*c*ROVER: Context-augmented Speech Recognizer based on Multi-Decoders' Output

by

Mohamed Kacem Abida

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Electrical and Computer Engineering

Waterloo, Ontario, Canada, 2011

Abstract

The growing need for designing and implementing reliable voice-based human-machine interfaces has inspired intensive research work in the field of voice-enabled systems, and greater robustness and reliability are being sought for those systems. Speech recognition has become ubiquitous. Automated call centers, smart phones, dictation and transcription software are among the many systems currently being designed and involving speech recognition. The need for highly accurate and optimized recognizers has never been more crucial. The research community is very actively involved in developing powerful techniques to combine the existing feature extraction methods for a better and more reliable information capture from the analog signal, as well as enhancing the language and acoustic modeling procedures to better adapt for unseen or distorted speech signal patterns. Most researchers agree that one of the most promising approaches for the problem of reducing the Word Error Rate (WER) in large vocabulary speech transcription, is to combine two or more speech recognizers and then generate a new output, in the expectation that it provides a lower error rate. The research work proposed here aims at enhancing and boosting even further the performance of the well-known Recognizer Output Voting Error Reduction (ROVER) combination technique. This is done through its integration with an error filtering approach. The proposed system is referred to as *c*ROVER, for context-augmented ROVER. The principal 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 within ROVER. The error detection technique consists of spotting semantic outliers in a given decoder's transcription output. Due to the fact that most error detection techniques suffer from a high false positive rate, we propose to combine the error filtering techniques to compensate for the poor performance of each of the individual error classifiers. Experimental results, have shown that the proposed *c*ROVER approach is able to reduce the relative WER by almost 10% through adequate combination of speech decoders. The approaches proposed here are generic enough to be used by any number of speech decoders and with any type of error filtering technique. A novel voting mechanism has also been proposed. The new confidence-based voting scheme has been inspired from the *c*ROVER approach. The main idea consists of using the confidence scores collected from the contextual analysis, during the scoring of each word in the transition network. The new voting scheme outperformed ROVER's original voting, by up to 16% in terms of relative WER reduction.

Acknowledgements

The road was longer and harder than it promised, but in the end I persevered. For those who have helped me along the way, know you have a place in my heart forever warmed by my gratitude.
So here I say thank you to...

- My supervisor, Professor Fakhreddine Karray, for his guidance, commitment and support throughout my research. I have learned a lot from his feedbacks on my research work as well as his dedication to research.
- The Pattern Analysis and Machine Intelligence (PAMI) research group for the fruitful discussions, as well as the Electrical and Computer Engineering Department (ECE) at the University of Waterloo for ensuring the best working conditions during the past several years.
- Voice Enabling Systems TEChnology Inc. (VESTEC) for providing valuable resources used in this thesis and for its courtesy to use its labs to carry out some of the reported experiments.
- Dr. Jiping Sun, the speech scientist, for his advice, help and kindness.
- My committee members, namely Dr. Kumaraswamy Ponnambalam, Dr. Oleg Michailovich, and Dr. William Bishop, as well as my external examiner Dr. Reda Alhajj for taking the time to assess my research work, as well as for their pertinent advice and feedback.
- My uncle's family for giving me confidence and encouragements.

To my parents and my wife

Contents

List of Figures

List of Tables

Glossary

AM	Acoustic Modeling or Model
ASR	Automatic Speech Recognition or Recognizer
CMU	Carnegie Mellon University
CN	Confusion Netowrk
CNC	Confusion Network-based Combination
cROVER	Context-augmented Recognizer Output Voting Error Reduction
EPPS	European Parliament Plenary Sessions
FN	False Negative
FP	False Positive
fWER	minimum Time Frame Error-based speech decoding
HM	Harmonic Mean
HMM	Hidden Markov Model
LM	Language Modeling or Model
LSI	Latent Semantic Indexing
LVCSR	Large Vocabulary Continuous Speech Recognition
MR	Mean Rank of the Semantic Scores
MSS	Mean Semantic Scoring
NIST	National Institute of Standards and Technology

PMI	Point-wise Mutual Information
ROVER	Recognizer Output Voting Error Reduction
SCTK	Speech Recognition Scoring Toolkit
TN	True Negative
TP	True Positive
WA	Weighted Average
WER	Word Error Rate
WTN	Minimal Cost Word Transition Network

Chapter 1

Introduction

1.1 Background and Motivation

During the past two decades, the automatic speech recognition technology has evolved to the point where a large number of speech-enabled commercial applications are now widely deployed and are becoming increasingly robust and accurate[2]. Speech recognition has become ubiquitous: applications ranging from systems for name-dialing[3, 4, 5], travel reservations[6, 7], automated directory assistance[8], dictation[9, 10], smart phones[11, 12, 13], Global Positioning System navigation, in-car infotainment systems[14, 15], meetings, podcasts, and broadcast news transcription[16, 17, 18], to name a few. Novel speech recognition applications are being developed at an increasing pace, especially motivated by the recent boom in mobile personal devices. The fact that these systems are working very satisfactorily for millions of people on a daily basis, is a testimony to the technological advances made in the field of automatic speech recognition to date.

However, even though the current technology seems to have matured, the problem of Large Vocabulary Continuous Speech Recognition (LVCSR) is far from being completely resolved. LVCSR relates to all aspects of speech recognition in the domain of spontaneous, human-human conversational speech (as opposed to planned, read, or human-machine dialog). In the field of LVCSR, the voice-enabled systems can fail for several reasons: channel distortion caused by background noise, or corruption of the transmission channels, accented speech, variation of the speaking style, speech overlapping, poor signal quality, ill-structured sentences, spontaneous and casual speech, etc. The need for highly accurate and optimized speech decoders has never been more crucial. The research community is actively involved in developing powerful techniques and solutions to resolve the high error rate in LVCSR. These solutions include the combination of the existing feature extraction techniques towards a more reliable information capture from the analog signal[19, 20, 21], hierarchical language modeling through the use of several dedicated and domain-specific grammars[22, 23], and acoustic models adaptation[24], among others. Currently most researchers agree that one of the most promising directions towards reducing the Word Error Rate (WER) in LVCSR applications is through a combination of decoder outputs. The idea is to compile two or more speech recognizers' outputs into a composite new transcript, in the hope that this yields a lower error rate. One of the earliest successful attempts dates back to 1997, when Jonathan Fiscus, from the National Institute of Standards and Technology (NIST), proposed the NIST Recognizer Output Voting Error Reduction (ROVER) system[25]. Because of its simplicity and its outstanding performance, ROVER is considered in the literature as the baseline technique in decoder combination. All the research work that followed in this field, compared every new proposed technique to ROVER's performance. Quite a bit of work and enhancement has been done around the original ROVER system. However, after a while, it appears that ROVER performance has reached a plateau and it has become quite difficult to achieve a substantial WER reduction. Our main goal in this research work is to produce an even more powerful approach with the aim of enhancing and improving the original ROVER performance by producing the lowest possible WER.

1.2 Objectives

The following objectives have been targeted for the research reported in this thesis:

Objective 1 A novel framework to improve on the performance of the original ROVER system. The framework is generic in terms of recognizers, and is application domain independent.

Objective 2 Scalable framework in terms of the number of decoders to be combined together.

Objective 3 Implementation and assessment of the proposed framework against existing approaches.

Objective 4 A novel voting scheme for the ROVER procedure

1.3 Contributions

The thesis makes several contributions while meeting the objectives stated in Section 1.2.

- This thesis proposes a novel approach to improve the ROVER procedure. The ROVER procedure consists of two main processes. First, the output from different speech decoders are combined together into a network of tokens. Second, a voting algorithm browses the composite network to select the winner token at each location. Our proposed approach is to implement a contextual analysis procedure right after the first process of building the composite network. This analysis aims at filtering transcription errors from the different speech decoders in order to facilitate the voting stage. The proposed approach is called *c*ROVER, where *c* stands for context.

- The proposed approach is generic and does not make use of any internal information about the speech decoders. Similarly to ROVER, the newly designed combination procedure is independent of the speech domain and application.

- The *c*ROVER approach is as scalable as the original ROVER procedure. In fact, the proposed approach did not alter the inner mechanism of ROVER, allowing it to keep the same characteristics.

- Three aggregation schemes were proposed to combine several automatic error detection techniques.

- A novel scoring scheme has been proposed. The confidence-based voting mechanism relies on confidence scores collected during the contextual analysis. The new scoring technique is a weighted combination of the original ROVER's score and the confidences from the contextual analysis of the transcription output.

1.4 Organization

The rest of the thesis is organized as follows:

Chapter 2 provides some background material on automatic speech recognition and a review of literature on the speech decoders' combination in large vocabulary transcription applications. The chapter also provides a review of the automatic error detection techniques.

Chapter 3 describes our proposed approach to improve the original ROVER system. The chapter starts by detailing the rationale behind the proposed approach. It then describes the proposed context-augmented ROVER, cROVER. A case study illustrating the approach is then provided. The chapter concludes with a description of a novel voting scheme for ROVER.

Chapter 4 presents a set of experiments to validate the performance of the proposed *c*ROVER approach. First the experimental framework is presented, followed by the assessment of the error filtering techniques used in the case study presented in the previous chapter. The approach is then validated on two test sets. An analysis of the computational requirements of the proposed approach is then presented. The chapter concludes with an assessment of the novel voting scheme for ROVER.

Chapter 5 concludes this thesis with a summary of the work presented along with the future direction of research.

Appendices A, B, C, and D provide the numerical values of the error rates in terms of the relative and absolute reduction, for the *c*ROVER approach and the new voting scheme.

1.5 Conclusion

This chapter has discussed the motivation and specified the objectives of this thesis. It has highlighted the impact of the recent advances in the automatic speech recognition, in terms of the successful deployment of commercially robust and reliable speech-enabled applications. The chapter also identified a set of research issues involved in LVCSR. The objectives as well as the contributions made in this research work were summarized.

The next chapter provides a survey of the literature pertinent to this work. It specifically highlights related research work on speech decoder combination, and automatic error detection in LVCSR systems.

Chapter 2

Literature Review

In this chapter, we start with a brief description of automatic speech recognition technology, followed by the current performance of LVCSR in different applications, namely read, broadcast, meeting, and conversational speech. The latest advances in decoders' combination and automatic error detection are detailed in the remainder of the chapter.

2.1 Automatic Speech Recognition Fundamentals

In this section, a broad description of automatic speech recognition major components is presented.

2.1.1 Speech Recognition: How Does It Work?

In very general terms, speech recognition produces the text of an uttered sentence from the continuous acoustic signal of a speaker. In a nutshell, the typical process of speech recognition can be divided into several stages, as shown in Figure 2.1. The front end of the speech recognizer is a module that transforms the speech waveform into a sequence of discrete observations, called feature vectors. These acoustic features are obtained in such a way as to preserve all the relevant information from the original signal. The next step is the transformation of these acoustic features into a time-sequenced lattice of phones. An important

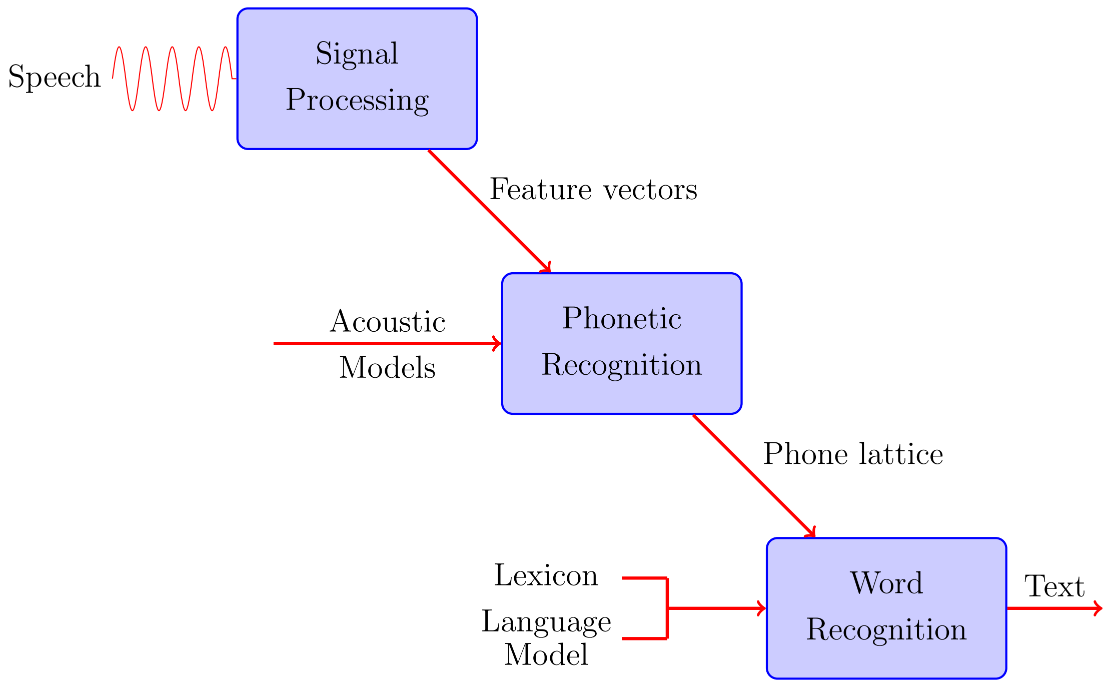

Figure 2.1: Bottom Up Approach For Speech Recognition

knowledge source is needed at this stage, namely the acoustic model. This is a statistical representation of the sounds which make up each word. The speech recognizer relies on this model to be able to transform the feature vectors into phonemes. Now that we have a lattice of phones in hand, the process of reconstructing the uttered words begins. As in the previous step, some additional sources of knowledge are needed. The lexicon is a set of words transcribed into their phonetic forms. The language model, or grammar in the case of isolated word recognition, is a key component. A grammar is a set of word patterns used to guide the recognizer as to what to expect from the speaker. By using the lexicon and the language model, the word recognition module walks through the phone lattice and constructs words. A more detailed description of both acoustic modeling and language modeling follows.

2.1.2 Acoustic Modeling

Acoustic Modeling (AM) of speech typically refers to the process of establishing statistical representations for the feature vector sequences computed from the speech waveform. Several techniques are commonly used for this. The Hidden Markov Model (HMM)[26] is one of the most common types of acoustic model. Other acoustic models include segmental models, suprasegmental models (including hidden dynamic models), neural networks, maximum entropy models, (hidden) conditional random fields, etc.

Acoustic modeling also encompasses "pronunciation modeling", which describes how a sequence or multi-sequences of fundamental speech units (such as phones or phonetic features) are used to represent larger speech units such as words or phrases, which are the object of speech recognition. Acoustic modeling may also include the use of feedback information from the recognizer to reshape the feature vectors of speech in order to achieve robust performance in noisy speech recognition problems[27].

Acoustic modeling is arguably the central part of any speech recognition system. For any given acoustic observation $X = X_1X_2 \dots X_n$, the goal of speech recognition is to find out the corresponding word sequence $\hat{W} = w_1w_2 \dots w_n$ that has the maximum posterior probability $P(W|X)$ as expressed by Equation 2.1:

$$\hat{W} = \arg_w \max P(W|X) = \arg_w \max \frac{P(W)P(X|W)}{P(X)} \tag{2.1}$$

Since the maximization of Equation 2.1 is carried out with the observation X fixed, the above maximization is equivalent to maximization of the following equation:

$$\hat{W} = \arg_w \max P(W)P(X|W) \tag{2.2}$$

The practical challenge is how to build an accurate acoustic model, $P(X|W)$, which can truly reflect the spoken language to be recognized.

2.1.3 Language Modeling

A language Model (LM) gives the probabilities of sequences of words. Language models are often used for dictation applications. A special type of language model is a regular grammar, which is used typically in desktop command and control or telephony IVR-type applications. The language model provides a description of the language. It is also a way to compute the $P(W)$ in Equation 2.1. Assume $W = w_1 \dots w_k$. Then $P(W)$ can be computed by Equation 2.3:

$$P(W) = P(w_1 \dots w_k) = P(w_1)P(w_2|w_1)\dots P(w_k|w_1 \dots w_{k-1}) \tag{2.3}$$

Estimating $P(w_l|w_1 \dots w_{l-1})$ for all the possible words and sentences in a given language is practically impossible. One approach to reducing this difficulty would be to approximate $P(w_l|w_1 \dots w_{l-1})$ by $P(w_l|w_{l-K+1} \dots w_{l-1})$ for a fixed value of K. This means that we only care about the K previous tokens. This type of language model is commonly called a K-gram language model, where K is generally either 2, 3 or 4. These probabilities are estimated during the training of the language model, which requires a large set of text data.

2.2 Current LVCSR Performance

The need for designing and developing highly robust, accurate, and efficient speech decoders is critical to the widespread adoption of a large number of commercial applications. These applications include, among others, automated call centers, broadcast news transcription, voice-activated car accessories, large-vocabulary voice-activated cell phone dialing, and navigation. The current advances in LVCSR have ensured a big deployment of these commercial applications. The fact that these services are being used on a daily basis by millions of people throughout the world is a testimony that the current state of the art in speech recognition has reached a high degree of maturity.

