cROVER: IMPROVING ROVER USING AUTOMATIC ERROR DETECTION

K. Abida* F. Karray* W. Abida†

* University of Waterloo
Department of Electrical and Computer Engineering
Waterloo, Ontario, CANADA
† VESTEC Inc.
Waterloo, Ontario, CANADA

ABSTRACT
Recognizer Output Voting Error Reduction (ROVER), is a well-known procedure for decoders’ combination aiming at reducing the Word Error Rate (WER) in transcription applications. However, it appears that this technique has reached a plateau in terms of performance. This paper presents a novel approach, cROVER, in order to boost the current ROVER performance, by relying on a contextual analysis to trim erroneous words. Experiments have proven that it is possible to outperform ROVER, despite the high false positive rate of the error detection technique.

Index Terms— ROVER, PMI, WER, Error Detection

1. INTRODUCTION

Many researchers agree that one of the most promising approaches for the problem of reducing the WER in Large Vocabulary Continuous Speech Recognition (LVCSR) is to combine two or more speech recognizers and compile a new output, in the hope that this latter will ensure a lower WER. ROVER[1], is a system developed at the National Institute of Standards and Technology (NIST), to produce a composite Automatic Speech Recognizer (ASR) output when multiple ASR outputs are available. ROVER is a two-step process. First, it combines the multiple outputs into a single, minimal cost word transition network (WTN) through dynamic programming. Once this alignment is done, the WTN is browsed by a voting process which selects the best output sequence (with the highest votes). Three voting mechanisms have been presented in [1], two of which involved the use of word-level confidence values. An exhaustive review in [2], has outlined the most recent advances in improving the baseline ROVER to overcome its limitations and achieve lower WER. However, the authors in [2], concluded that this technique’s performance has reached a plateau, and researchers have been trying hard to enhance the performance, using various advanced computational intelligence approaches. This paper proposes a novel approach towards boosting the baseline ROVER performance, via its integration with an automatic error detection technique. The proposed system is referred to as cROVER, for context-augmented ROVER. The idea is to flag erroneous words following the combination of the word transition networks through a scanning process at each slot of the resulting network. This step aims at eliminating some transcription errors and thus facilitating the voting process in ROVER. The error detection technique consists of spotting semantic outliers within a given window of context words. Even though the error detection techniques in the literature suffer from a high false positive rate, our experimental results have shown that this approach is able to improve the performance of the ROVER procedure. This paper is organized as follows. The proposed cROVER method is detailed in section 2, followed by a set of experimental results and analysis in section 3. The conclusion then highlights our findings.

2. PROPOSED APPROACH: cROVER

This research work aims at removing errors out of the recognizers’ output before applying any voting during ROVER combination of the different decoders. The Point-wise Mutual Information-based error filtering technique has been used to spot recognition errors.

2.1. PMI-based Error Detection

We need to define a few items before describing the error-detection procedure in detail. The neighborhood $N(w)$ of a word $w$ is the set of context tokens around $w$ that appear before and after it. This concept is defined within a window of tokens. The Point-wise Mutual Information (PMI)-based semantic similarity is a measure of how similar and how close in meaning $w_i$ and $w_j$ are. In a nutshell, the PMI score in equation 1 below is defined as the probability of seeing both words ($w_i$ and $w_j$) together, divided by the probability of observing each of these words separately.

$$PMI(w_i, w_j) = \log \left( \frac{P(w_i, w_j)}{P(w_i)P(w_j)} \right)$$

(1)
Given a large textual corpus with size \( N \) tokens, the probabilities introduced in 1 can be computed using

\[
P(w_i) = \frac{c(w_i)}{N},
\]

and

\[
P(w_i, w_j) = \frac{c(w_i, w_j)}{N},
\]

where \( c(w_i) \) and \( c(w_i, w_j) \) are the frequency counts collected from the corpus. The process of detecting an error using the PMI-based technique\[3\] is described in algorithm 1.

Algorithm 1 PMI-based Error Detection

1. Identify the neighborhood \( N(w) \).
2. Compute PMI scores \( PMI(w_i, w_j) \) for all pairs of words \( w_i \neq w_j \) in the neighborhood \( N(w) \), including \( w \).
3. Compute Semantic Coherence \( SC(w_i) \) for every word \( w_i \) in the neighborhood \( N(w) \), by aggregating the \( PMI(w_i, w_j) \) scores of \( w_i \) with all \( w_j \neq w_i \).
4. Define \( SC_{avg} \) to be the average of all the semantic coherence measures \( SC(w_i) \) in \( N(w) \).
5. Tag the word \( w \) as an error if \( SC(w) \leq K \cdot SC_{avg} \).

