform controlled experiments on a very strong baseline English CTS system, comparing three methods for simulating ASR output, and compare the results with training with "real" n-best list output from the baseline recognizer. We find that methods based on extracting phrasal cohorts – similar to methods from machine translation for extracting phrase tables – yielded the largest gains of our three methods, achieving over half of the WER reduction of the fully supervised methods.

Index Terms— language modeling, automatic speech recognition, discriminative training, semi-supervised methods

1. INTRODUCTION

Standard generative language modeling methods for automatic speech recognition (ASR) involve counting n-grams from large corpora, normalizing them to produce multinomial models over a fixed vocabulary and smoothing (regularizing) the model to avoid assigning zero probabilities. Other than perhaps selecting training corpora to skew the model towards one domain or another, there is nothing in this modeling approach that is optimized for a particular task objective. Regardless of whether the application making use of the language model is automatic speech recognition or machine translation, the same parameterizations will be derived from the corpora, despite the fact that the kinds of ambiguities that the model is used to resolve are radically different in the two applications. In speech recognition, alternative transcriptions will be acoustically confusable; in machine translation, acoustic confusability has nothing to do with the kinds of alternative translations that the language model is scoring within the system.

Discriminative language modeling has been used to optimize large vocabulary speech recognition performance, both with standard n-gram features [1, 2], as well as with morphological, syntactic and trigger features across a variety of languages and tasks [3, 4, 5, 6]. The paradigm in this approach is to run the baseline recognizer over training data and optimize a log linear model using some discriminative objective, such as global conditional log likelihood or with the perceptron algorithm. Because transcribed speech is required to train models with these methods, the amount of data that is available for such an approach is typically a small fraction of what is available to train generative language models, which only require text. The current paper is focused on the problem of applying discriminative language modeling methods to training data that consists of just text. Should such methods prove successful, they could be applied to large scale text resources just as generative models are.

Simulating ASR errors has been pursued in the past, using a number of methods. Printz and Olsen [7] presented methods for calculating the acoustic confusability of words, for the purpose of estimating the word error rate (WER) of a model in a new domain. Chen et al. [8] provide some new methods in this vein. More recently, Tan et al. [9] pursued methods that exploited parallel corpora of ASR output and reference text to learn models of errors, using methods popularized for machine translation (MT). None of these methods have been directly applied to improving system performance.

Simulated errors have been used to improve MT systems [10, 11]. Exploiting simulated errors for ASR system improvements was explored in Jyothi and Fosler-Lussier [12] using weighted finite-state transducer approaches for generating sequences confusable with the reference. Kurata et al. [13] also used weighted finite-state transducers, modeling acoustic similarity to generate confusable strings in a process they call "pseudo-ASR," which creates training data for a discriminative model that produced improved WER in Japanese call center data [14]. We will explore similar methods in this paper, applied to large vocabulary continuous telephone speech systems, and contrast them with methods closer to the Tan et al. [9] methods cited above and the closely related methods of cohort extraction [15].

The main purpose of this paper is to perform controlled experimentation under a number of conditions, to determine how much WER reduction can be achieved with these semi-supervised methods, versus fully supervised training on the same data. We carefully prepared a large discriminative language modeling corpus, and examined performance when training models on real n-best lists produced under standard conditions versus when training models on hallucinated n-best lists produced under various protocols for producing such lists. We find that methods for direct extraction of phrasal cohorts yield the largest WER reductions of our three examined methods, just over half the gain achieved with fully supervised methods, and that improvements can also be achieved by modeling confusability at the phone level. Given that these semi-supervised methods can be applied to arbitrarily large text corpora, this bodes well for additional system gains beyond what was achieved for these moderate sized training sets.

2. DATA

For this paper, we focused on English conversational telephone speech (CTS), and used the IBM Attila speech recognition software library [16] to build a baseline system. The training data for acoustic models consists of about 2000 hours of speech segments from about 14500 telephone conversations (25M words), of which about 11000

Most of the work presented here was done as part of a 2011 CLSP summer workshop project at Johns Hopkins University. We acknowledge the support of the sponsors of that workshop. This work was also partially supported by NSF Grants #IIS-0963898 and #IIS-0964102, and by TUBITAK (Project No: 109E142). Any opinions, findings, conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the sponsors.

conversations are from the Fisher corpus and 3500 from the Switchboard corpus. We utilized the NIST RT04 Fall development and evaluation test for benchmarking the performance of our systems. The development and evaluation sets consist of about 38K words and 37K words respectively, spanning about 2 hours.