Figure 2.2 reports the NIST Speech To Text (STT) benchmark test history for all NIST standard test frameworks in LVCSR, including read, conversational, and meeting speech[28]. The benchmark test history begins from the early nineties up to 2009. WER is reported for each standard test framework through the years, illustrating the advances in the field and therefore the impact on the WER reduction. Impressive achievements can be seen in the read speech type of applications, where the current WER is within the same range as human error in transcription. However, for conversational, broadcast, and meeting speech, the error rates are still far from the human range. This can be explained by the fact that several

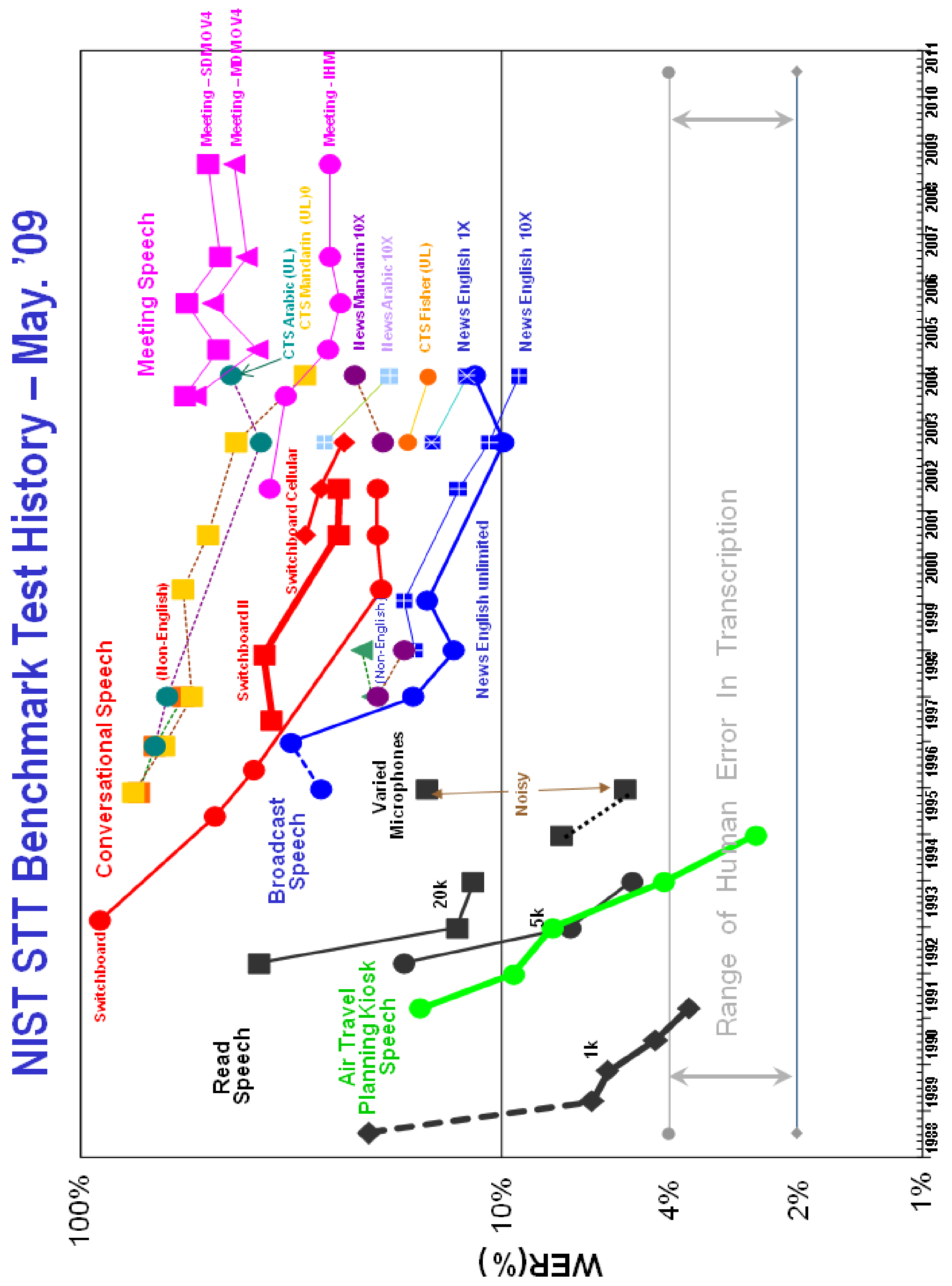

Figure 2.2: NIST STT Benchmark Test History[1]

problems in LVCSR are still not resolved. These problems include channel distortion caused by background noise, or corruption of the transmission channels, large variety of accents, variations in speaking style, speech overlapping, poor signal quality, badly structured sentences, spontaneous speech, etc[29]. Researchers are actively working on tackling these issues at different levels, namely the front end signal processing, the acoustic and language modeling, and the search mechanism. Researchers are also actively investigating several post-decoding strategies to improve and tune the final systems' output. These strategies include system combinations, as well as automatic error detection and correction. Since the scope of our work lies in this field, a detailed review of these post-decoding techniques is presented in Sections 2.3 and 2.4, where a review of the recent advances in systems' combination and automatic error detection is presented.

2.3 Decoders' Combination for LVCSR

In recent years, it has become common practice to reduce WER by combining the outputs from several recognition sites[30]. This section is meant to be a survey of the latest techniques in the field of system combination toward improving recognition performance in LVCSR, where the most significant work and development witnessed in the last two decades is presented.

2.3.1 The ROVER System

Recognizer Output Voting Error Reduction, also known as ROVER [25], is a system developed at NIST in 1997 to produce a composite of decoders' output when the outputs of multiple ASR systems are available. The goal of the combination is to produce a lower error rate in the final composite output. This is done through a voting mechanism to select the winner word from among the different decoders' output. The voting schemes relied on the frequency of occurrence as well as on the decoders' confidence scores. ROVER is considered in the literature to be the baseline technique in system combination. All the research work that follows it has evaluated the new proposed technique and compared it to the performance of ROVER.

The proposed ROVER technique in [25] has been evaluated on the LVCSR HUB-5E of 1997 testing paradigm[31]. Up to 5 ASR outputs have been combined, and the three voting schemes yielded similar WER reductions (the slight difference between the voting algorithms was judged to be insignificant) for the chosen evaluation corpus.

ROVER's Shortcomings

Even though the improvement in terms of WER, obtained by using the ROVER technique can be considered outstanding throughout the literature, a few problems still need to be investigated. In fact, the ROVER scoring approach can only succeed if and only if the errors produced by each ASR system are different from one system to the other. Otherwise, the combination will yield the same WER since there is no point in voting between erroneous outputs at each location. The ASR systems to be combined must output different errors, at each position, from each other. Therefore, a careful selection of the different systems to combine is critical to ensure a rich selection for the voting algorithms at each slot in the Word Transition Network (WTN).

It is also worth mentioning that the iterative combination of word transition networks does not guarantee the optimal composite output. In fact, the order of combination is important and affects the end result. It is thus worth looking for techniques to optimize the building of the composite word transition network. In [32], it has been reported that the best results are obtained when systems are ordered by increasing WER. This observation is empirical though, and has been observed after exhaustive experimentation.

The fact that ROVER relies on word level confidence values in the voting process, makes it a vulnerable technique. In fact, it is not safe to assume that the word level confidences are reliable. Much research is still underway into trying to come up with a robust and effective technique to provide a decent confidence measure for LVCSR. Even in [25], neither algorithm that relies on a confidence score achieved a significant error reduction compared to the frequency-based voting algorithm.

Another limitation of the original ROVER is the usage of only 1-Best word sequence. Even if each participating ASR system is providing N-Best output,

ROVER is unable to make use of this information due to the inherent nature of its combination process.
Finally, ROVER is unable to outvote the erroneous ASR systems when only one single ASR is providing the correct output. The proposed scoring schemes make it difficult to boost a single ASR output since both occurrence and confidence are used to score each word at a specific location.

2.3.2 The After-ROVER Era

Most HMM-based speech recognition systems use the sentence level Maximum A-Posteriori (MAP) criterion to select the best word sequence. However, this criterion is only optimal if we want to minimize the Sentence Error Rate (SER), whereas the actual goal in speech recognition development is usually to reduce the word error rate. The general framework [33] for a minimum WER decoder can be provided through Bayes decision rule with a *Levenshtein* cost function $\mathcal{L}$ as shown by Equation 2.4.

$$\{w_1^N\}_{opt} = arg \min_{w_1^N} \left\{ \sum_{v_1^M} \mathcal{L}(w_1^N, v_1^M) p(v_1^M | x_1^T) \right\} \tag{2.4}$$

with a word sequence w_1^N and the posterior probability $p(v_1^M|x_1^T)$ for word sequence v_1^M given the acoustic observation x_1^T. The *Levenshtein* distance measures the amount of difference between the two sequences of strings. Judging from Equation 2.4, it is basically impossible to apply this framework for large vocabulary continuous speech recognition systems due to the large search space. The Confusion Network (CN) and minimum Time Frame Error (fWER) decoder are two approaches using different approximations in order to achieve the minimization of word error rate on word lattices[33, 34]. Both approaches have achieved relative improvements of up to 5% in terms of WER[32].
CN and fWER constitute the roots of two of the (after-ROVER) most common approaches to system combination, namely Confusion Network Combination (CNC) and fWER-based combination. Most of the research work in the area of system combination focused on either improving ROVER, or providing different variants of either CNC or fWER-based combination techniques. What follows is a review of the most important work in this direction.

2.3.3 Confusion Network Combination

A confusion network[34] is a very compact representation of the most likely word hypotheses in the lattice generated by the Viterbi algorithm. It is a directed graph where all the outgoing arcs of a given node have the same target node. The process to convert a word lattice into a confusion network is two-staged. First, all the links that correspond to the same word and overlap in time are combined and the word graph is updated accordingly. Second, the remaining links corresponding to different words are grouped in confusion sets. The order of grouping is ruled by the phonetic similarity, the time overlap and the posteriors of the words[35]. For each word, the posterior is the weighted average of the individual slot-wise posteriors. This procedure is repeated till the lattice structure becomes linear, and thus the confusion network is obtained. In CN decoding, the words with the highest posteriors are selected as winners.

The main idea of confusion network combination is to align multiple confusion networks which have been built from the individual lattices. This combination results in a new confusion network with updated word posteriors at each slot. The fact that CNC is converting the word lattices into a linear confusion network makes it suitable to take into account alternative hypotheses (NBest), which represented one of the main limitations in the original ROVER. Other advantages of using CNC include the availability of word-posterior probabilities, and the inherent lower WER because of the usage of CN - which as we have described earlier, is a technique to optimize WER rather than SER. Besides, at each slot in the confusion network, there are many more hypotheses than with ROVER. Combination will then lead to a much lower oracle WER[36].

Even though the CNC seems to be addressing some of the limitations of the original ROVER, the experimental results that we have found in the literature were rather disappointing. In [35], experimental results were presented based on the CU-HTK conversational telephone speech evaluation system. The improvement over the single best compared to ROVER is very small, even negligible. In [33], the experimental results were conducted on the European Parliament Plenary Sessions (EPPS) of 2005 for both English and Spanish. However, the improvement of system combination using CNC is not substantial compared to ROVER-

and fWER-based system combination. In [32], extensive experiments have been carried out to compare CNC, ROVER, and fWER, using the EPPS 2006 English corpus. It has been found that when more than two participating sites are involved in the combination, ROVER tends to achieve better performance. However, when only two outputs are combined, CNC achieves a lower WER compared to the original ROVER. But as in earlier research work, the improvement is little. The authors in [32] highlighted a very important feature of ROVER, which is the ability to provide robust combination with more than two ASR system outputs. Based on these observations, it appears that it is quite difficult to achieve significant improvement using confusion network combination. We think that this is due to the fact that CN decoding is already optimizing WER, therefore a combination based on this type of decoding is unlikely to produce a substantial boost, as each of the ASR systems is already providing near optimal output.

2.3.4 fWER-based Combination

Minimum frame WER decoding is another approximation of Equation 2.4 to achieve the minimization of word error rate in speech decoding. Confusion networks achieve this goal by changing the structure of the initial word lattice and collapse it into a linear graph where the search space is reduced and computation time is limited. However, the fWER decoding approach aims at changing the computationally expensive *Levenshtein* distance $\mathcal{L}$ in Equation 2.4 by another cheaper cost based on the covered time frames[37]. The authors in [37] have proved that the fWER decoding leads to a minimization of WER.
One of the main advantages of this new cost function is that there is no need for word sequences alignment, which makes it computationally cheap. The fWER decoding approach preserves the lattice structure and the word boundaries, compared to CN, which collapses the lattice structure to a linear network. This saves a post-processing stage to produce the word time boundaries.
fWER decoding can be easily extended from a single word lattice to multiple word lattices output from several ASR systems. The idea to use fWER decoding to perform system combination is similar to the one that used CN for CNC. The fWER based combination was first proposed in [33]. Experiments have been

carried out on the EPPS 2005 Spanish and English corpora. Even though the proposed approach seemed to be well formulated, it did not provide substantial improvements compared to both ROVER and CNC. The authors claimed that this might be due to the fact that all the combined systems were internal, and thus not different enough from each other. In other words, most of the recognition errors were common among all the combined systems, and therefore it was difficult to obtain a significant reduction in WER after combination using ROVER, or CNC, or a fWER-based technique.
In [32], a thorough comparison of the three combination techniques has been attempted as described earlier. A fWER based technique didn't achieve reliably superior performance compared to both ROVER and CNC. Experiments showed the same level of performance as CNC. Both CNC and fWER failed to outperform ROVER when more than two Automatic Speech Recognition (ASR) systems outputs were combined.

2.3.5 Soft Computing Towards Enhancing ROVER and CNC

Since 2006 a new trend in research has emerged, aimed at enhancing the existing system combination techniques using tools of supervised learning. In this section, we report briefly on some of these efforts. The first attempt led by Zhang in [38] proposed an enhancement to the original simplistic voting mechanism in ROVER, using neural networks-based classifiers. The proposed scoring algorithm is a two-stage process. First, a trained neural network attempts to determine whether the current node in the word transition network is an insertion error. When the current node is classified as a non-insertion, each word is scored based on a variety of features that are extracted from multiple information sources. The word that gets the highest score is chosen as the decoding result. To train the neural network, the authors in [38] have used several features, including the average frequency of occurrence of real words and filler model, the language model back-off mode, the utterance level posterior, etc. This proposed technique has been evaluated on the continuous speech recognition task in a meeting environment using the ICSI Bro series of meetings. It has been shown that this enhanced ROVER scoring

mechanism boosts the performance by a reduction in WER of 2.18%.
In [39], the authors proposed a similar approach to boost ROVER-based combination. They called the new system *i*ROVER, for improved ROVER. The authors used the Boostexter classifier trained on a set of features from the ASR system lattices, in order to select the ASR system most likely to be correct at each location in the word transition network. Six features were used to train the classifier to pick up the correct word at each location. These features include character length, frame duration, frames per character, top error words in the development set, character distance between systems, etc. This team of researchers evaluated *i*ROVER on the EPPS 2006 English corpus. The improvement compared to the original ROVER on this corpus was the largest known by 2007.
Exhaustive experiments were carried out in [36] to prove that using supervised learning tools can indeed improve ROVER and CNC-based system combination. The idea is to train a classification technique at each location, to decide which of the provided alternatives is most likely correct. Three different classifiers were used, namely Boostexter, random forests, and maximum entropy models. Several feature sets were attempted as well. For each word hypothesis, a set of features are computed including acoustic and language model scores, word duration, and whether the word is in a list of the ten, twenty or one hundred words causing the most errors. For the CNC, the features used to train the classifiers included CN confidence, CN slot entropy and confidences based on frame wise posterior probabilities across all systems. Experiments were carried out on the EPPS 2007 English corpus. The results of combination of up to four lattice sets were presented. It was shown that improvement over highly optimized ROVER and CNC baselines is rather small, and the authors raised an important question, which is whether or not the limit of improving combination with classification has been already reached.
Based on the previous review of some of the efforts towards reducing WER by systems combination, we can conclude that quite a bit of research work still can be done. We have noticed that it is more and more difficult to outperform the improved ROVER and the highly optimized confusion network and fWER-based combination techniques. This situation led the authors in [36] to think that current approaches have reached a plateau and are unable to achieve any substantial

improvement. However, we don't think that this is totally true. In fact, only a limited number of classification techniques has been investigated. There are several other tools that can be used to better address the problem of selecting the best system at each location. Among them, we can cite support vector machines, neural networks, decision trees, etc.
Further, most of the experiments reported in the literature, have been conducted by the same research team ([32, 33, 36, 39]) and with the very same testing corpus, namely the EPPS English corpus. Most of the proposed techniques have not been tested with many different corpora. Ideally, these approaches are to be tested on a standard NIST Hub testing paradigm, so that other researchers are able to test with the same data and compare their reported results. We definitely think it would be really beneficial to replicate all the experiments already presented on different corpora and to check whether all the claims still hold.
Another crucial item, is the importance of combining sites that are different enough. Reporting combination results using internal systems is not relevant. As we have already stated, the technique of combining ASR systems to improve recognition relies on the fact that the combined systems have different errors from each other. This will ensure that the oracle error will be very low which allows the different combination approaches to work on a bigger margin of WER. This is one of the biggest constraints of this research direction. Therefore, the choice of ASR systems to be combined and reported in experiments is to be carefully studied. Otherwise, wrong conclusions will be drawn and research leads will be wasted.
Speaking of the latest work involving the use of tools of soft computing to boost ROVER, CNC, and fWER-based combination, several features have been investigated to train the different classifiers. Among these features, we can cite confidence information, frame duration, character length, word length, and acoustic and language models scores. We think that there are plenty of features that could be used, especially in the case of CNC and fWER where we have most of the lattice information available. Besides, there has been intensive research work towards combining features to boost accuracy in speech recognition.
In the same spirit of using new sources of information to help select reliable features for the training of the soft computing tools, we would like to mention a

very promising direction: that is using the context and semantics in the process of combination. Researchers have been persistent over several decades in trying to involve semantics to optimize WER in LVCSR. Several approaches and techniques have been developed and promising performances were achieved. These attempts ranged from using the language model statistics to computing semantic similarities and using phonetic similarities, to accurately detect and sometimes fix the recognition errors in a given transcription.
The next section reviews the latest advances in automatic error detection in LVCSR.

