Estimating Speech Recognition Error Rate without Acoustic Test Data

Yonggang Deng, Milind Mahajan[†], Alex Acero[†]

Center for Language and Speech Processing
Johns Hopkins University, Baltimore, MD 21218, USA
`dengyg@jhu.edu`
[†]Speech Technology Group
Microsoft Research, One Microsoft Way, Redmond, WA 98052, USA
`{milindm,alexac}@microsoft.com`

Abstract

We address the problem of estimating the word error rate (WER) of an automatic speech recognition (ASR) system without using acoustic test data. This is an important problem which is faced by the designers of new applications which use ASR. Quick estimate of WER early in the design cycle can be used to guide the decisions involving dialog strategy and grammar design. Our approach involves estimating the probability distribution of the word hypotheses produced by the underlying ASR system given the text test corpus. A critical component of this system is a phonemic confusion model which seeks to capture the errors made by ASR on the acoustic data at a phonemic level. We use a confusion model composed of probabilistic phoneme sequence conversion rules which are learned from phonemic transcription pairs obtained by leave-one-out decoding of the training set. We show reasonably close estimation of WER when applying the system to test sets from different domains.

1. Introduction

The performance of an ASR system is closely tied to the training data used to train the acoustic model and the language model (LM). Consequently, in certain task domains, the speech recognition system will perform better than in other task domains. In order to determine how ASR system will work in a particular task domain, transcribed acoustic test data for that domain is needed, in addition to the dictionary and LM for the domain. Collecting a sufficient amount of transcribed acoustic test data to determine the error rate of the system is expensive and time-consuming and forms a barrier to developing speech enabled computer applications.

We assume that representative text data from the test domain is available. Given this, one solution would be to use lexical perplexity. However, lexical perplexity cannot be directly translated into word error rate (WER). This could be due the fact that it ignores the acoustic confusability of the words in the text and the base WER of ASR. For a given LM, it is possible to have poor correlation between WER and perplexity as shown in [1].

Fig. 1 shows block diagram of ASR operation at a very abstract level. Speaker speaks the intended word sequence W_c creating an acoustic realization A which is then decoded by ASR into hypothesis W_h.
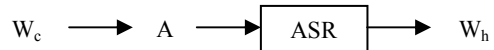

Figure 1: High-level block diagram of ASR process

It should be noted that the mapping from W_c to A is one-to-many due to speaker and acoustic channel variation. Mapping from A to W_h is deterministic many-to-one mapping for a given ASR system with fixed parameters. In other words, many acoustic realizations get mapped to the same word sequence but a given acoustic realization always gets mapped to the same word sequence.

In this paper, we describe a system which we call "Text Decoder" which can simulate ASR without acoustic data. Fig. 2 shows a block diagram of the Text Decoder. Text Decoder therefore encapsulates the speech production and ASR process as a black box.
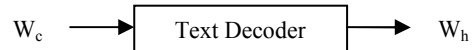

Figure 2: High-level block diagram of text decoding

In order to faithfully simulate the mapping from W_c to W_h, Text Decoder needs to take into account all the acoustic realizations which are possible and to estimate the mapping as a probability distribution.

Being able to simulate ASR has other applications besides WER estimation as well. Available acoustic training data is typically a small fraction of the text training data. Discriminative LM training can use the larger amount of text data if the confusing word sequences which need to be discriminated against can be predicted. Ability to predict the errors made by ASR may also lead to quicker identification of the parts of ASR system which need improvement.

Section 2 describes the framework which we use to estimate WER. Section 3 describes the structure and training of Confusion Model. Section 4 describes the search algorithm of the Text Decoder. Section 5 contains the experimental results followed by conclusions in Section 6.

2. WER estimation framework

Given the joint probability distribution $P(W_c, A, W_h)$ for ASR, WER can be calculated as:

$$WER = \frac{\sum\limits_{W_c, A, W_h} P(W_c, A, W_h) * ErrCount(W_h, W_c)}{\sum\limits_{W_c, A, W_h} P(W_c, A, W_h) * |W_c|} \qquad (1)$$

where, $ErrCount(W_h, W_c)$ is the number of word errors obtained by aligning the word sequences. For notational convenience, we treat A as being discrete even though it is actually a continuous variable. This can be re-written as:

$$WER = \frac{\sum\limits_{W_c} P(W_c) \sum\limits_{W_h} P(W_h | W_c) * ErrCount(W_h, W_c)}{\sum\limits_{W_c} P(W_c) * |W_c|} \qquad (2)$$