The acoustic models contain 41 phones, a 3-state left-to-right HMM topology for phones, 4K clustered quinphone states, 150K Gaussians, a linear discriminant transform, and a semi-tied covariance transform. The acoustic features consist of 13 coefficient perceptual linear coding vectors with speaker-specific vocal tract length normalization (VTLN). This training approach is as described in [17], and we refer readers there for further details. The baseline 4-gram language model defined over 50K word vocabulary was estimated by interpolating the transcripts and similar data extracted from the web as described in [18].

In order to produce training data for discriminative language modeling, we decoded the training data using 20-fold cross validation. That is, while decoding a fold, we ensured that the transcripts from that fold were kept aside in estimating the language model used for decoding the fold. All three stages of the decoding were performed in this 20-fold cross-validation manner. Thus, we took more care to avoid biasing the decoder than the standard process of employing 20-fold cross-validation just in the last stage of decoding.

For the controlled studies of the current paper, we used up to eight of these folds to train our discriminative language models. In fully supervised conditions, the lattices output by the above cross-validation scenario were used to produce the n-best lists directly. To compare with these fully supervised conditions, we generated hallucinated n-best lists for the same utterances. To produce these simulated n-best lists, we followed another cross-validation procedure: For each of the k folds in the given training set, we trained confusion models (using one of our three methods detailed below) on the other $k-1$ folds, and used them to produce n-best lists for that fold. In such a way we produce alternative n-best lists for the same data as in the fully supervised scenario, to provide to the discriminative training algorithm. In the results presented below, we investigated conditions with 2, 4 and 8 folds.

3. METHODS

We present three methods for generation of simulated ASR n-best lists from text to serve as training material for discriminative language models. The first is based on finite-state transducers that model phone confusion in the output of the baseline system. The other two methods consider confusability at the level of words and phrases directly, using techniques developed for machine translation.

3.1. Finite-state transducers for modeling phone confusion

Our first proposed method for generation of simulated n-best lists is to model confusions at the phone level using a process that can be thought of as a simulation of an ASR system without actual speech. The more faithful our simulation is to the targeted ASR system, the more likely it is that the resulting simulated n-best lists will be effective training material for discriminative language modeling.

Given a sequence of words S and a weighted finite-state transducer X that maps phone sequences to confusable phone sequences, a list of phonetically confusable word sequences can be generated using a pronunciation dictionary L and a generative language model G (also encoded as finite-state transducers) through the composition $S \circ L \circ X \circ L^{-1} \circ G$. The initial sequence of words is composed with the pronunciation dictionary, resulting in a corresponding lattice of phone sequences (since words in the dictionary may have multiple corresponding pronunciations). Composing this lattice with a phone confusion transducer produces a lattice encoding several confusable phone sequences. An inverse pronunciation dictionary is then used to create a lattice of words from the lattice of phones, and finally the generative n-gram language model is composed with this lattice of words. A list of word strings confusable with the input string S, scored according to phonetic confusability and the n-gram language model, can then be generated from this resulting lattice. Since the same dictionary and generative language model used in an ASR system can be used in this approach, whether or not the resulting lists are similar to what the ASR system would produce depends crucially on the phone confusion transducer X.

To create a phone confusion transducer, we use real n-best lists produced by the ASR system we want to simulate, coupled with the reference transcription corresponding to each n-best list. Using the dictionary L, the reference strings and each of the strings in the n-best lists are mapped to corresponding sequences of phones. If multiple pronunciations are listed for words in L, only one is chosen arbitrarily, so that each word sequence is mapped to exactly one phone sequence. For each n-best list, we then produce Levenshtein alignments between the reference phone sequence and each n-best hypothesis phone sequence. Based on each of these individual alignments of pairs of phone strings, each phone in a reference string corresponds to the same phone in the hypothesis string (phone identity), to a different phone in the hypothesis string (a substitution), or to no phone (ϵ) in the hypothesis string (a deletion). Additionally, a phone in the hypothesis string might not correspond to any phones in the reference string, in which case ϵ in the reference string is aligned with the phone in the hypothesis string (an insertion). Each alignment between two phone sequences is then composed of a sequence of such phone-to-phone (or ϵ-to-phone, or phone-to-ϵ) mappings, represented as $X{:}Y$, where X and Y can be any phone or ϵ. Using these sequences, we estimate an n-gram model of phone identity, substitutions, insertions and deletions, following [19]. We encode this n-gram model as a weighted finite-state acceptor with $X{:}Y$ pairs in the transition arcs, and by separating these pairs so that X is an input symbol and Y is an output symbol, we obtain a phone confusion transducer X. In our experiments, we used 5-gram phone confusion models.