2.4 Automatic Error Detection in LVCSR

The advancements in the signal processing field, along with the availability of powerful computing devices, have led to the achievement of decent performance in the speech transcription field. However this performance could only be obtained under restrictive conditions, such as broadcast speech data, noise-free environments, etc. Making speech recognition effective under all conditions is the ultimate goal, which can be achieved if we are able to minimize or even eliminate and/or correct most of the recognition errors. In general, there are five recognition errors that can arise in a decoder's transcription output:

- Insertion errors: a new word, not part of the reference utterance, is inserted in the transcription output.
- Deletion errors: a word in the reference utterance is missing in the transcription output.
- Merging errors: two or more words are erroneously merged together in a new single word (wreck a nice $\rightarrow$ recognize)
- Splitting errors: a word in the original utterance is split into two or more words in the transcription output (baby $\rightarrow$ bay be)
- Substitution errors: a word is replaced by another (for $\rightarrow$ four)

In large vocabulary speech transcription, these recognition errors can be caused by a variety of factors. The most common of these causes are:

- Out Of Vocabulary (OOV) terms are words that are missing from the language model and therefore are falsely mapped to (relatively close in pronunciations) vocabulary words.
- The Viterbi decoding procedure includes some heuristics to trade off between speed and accuracy. Therefore, potential correct hypotheses can be pruned at early stages and be eliminated from the search space, leading to the propagation of an erroneous hypothesis to the final output.
- Grammar and language model building procedures are not perfect. This leads to misguiding the Viterbi search while processing an acoustic utterance. This is besides the fact that it's impossible to build a generic language model that can model all the typical usage of the language for various applications.
- The acoustic model is meant to map the acoustic features of the speech signal to phonemes. However, deficiencies in the training stage and missing pronunciations can hinder this mapping, causing false mappings in the final recognizer's output.
- Noisy environments can deteriorate the original speech signal, leading to wrong mappings of the signal to a phoneme, or even the loss of the whole noisy portion of the signal.

Since there is no foreseeable solution to counteract all of these factors, researchers investigated ways to automatically detect possible recognition errors and eliminate them or even attempt to correct them. There exists two main approaches to error detection in large vocabulary continuous speech recognition: methods that are recognizer-independent, and others that are recognizer-dependent. Recognizer-dependent techniques tend to rely heavily on the internal decoder's information, thus making them tied to the recognizer's implementation[40, 41, 42]. In this research work, we are only interested in recognizer-independent (generic) approaches for error detection. In this type of approach, the speech decoder is a black box. Details on recognizer-dependent error detection approaches can be found in [43]. Throughout the literature, we could distinguish two main directions under the recognizer-independent approaches: probabilistic and non-probabilistic techniques.

2.4.1 Non-Probabilistic Approaches

The most common non-probabilistic technique is pattern matching. The idea is to identify and collect error patterns that occur in the speech transcription. A database of common error patterns is created from transcripts relevant to the application domain. Rules are then used to compare the decoder's output to the patterns stored in the database. This technique suffers from many issues. First, pattern matching is unable to cope with unseen patterns. Second, for a broad domain, it's quite difficult to collect all possible error patterns due to the language variation[44]. Finally, pattern matching is susceptible to false positives in cases where correct words occur in a known error context[41]. In [44], a database of common errors and their correction in Japanese has been created. Whenever an error is spotted in the transcription output, it was replaced by the corresponding correction in the database.
Concepts comparison is another non-probabilistic approach used to identify outliers in a given utterance. A similarity score is derived between concepts to measure the degree of relatedness between them. Conceptual similarity can generally be determined, given a hierarchical knowledge base, using either edge- or node-based similarities. These techniques are usually used in the fields of data mining, vocabulary development, and decision support, but they have been also applied to the field of automatic error detection in the decoders' output in order to spot semantic outliers. [45] exploited the hierarchical structure of the Unified Medical Language System, and the fact that similar concepts are closer to each other, in order to derive an edge-based metric. The authors in [46] used weighted projections of two concepts to compute the semantic distance through a vector distance measurement. Edge-based approaches assume that the links between concepts represent uniform and symmetric distances, but that is not always true[47]. Researchers tend then to include a weighting factor to reflect

the information content of a node (the concept). The authors in [47] proposed an approach to compute this informational content of a given concept, using information theory. Further details can be found in [43].

2.4.2 Probabilistic Approaches

According to [48], speech recognition errors tend to occur in regular patterns rather than at random. In fact, if we are given a large enough training data from a certain domain, the collected frequencies can be very well extended beyond the training corpus to cover the whole domain. These techniques are also called corpus-based techniques, because they rely on large textual corpora in order to collect frequencies and mine for knowledge to spot outliers in a given transcription output. The most commonly used techniques in this regard are Co-occurrence relations, Latent Semantic Indexing (LSI)[49] and Point-wise Mutual Information[50] (PMI)-based error detection. The co-occurrence relations are used to determine the frequency of a word in a given context[26]. These word and context statistics are then used in order to determine how likely a given word is to occur in a specific context. The authors in [51] used this approach to spot errors in a given query. LSI is an information retrieval tool that relies on the terms co-occurrences to identify the degree of similarity between them. It represents the terms and documents in a reduced dimensionality, without losing much of the relationship information between them. Each term is represented with a limited set of features, that are used afterwards to compute distances between terms. LSI was first applied to the field of error spotting in [49] by exploiting this semantic analysis. Experiments showed that it's possible to achieve high recalls, but with a low precision rate. PMI computes similarity scores between terms using word frequencies from a given corpus. Those similarity scores are used to identify a semantic outlier in a given context of terms. PMI was first applied to the field of automatic error detection in [50]. Most of the error detectors have achieved high precision and low recall rates. This is a common issue in the field of automatic error detection in speech transcriptions. Only a few attempts already have been successful in combining error detection techniques to improve both precision and recall ratios. The authors in [52, 53] relied on a direct aggregation scheme, in which only one single heuristic is needed to tag a given word as an erroneous output.

2.5 Conclusion

This chapter has highlighted the basics of automatic speech recognition technology, specifically the front end processing module, as well as the acoustic and language modeling components. Then a brief survey of research on systems' combination has been provided. Advances and recent work on the ROVER procedure, as well as the confusion network and fWER-based combination were presented. The chapter has reviewed the latest work in automatic error detection in LVCSR transcriptions, especially approaches that are speech decoder-independent. The following chapter describes our proposed approach to improve the ROVER combination procedure, using techniques from the automatic error detection field.

Chapter 3

Proposed Approach: *c*ROVER, the Context-Augmented ROVER

In this chapter, we start by presenting the original ROVER system. The motivations behind our approach to improve on the current ROVER are then presented. The new approach is then described in details, followed by a case study using two widely-known error detection techniques. The chapter concludes with a novel voting mechanism for the original ROVER. The new voting mechanism is an alternative to the current ROVER's voting schemes, and has been inspired by our proposed approach to augment ROVER with automatic error filtering.

3.1 The ROVER Procedure

ROVER is a two-step process, as shown in Figure 3.1. First, it combines the multiple outputs into a single, minimal cost word transition network, WTN, through dynamic programming. Once this alignment is done, the resulting network is browsed by a voting process which selects the best output sequence (with the highest votes). Since implementing an algorithm to optimally align more than two WTNs is difficult, an approximate solution has been proposed using the traditional two-dimensional dynamic programming alignment procedure. The approximation works by iteratively merging the composite WTN with the next linear word hypothesis, till all decoders' outputs have been merged in the composite final WTN.

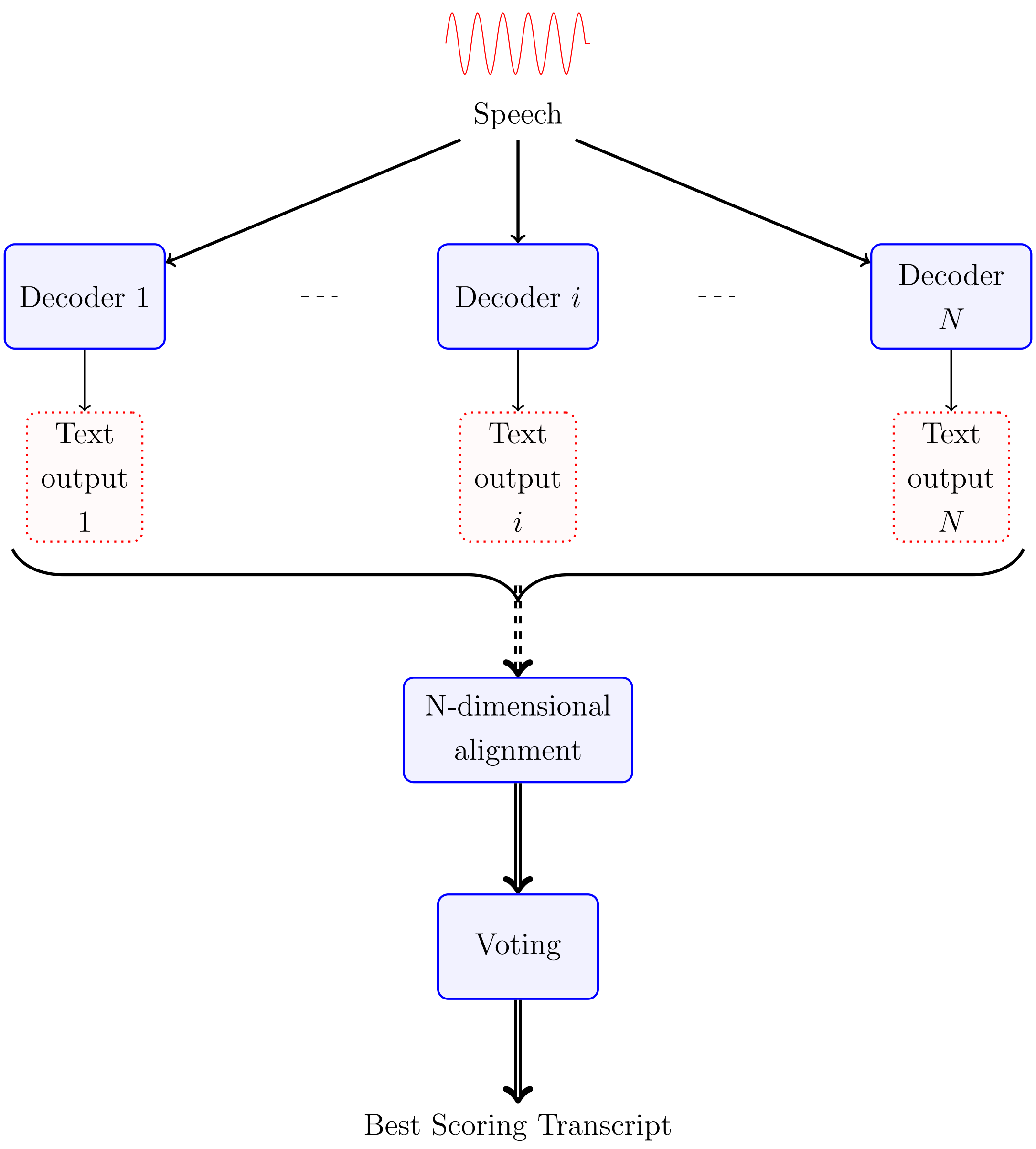

Figure 3.1: ROVER Procedure

Three voting mechanisms have been presented in [25], two of which involved the use of word level confidence values. At each location in the composite word transition network, a score is computed for each word using Equation 3.1.

$$Score(w) = \alpha(\frac{N(w,i)}{N_s}) + (1-\alpha)C(w,i) \tag{3.1}$$

where i is the current location in the WTN, N_s is the total number of combined systems, $N(w,i)$ the frequency of word w at the position i and $C(w,i)$ is the confidence value for word w at the position i. The parameter α is set to be the trade-off between using word frequency and confidence scores. In the case there is an insertion or deletion, the NULL transition, noted as @, in the word transition network, will have the confidence $conf(@)$. A training stage is therefore needed to optimize both the α parameter and the `NULL` transition confidence value. This is commonly done through grid-based searching.

- `Frequency of Occurrence`:
  In this voting schema, the word confidence values are not used, and therefore α is set to 1. Only occurrence information is used to select the winning word at the given location in the word transition network. According to [25], this scoring leads to a major problem: ties tend to occur very frequently. There is no way to determine any reliable knowledge source to break them, and thus ties are broken randomly.

- `Frequency of Occurrence and Average Word Confidence`:
  This voting schema has been introduced to overcome the problem of arbitrarily broken ties. Both the occurrence and the confidence values are used in the scoring of each word at a given location in the word transition network. For each word, the overall confidence is obtained by averaging the confidence values of all the occurrences of that word at the same location.

- `Frequency of Occurrence and Maximum Confidence`:
  The scoring is very similar to the earlier schema. However, instead of using the average confidence, the maximum value is selected as the composite confidence for each word at a specific location in the word transition network. It is worth mentioning that both parameters have to be optimized for each voting schema.

3.2 Motivations behind *c*ROVER

Researchers have attempted to improve on the performance of the original ROVER for several years now. As detailed in Section 2.3.5, these attempts, such as the use of new features and machine learning tools to select the winner word at each slot of the transition network, have achieved some WER reduction. However, it appears that ROVER performance has reached a plateau and that it has become quite difficult to achieve a substantial WER reduction. Nevertheless, we strongly believe that improving ROVER is still possible, and even more practical than improving CNC and fWER-based combination. In fact, these combination procedures cannot guarantee any large WER reduction because the baseline systems are already very optimized. Per our discussion in Section 2.3.2, CN and fWER are approximations to a different paradigm in the speech recognition, aiming at reducing the WER directly. However, ROVER works on baseline systems, aiming at reducing the SER instead of the WER. That's where we see the potential. It is therefore easier to improve and tune ROVER, rather than optimizing CNC or fWER-based combination. For this reason, we have chosen to focus on the ROVER system, since we strongly believe that there is still room for research and improvment.

We have noticed upon our review of automatic error detection in LVCSR, that researchers are still struggling to come up with a decent approach that works reasonably well. In fact, the problem of erroneous transcriptions has been around since the very first days of speech recognition technology, and we are still unable to efficiently spot errors in the decoders' outputs. Current error detectors suffer from low precision and/or low recall rates. So the question that kept arising, was how to make use of what has been done so far in this area without having to suffer from this poor performance. The answer to this question can be found right in the next question: why do we have to apply the error detector on the whole transcription output? If we know exactly when to involve the error classifier to check whether the word in question is an error or not, we would be able to achieve better performance and avoid falsely tagging correctly recognized words as erroneous outputs. Based on our review of the current advances in automatic error detection in Section 2.4, we have decided that the probabilistic approaches

to errors detection in LVCSR are the most suitable for large vocabulary, domain-independent applications. In fact, it is almost impossible for pattern matching techniques to collect all types of errors when dealing with applications of this nature.
Based on the conclusions and motivations presented above, our research work objectives are aimed at augmenting the original ROVER with an error detection approach, in order to reduce the WER of the composite WTN. The next section describes this proposed approach.

3.3 *c*ROVER: the Context-augmented ROVER

In this section, we describe our proposed approach to improve ROVER's performance. We called it *c*ROVER, where the *c* stands for context[61, 63, 64]. The idea is to embed a contextual analysis within the ROVER procedure to eliminate as many errors as possible.

3.3.1 Objectives

As described in Section 2.3.1, ROVER is a two-step procedure. First, a composite WTN is built from the outputs of different decoders. Then a voting algorithm browses the WTN to select the winner word at each slot. It is worth mentioning here that all errors are being propagated through the composite WTN, which led us to think of involving an automatic error detection technique before reaching the voting step. The objective of *c*ROVER is to eliminate erroneous words from the composite WTN in order to guarantee a smoother selection of the winner token at each slot.