Different window sizes have been used for the first step. In step 3 of the algorithm, the semantic coherence has been computed using different aggregation variants: Harmonic mean: \( SC(w_i) = \sum_{i \neq j} \frac{1}{PMI(w_i, w_j)} \), Arithmetic mean: \( SC(w_i) = \frac{1}{n} \sum_{i \neq j} PMI(w_i, w_j) \), Maximum: \( SC(w_i) = \max_{i \neq j} PMI(w_i, w_j) \), and Sum: \( SC(w_i) = \sum_{i \neq j} PMI(w_i, w_j) \). The filtering parameter \( K \) is used to control the error detection rate. The higher \( K \) is, the more aggressive the error detection, and vice versa. If \( K \) is set quite low, more erroneous tokens slip past the detector, and get tagged as correctly transcribed.

2.2. Integrating Error Detection within ROVER

ROVER works as follows: first, the output of the different recognizers are combined into a composite transition network through a dynamic programming alignment procedure. Then a voting schema is applied at each slot of the network, to select the best hypothesis and build a new transcription output. The problem with this process is that errors contained in each recognizer output are kept in the composite network, which may trick the voting algorithm and lead to errors propagating into the final composite output. We propose a solution, where we introduce a pre-filtering stage right after the different outputs are aligned, and then eliminate the errors to facilitate the voting. This way we are hoping for fewer mistakes in the final output. To do this, each word’s surrounding context at each slot in the WTN is used, to determine whether that word is a semantic outlier and therefore should be deleted. The cROVER procedure is presented in algorithm 2.

Algorithm 2 cROVER Procedure

1. Create the composite WTN by aligning the WTNs from the different recognizers
2. for all slots in the composite WTN do
   3. Apply the PMI-based error detection
   4. Remove the detected erroneous words from the slot, and replace them with the NULL transitions.
5. end for
6. Apply ROVER voting mechanism on the new composite WTN.

The first step consists of merging the different recognizer WTNs into a composite transition network. Then instead of applying the voting mechanism at each slot, a pre-filtering stage is introduced, and the composite network is scanned slot by slot. At each slot, the PMI-based error detection method is used to spot errors. If a token is flagged as a potential error, the algorithm updates the slot being processed by removing the arc of the erroneous word, and replacing it by a NULL transition network. This will simulate a deletion, and the ROVER voting schema will handle it accordingly. Once all slots are pre-processed, the voting algorithms are used to select the most appropriate token at each slot in the updated transition network.

3. SIMULATION RESULTS AND ANALYSIS

3.1. Experimental Framework

Two widely-known decoders have been used: Nuance ASR (v9.0), and CMU Sphinx-4. The HUB4 (LDC98T28) testing framework was used. HUB4 transcriptions have been used to train the language model for all the baseline systems except Sphinx-4, where another language model has been used. This language model is referred to as LM-98T28 hereafter. Without access to the HUB4 evaluation data, it was decided to exclude a subset of the training data for evaluation purposes: CNN headlines (30 min) of Feb 17, 1998 and CSPAN Washington Journal (96 min) of Feb 17, 1998. For Nuance v9.0, the English US acoustic model has been used, along with the LM-98T28. With Sphinx-4, two different language models have been used: the LM-98T28, and an open source model for broadcast news transcriptions from CMU, referred to as LM-BN999 hereafter. The already trained HUB4 acoustic model provided on the Sphinx download site has been used. Google Inc.’s one trillion-token corpus[4] was used to collect uni-gram and bi-gram frequency counts required by the PMI-based error detector: the 13.5 million uni-grams and the 314.8 million bi-grams counts were used. Errors found in each of the decoders’ transcriptions were automatically flagged, and correct words were selected at random, to obtain a testing
dataset to assess the performance of the error detection algorithm. In terms of metric measures, both the F-measure, and the positive and negative predictive values were reported. For a two-class problem (Error/Not Error), we can define the following quantities: precision, $\text{Prec} = \frac{TP}{TP + FP}$, and recall, $\text{Rec} = \frac{TP}{TP + FN}$, where TP, FP, TN, and FN represents true positives, false positives, true negatives, and false negatives respectively. The F-measure is the harmonic mean of both the precision and the recall, $\text{F-measure} = \frac{2 \times \text{Prec} \times \text{Rec}}{\text{Prec} + \text{Rec}}$.

The positive predictive value (PPV) is the precision $\text{Prec}$. The negative predictive value (NPV) is defined as $\text{NPV} = \frac{TN}{TN + FN}$.

To report the recognition performance, we used the WER metric. Let $N$ be the total number of tokens in a transcription. $D$ is defined as the number of deleted tokens from the recognition output, $I$ is the number of insertion and $S$ is the number of substitutions. The WER is then defined by $\text{WER} = \frac{D + S + I}{N}$.