Here $P(W_c)$ is the distribution of the correct text sentences from the test domain and,

$$P(W_h | W_c) = \sum_A P(W_h | A) * P(A | W_c) \qquad (3)$$

So $P(W_h | W_c)$ is the expected distribution of the hypothesis W_h which would be generated by ASR given all the possible acoustic realizations corresponding to the test sentence text W_c. It should be noted that $P(W_h | A)$ is not the posterior probability of the word sequence but instead is the probability that ASR will output W_h as the 1-Best hypothesis. In other words,

$$P(W_h | A) = \delta(W_h, \arg\max_{W_h'} Score(W_h' | A)) \qquad (4)$$

We use Equation (2) as the basis for implementing the Text Decoder. Since, $P(W_c)$ is not known, we use the relative frequencies of the text data in the test corpus $\tilde{P}(W_c)$. $\tilde{P}(W_c)$ will tend towards $P(W_c)$ as the size of the test set grows if the test text corpus is drawn according to $P(W_c)$.

Robust estimation of $P(W_h | W_c)$ directly at word sequence level would be difficult given the sparseness of data for multi-word sequences. Also, it would not generalize well in the cases where the test domain vocabulary is different from the vocabulary of the training set. It would, therefore, be better to decompose it further using sub-word units such as phonemes. Equation (5) gives a phoneme level decomposition using chain rule and reasonable approximations.

$$P(W_h | W_c) \approx \sum_{\varphi_h, \varphi_c} P(W_h | \varphi_h) P(\varphi_h | \varphi_c) P(\varphi_c | W_c) \qquad (5)$$

Here, φ_c and φ_h represent the phoneme sequences in the correct and hypothesis word sequences respectively. Approximation is due to the assumption that the correct phoneme sequence captures all the relevant information in the correct word sequence. In using, Equation (5), Text Decoder gets $P(\varphi_c | W_c)$ from a dictionary or a pronunciation model. We refer to $P(\varphi_h | \varphi_c)$ as the Confusion Model. Since,

$$P(W_h | \varphi_h) = \frac{P(\varphi_h | W_h) * P(W_h)}{\sum\limits_{W_h'} P(\varphi_h | W_h') * P(W_h')} \qquad (6)$$

a grammar or LM is required in addition to a dictionary to disambiguate between the word sequences which result in identical phoneme sequences such as "way to" and "weigh two". Text Decoder uses these components as depicted in Fig.

3. It takes test text as input, and predicts the probability distribution of the ASR 1-Best hypotheses.
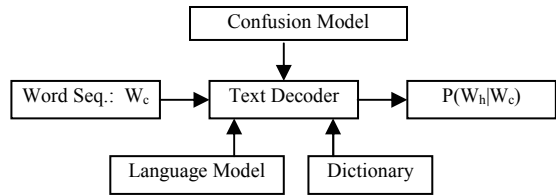

Figure 3: Text decoder

2.1. Confusion model alternatives

The critical problem is how the acoustic confusion complexity can be modeled and obtained. One possible way is to use acoustic encoding probability $P(x | w)$ proposed by [2]. The expected log-likelihood of model x's acoustic observation when evaluated by model w can be theoretically analyzed. The desired probability can be derived from log-likelihood. The advantage of this approach is that it works directly from the trained acoustic model and does not need access to the acoustic training data. A possible drawback is that it makes the same assumptions that the acoustic model makes. Some of the independence assumptions made by the acoustic model may lead to a significant under-estimation of the actual confusability of the models using real acoustic data.

An alternative is to learn the confusion model from the training data. If training sentences and the corresponding ASR hypotheses are taken as input string pairs, the edit operation distribution can be learned with a certain criterion and learning algorithm. [3] proposed an EM-algorithm to learn stochastic model for string edit distance. String pair probability is defined from edit operation probability and Forward-Backward algorithm is employed to maximize likelihood of training data.

Context dependent phoneme conversion rules with probabilities were used in [4] for pronunciation modeling. We decided to use a similar structure for the Confusion Model. Other techniques in the field of pronunciation modeling may also be useful for Confusion Model. [5] provides a good overview of the field.

Tsai and Lee [6] use a framework similar to ours. They address the different problem of minimizing WER for a given domain by improving the pronunciation dictionary. They choose to model the confusion entirely within the pronunciation dictionary and not at a general phoneme sequence level as we do. Consequently, our confusion model is more general. It is independent of the training vocabulary and the domain. It also allows us to model confusions at the whole sentence level rather than at the word level alone.

3. Confusion model

3.1. Training data for confusion model

To obtain the phonemic string pairs needed for training the confusion model, we start with acoustic training data transcribed at word level. We divide acoustic training data into multiple parts in order to use leave-one-out method.

In turn, all but one (the left-out) part of the data is used to train a state-tied tri-phone acoustic model from scratch using

HTK toolkit [7]. The left out part is then decoded by ASR using the acoustic model described above to get word hypotheses. Reference and hypothesis phoneme sequences are then obtained from reference (transcribed) and hypothesis word sequences using forced alignment of the acoustic data against the corresponding phoneme networks. These phoneme sequence pairs serve as the training data for the confusion model.