3.3.2 *c*ROVER Architecture

The architecture of the *c*ROVER procedure is shown in Figure 3.2. The augmented ROVER with error filtering works as follows: first, the building of the WTN from the outputs of different decoders. Second, an error filtering stage is introduced to the newly built WTN to eliminate erroneous words. Finally, the voting algorithm browses the newly updated WTN to select the winner word at

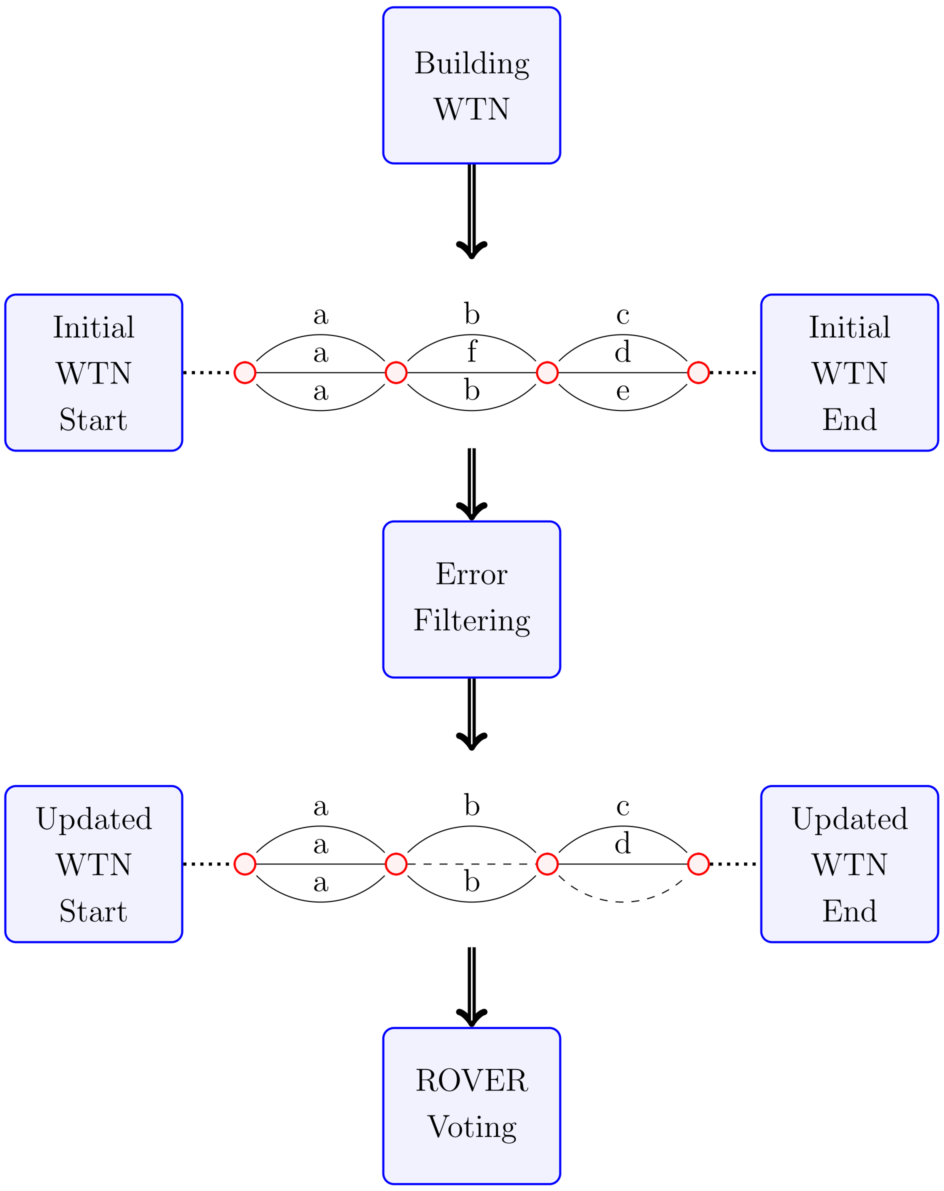

Figure 3.2: *c*ROVER System Overview

each slot of the WTN. Now the question is how to apply the error filtering on the composite WTN? As discussed in Section 3.2, the error filtering shouldn't be applied on the whole transcription to minimize false tagging correctly transcribed words as errors. Therefore, upon building the WTN, the error detector is only applied on the slot with discrepancies between the different speech decoders. In other words, when all recognizers agree on the same word at a specific slot, there is no need to check whether the word in question is an error or not. In the sample example shown in Figure 3.2, error detection is only applied on the second and third slot of the composite WTN. For this reason, we start by building the WTN before involving the contextual analysis, in order to spot the slots with discrepancies. Once such a slot has been detected, the error classifier is involved. If an arc (word) in the slot is tagged as an erroneous token, the arc in question is removed and replaced by the NULL transition. The NULL transition is later on handled by the voting algorithm as a deletion.
The architecture we have described so far is generic. That is, any error detection approach, whether probabilistic, non-probabilistic, or hybrid, could be used. The approach is independent from the error filtering procedure. The next section provides details about this error spotting stage. The focus will be on the probabilistic based approaches, per our discussion in Section 3.2.

3.3.3 Automatic Error Detection

The low recall and precision ratios of the current automatic error detection techniques in speech transcription led us to investigate ways to improve both of these ratios[62]. The idea is to combine different error detection approaches in the hope that the new technique achieves higher recall, without degrading the precision ratio. The ultimate goal is to be able to improve both ratios simultaneously upon the combination of the error detection techniques. The implicit assumption in this thinking, is that the error detection techniques have to have different performance in terms of these two ratios. In other words, when one approach achieves high recall and low precision, the other technique to be combined with it, needs to achieve high precision and low recall to ensure improvement in both ratios with the new technique. The logic of our proposed approach is to preserve each technique's advantage or powerful characteristics in the final combination. Figure 3.3 describes the flow of the error detection combination approach. All probabilistic based approaches rely on thresholding a confidence score to decide whether or not a given word is an error or not. However, the scale of each error detection technique's confidence score is different. Therefore a score normalization stage is needed to standardize all confidence scores from the various detection techniques to lie between zero and one. Equation 3.2 is used to normalize the confidence scores, where X is the score to be normalized, and min, respectively max, is the minimum, respectively maximum, value of the technique's confidence score.

$$X_{scaled} = \frac{X - min}{max - min} \tag{3.2}$$

Once all the confidence scores have been normalized, a score combination formula is then applied to build a new score. The classification threshold, K, is then applied to this new score to detect erroneous output. Two score combination formulas were used, namely Weighted Average (WA) and Harmonic Mean (HM), as shown in Equations 3.3 and 3.4, where $Score_i$ refers to the confidence score of the ith error detection technique, N is the total number of combined error detection techniques, and α_i are weighting scales to each technique in such a way $\sum_{i=1}^{N} \alpha_i = 1$

$$Score_{WA} = \sum_{i=1}^{N} \alpha_i Score_i \tag{3.3}$$

$$Score_{HM} = \frac{N}{\sum_{i=1}^{N} \frac{1}{Score_i}} \tag{3.4}$$

The weighting factors play an important role in realizing a trade off between various detection techniques to optimize recall and precision ratios. These coefficients need to be optimized a priori during training. Besides the two score aggregations, we have also used direct combination, which means that a given token in a transcript is tagged as an error if at least one of the error detectors tags it as an error.

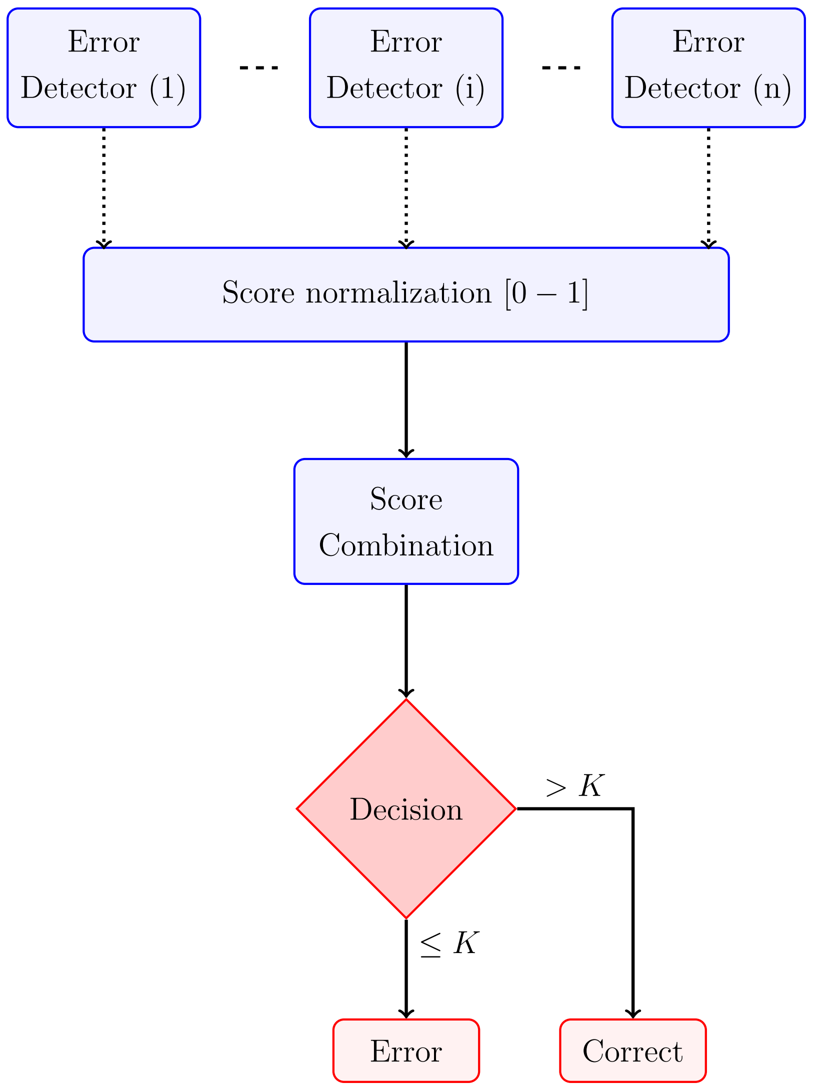

Figure 3.3: Combination Procedure for n Error Detection Techniques

Implicitly, this type of combination is in favor of the recall rate because it only takes one decision to classify a given word as an erroneous output. Algorithm 1 summarizes the combination procedure for N different speech transcription error detection techniques.

Algorithm 1 Combination of Error Detection Techniques

```
1: Compute the score of the word $w$, $Score_i$, for each technique.
2: Scale the confidence score to the [0,1] interval, using Eq.3.2.
3: if Direct Combination then
4:     if $\exists Score_i \leq K$ then
5:         Tag the word $w$ as an error.
6:     end if
7: else
8:     Compute the new confidence score, $Score_{comb}$, using Eq.3.3 or Eq.3.4.
9:     Tag the word $w$ as an error if $Score_{comb} \leq K$.
10: end if
```

In Step 4 and 9 of Algorithm 1, the threshold parameter K is to be optimized through a training stage. It is used to control the error detection rate. The higher K is, the more aggressive the error filtering, and vice versa. If K is quite low, more erroneous words slip past the combined error detector. Figure 3.4 reports a detailed architecture of the *c*ROVER technique, when augmented with a combination of error detection approaches. The n error classifiers cooperate together to decide whether or not an arc in a given WTN slot is correct or not. Based on this cooperative decision, the arc in question can be deleted and replaced by the NULL transition if the token has been flagged as an error. Otherwise, the arc is left unchanged.

3.3.4 Error Filtering Integration 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

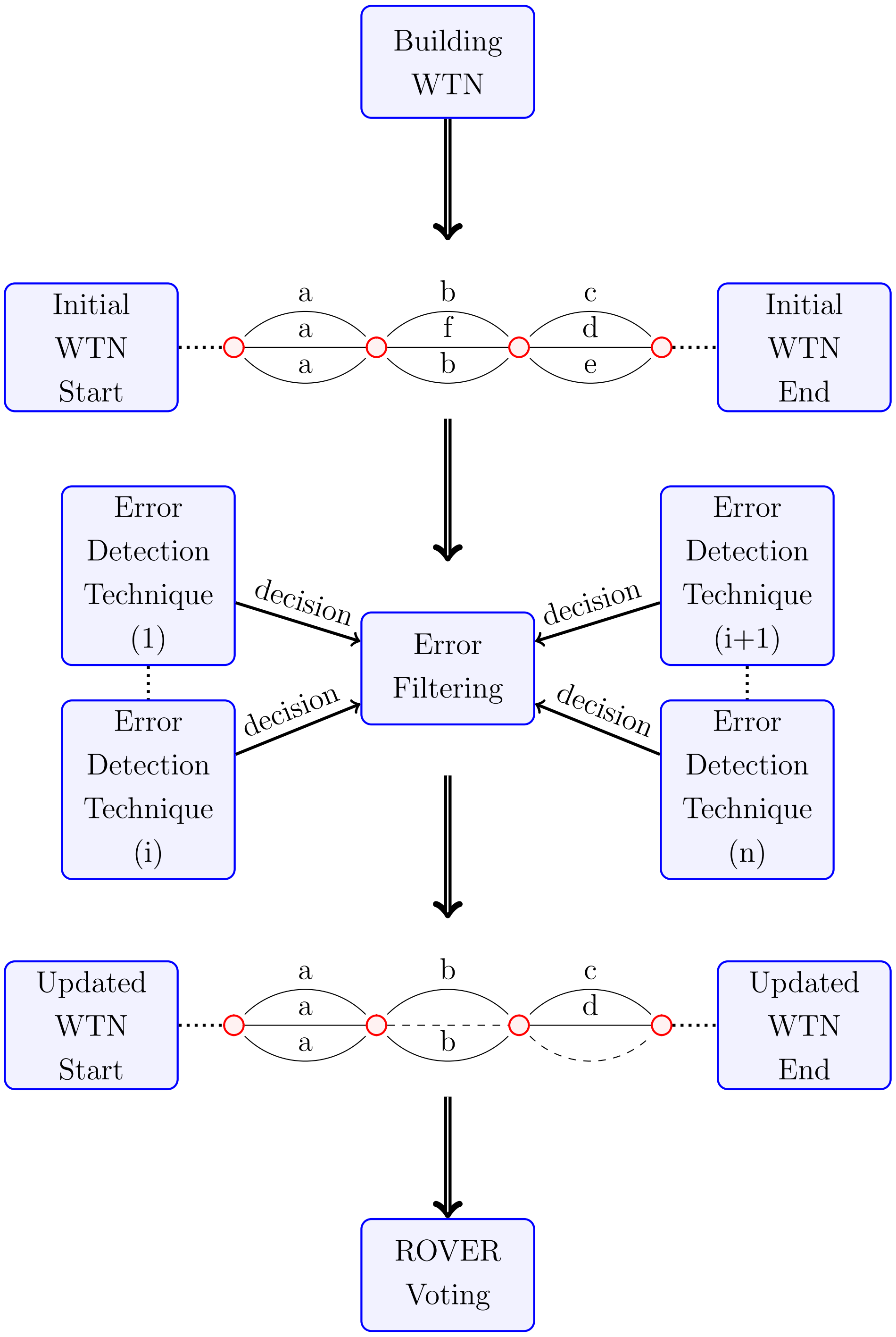

Figure 3.4: *c*ROVER Detailed System Overview

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 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 augmented ROVER with error filtering is detailed in Algorithm 2.

Algorithm 2 Augmenting ROVER with Error Detectors

```
1: Create the composite WTN by aligning the WTNs from the different recog-
   nizers.
2: for all slots with discrepancies in the composite WTN do
3:     Apply the error filtering.
4:     Remove the detected erroneous words from the slot, and replace them with
       the NULL transitions.
5: end for
6: Apply voting on the new WTN.
```

Once the composite WTN is built, and instead of applying the voting mechanism at each slot, a pre-filtering stage is introduced. At each slot with discrepancies, the error detector is used to spot errors. If a token is flagged as an error, the algorithm updates the slot by removing the arc of the erroneous word, and replacing it by a NULL transition. This simulates a deletion, and the ROVER voting schema handles it accordingly. Once all slots are pre-processed, the voting algorithms are used to select the most appropriate token at each slot in the new network.

The next section presents a case study of the whole approach with two widely-used probabilistic error detection techniques: Point-wise Mutual Information, and Latent Semantic Indexing-based approaches.

3.4 A *c*ROVER Implementation with Probabilistic-based Approaches

In this section, we present a case study of the newly proposed *c*ROVER. The example describes the integration of ROVER with two widely-known error detection techniques, namely, PMI- and LSI-based detectors.

3.4.1 PMI-based Error Detection

The PMI-based error detector aims at spotting outliers in a given decoder's output. A word is tagged as an outlier if the score computed by the PMI-based procedure is lower than a given threshold. The PMI-based detector computes a confidence score through the aggregation of PMI scores of word pairs in the given transcription. Before we go into detailing this procedure, let us first define a few terms:

- 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. For instance, a neighborhood with a window of 2 around w implies two left side context tokens, and two right side ones, along with the word w itself.

- Pair-wise Semantic Similarity $S(w_i, w_j)$ 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 3.5 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.

$$PMI(w_i, w_j) = \log\left(\frac{P(w_i, w_j)}{P(w_i).P(w_j)}\right) \tag{3.5}$$

  Given a large textual corpus with size N tokens, the probabilities introduced in Equation 3.5 can be computed using Equations 3.6, where $c(w_i)$ and

$c(w_i, w_j)$ are the frequency counts collected from the corpus.

$$\begin{aligned} P(w_i) &= \frac{c(w_i)}{N} \\ P(w_i, w_j) &= \frac{c(w_i, w_j)}{N} \end{aligned} \tag{3.6}$$

The process of detecting an error using the PMI-based technique[50] is detailed in Algorithm 3.