### 3.2. Experimental Results

#### 3.2.1. Assessment of the Error Detection

The PMI-based error detection requires the optimization of the filtering parameter $K$, the aggregation method, and the size of the context window. Fig. 1 reports the F-measure in terms of the threshold $K$, and for few window sizes. All the aggregation methods’ plots have been included in each subplot. Note that these plots included a limited set of values for the different parameters, to make them reader-friendly. We notice that the best aggregation method depends on the window size. In fact, for a larger window size, the Maximum aggregation outperforms all the other methods, whereas for smaller window sizes, the Harmonic Mean guarantees better F-measure scores. Besides, for bigger windows, the Sum and the Harmonic Mean aggregation perform very similarly. It can be concluded that the PMI-based classifier performs better with larger window sizes. In fact, this observation is in concordance with expectation, because the PMI scores rely heavily on the surrounding context. Therefore the larger the context, the more reliable the classification can be: it is easier to flag semantic outliers when the surrounding context is large enough. Fig. 2 shows the behavior of the PPV and NPV rate as a function of $K$, for a window size of 20, with the Maximum aggregation. This plot is very important, since it shows the effect of detecting many errors on the correctly transcribed keywords. In fact, the higher the PPV is, the more degradation (mistakenly tagging correctly transcribed words as errors) is induced to the correct segments of the transcription. This undesired behavior is expected, and is a result of the limitations of the current error detection techniques in speech recognition, widely known throughout the literature[5]. In this work, it was decided to use a ratio of 25% and 90% for the cROVER experiments: that is, aiming to spot only 25% of the errors in the transcript while guaranteeing to preserve 90% of the correctly transcribed words.

#### 3.2.2. Assessment of cROVER

cROVER experiments were run, considering whether or not to exclude stop words from the processing. A 200- frequent stop words list was collected. During the error detection stage, if a word is a stop word, the PMI filter is not applied, and the next token in the WTN slot is processed instead. Further, in the case of more than two recognizers being combined, the PMI-based filtering was applied only in specific cases: if all the words at a specific slot in the WTN are different, and if only two of the recognizers agree on the same word at a WTN slot, the PMI filtering was only applied on the third one. This experiment has been carried out to minimize the effect of mistakenly tagging correct words as recognition errors. Hereafter, this experiment is referred to as cROVER*. In this paper, only the frequency-based voting mechanism of ROVER was used. This is due to the poor confidence measure of both Nuance v9.0 and Sphinx-4 decoders. Performance figures for all combinations of two and three recognizers at hand were carried out. Experiment 1 to 6 represent the
combination of the three decoders settings, V9-LM-98T28, Sphinx4-LM-98T28, and Sphinx4-LM-BN99. The remaining experiments, experiment 7 to experiment 12, refer to the different combinations of only two of the three decoders. Note that the combination order matters (an inherent problem from ROVER WTN-building stage). The obtained error rates prove that the cROVER is outperforming the baseline ROVER in all configurations. To highlight this improvement, Fig. 3 shows the WER reduction over the ROVER baseline when only two decoders were combined (experiments 7 to 12). WER reductions with stop words (filtering only applied on non stop words), and without Stop words (filtering applied on all the words) have been reported. The absolute WER reduction, achieved with the proposed cROVER, reached up to 1.55%. This is a very promising result, because the existing techniques have reached a plateau, and even an apparently-small improvement in percentage is important. Note that WER reduction is especially notable when the combined decoders are not very optimized. When decoders are not highly optimized, more improvement is gained when filtering is applied on all words. However, when both decoders’ WER is already too low, applying error filtering on non stop words gives much better enhancement. This is because when both recognizers are optimized, there is a little margin to work with in terms of error detection. Therefore, it is wiser to only focus on keywords, and discard stop words to limit the impact of false positives. This is clearly shown in Fig. 3, where it can be seen that discarding stop words guarantees lower WER (experiments 8 and 10), whereas, for experiments 7, 9, 11 and 12, applying the PMI-based error detection filtering on all words (without excluding stop words) gave higher WER reduction. Fig. 4 reports the reduction in terms of WER compared to the original ROVER baseline for the experiments on which all three decoders were combined. When the number of combined decoders increases, the initial cROVER starts losing its performance edge over the ROVER baseline for all six experiments, but doesn’t fall below it. However, when using the special cROVER*, things become similar to the enhancement observed in the two-decoder combination - which seems intuitive. That is, if two out of three recognizers agree on the same word, it is better not to apply pre-processing on the word to avoid losing accuracy to the false-positive impact of filtering. In this case, the filtering is only applied on the the word obtained from the third recognizer. This shows how this little trick (cROVER*), when added to the original cROVER, ensured lower WER.

4. CONCLUSION

In this work, a framework was proposed to improve on ROVER, by augmenting it with semantic analysis. A contextual analysis is applied to the composite WTN to remove the semantic outliers, and boost the performance of the voting schemes. The PMI-based error detection technique has been used in the pre-filtering stage within the ROVER procedure. Experimental results have proved that cROVER, outperforms the ROVER baseline for all configurations, reaching more than 1.5% in some cases, in terms of absolute WER reduction. Future work will study the computational complexity and the scalability of the augmented ROVER with error detection techniques.

5. REFERENCES


