Stacked Auto-Encoder for ASR Error Detection and Word Error Rate Prediction

Shahab Jalalvand\textsuperscript{1,2}, Daniele Falavigna\textsuperscript{1}

\textsuperscript{1} FBK - Fondazione Bruno Kessler, Via Sommarive 18, 38123 Trento, Italy
\textsuperscript{2} ICT Doctoral School, University of Trento, Italy
\{jalalvand,falavi\}@fbk.eu

Abstract

Recently, Stacked Auto-Encoders (SAE) have been successfully used for learning imbalanced datasets. In this paper, for the first time, we propose to use a Neural Network classifier furnished by an SAE structure for detecting the errors made by a strong Automatic Speech Recognition (ASR) system. Error detection on an automatic transcription provided by a "strong" ASR system, i.e. exhibiting a small word error rate, is difficult due to the limited number of "positive" examples (i.e. words erroneously recognized) available for training a binary classifier. In this paper we investigate and compare different types of classifiers for automatically detecting ASR errors, including the one based on a stacked auto-encoder architecture. We show the effectiveness of the latter by measuring and comparing performance on the automatic transcriptions of an English corpus collected from TED talks. Performance of each investigated classifier is evaluated both via receiving operating curve and via a measure, called mean absolute error, related to the quality in predicting the corresponding word error rate. The results demonstrate that the classifier based on SAE detects the ASR errors better than the other classification methods.

Key-words: automatic word error detection, stacked autoencoder, word error rate prediction.

1. Introduction

An important aspect of Automatic Speech Recognition (ASR) systems is to try to predict the probable erroneous words that have been recognized. In the real applications where the reference is not available, the need for an accurate error detection is more tangible. Word Error Rate (WER) prediction and ASR Quality Estimation are some of these applications [1, 2, 3]. In spoken language translation, where the output of an ASR system is directly transferred to a Machine Translation (MT) module, knowing the erroneous words and predicting the quality of the transcription can saliently improve the final translation performance [4, 5].

Only from the ASR side, detecting the errors, correcting them and selecting the probable alternatives help in purifying the final output. In addition, Out-Of-Vocabulary and Named-Entity detection [6] are the other tasks, in which the error detection algorithms can be utilized. So far, the most ASR error detection research has been carried out in the problematic speech recognition conditions, such as conversational speech, dialog systems, telephone and noisy environments [4, 5]. Whereas, in the current paper, we work with an ASR system, based on DNN-HMM acoustic model [7], which is trained and tested on a clean, spontaneous speech corpus of English lectures, namely TED talks (see section 3 for the details). Detecting the errors of the latter ASR system is a difficult task, since the number of erroneously recognized words is much lower than the corresponding number of correctly recognized ones. This faces us to the problem of learning from imbalanced data sets [8], which is by itself a challenging research objective.

In this work, we cast ASR word error detection as a classification problem and we introduce a deep neural network structure which allows detecting the minor mistakes of an ASR system better than the traditional word error detectors. This is inspired by [9, 10] that have recently reported a considerable performance of Deep Blief Network and Stacked-Auto Encoder (SAE) on various skewed and imbalanced datasets.

During the experiments, we compare three baseline classifiers named Support Vector Machine (SVM), Extreme Randomized Tree (XRT) and Maximum Entropy (MAXENT) with our proposed SAE approach. As the comparison measure, we first investigate the Receiver operating characteristic (ROC) curves resulted by each classifier and we show that SAE outperforms the others. Furthermore, additional comparative evaluations were carried out on WER prediction task, a most recent research field that we started to study in a more general ASR quality estimation framework [1, 2]. For the latter evaluation, we count the detected errors in each test utterance and we compare the Mean Absolute Error (MAE) between the real error rate and the error rate predicted by the detector. Again, we observe that SAE predicts the error rate better than the other approaches. It’s worthwhile to note that usage of MAE for comparing ASR error detectors is another contribution of this paper.

The rest of this paper is organized as follows. After a brief review on the related works in Section 2, in Section 3, we describe the ASR system employed in this work and the data sets used for the experiments. In Section 4 we describe the classifiers adopted for error detection, their architectures and the features used. In Section 5 we report the experiments, the metrics used for performance evaluation and the results achieved. Finally, Section 6 gives the conclusions.