Algorithm 3 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 the token $w$. Scale up the PMIs in such a way
   that they are all non-negative.
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 $N(w_i)$.
5: Tag the word $w$ as an error if $SC(w) \leq K.SC_{avg}$.
   $K$ is a filtering parameter to control the tolerance of the algorithm for recog-
   nition errors.
```

In Step 2 of the algorithm, the scaling is done through normalization of the PMI scores. Different window sizes have been used for the first step. In other words, the left and right context are expanded to evaluate the impact on the performance of the error detection. This parameter is key, since all the PMI scores heavily depend on its value. In Step 3 of the algorithm, the semantic coherence has been computed using different aggregation variants:

- Harmonic mean: $SC(w_i) = \dfrac{n}{\displaystyle\sum_{i \neq j} \frac{1}{PMI(w_i, w_j)}}$

- Arithmetic mean: $SC(w_i) = \dfrac{1}{n} \displaystyle\sum_{i \neq j} PMI(w_i, w_j)$

- Maximum: $SC(w_i) = \max_{i \neq j} PMI(w_i, w_j)$
- 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. This switch is to be handled with care, as it has a direct impact on the false positive rate of the error classifier. 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 are tagged as correctly transcribed.

3.4.2 LSI-based Error Detection

LSI determines the similarity between terms by analyzing their co-occurrences within several documents. LSI assumes that when words co-occur together within the same set of documents, these words are generally semantically related to one another. The LSI procedure mines for features that highlight the similarities between words. These features are obtained by applying a dimensionality reduction technique to a high dimensional word-feature matrix. Given a large textual corpus, a term-document matrix is built where rows stand for words, and columns stand for documents. The value in the cell (i, j), $w_{i,j}$, holds the weighted frequency of occurrence of word i in the document j. These weights are a combination of local and global weighting schemes, as shown in Equation 3.7.

$$w_{i,j} = localweight(i, j) * globalweight(i) \tag{3.7}$$

The local weights are used to reflect the importance of a word in a given document, whereas the global weighting calibrates its importance across all the documents. Several combinations of local and global weighting methods have been used throughout the literature[43], and the authors found that the combination of entropy (as global weighting) and the logarithm term frequency (as local weighting) produced the best performance on information retrieval tasks. In this research work, we have used this specific combination as shown by Equation 3.8 and Equation 3.9, where $freq(i, j)$ denotes the frequency of occurrence of term i in document j, N is the total number of documents constituting the corpus,

$H(d)$ is the entropy of document distribution, and $H(d|i)$ is the entropy of the term i across all documents.

$$\begin{aligned} localweight(i,j) &= log(1+freq(i,j)) \qquad (3.8) \\ globalweight(i) &= 1-entropy(i) \qquad (3.9) \\ &= 1-\frac{H(d|i)}{H(d)} \\ &= 1-\frac{-\sum_{j=1}^{N} P(i,j)*log(P(i,j))}{log(N)} \\ &= 1+\frac{\sum_{j=1}^{N} P(i,j)*log(P(i,j))}{log(N)} \end{aligned}$$

Singular Values Decomposition (SVD) is applied on the large term-document matrix. This is carried out to reduce dimensionality of the term and document vectors. Since the term-document matrix is very sparse and its rank is much more lower than its actual dimensionality, it is safe to represent the terms and documents vectors in a much lower dimensional space with little loss of information. In other words, if two terms are similar in the original highly dimensional space, they will still be semanticaly close in the reduced space. This new reduced space is basically obtained by selecting the first several dimensions of the SVD-decomposed matrix, since the eigenvalues are sorted by order of importance. Therefore the first few dimensions hold much of the information, and we can then derive reliable similarity measures between terms using only these dimensions. The most commonly used measure between vectors is the cosine metric, which basically measures the overlap along each dimension. The cosine similarity is computed using Equation 3.10, which is the angle between the two vectors u and v.

$$\cos(u,v)=\frac{<u,v>}{||u||*||v||}=\frac{\sum_{i}^{n}(u_i*v_i)}{\sqrt{\sum_{i}^{n}u_i^2}*\sqrt{\sum_{i}^{n}v_i^2}} \qquad (3.10)$$

Now that we have collected the most significant word features (through the SVD decomposition), and selected a metric to measure the similarity between any two words (through the cosine similarity measure), all that remains is to compute the semantic similarity score of a given word in an utterance of length M. Two different aggregations have been used: the Mean Semantic Scoring (MSS) and the Mean Rank of the Semantic Scores (MR)[49]. MSS and MR scores are computed as shown in Equation 3.11 and Equation 3.12 respectively.

$$MSS_i = \frac{1}{M}\sum_{j=1}^{M}\cos(w_i, w_j) \tag{3.11}$$

$$MR_i = \frac{1}{M}\sum_{j=1}^{M} RANK(\cos(w_i, w_j)) \tag{3.12}$$

The $\cos(w_i, w_j)$ is computed using Equation 3.10. The rank of the semantic score shown in Equation 3.12 is computed as follows. First, the set of semantic scores L_i is computed. L_i is the set of cosine scores between the word w_i and all the remaining words w_j in the corpus. The MR_i score is then the mean of the rank of each $\cos(w_i, w_j)$ score in the set of L_i. The LSI-based error detection works as follows: given a recognizer's transcription output, a word is tagged as erroneous if and only if its MSS (respectively MR) is below a threshold K. The error filtering procedure applied on a word w_i in a given transcription output, is detailed in Algorithm 4.

Algorithm 4 LSI-based Error Detection

```
1: Compute cosine scores between $w_i$ and all other words in the transcription
   output.
2: Compute $MSS_i$ score (Equation 3.11) or $MR_i$ score (Equation 3.12).
3: Tag the word $w_i$ as an error if $MSS_i \leq K$ or $MR_i \leq K$.
```

The threshold K is to be optimized through a training stage. It is used to control the error detection rate. The higher K is, the more aggressive the error filtering, and vice versa. If K is quite low, more erroneous words slip past the error detector.

3.4.3 Combination of Error Detectors

Both error detectors compute a confidence score, on which a threshold parameter K is applied to identify whether the token is an error or not. Before we are able to identify the composite decision of these two error classifiers, these scores need to be scaled to lie within the same range, as demonstrated by Equation 3.2. Once normalization is done, the aggregation of the two confidence scores can be done either through score weighting or harmonic averaging, as shown by Equations 3.3 and 3.4 respectively. The aggregated score can now be thresholded to decide whether or not the token is a recognition error. Obviously, this threshold needs to be optimized through a training stage. If the direct combination scheme is used, score aggregation is not required. Upon score normalization, each technique's score is thresholded, and a word is tagged as an error if at least one of the techniques classifies it as an error.

3.4.4 *c*ROVER with PMI and LSI Classifiers

Figure 3.5 reports the whole system architecture once both PMI- and LSI-based error detectors are integrated within the ROVER procedure. Two knowledge sources are required by the error filtering techniques, namely a textual corpus to extract the unigrams and bigrams counts for the PMI-based classifiers, and a term document matrix for the LSI-based classifier. Once the composite WTN network is built, it is browsed slot by slot to identify discrepancies. For each token in a given slot, the error filtering techniques compute a confidence score, which will then be normalized. Then depending on whether or not the direct combination is used, these scores are aggregated together. If the final decision tags the token as an error, the arc representing this token is removed and replaced by the NULL transition. Otherwise, the arc is left unchanged, and the same procedure is repeated on the remaining arcs of the slot. Once all arcs have been processed, the procedure is then reiterated on the next slot with discrepancies.

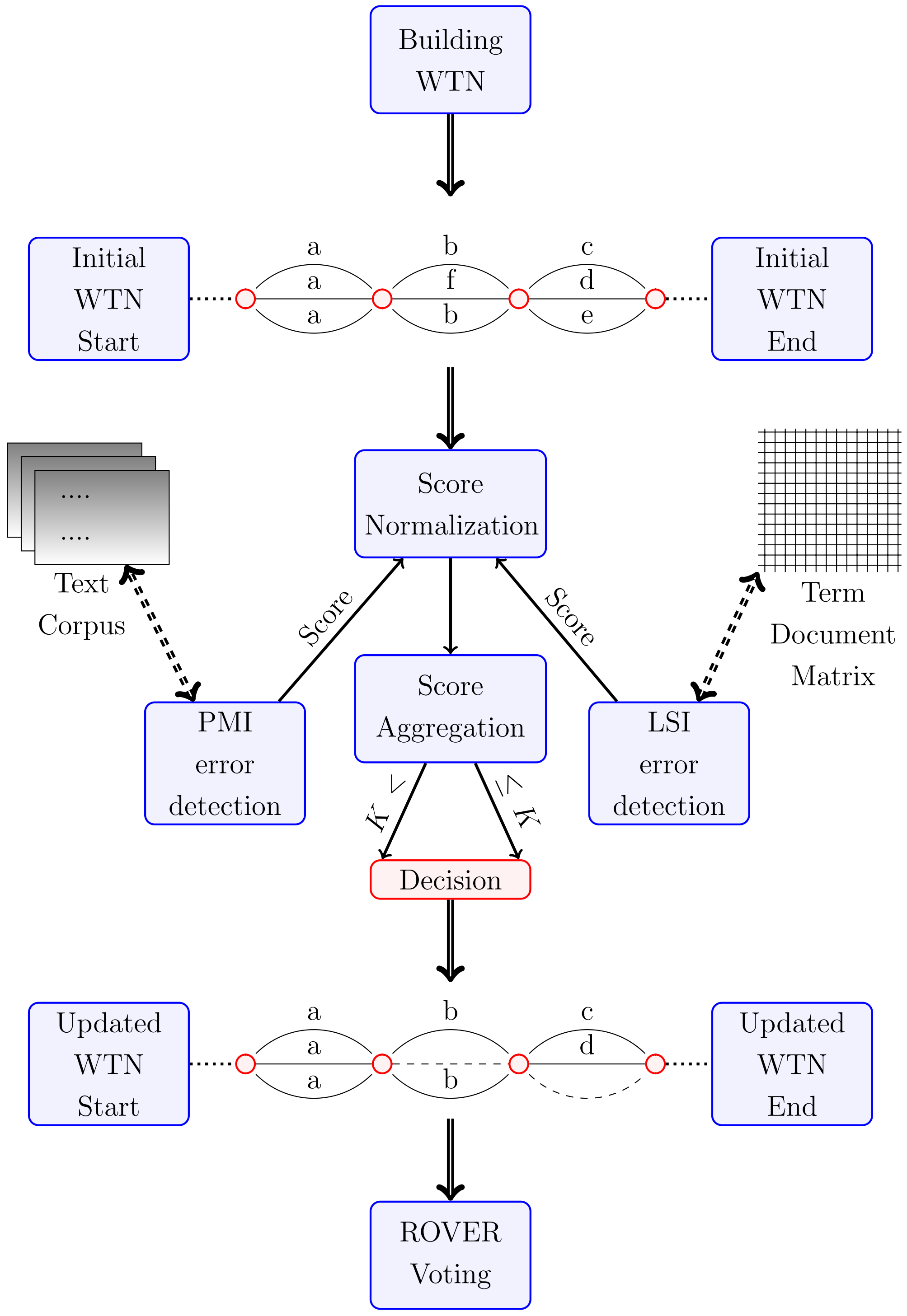

Figure 3.5: Case Study: *c*ROVER with PMI and LSI classifiers

3.5 New Voting Mechanism for ROVER

The proposed cROVER approach has inspired us to investigate a new scoring mechanism for ROVER. The new voting scheme consists of using the confidence scores collected during the contextual analysis (error filtering stage within the cROVER procedure) to compute the word scores at each slot of the WTN. This section starts by highlighting the voting issues of the ROVER procedure. It then proceeds with the description of the new scoring scheme.

3.5.1 Issues Related to ROVER's Voting

The original ROVER voting mechanisms have several shortcomings. The fact that two of the ROVER schemes rely on the speech recognizers' confidence score, make the whole procedure vulnerable. Issues in the confidence measures of speech recognizers are far from being solved. In our experiments, for example, we were quite unable to use them as they turned out to be all equal to one (especially the scores coming from the most widely-deployed and trusted commercial engine, Nuance N9). Furthermore, the frequency-based voting scheme proposed in [25] suffers from the random tiebreaker problem. In fact, ties tend to occur frequently, and the only way to resolve this issue, is to randomly select a winner whenever these ties occur.
In this section, a novel voting algorithm for the ROVER combination procedure is proposed. The voting mechanism relies on the contextual analysis that has been used, in the cROVER framework.

3.5.2 Confidence-based Voting Algorithm

The confidence-based voting scheme for the original ROVER consists of trading off between the original ROVER's scoring and the confidence score obtained upon the error filtering procedure. In other words, we are computing a new confidence measure besides the ones coming from the speech decoders. These new confidence scores are obtained through a contextual analysis using a confidence-based error filtering technique.
The original ROVER's scoring, given by Equation 3.1, is a weighted sum of

both the frequency of occurrence and the speech decoder's confidence scores. We are proposing to alter this equation to take into account the confidence scores computed during the contextual analysis. Equation 3.1 is used to compute a score for each word at each slot in the composite WTN.

$$Score(w) = \beta \left[\alpha(\frac{N(w,i)}{Ns}) + (1-\alpha)C(w,i) \right] + (1-\beta)Err(w,i) \tag{3.13}$$

$Err(w,i)$ is the aggregated score from different error detection algorithms assigned to the word w at slot i, and β is a parameter to balance between the original ROVER scores and the composite error filter scores. Similarly to the original α parameter, the β factor needs to be optimized through training.
Following the same logic as the *c*ROVER approach, the newly proposed voting algorithm is only applied on nodes with discrepancies. This is done to limit the incidence of false positives of the error filtering techniques. It is worth mentioning here, that we do not require the decision from each error classifier; only the scores are used in this voting scheme. In other words, the threshold factor K is irrelevant in this voting scheme.
The confidence-based voting scheme is described in algorithm 5.

Algorithm 5 ROVER's New Confidence-based Voting Scheme