Leave-one-out method is essential to avoid bias in the phoneme pairs. Otherwise, acoustic model used to obtain the hypotheses phoneme sequences will make substantially less mistakes on the acoustic data which it has already seen during training. If acoustic data transcriptions are used for LM training, then a similar leave-one-out procedure is also necessary for LM training.

The phonemic string pairs are expected to be able to fully expose internal acoustic confusion. However, it is not clear how ASR LM should be configured. By intuition, a weaker LM would do a better job in terms of exposing the underlying acoustic confusability. The learned model will thus be influenced less by the training domain specific language patterns. The results in section 5 confirm this intuition.

3.2. Phoneme conversion rules

We characterize the mapping from reference phoneme sequence to hypothesis phoneme sequence by learning probabilistic rules similar to those described in [4]. Each rule provides the probability of conversion from zero or one phoneme in the reference to zero, one, or more than one phoneme in the hypothesis sequence within a certain context.

Let Ω be the phoneme set including the silence phoneme *sil* which we also use to mark word boundaries. Let ε denote an empty phoneme sequence. Then, each rule has the form: $L - F + R \rightarrow F'$, p, which means the focus phoneme $F \in \{\varepsilon\} \cup \Omega$ would be replaced by $F' \in \Omega^*$ with probability p when it occurs in the context of phoneme sequence L to its left and phoneme sequence R to its right. We constrain the context phoneme sequences L and R to be of length 2 or less. In other words , L and R belong to $\{\varepsilon\} \cup \Omega \cup \Omega^2$. The probability p associated with each rule is $P(F'|L - F + R)$. The rules of this form can model insertion, deletion and substitution through the appropriate use of ε for F or F' .

To learn the probability rules, we examine the alignment between the ASR decoded phoneme sequence and the reference phoneme sequence. For each phoneme in the reference sequence, we create a rule of the form indicated above, using the alignment and all possible contexts of length 2 or less. We also create rules with ε as the focus for every position in the reference phoneme sequence to model insertions. The probability associated with each rule is simply the relative frequency: $C(L - F + R', F') / C(L - F + R')$. We use count-cutoffs to prune the rule set to ensure robustness and efficiency.

4. Text decoding

For a task domain, the Text Decoder will take test text as input, apply dictionary and LM for the test domain that would

also be used by ASR, and output the estimated distribution of hypotheses that would be generated by ASR.

For simplicity of implementation, we assume that there is a unique pronunciation φ_c for a given reference test text sentence. Given φ_c , we insert *sil* word boundary marker between words. The Text Decoder then finds out all possible rules that could apply to every position within φ_c by matching left hand sides (LHS) of all the rules against φ_c . If multiple LHS match at a particular position, Text Decoder selects a single LHS with the largest context length. Linear interpolation is also possible, although we did not experiment with it. Application of the selected rules to φ_c creates a network which represents the set of paths representing all the possible hypothesis phoneme sequences φ_h that would be generated using the Confusion Model i.e. $P(\varphi_h | \varphi_c)$. Fig. 4 shows an example of a partial phoneme network. Notice that *sil* (word boundary) between *ey* and *t* can be deleted with probability 0.18, leading to the possible errors caused by the replacement of the word *way* with *wait* or *weight*.
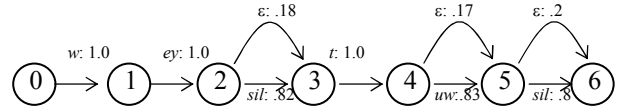

Figure 4: Partial phoneme network for reference phrase "way to"

To generate hypothesis words W_h , the Text Decoder explores the paths in the φ_h network by traversing the network using standard depth-first-search strategy. To increase the efficiency and to restrict the number of paths explored, the Text Decoder prunes partial paths whose partial likelihood falls below certain threshold from the best complete path explored so far. Also, partial paths which do not correspond to a valid word sequence as indicated by the dictionary are also pruned.

Due to the presence of word boundary *sil*, different paths in the hypothesis phoneme network may correspond to the same pronunciation. The Text Decoder uses Equations (5) and (6) simplified using the single pronunciation per word assumption to estimate $P(W_h | W_c)$.

Correct Word Sequence: What is the cheapest way to fly from …	Probability
What is the cheapest way to fly from …	0.80
What is the cheapest way to **flight** from …	0.12
What is ~~the~~ cheapest way to fly from …	0.04
What is the cheapest **weight** fly from …	0.02
What is the cheapest **wait** fly from …	0.02

Table 1: Word sequences predicted by the Text Decoder with associated probabilities

Table 1 shows a typical example of hypothesis distribution. As pointed out in Section 2, this is not an N-Best

list with posterior probability but an estimate of ASR one-best hypothesis probabilities.