2. Related Works

This work comprises three different subjects: ASR error detection, imbalanced dataset learning and WER prediction. ASR error detection has been tackled by many researchers aim to improve the performance of ASR systems. In [11], a set of decoder-independent features are studied for identifying the errors. These features are extracted based on: first, disagreement of two complementary ASR systems; second, the number of bi-gram occurrences provided by a web search engine and third, the topic related to each word. The work described in
The English ASR system used in the experiments is developed an out-of-domain one (formed by the modified shift-beta approach) 4-grams of two collections: using the IRSTLM toolkit [15] by mixing the smoothed (with model (LM) used for decoding is a back-off 4-gram LM built Model (DNN-HMM) hybrid architecture [7]. The language performance thanks to its Deep Neural Network Hidden Markov combination of all of the available LMs, as explained in [17, 18]. Performance was measured on two different speech corpora: "Dev2010", that is the development set of the ASR track evaluation of IWSLT2010 [19], and "Test2013" that is the test set of IWSLT2013 [20].

Eventually, the obtained performance was 12.6% and 16.3% WER on Dev2010 and Test2013, respectively. The main reason for degrading the performance from Dev2010 to Test2013 is due to the fact that Dev2010 consists of mostly native American-English speakers, while Test2013 is more problematic since it includes non-native and stuttering speakers, as well. Table 1 gives the statistics of the two mentioned corpora as well as the corresponding performance values.

This work aims at both: 1) detecting the erroneous words in the automatic transcription and 2) estimating the error rate in the transcription by counting the number detected errors. Note that, since detection of deletions is not considered for this work, we introduce a new measure named Insertion and Substitution Error Rate (ISER), which is the same as WER but without considering deleted words. The %ISER obtained on Dev2010 and Test2013 was 11.2% and 14.5%, respectively (see Table 1).

3. Baseline ASR System

The English ASR system used in the experiments is developed in our Labs for the IWSLT 2013 evaluation campaign1, where the ASR track focused on the transcription of TED talks. The latter is a global set of conferences whose audio/video recordings are available through the Internet2. The main challenges for automatic transcriptions of TED talks include: variability in acoustic conditions, large variability of topics (hence a large, unconstrained vocabulary), presence of non-native speakers and a rather informal speaking style.

The baseline ASR system was trained on 144 hours of in-domain data (i.e. TED talk videos released before the cut-off date, 31 December 2010). For both training and decoding we used the KALDI toolkit [14] which has shown excellent performance thanks to its Deep Neural Network Hidden Markov Model (DNN-HMM) hybrid architecture [7]. The language model (LM) used for decoding is a back-off 4-gram LM built using the IRSTLM toolkit [15] by mixing the smoothed (with the modified shift-beta approach) 4-grams of two collections: an out-of-domain one (formed by ~1 billion of words) and an in-domain one (consisting of ~2.7 million of words). In addition, we exploited two RNNLMs [16]: one trained on the in-domain corpus (~2.7M words) and the other trained on a collection of ~13M words built adding to the in-domain corpus ~10M words automatically extracted from the out-of-domain text data. Details on both procedure for automatic selection of text data and RNNLM training approach can be found in [17].

For decoding we employed a graph built using a "pruned" version of the 4-gram mix-adapted LM introduced above. The word lattices generated for each utterance by the DNN-HMM hybrid system were finally rescored using a linear combination of all of the available LMs, as explained in [17, 18]. Performance was measured on two different speech corpora: "Dev2010", that is the development set of the ASR track evaluation of IWSLT2010 [19], and "Test2013" that is the test set of IWSLT2013 [20].

Eventually, the obtained performance was 12.6% and 16.3% WER on Dev2010 and Test2013, respectively. The main reason for degrading the performance from Dev2010 to Test2013 is due to the fact that Dev2010 consists of mostly native American-English speakers, while Test2013 is more problematic since it includes non-native and stuttering speakers, as well. Table 1 gives the statistics of the two mentioned corpora as well as the corresponding performance values.

This work aims at both: 1) detecting the erroneous words in the automatic transcription and 2) estimating the error rate in the transcription by counting the number detected errors. Note that, since detection of deletions is not considered for this work, we introduce a new measure named Insertion and Substitution Error Rate (ISER), which is the same as WER but without considering deleted words. The %ISER obtained on Dev2010 and Test2013 was 11.2% and 14.5%, respectively (see Table 1).

4. ASR Error Detection Methods

In this section, we first introduce the feature sets utilized to feed the various investigated classifiers and then, we describe our proposed SAE structure for classifying errors.

4.1. Features

The features that have been used for error detection could be divided into three main categories: ASR-based features, hybrid features and textual features [1, 21, 12, 11].