```
1: for all nodes $i$ in the composite WTN do
2:   if $\exists$ Discrepancies then
3:     Compute ROVER's original score using Eq. 3.1.
4:     Compute the score $Err(w,i)$ .
5:     Aggregate both scores, using Eq. 3.13.
6:   else
7:     Compute ROVER's original score using Eq. 3.1 .
8:   end if
9:   The winner word is the word with the highest aggregated score.
10: end for
```

Algorithm 5 is used once the composite WTN is built from the different decoders' outputs. Similar to ROVER's original voting schemes, if ties occur, the algorithm will choose randomly one single best word. Line 7 is for nodes with only one single

token in all arcs, as well as the NULL transition. This special type of transition has a predefined confidence score, that is optimized beforehand through training, as proposed in [25].
Several error filters can be used in Step 4 of Algorithm 5. Similar to what has been proposed in the *c*ROVER approach, error filters are combined together through three aggregation schemes, specifically, direct combination, and weighted and harmonic aggregations. In the voting framework, only harmonic and weighted aggregations are used. When multiple error detection techniques scores are available, these scores are aggregated using Equation 3.3 or 3.4.
It is worth mentioning that the proposed voting scheme only works with error detection techniques that rely on thresholding a confidence score to decide whether or not a given word is an erroneous decoder's output. Therefore the voting scheme does not extend to all error detection techniques, such as pattern matching-based error filters for example.
The solution proposed in the original ROVER[25], to solve the problem of arbitrary broken ties, is to introduce the confidence scores from the speech decoders in the scoring formulas. That way, ties occur considerably less frequently. Our proposed confidence-based voting scheme outperforms the original ROVER voting in two ways. First, if the speech decoders' confidence measures are unavailable, the scores from the contextual analysis (error classifiers) compensate for these missing confidences, and thereby considerably lower the risk of ties during voting. Second, even when decoders' confidences are available, our proposed voting scheme lowers even further the risk of ties, as well as the risk of using the unreliable decoders' confidences, because of the second set of confidences that are introduced from the contextual analysis. For these reasons, the new voting scheme is more robust than the original ROVER voting schemes.

3.6 Conclusion

This chapter proposed a novel approach to improve on the original ROVER performance. Automatic error filtering techniques have been integrated within the ROVER procedure to filter out erroneous words at each slot of the composite WTN. An implementation using two probabilistic error detection techniques, namely, LSI, and PMI-based detectors, has been presented. The chapter concluded with a novel voting scheme for the original ROVER procedure inspired from the proposed *c*ROVER framework. Semantic scores from the error filtering techniques were used during the voting stage to select the correct word at each slot. The next chapter reports the experimental framework as well as the results and analysis of the case study presented in this chapter.

Chapter 4

Performance Evaluation

This chapter focuses on the performance evaluation of the proposed approach to augment ROVER with a contextual analysis through the use of automatic error detection techniques. The chapter starts by describing the experimental framework, followed by the assessment of the error filtering techniques. The assessment of the *c*ROVER procedure is carried out using two different test sets. The chapter then proceeds with a study of the *c*ROVER computational requirements. Experiments pertaining to the assessment of the novel voting procedure are described at the end of the chapter.

4.1 Evaluation Criteria

In the area of LVCSR, researchers usually rely on the WER metric[54]. Let N be the total number of tokens in a reference transcript. D is defined as the number of deleted tokens from the recognition, I is the number of insertion and S is the number of substitutions. The WER is then defined by Equation 4.1, as:

$$WER = \frac{D + S + I}{N} \tag{4.1}$$

Obviously, the lower the error rate, the better the recognition. A sample example is provided below in Figure 4.1. In this example, there are four words in the reference transcript ($N = 4$). After aligning the hypothesis with the reference, through dynamic programming, we notice that only one single word has been

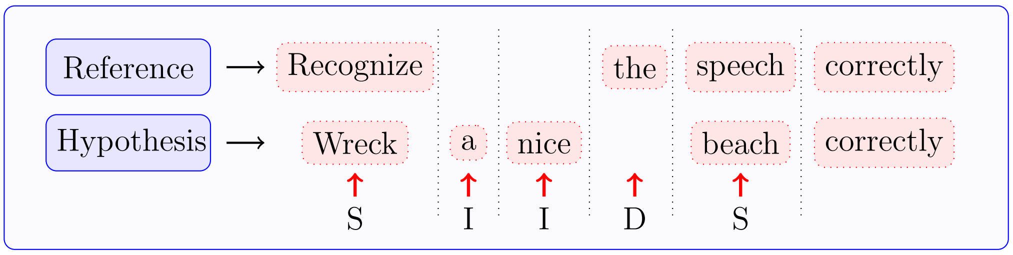

Figure 4.1: How to Compute the WER?

recognized properly (specifically the word "correctly"). Substitutions, deletions, and insertions are shown in the figure by S, D and I respectively. There are two substitutions ($S = 2$), one deletion ($D = 1$) and two insertions ($I = 2$). The resulting WER in this example is 125%.

In order to assess the performance of the error detection module, metrics from the machine learning theory were used. The F-measure, the precision, and the recall were reported. For a two-class problem (`Error`/`Correct`), we can define the following quantities: true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN), as illustrated in Table 4.1.

		Reference	
		`Error`	`Correct`
Hypothesis	`Error`	TP	FP
	`Correct`	FN	TN

Table 4.1: Two-Class Classification Terminology

The precision and recall, can then be defined as in Equations 4.2 and 4.3 respectively,

$$Precision = \frac{TP}{TP + FP} \tag{4.2}$$

$$Recall = \frac{TP}{TP + FN} \tag{4.3}$$

The F-measure, also called the F_1 score, is nothing else but the harmonic average of the precision and the recall ratios, as shown in Equation 4.4.

$$F_1 = 2 \ . \ \frac{Precision \ . \ Recall}{Precision + Recall} \tag{4.4}$$

4.2 Experimental Setup

In this section, we provide details on the experimental framework used to assess the cROVER approach. The baseline WER were obtained using the NIST implementation of ROVER, within the Speech Recognition Scoring Toolkit (SCTK)[55]. The tool has been later modified and upgraded to integrate the automatic error filtering stage within the ROVER procedure.

4.2.1 Automatic Speech Decoders

The experiments were conducted using recognition outputs obtained from two widely-known automatic speech decoders. The first is the latest version of Nuance Communication Speech recognizer (Version 9.0). This commercial decoder is state of the art in terms of performance and technology, and is currently deployed in hundreds of call centers around the globe. This engine comes with its own highly enhanced acoustic models. We have built our own language models, compatible with this engine.
The second speech decoder we have used is the Carnegie Mellon University (CMU) Sphinx 4, which is a speech recognition system written in Java. This engine has been designed to be very flexible and thus became an excellent platform for speech research. In fact, any front end processing can be used thanks to the plugin-based design of this open source decoder. Furthermore, acoustic models and language models can be of any formats. This is a major advantage because it does not constraint researchers to limited predefined formats.

4.2.2 Testing Sets

In terms of data, we have considered the English Broadcast News Speech (HUB4) [56] testing framework. This corpus is composed of both speech data (LDC98S71) and transcripts (LDC98T28). It is a total of 97 hours of 16000 Hz recordings from radio and television news broadcasts. Transcriptions of this HUB4 corpus have been used to train the language model for all the baseline systems except the Sphinx 4, where another freely available language model has been used. From

now on, the language model trained using the LDC98T28 is referred to as LM-98T28. For Nuance V9.0, the default acoustic model for US English has been used, along with the LM-98T28 trained with the decoder's statistical language modeling toolkit. With Sphinx 4 three different language models have been used: the LM-98T28 defined earlier, an open source model for broadcast news transcriptions from CMU[57], referred to as LM-BN99 hereafter, and a third language model created from the English Gigaword corpus[58], referred to as LM-GIGA hereafter. The vocabulary size of LM-GIGA and LM-BN99 is $64,000$ words. The LM-98T28 vocabulary size was limited to just $20,000$ words due to the constraints of Nuance V9. In terms of Sphinx 4 acoustic modeling, the already trained HUB4 model provided on the Sphinx download site has been used.
Without access to the HUB4 evaluation corpus, it was decided to select two subsets of the LDC98S71 training data, for evaluation purposes. The first set, Set 1, consisted of 2 hours and 36 minutes of speech data, whereas the second set, Set 2, consisted of 8 hours and 30 minutes. The average sentence length in both testing sets is 55 words. Both testing sets were not included in the data used to train the language models.
In order to simulate speech decoders' outputs from several sites, we have created different combinations of speech decoders and language models. A total of four configurations has been set up:

- **s4-LM-98T28**: Sphinx 4 with the LM-98T28 language model.
- **s4-LM-BN99**: Sphinx 4 with the LM-BN99 language model.
- **s4-LM-GIGA**: Sphinx 4 with the LM-GIGA language model.
- **v9-LM-98T28**: Nuance v9 with the LM-98T28 language model.

In the first experiments with the Set 1 test set, we chose to use the following decoder configurations: s4-LM-98T28, s4-LM-BN99 and v9-LM-98T28, whereas in the second experiments with the Set 2 test set, the following configurations were used: s4-LM-98T28, s4-LM-BN99, and s4-LM-GIGA. In both experiments, all two- and three-recognizer combinations were carried out. Table 4.2 reports these settings for both the first and second experiments. These IDs will be used