One practical difficulty with the computation of $S \circ L \circ X \circ L^{-1} \circ G$ for a given text string S is that X allows nearly arbitrary deletions, insertions and substitutions in nearly any context. While the vast majority of deletions, insertions and substitutions have extremely low probability in most contexts, they are still encoded in X, making any composition involving X and L^{-1} infeasible. We approach this problem with two strategies. First we prune low probability transitions from X, reducing the size of the transducer by 60% (the amount of pruning was determined empirically based on performance on a small held-aside set). The second strategy is to modify the original composition slightly: given a string S, we first compute $S \circ L \circ X$, then find the 300 best paths P (which correspond to the highest scoring confusable phone sequences according to X), and finally proceed with $P \circ L^{-1} \circ G$ (the number of confusable phone sequences to keep was also determined empirically). Extraction of n-best paths from the resulting transducer produces the simulated n-best lists.

3.2. Machine translation methods

Above we discussed methods for generating hallucinated ASR n-best lists using phone-based finite-state transducers. We now turn

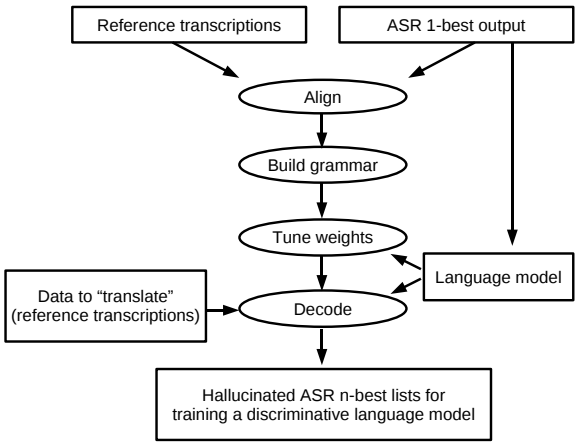

Fig. 1. "MT" pipeline for hallucinating ASR n-best lists.

to generating confusions at the word level. In this section, we discuss using standard phrase-based statistical machine translation techniques (MT) to "translate" from reference transcriptions to what those strings might be as ASR output.

Figure 1 presents the basic "MT" pipeline used to hallucinate ASR output. There are three input data sources required: 1) a parallel training corpus consisting of source sentences of human-generated reference transcriptions and target sentences of true ASR output; 2) data on which to build a language model for the target language output; and 3) data in the source language to be "translated" into the target language. We first generate word-level alignments of the sentences in the parallel corpus. From these word alignments, we build a phrase table and tune the weights of the feature functions for the phrase translation rules. Using the phrase table and associated weights, we decode the data to be "translated" to generate the hallucinated ASR n-best lists that can then be used as training data for the perceptron.

In the experiments presented here, we use only the top candidate from the ASR n-best lists as the target side of our parallel corpus. The language model for translation was built on the source side (i.e., the reference transcriptions) of the parallel training corpus. Giza++ ((http://code.google.com/p/giza-pp/) was used to train the word alignments. Grammar extraction, tuning, and decoding were performed with Moses (http://www.statmt.org/moses) using default settings with a few exceptions: since this kind of "translation" is monotonic, it was not necessary to build a reordering model; we also set the distortion limit of the decoder to zero in order to prohibit reordering during decoding.

3.3. Phrasal cohort methods

The approach described in this section bears some similarity to the MT-based confusion generation. As above, we think of the confusion generation problem as a kind of translation, but one that is in many ways simpler than translation between two natural languages. This allows us to use a more straightforward approach than in a full MT system, rather than configuring existing MT software for this task. The key observation here is that ASR confusions do not involve any reordering: the entire recognition pipeline is a time-synchronized process. Unlike in MT, where the translation rules have to be extracted in more complex ways, it is easy to identify the mistakes that an ASR system makes from simple string alignments.

p	$C(p)$
that to you	that too you
<s>she	<s>and she, <s>oh she
to you	to be you

Table 1. Examples of cohort

Central to our methodology is the definition of *cohort*. The cohort for a phrase p, denoted as $C(p)$, is the set of the phrases that are potential erroneous outputs where the correct transcription is p. Table 1 contains a few examples of the cohorts that we extracted from our baseline ASR system. Note that the phrases in Table 1 contain identical left and right contexts, which we call *pivots*, thus the confusions that we learned can be thought of as context-dependent. It is not necessary to constrain the pivots to be single words; we could also have multiple words or phone-level contexts serving as pivots. If we remove the pivots, we then get a set of context-independent rules. In either case, the key to extracting cohorts is identifying pivots, which can be obtained from Levenshtein string alignments.