ASR features aim to capture the confidence of the speech recognizer and the reliability of the whole decoding process. In this work they are 10 and include: log of the current word posterior probability, log of the posterior of the previous word, log of the posterior of the next word, mean of the posteriors in the confusion network (CN) bin, standard deviation (std) of the posteriors in the CN bin, number of alternatives in the CN bin, number of the alternatives whose posterior is less than a threshold, relative position of the word in the sentence. In addition, two binary features are employed answering the questions: "is the previous word silence?" and "is the next word silence?".

Hybrid features provide a more fine-grained way to capture the difficulty of transcribing the signal. This is done by considering information about both energy and pitch in each hypothesized word segment, as well as the respective duration.
Hybrid features exploited here are 22 including: duration of the word in second, means of 12 Mel Frequency Cepstral Coefficients (MFCC), mean/max/min/std of energies of the word frames, mean/max/min/std of pitch values of the word frames, the corresponding ratio between max and min pitch values. Pitch features have been computed with the Praat software tool [22].

**Textual features** aim to capture an a-priori plausibility of an output transcription. To this aim, we consider information about LM probability of each hypothesized word (both at the level of words and parts of speech). The part of speech (POS) has been obtained by processing with the TreeTagger [23]. Textual features are 6, namely: the LM probability given by the above mentioned mix 4-gram LM, two RNNLM probabilities, number of phonemes of the word, POS tag and POS score given by the POS tagger.

We evaluated these feature sets with regard to their information gain for classifying correct/error words on Dev2010 transcription. Figure 1 shows the gain of each feature set. Unsurprisingly, the ASR features own up the highest information gain and, after that, also textual features carry a significant level of information. Hybrid features do not show any level of importance, although more effective hybrid features, as the ones described in [3, 11], could be exploited. Hereinafter, in the experiments we will only use as input features the union of ASR and textual features.

![Figure 1: Information Gain](image)

### 4.2. SAE for ASR Error Detection

According to the literature, the most popular classifiers for ASR error detection are SVM, XRT, MAXENT. However, the performance of these classifiers degrades dramatically when the dataset is highly imbalanced. So that, in this paper, for the first time to the best of our knowledge, we use a Neural Network classifier furnished by the Stacked Auto-Encoders (SAE) as the hidden layers. The SAE helps in learning the error word representation.

SAE is a deep neural network structure consisting of a stack of Auto-Encoders (AE) building the hidden layers. Figure 2 shows an example of a 2-layer NN classifier whose hidden layers are formed by SAE structures. An AE is usually a single hidden layer network, in which, the input and output layers are the same. Therefore, the AE learns to represent the input layer in a new form which is defined by its hidden layer. The AEs are usually pre-trained using Restricted Boltzmann Machine (RBM) [24] algorithm to set the output with the same size as the input.

![Figure 2: An example of a Neural Network classifier furnished by SAE representation learning](image)

### 5. Experiments

In this section, we first introduce the evaluation metrics used for comparing the error detection approaches and, then, we report the results. Training of all of the classification systems used in the experiments was carried out exploiting data derived from "Dev2010" speech corpus, that is from the alignment between the automatic word transcription of each utterance with the corresponding reference one. Labels corresponding to insertion and substitution errors provide the "true" samples, while labels corresponding to correctly recognized words furnish the "negative" samples. Input features are extracted, as previously mentioned, from each word segment to be classified. Performance evaluation was led on "Test2013" corpus, where reference labels are obtained via alignment between automatic and reference transcriptions.

#### 5.1. Evaluation Metrics

When the data is highly imbalanced, some assessment metrics such as simple accuracy (hit rate) are not reliable. Instead, balanced accuracy or G-mean provide better understanding of the classification performance. However, these metrics suffer from an crucial disadvantage in ASR error detection: the higher balanced accuracy, the higher number of correct words mistakenly assigned as errors, the worse prediction of WER. Despite, in the experiments, we first use ROC curve for comparing the performance of the classifiers. Then, we explore the performance of different error detectors for error rate prediction in each individual utterance. As mentioned before, since the deletion errors are not known in the transcription, we cannot estimate the exact WER. Instead, we define another term, ISER-Insertion and Substitution Error Rate, which only takes into account the in-
sions and substitutions. Hence, for the \(i^{th}\) utterance \(S_i\), the predicted ISER (pISER) is computed by:

\[
p_{ISER}(S_i) = \frac{E(S_i)}{L(S_i)}, \quad i = 1..N
\]

where, \(E(S_i)\) is the number of insertions and substitutions in \(S_i\) and \(L(S_i)\) is the total number of recognized words in for \(S_i\). Likewise, we define the real ISER (rISER) by considering the oracle error detector. That is, rISER is considered to detect the exact number of errors occurred in the transcription. This oracle number of errors is shown by \(\hat{E}(S_i)\):

\[
r_{ISER}(S_i) = \frac{\hat{E}(S_i)}{L(S_i)}, \quad i = 1..N
\]