5. Experimental results

ATIS data was used for building confusion models. ASR system to be predicted used Viterbi decoder with state-tied triphone acoustic model trained using all ATIS training data.

We used 5-way leave-one-out training on acoustic model and LM to obtain phonemic transcription pairs described in section 3.1. We tried three types of LMs during the process of generating hypothesis phoneme sequences by decoding the left-out part of the training data. The LMs were: phone-bigram LM, word unigram LM, and word trigram LM. These resulted in three sets of phoneme transcription pairs. We then built a confusion model from each set of phoneme transcription pairs. We will refer to these confusion models as phone-bigram, word unigram and word trigram confusion models respectively.

5.1. WER estimation within training domain

Since the acoustic model of ASR was built from all ATIS training data, naturally one would like to know how close the predictions on ATIS test task would be.

We created 3 different test conditions within ATIS domain by using ASR with unigram, bigram, and trigram LM to decode the test data. We will refer to LM used in each test condition as a Test LM. For every test condition, the Text Decoder and ASR used the same test LM. For each test condition, we compare the WER predicted by each of the 3 confusion models described above with the actual WER of ASR. The results are shown in Table 2 with the closest matching Text Decoder prediction for each test condition in boldface.

It is clear that confusion model generated using phone-bigram and word-unigram estimates ASR WER much better than the confusion model generated using word-trigram. This confirms the intuition that a weaker LM would be better at discovering acoustic confusion.

Test LM	ASR	Text Decoder with Confusion Model generated using		
		Phone-Bigram	Word-Unigram	Word-Trigram
Unigram	15.4	**9.8**	9.0	6.2
Bigram	4.8	5.6	**4.6**	1.9
Trigram	3.9	5.4	**4.5**	1.7

Table 2: Comparison of WER estimate by Text Decoder with real ASR WER

5.2. WER estimation for new domains

We of course want the confusion model to be able to estimate WER for applications in new test domains. Our confusion models were trained from ATIS corpora. We applied the models to predict ASR WER in other test domains such as: Wall Street Journal dictation task (WSJ5K) and TI Digits. The results in Table 3 show that the predictions are reasonably close.

What deserves mention here is the prediction on WSJ5K task which has a much larger vocabulary size and certainly contains more acoustic phenomena than the training corpus. Yet the estimation is pretty close.

		WSJ5K	TI_DIGITS	
			MALE	FEMALE
Vocabulary size |V|		4986	11	11
Test LM		Bigram	Uniform	Uniform
# of test utterance.		318	4K	4K
W E R	ASR	18.3	3.8	1.6
	Phone-Bigram Confusion Model	**19.3**	**2.7**	2.7
	Word-Unigram Confusion Model	15.1	1.3	**1.2**

Table 3: Comparison of WER estimate by Text Decoder with real ASR WER in New Domains

6. Conclusion and future work

In order to estimate ASR WER in a task domain without acoustic data, we proposed a Text Decoder architecture which estimates the distribution of ASR 1-best hypotheses using only a text test corpus. We showed how a phoneme-level confusion model based on context-dependent phoneme conversion rules can be used to capture acoustic model confusion. We experimentally validated Text Decoder by showing reasonably close ASR WER prediction results both on the training domain on which the confusion models were built and for new domains.

There are other potential areas of application of the proposed Text Decoder such as: a development tool which will allow the designer of ASR enabled application to identify parts of the application grammar which are likely to lead to high WER; and discriminative LM training, where the LM model parameters can be estimated while taking into account the potentially confusable competing word sequences that are discovered by the Text Decoder without requiring acoustic data for the LM training corpus.

7. Acknowledgements

8. References

[1] P. Clarkson and T. Robinson, "The applicability of adaptive language modelling for the broadcast news task" *ICSLP*, 1998.

[2] H. Printz and P.Olsen, ``Theory and practice of acoustic confusability" in *ISCA ITRW ASR2000*, pp. 77—84, 2000.

[3] E. Ristad and P. Yianilos, ``Learning string edit distance" in *IEEE Trans. PAMI*, 20, pp. 522--532, 1998.

[4] N. Cremelie and J. Martens, "In Search of Better Pronunciation Models for Speech Recognition", Speech Communication, 29, pp. 115-136, 1999.

[5] H. Strik and C. Cucchiarini "Modeling pronunciation variation for ASR: A survey of the literature", Speech Communication, 29, pp. 225—246, 1999.

[6] M. Tsai and L. Lee, "Pronunciation modeling for spontaneous speech by maximizing word correct rate in a production-recognition model", IEEE and ISCA Workshop on Spontaneous Speech Processing and Recognition, 2003.

[7] S. Young et al, ``The HTK Book, Version 3.0", 2000.