To extract cohorts, we assume there is a set of *transcribed* speech data where we can learn phrase-level transformations. By aligning the reference string with the ASR hypothesis based on edit distance, we can easily find the errors that the ASR system produces. Below is the alignment between the reference and the 1-best ASR output.

Reference: <s>What kind of *a company is it* </s>
1-best: <s>What kind of *the campaign that* </s>

The pivot words are underlined. The phrases between the pivots form a phrase-level transformation, where "a company is it" is incorrectly recognized as "the campaign that". Such alignments can be done between the reference text and each one of the n-best hypothesis, where n can be a parameter to tune. We can also assign weights to the translation rule based on the number of times such errors appear in the n-best hypotheses. In the example above, let n be 10, if "a company is it" becomes "the campaign that" in all of the 10-best hypotheses, then the rule has probability 1, if 5 of them make this error, the probability is 0.5.

With the set of weighted translation rules obtained from the transcribed speech, we can hallucinate n-best lists from the text only. The set of cohorts essentially defines a phrase table which describes possible transformations from the reference to the ASR output. These translation rules can be applied easily to the text and generate n-best output. Application of these rules to a particular input string produces a confusion network. Each path through this network is a confusable string, which is scored according to the translation rule probabilities. Although it is possible to take generative n-gram language modeling scores into account in this framework, in this paper we used only scores derived from the translation rules.

4. EXPERIMENTAL RESULTS

In our experiments, we used the perceptron algorithm to train discriminative language models (DLM) for re-ranking n-best list output (100-best lists, more precisely) from the baseline ASR system. We follow the approach outlined in [1, 2], using unigrams, bigrams and trigrams as features in the model, and using the averaged perceptron for evaluation on held-aside, development and evaluation sets. The perceptron is an on-line algorithm, and after every epoch over the training set as a whole, performance was evaluated on a small held-aside set. Training stopped when performance on the held-aside set failed to improve for 5 epochs, and the averaged perceptron model for the lowest held-out WER epoch was output.

		dev	eval
1-best ASR		22.8	25.7
Real n-best lists	2 folds	22.3	–
	4 folds	21.8	–
	8 folds	21.7	24.8
Phone model lists	2 folds	22.4	–
	4 folds	22.3	–
	8 folds	22.3	25.3
MT model lists	2 folds	22.5	–
	4 folds	22.5	–
	8 folds	22.4	25.4
Cohort model lists	2 folds	22.3	–
	4 folds	22.4	–
	8 folds	22.2	25.2

Table 2. WER results using various n-best lists.

One difference from the methods in [1, 2] is that we did not use the baseline ASR system score as a feature during training of the n-gram DLM. Rather, we empirically optimized the scaling parameter for the baseline ASR score when adding it to the score of the DLM, after training the DLM independently. For the current results, we optimized the mixing factor on the development set, and used that mixing parameterization with both development and evaluation sets.

Table 2 presents our results obtained using 100-best lists hallucinated from text using each of the methods described in section 3. We show results obtained on development data with varying amounts of training data to illustrate the effect of dataset size, but test each method on the evaluation set using only the configuration with lowest WER on the development set, which, not surprisingly, is the configuration using the larger portion of the data for each method. For comparison, we also show the error rates for the 1-best result of the baseline ASR system, and for fully supervised DLMs trained on real n-best lists produced by the baseline ASR system. Using real n-best lists, we obtain an absolute improvement of 0.9% in word error rate over the baseline, or a 3.5% relative improvement. Our approach based on phrasal cohorts produced an absolute improvement of 0.5% over the 1-best baseline, which is statistically significant ($p < 0.001$) and more than half of the improvement obtained using real n-best lists. Our phone confusion approach produced an improvement of 0.4% ($p < 0.005$). Because these two approaches work at different levels (phones vs. words and phrases), it is possible that there might be ways to leverage both approaches for further improvements. The improvement over the baseline obtained using our MT-based approach was smaller, at 0.3%, but still statistically significant at the $p < 0.05$ level.