	ID	Decoders' Combination Configuration
SET 1	C1	v9-LM-98T28 - s4-LM-98T28
	C2	v9-LM-98T28 - s4-LM-BN99
	C3	s4-LM-98T28 v9-LM-98T28
	C4	s4-LM-98T28 - s4-LM-BN99
	C5	s4-LM-BN99 - s4-LM-98T28
	C6	s4-LM-BN99 - v9-LM-98T28
	C7	v9-LM-98T28 - s4-LM-98T28 - s4-LM-BN99
	C8	v9-LM-98T28 - s4-LM-BN99 - s4-LM-98T28
	C9	s4-LM-98T28 - v9-LM-98T28 - s4-LM-BN99
	C10	s4-LM-98T28 - s4-LM-BN99 - v9-LM-98T28
	C11	s4-LM-BN99 - s4-LM-98T28 - v9-LM-98T28
	C12	s4-LM-BN99 - v9-LM-98T28 - s4-LM-98T28
SET 2	C1	s4-LM-GIGA - s4-LM-98T28
	C2	s4-LM-GIGA - s4-LM-BN99
	C3	s4-LM-98T28 s4-LM-GIGA
	C4	s4-LM-98T28 - s4-LM-BN99
	C5	s4-LM-BN99 - s4-LM-98T28
	C6	s4-LM-BN99 - s4-LM-GIGA
	C7	s4-LM-GIGA - s4-LM-98T28 - s4-LM-BN99
	C8	s4-LM-GIGA - s4-LM-BN99 - s4-LM-98T28
	C9	s4-LM-98T28 - s4-LM-GIGA - s4-LM-BN99
	C10	s4-LM-98T28 - s4-LM-BN99 - s4-LM-GIGA
	C11	s4-LM-BN99 - s4-LM-98T28 - s4-LM-GIGA
	C12	s4-LM-BN99 - s4-LM-GIGA - s4-LM-98T28

Table 4.2: Decoders' Combinations ID for Set 1 and Set 2

later on in this chapter instead of the whole configuration to make the plots more reader friendly. Note here that the combination order matters. This is an inherent problem from the ROVER composite WTN-building stage as discussed in Section 2.3.1. For example, the combination with id C3 for Set 1 in Table 4.2, means that when building the composite WTN, we start with the WTN, output of s4-LM-98T28 first, then we align the second WTN, output of V9-LM-98T28,

on top of the first WTN.
It is worth mentioning that because of the use of different LMs, a normalization step is required to standardize the output of the different speech decoders. In other words, the same word can be written in different manners and therefore all these forms have to be unified under a single form. This is a tedious task, which if not done properly, can lead to wrong WER figures, when the outputs of all decoders are aligned against each other as shown in Figure 4.1. Examples of these issues include:

- CNN can be written as c. n. n. (three distinct letters), c.n.n. (one single word), cnn (one single word), c n n (three letters), etc. All of these forms must be converted to one single form.
- Vote, voTe, vote, etc should all be considered as the same word.
- It's, its and it s have been all considered as the same word (even though it is not quite true).

4.2.3 PMI-based Error Detector

The PMI-based error detection technique used in this paper requires uni-gram and bi-gram frequency counts. These counts need to be collected from a very large textual corpus. This was first investigated using the Wikipedia XML dumps; however, after collecting bi-gram counts, 39% of the bi-grams were not found. This led to considering a much larger corpus. But collecting word counts from big corpora is not an easy task, in terms of memory, storage and processing power requirements. In fact, the corpus by itself would require several terabytes - therefore, we had to look for a pre-compiled collection of word and bi-gram counts. Google Inc.'s trillion-token token corpus (LDC2006T13) was the solution[59]. This is a dataset of six DVD-ROMs, totaling 24 GB of compressed n-gram counts. These n-gram counts were generated from approximately one trillion word tokens of text from publicly accessible Web pages. The length of the n-grams range from uni-grams to five-grams. For the purposes of this paper, only the 13.5 million uni-grams and the 314.8 million bi-grams were used. With the Google corpus, only 13% of the bi-grams sought were not found.

4.2.4 LSI-based Error Detector

The term-document matrix, required by the LSI based approach, has been built using the latest Wikipedia XML dump. A total of 3.2 million documents, and $100,000$ unique terms have been identified. Obviously, the matrix is very sparse because not all the tokens exists in all the documents of Wikipedia. The SVD decomposition on such a large sparse matrix has been carried out using the SVDLIBC[60] library from the Massachusetts Institute of Technology. Out of the $100,000$ tokens identified on Wikipedia, only the feature vectors of 5000 words were needed in the experiments carried out on Set 1 and 2.

4.3 Automatic Error Filtering Assessment

The PMI- and LSI-based error detectors, as well as their combinations' have been experimented with using Set 1. The output from different recognizers has been aligned against the transcript, then errors were manually spotted to serve as the testing set for the first class (`Error`). Correctly recognized tokens were also identified to serve as the testing set for the second class (`Correct`).

4.3.1 PMI-based Error Filtering

The PMI-based error detection requires the optimization of a few parameters. These parameters include the filtering parameter K, the aggregation method (arithmetic mean, summation, maximum and harmonic mean) used to compute the semantic coherence score from the PMI measures, and the size of the context window (the neighborhood). To optimize these parameters, a grid search was carried out. The filtering parameter ranged from 0.1 to 5 with different increments, and the window size ranged from a single token either side, up to as many as twenty neighbours to either side. The grid search totaled 5040 different configurations. It is worth noting that when the size of the window increases, it necessarily implies heavy computations - but since the intended application (speech transcription) is usually carried out off-line, it was felt there was no need to take care of the on-line and real time issues. This is detailed later in Section 4.6 of this chapter.

Figure 4.2 reports the behavior of the F-measure in terms of the filtering switch K and for a few window sizes. All the aggregation methods' plots have been included in each subplot. Note that these plots included a limited range and set of values for the different parameters, to make them reader-friendly. The first thing to notice here is that the best aggregation method depends on the window size. In fact, for a larger window size, the maximum aggregation method outperforms all the other methods, whereas for smaller window sizes, the harmonic mean guarantees better F-measure scores. Besides, for bigger windows, the summation and the harmonic mean aggregation perform very similarly. It can be concluded as well 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.

4.3.2 LSI-based Error Filtering

We have experimented with several word feature vector dimensions: 50, 100, 200, 300, 400, and 500. All the performance results reported below have been obtained using the logarithm base 2 in Equation 3.8 and Equation 3.9. The LSI-based error filtering technique requires the optimization of the threshold parameter K as well as the term features' dimensionality, that is the length of the feature vector representing each word in our corpus. Figure 4.3 reports the F-measure of the error filtering classifier with both aggregations, MSS and MR, as a function of the threshold K as well as the feature vectors dimensionality. We notice that the bigger the dimensionality, the higher the F-measure is, mainly for the MSS aggregation schemes. However, the larger the dimensionality is, the slower the detection. Speeding up this process is not the subject of this research work. In other words, the error filtering improves as the amount of information representing each word gets larger. This is expected since the LSI-based error detector relies on the similarity between words which is measured by using the feature vectors representing each word. This behavior is not manifested with the MR aggregation though. It appears that when we convert the semantic scores (cosine measures)

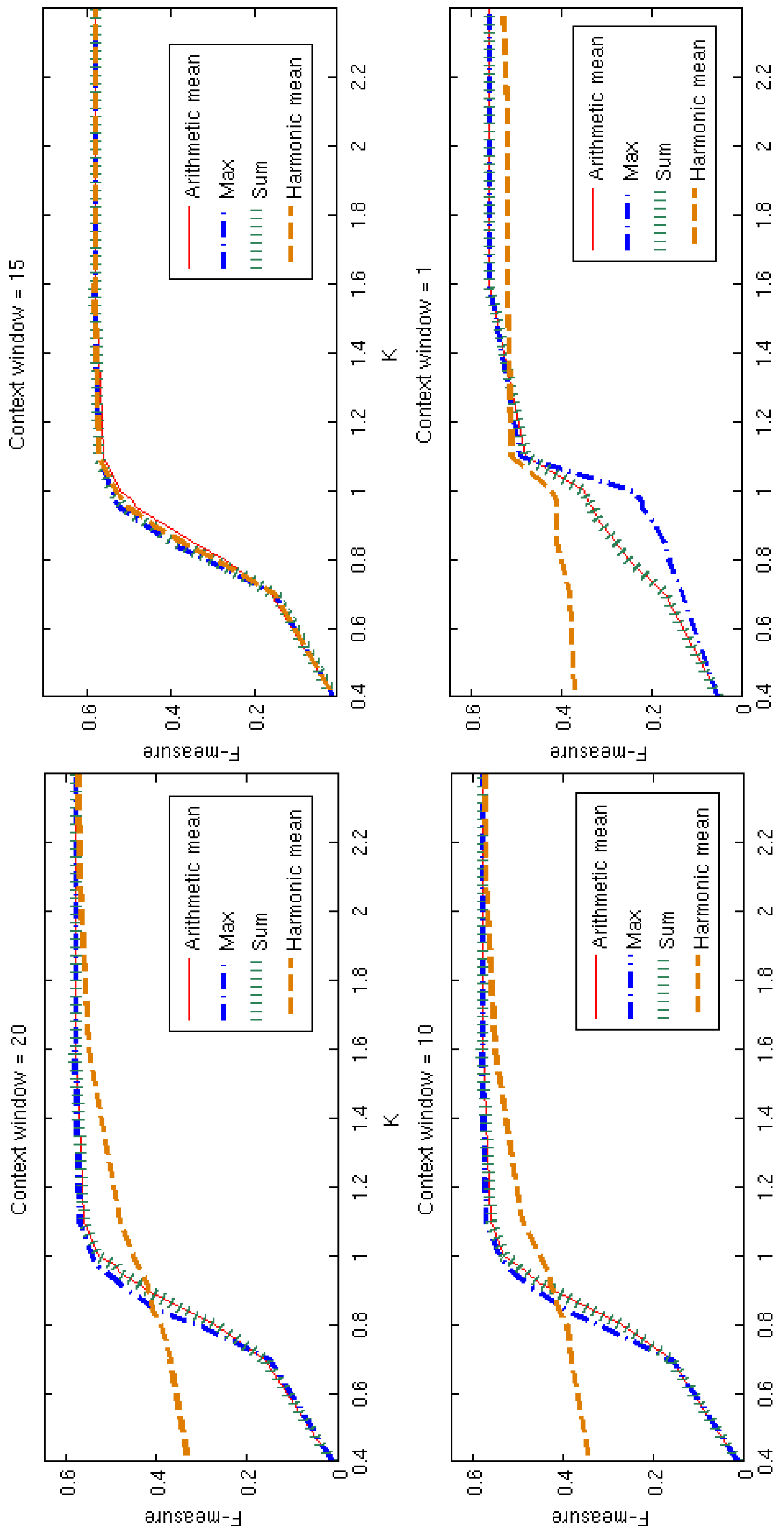

Figure 4.2: F-measure as a function of K and Aggregation methods with different Context Window Sizes

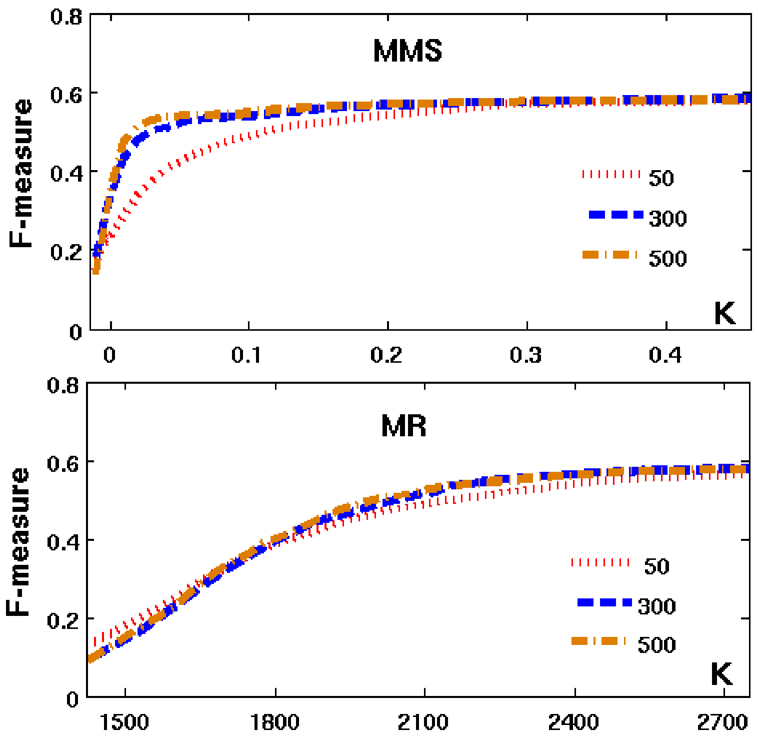

Figure 4.3: F-measure as of function of K and dimension

to ranks, this dependency to the dimensionality is somehow lost. In fact, we can observe that almost all dimensions lead to the same F-measure score. Also the increase in the F-measure is smoother with MR than with MMS. This is actually better for us, because it gives us a larger range to select the threshold K. The filtering threshold K is used to control the degree of filtering. That is, as it gets bigger, the filtering becomes more prone to errors and the precision starts decreasing drastically. This is demonstrated in Figure 4.3, where the F-measure reaches a steady state when K becomes large. Therefore, we have to select a threshold to better trade-off between the precision and recall.

4.3.3 Combination of Error Detectors

In this section, we study the impact of the combination of error detection techniques, namely weighted average, harmonic average and direct combination. Five hundred-feature vectors long have been selected to represent the tokens in the LSI based error detection technique. A neighborhood of twenty words has been selected to compute the PMI scores, and the maximum aggregation schema has been used to aggregate these scores. Figure 4.4 shows the precision vs recall graph for the different error classifiers.

The precision vs recall graph in Figure 4.4, highlights the effect produced by combining error detectors. In fact, all three combination scenarios outperform the individual error detection technique. The graph also shows that we cannot guarantee decent recall if precision is high with each individual error classifier. However, when we combine the classifiers, we could improve the recall ratio without sacrificing too much precision. Our main goal in error detection, is to capture as many errors as possible without falsely tagging correct tokens as erroneous output. Therefore, we want high precision with as high recall as possible. It appears now that when error detectors are combined, it is possible to achieve a better trade off between the recall and precision. The next sections tackle the impact of augmenting ROVER with the contextual analysis to filter out errors.