Finally, as a comparison metric, we compute the Mean Absolute Error (MAE) between the real ISER (rISER) and the predicted ISER (pISER):

\[
MAE = \frac{1}{N} \times \sum_{i=1}^{N} |p_{ISER}(S_i) - r_{ISER}(S_i)|
\]

5.2. Results and Discussion

![Figure 3: ROC curve of ASR error detection on Dev2010.](image)

As mentioned above we trained different classifiers on data derived from Dev2010 speech corpus. To tune the parameters of each classifier we first applied k-fold cross validation, selecting \(k = 5\) in order to have enough number of error samples in each fold for both training and validation. We compare linear SVM, XRT and MAXENT to SAE, by reporting the miss-classification probability vs. false alarm rate in Figure 3.

As can be seen from Figure 3, the curve resulted by SAE is significantly better than the one resulted from SVM and slightly better than those corresponding to XRT and MAXENT. For training the SVM classifier, we used libSVM [26], with radial basis function (RBF) kernels. For XRT, we used a MATLAB ExtraTrees package [27] with 10 trees, 3 randomly selected attributes per tree and minimum number of 10 samples per leaf. The MAXENT classifier was trained using a Maximum Entropy package described in [28]. For SAE method, we utilized a deep learning toolbox [29] to train a FFNN-SAE with 4-layer of 16-10-10-2 neurons in the Input-Hidden1-Hidden2-Output layers. Note that in the experiments we use 16-dimension feature vectors including the ASR and textual features.

<table>
<thead>
<tr>
<th></th>
<th>Dev2010</th>
<th>Dev2010/Test2013</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVM</td>
<td>8.22</td>
<td>11.21</td>
</tr>
<tr>
<td>XRT</td>
<td>7.30</td>
<td>10.22</td>
</tr>
<tr>
<td>MAXENT</td>
<td>6.90</td>
<td>9.48</td>
</tr>
<tr>
<td>SAE</td>
<td>6.68</td>
<td>8.41</td>
</tr>
<tr>
<td>MAXENT+SVM</td>
<td>6.48</td>
<td>8.01</td>
</tr>
</tbody>
</table>

It’s worthwhile to mention that we exploited an input feature vector formed by the concatenation of only ASR and textual features, i.e. hybrid features were discarded since, according to Figure 1, they to not carry useful information. Furthermore, for the sake of brevity, we avoid reporting the details of different learning procedures experimented by us. Figure 3, only shows the best performance of each classifier.

Once classifier training and optimization was completed, we applied error detection to both Dev2010 and Test2013 data sets and with the achieved results we computed the predicted ISER, using the formula defined in (1), for each individual utterance. Then, we evaluated the MAE between the predicted and real ISERs using the formula in equation (3). Table 2, shows the MAE measure obtained by each classification method. The second column (Dev2010/Test2013) shows the results obtained on the development set by averaging the corresponding cross validation performance, the third column (Dev2010/Test2013) shows the results obtained on the test set. We observe in the second column that SAE outperforms the others classification approaches, i.e. SVM, XRT and MAXENT, by 18.7%, 8.4% and 3.1% relative improvement, respectively.

The results of the third column in Table 2 is more interesting and realistic at same time, since it refers to the case where training and test conditions do not match. In particular it is important to observe that none of the speakers in the test set (Test2013) is included in the training set (Dev2010). In this latter case, SAE outperforms SVM, XRT and MAXENT by relative improvements of 24.9%, 17.7% and 11.2%, respectively. The latter result not only proves that SAE is able to learn the errors made by an ASR system much better than the other classifiers, but it also exhibits generalization capabilities, not being biased towards the training data. Finally, further improvements (last row of Table 2) are obtained by combining the best two classifiers, i.e. SAE and MAXENT, by simply applying majority voting.

6. Conclusions

A Feed Forward Neural Network classifier furnished by a Stacked Auto-Encoder (SAE) structure is proposed in this paper, for detecting the insertion and substitution errors committed by a precise DNN-HMM based ASR system. On English speech corpora, we showed that the proposed structure outperforms traditional ASR error detection methods. Moreover, better error rate prediction is obtained after identifying the ASR errors by means of SAE.

As the future work, we will explore a larger set of features, especially hybrid features for training the classifiers. Additionally, the usage of the complementary ASR systems for extracting new features, as well as evaluating the more efficient SAE topologies are underway.
7. References