5. CONCLUSION

We presented three approaches for generation of hallucinated n-best lists from text that mimic ASR n-best lists produced from speech, and showed that discriminative language models trained on these hallucinated n-best lists can improve ASR word error rates from a strong baseline. One advantage of the use of hallucinated n-best lists is that it allows training of discriminative language models using arbitrary text, instead of transcribed speech.

In contrast to previous work, we compared the use of n-best lists produced using phone-level and word/phrase-level confusion modeling. We showed that approaches based on either level of confusion can be beneficial in discriminative language modeling, suggesting that an approach that works at both levels might produce even further gains.

6. REFERENCES

[1] B. Roark, M. Saraçlar, M. J. Collins, and M. Johnson, "Discriminative language modeling with conditional random fields and the perceptron algorithm," in *Proc. ACL*, 2004, pp. 47–54.

[2] B. Roark, M. Saraçlar, and M.J. Collins, "Discriminative n-gram language modeling," *Comput. Speech. Lang.*, vol. 21, no. 2, pp. 373–392, 2007.

[3] M. Collins, M. Saraçlar, and B. Roark, "Discriminative syntactic language modeling for speech recognition," in *Proc. ACL*, 2005, pp. 507–514.

[4] I. Shafran and K. Hall, "Corrective models for speech recognition of inflected languages," in *Proc. EMNLP*, 2006.

[5] N. Singh-Miller and M. Collins, "Trigger-based language modeling using a loss-sensitive perceptron algorithm," in *Proc. ICASSP*, 2007.

[6] E. Arisoy, M. Saraclar, B. Roark, and I. Shafran, "Discriminative language modeling with linguistic and statistically derived features," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 20, no. 2, pp. 540–550, 2012.

[7] H. Printz and P. Olsen, "Theory and practice of acoustic confusability," *Computer Speech and Language*, vol. 16, no. 1, pp. 131–164, 2002.

[8] J.-Y. Chen, P.A. Olsen, and J.R. Hershey, "Word confusability - measuring Hidden Markov Model similarity," in *Proc. Interspeech*, 2007.

[9] Q.F. Tan, K. Audhkhasi, P. Georgiou, E. Ettelaie, and S. Narayanan, "Automatic speech recognition system channel modeling," in *Proc. Interspeech*, 2010.

[10] Z. Li, Z. Wang, S. Khudanpur, and J. Eisner, "Unsupervised discriminative language model training for machine translation using simulated confusion sets," in *Proc. Coling*, 2010.

[11] Z. Li, Z. Wang, J. Eisner, S. Khudanpur, and B. Roark, "Minimum imputed-risk: Unsupervised discriminative training for machine translation," in *Proc. EMNLP*, 2011.

[12] P. Jyothi and E. Fosler-Lussier, "Discriminative language modeling using simulated ASR errors," in *Proc. Interspeech*, 2010.

[13] G. Kurata, N. Itoh, and M. Nishimura, "Acoustically discriminative training for language models," in *Proc. ICASSP*, 2009.

[14] G. Kurata, N. Itoh, and M. Nishimura, "Training of error-corrective model for ASR without using audio data," in *Proc. ICASSP*, 2011, pp. 5576–5579.

[15] P. Xu, D. Karakos, and S. Khudanpur, "Self-supervised discriminative training of statistical language models," in *Proc. IEEE ASRU*, 2009.

[16] H. Soltau, G. Saon, and B. Kingsbury, "The IBM Attila speech recognition toolkit," in *IEEE Workshop on Spoken Language Technology*, Dec. 2010, pp. 97–102.

[17] M. Lehr and I. Shafran, "Learning a discriminative weighted finite-state transducer for speech recognition," *IEEE Transaction on Audio, Speech and Language Processing*, vol. 19, no. 5, pp. 1360–1367, 2011.

[18] I. Bulyko, M. Ostendorf, M. Siu, T. Ng, A. Stolcke, and Ö. Çetin, "Web resources for language modeling in conversational speech recognition," *ACM Transactions on Speech and Language Processing (TSLP)*, vol. 5, no. 1, pp. 1–25, 2007.

[19] A. Ghoshal, M. Jansche, S. Khudanpur, M. Riley, and M. Ulinski, "Web-derived pronunciations," in *Proc. ICASSP*, 2009.