4.4 *c*ROVER Assessment with Set 1

In this section, we study the performance of *c*ROVER using the first testing set. It is worth mentioning that both speech decoders' confidence scores were not reliable, and therefore we have decided not to use them during the voting stage of the original ROVER process. In fact, almost all the confidence values were equal to 1 for both decoders. Therefore, if these scores were used during the voting procedure, no change in the voting outcome was recorded. For this reason, we only used the frequency-based voting algorithm in the reminder our experiments. A subset of Set 1, specifically 10% of the data, has been used to optimize the filtering threshold K. The neighborhood has been fixed to twenty words in the left and right context, and the maximum aggregation was used during the computation of the semantic coherence scores. For the LSI based error filtering, the dimensionality of the feature vectors were set to 300. That is, every word is represented by a feature vectors of 300 float values.
The baseline WER for the different decoder' combinations is shown in Table 4.3. The combination IDs for Set 1 have been defined previously in Table 4.2. Note

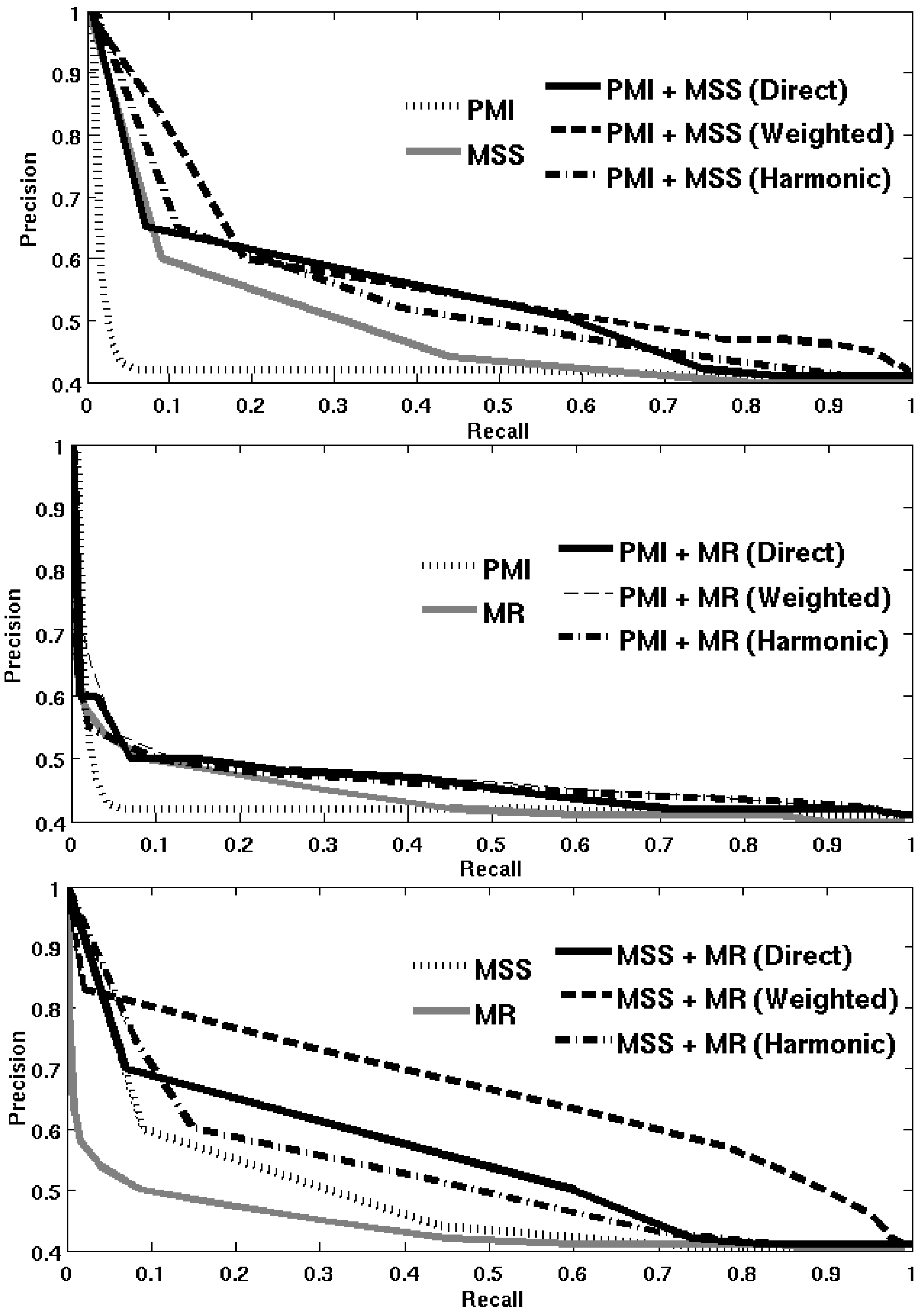

Figure 4.4: Error Filtering Combination: Precision vs Recall Graph for all Combinations of Error Filters

ID	WER (%)
C1	18.94
C2	19.55
C3	19.56
C4	20.36
C5	24.67
C6	24.45
C7	16.79
C8	16.92
C9	18.21
C10	18.78
C11	19.02
C12	18.56

Table 4.3: Set 1: ROVER's Baseline WER

that decoders' combinations C1 to C6 are binary combinations (only two decoders were combined), whereas experiments C7 to C12 involve trinary combinations. It is worth mentioning that these WER were lower than the WER of each single decoder; in other words, the ROVER process always yielded a lower WER for two and three decoders' combinations. These error rates are already very low. In fact, in the context of broadcast news LVCSR transcription, the current state of the art in terms of WER ranges between 15% to 30%. The lower end of this range can only be achieved with a limited vocabulary size of up to $10,000$ words. As explained in the previous chapter, the *c*ROVER approach aims at augmenting the original ROVER process with a pre-filtering stage to remove the erroneous words in the composite WTN. Three error detectors were experimented with namely, the PMI-based detector, LSI-based detector with the MSS aggregation scheme, and the LSI-based detector with the MR aggregation scheme. Figures 4.5 reports the absolute WER reduction achieved with the *c*ROVER when only two decoders are combined (experiments C1 to C6). The corresponding numerical values can be found in Appendix A.

Figure 4.6 reports the absolute WER reduction achieved with the *c*ROVER when

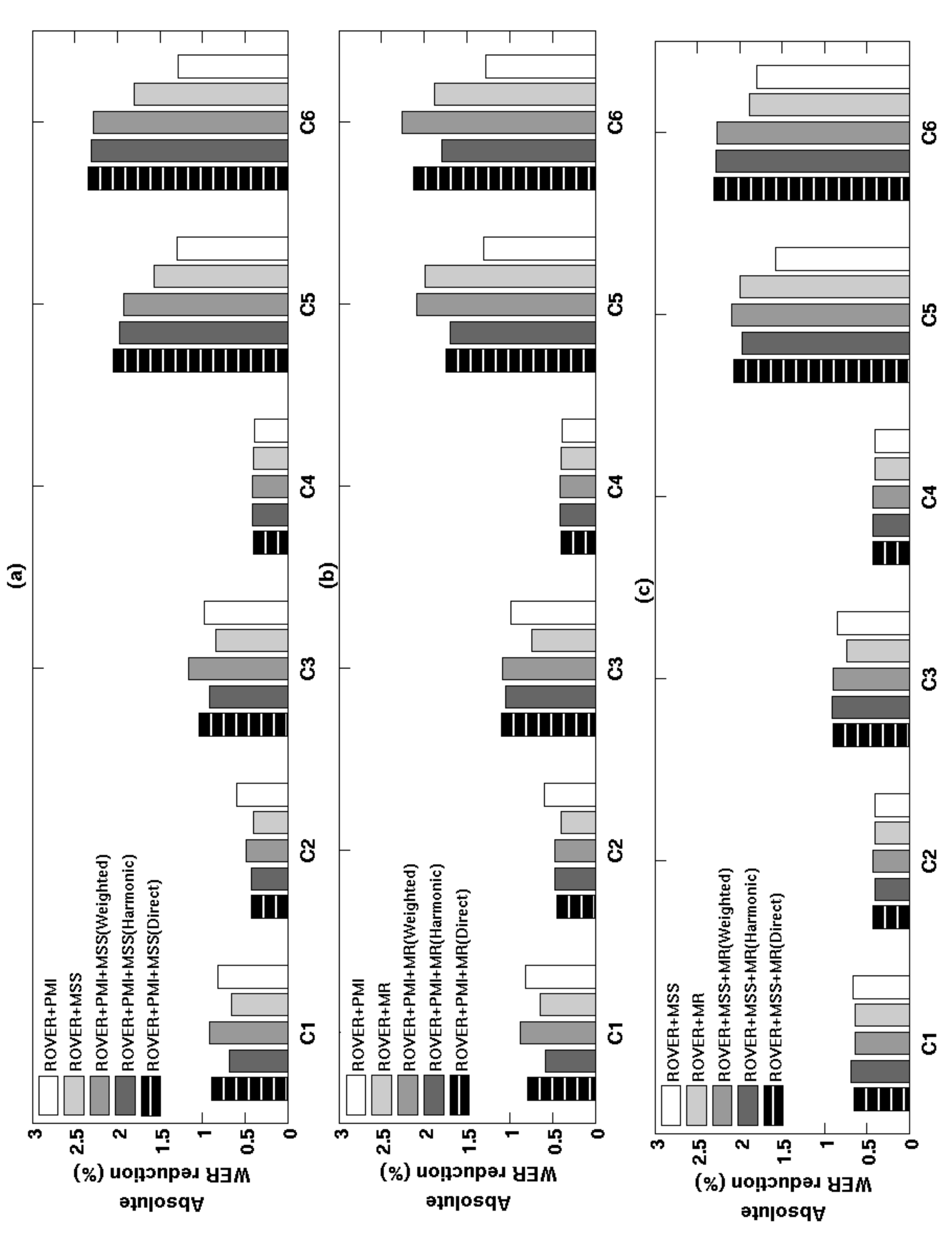

Figure 4.5: Set 1: Absolute WER Reduction with 2-decoder Combination

all three decoders are combined (experiments C7 to C12). The corresponding numerical values can be found in Appendix A. In Figures 4.5 and 4.6, the X axis is for the experiment ID, and the Y axis is for the percentage of the WER reduction. For each of the experiments (C1 to C12), we have reported the absolute reduction achieved when ROVER was augmented with the PMI-based error filtering, the MSS-based error filtering (LSI), the MR-based error filtering (LSI), and with the three types of combinations of two error detectors, (namely weighted, harmonic and direct type of error filtering combination schemes), respectively. In order to make the plots more reader friendly, we have divided each of the figures into three subplots, namely, subplot (a), (b), and (c). In subplot (a), we reported the absolute reduction when the PMI- and MSS-based error classifiers were used. In subplot (b), we reported the absolute reduction when the PMI- and MR-based error classifiers were used. In subplot (c), we reported the absolute reduction when the MSS and MR based error classifiers where used.
The first conclusion we can draw, is that *c*ROVER outperformed the original ROVER for all the experiments. There is some WER reduction in all configurations, with two and three decoders' combinations. The analysis of the absolute WER reduction plots is quite tedious, because we have to always keep in mind the baseline percentage when reading the absolute reduction from the plots. In order to highlight further findings, we have decided to plot the relative WER reduction instead because it encapsulates both the baseline WER of each experiment and the absolute WER reduction. Figure 4.7 reports the relative WER reduction when the output of two decoders were combined (experiments C1 to C6). The corresponding numerical values can be found in Appendix A. Figure 4.8 reports the relative WER reduction when the output of three decoders were combined (experiments C7 to C12). The corresponding numerical values can be found in Appendix A. In Figures 4.7 and 4.8, the Y axis now represents the relative WER reduction instead. Every thing else in these figures is the same as in Figures 4.5 and 4.6 in terms of the X axis and the reported values for each combination configuration.
When two decoders' outputs are combined (Figure 4.7), the highest relative WER reduction has reached 9.57%, specifically in experiment C6 subplot (a) of Figure

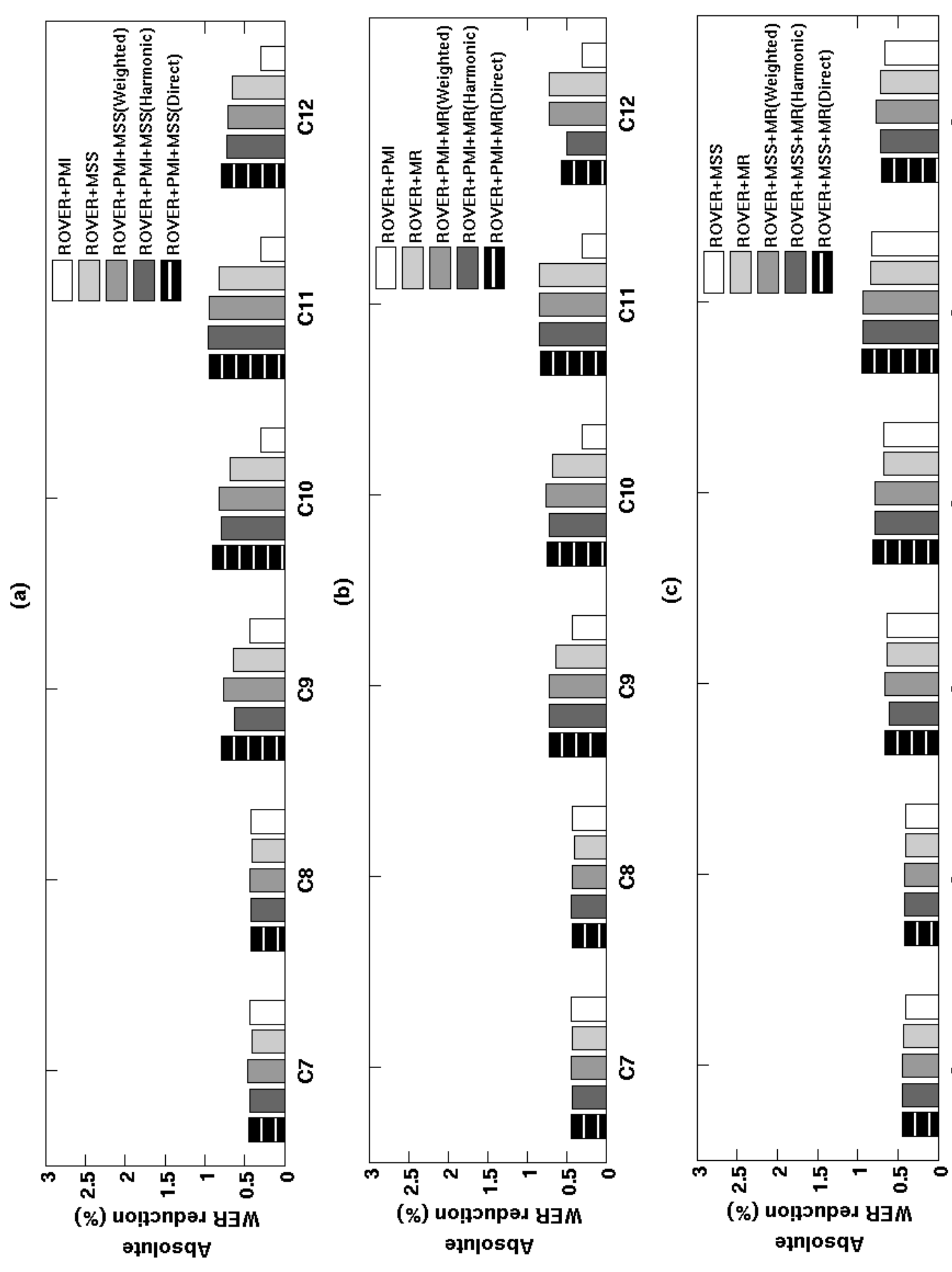

Figure 4.6: Set 1: Absolute WER Reduction with 3-decoder Combination

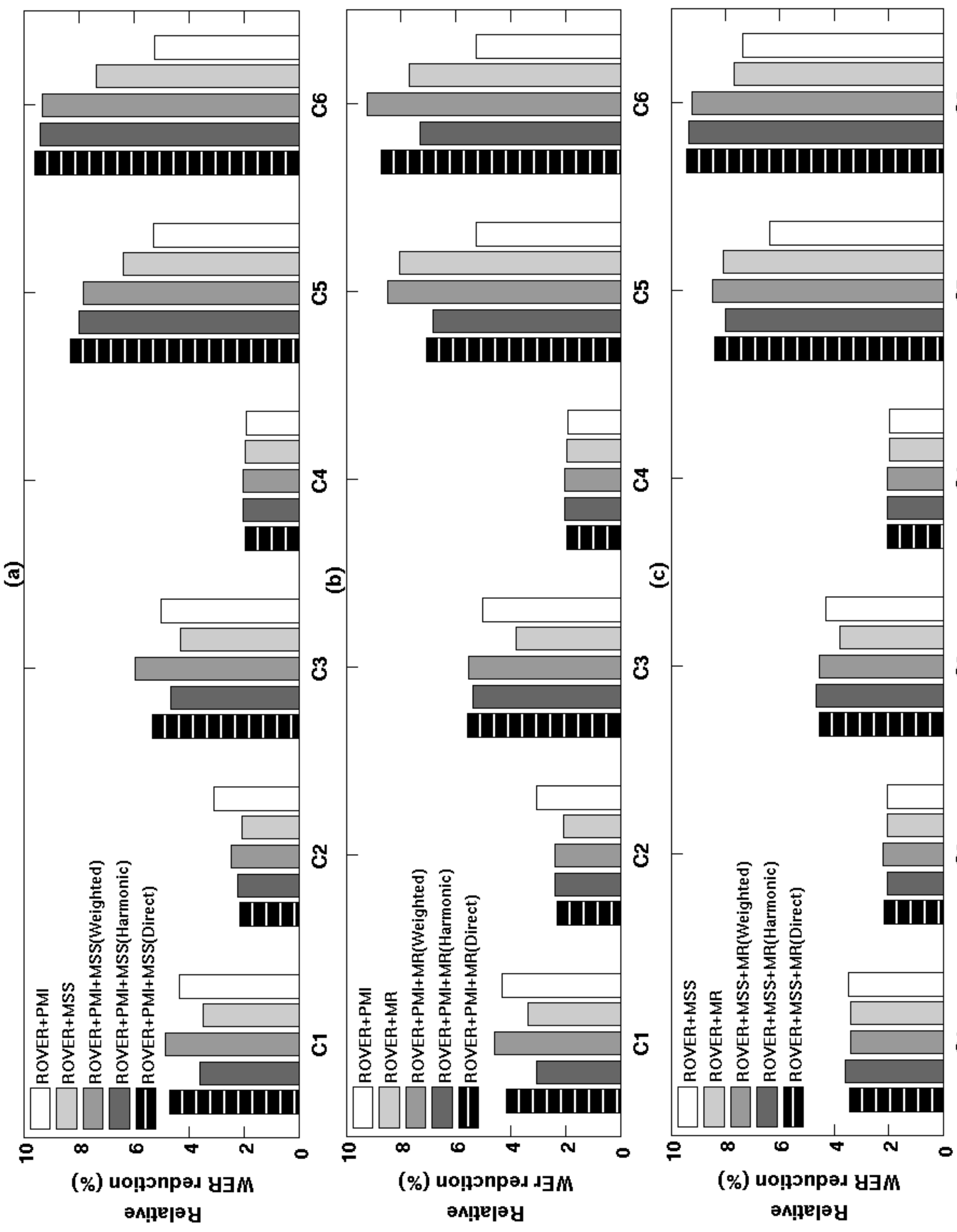

Figure 4.7: Set 1: Relative WER Reduction with 2-decoder Combination

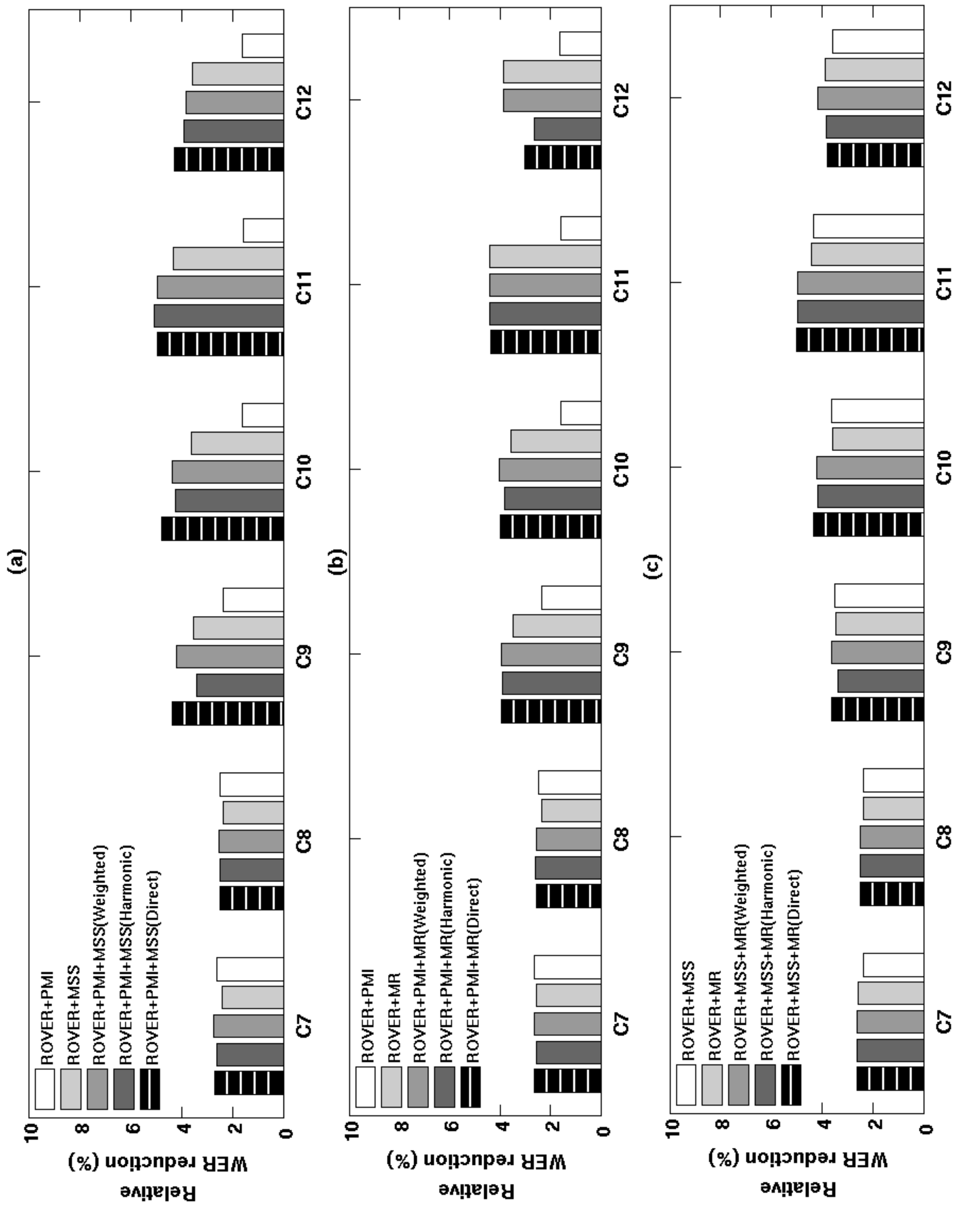

Figure 4.8: Set 1: Relative WER Reduction with 3-decoder Combination

4.7, when PMI- and MSS-based error filtering were combined through the harmonic aggregation scheme. It is worth mentioning, though, that when the baseline WER is low, it is more difficult to achieve a big WER reduction. This explains why the reduction for both C2 and C4 is the lowest, because the baseline WERs were already quite low.

We now discuss the impact of combining error detectors. In most experiments (C1 to C6) and in the three subplots of Figure 4.7, the aggregation of error classifiers yielded a bigger WER reduction compared to the ROVER baseline WERs. The direct and the harmonic score aggregation schemes usually yielded better results. However, in some cases, the difference between all three aggregations is not significant. This is because when the WER reduction is not substantial, the number of errors that are being processed is smaller, and therefore it is difficult to see significant difference between the single error classifier and their combination. To summarize, it is possible to achieve a higher WER reduction, through *c*ROVER, when combining different automatic error detection techniques.

When three decoders' outputs were combined (Figure 4.8), we still see a WER reduction compared to the ROVER baseline, but it is not as substantial as the reduction recorded when two decoders were combined. The highest relative WER reduction has reached 5.04%, specifically in experiment C11 subplot (a) of Figure 4.8, when PMI- and MSS-based error filtering were combined through the direct aggregation scheme. Furthermore, the three aggregation schemes of the error detectors yield approximately the same WER reduction in all experiments.

4.5 *c*ROVER Assessment with Set 2

To validate our findings in Set 1, we have created a larger test set, Set 2. This set is 3.5 times larger than Set 1. A subset of Set 2, specifically 10% of the data, has been used to optimize the filtering threshold K. The neighbourhood has been fixed to twenty words in the left and right context, and the maximum aggregation was used during the computation of the semantic coherence scores. For the LSI-based error filtering, the dimensionality of the feature vectors were set to 300.

The baseline WER for the different decoders' combinations is shown in Table

4.4. Notice how the WER is higher than Set 1, because all the LMs used in this

ID	WER (%)
C1	37.12
C2	36.45
C3	37.32
C4	35.51
C5	32.45
C6	33.10
C7	31.69
C8	31.39
C9	31.45
C10	31.15
C11	31.06
C12	31.30

Table 4.4: Set 2: ROVER's Baseline WER

experiments have a vocabulary size of $64,000$. The LM-98T28 used in Set 1 is different because the training data used to create this language model is not the same one used to train the grammar of Set 1. The combination IDs for Set 2 have been defined previously in Table 4.2. Similar to Set 1, the decoders' combinations C1 to C6 are binary combinations (only two decoders were combined), whereas experiments C7 to C12 are trinary combinations. Furthermore, these baseline WER were lower than the WER of each single decoder; in other words, the ROVER process always yielded a lower WER for two and three decoders' combinations.

Figure 4.9 reports the absolute WER reduction achieved with the *c*ROVER when only two decoders are combined (experiments C1 to C6). The corresponding numerical values can be found in Appendix B. Figure 4.10 reports the absolute WER reduction achieved with the *c*ROVER when all three decoders were combined (experiments C7 to C12). The corresponding numerical values can be found in Appendix B. The plots are similar to the ones in Set 1, in terms of axis representation and subplots division. Similar to the findings with Set 1, *c*ROVER

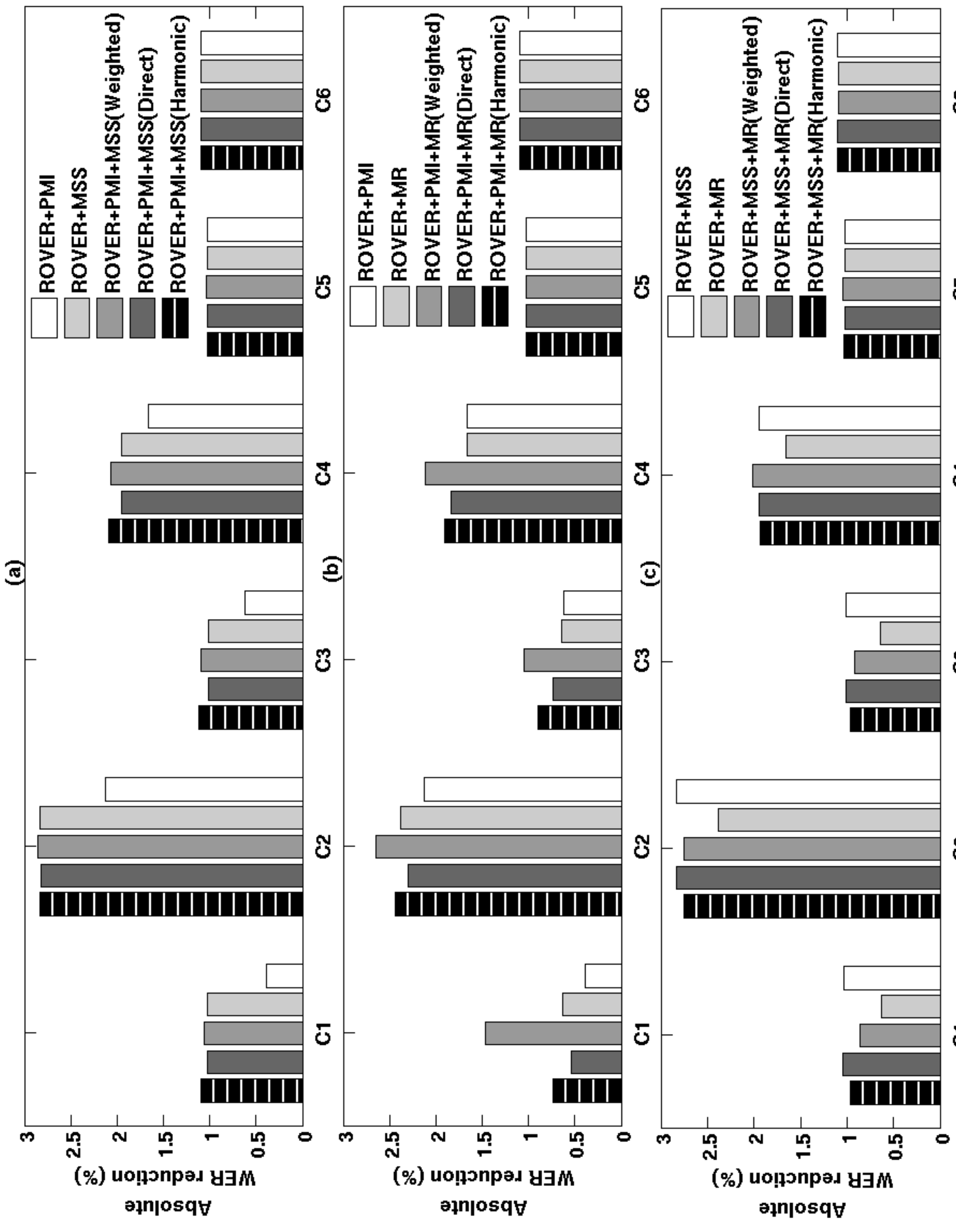

Figure 4.9: Set 2: Absolute WER Reduction with 2-decoder Combination

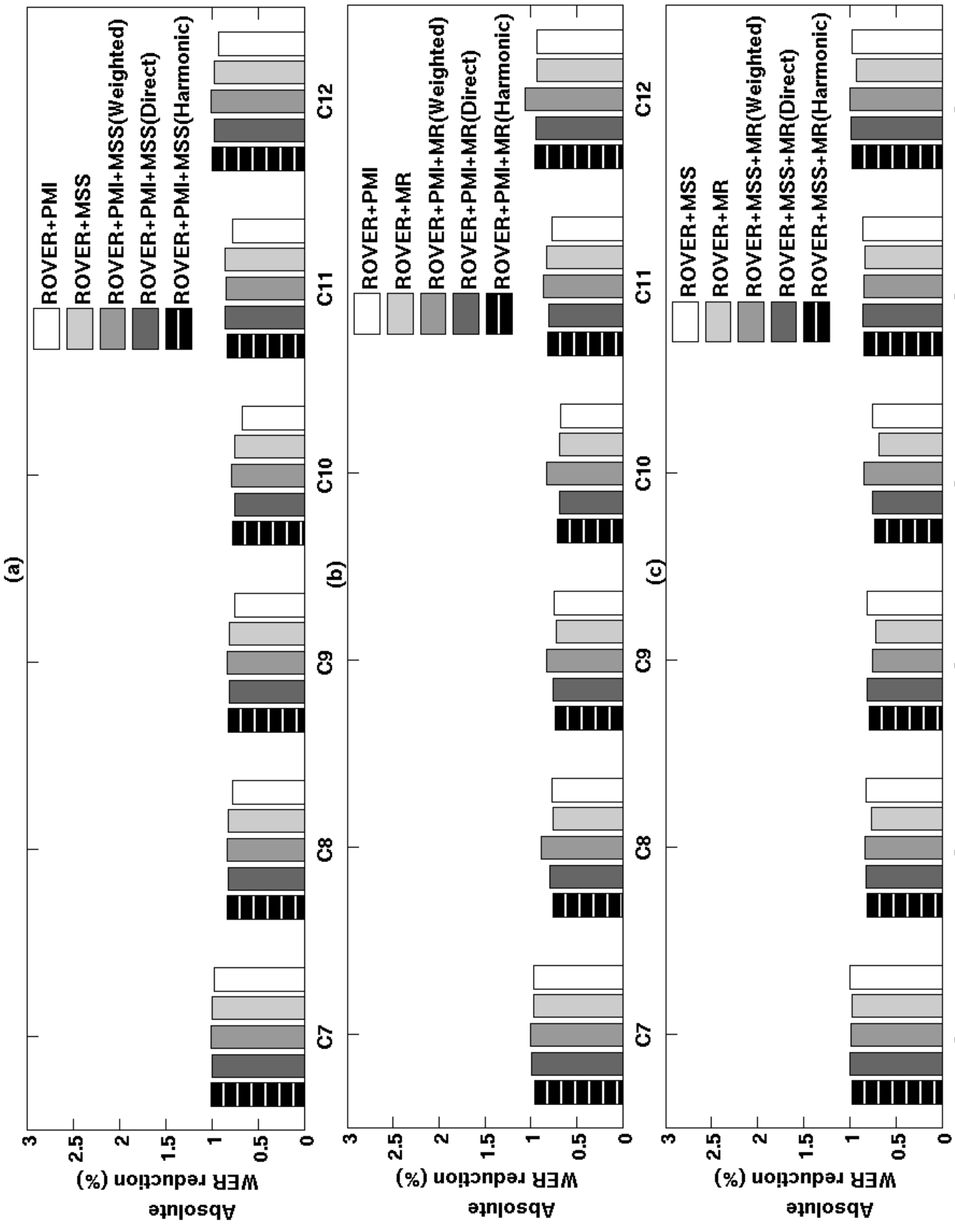

Figure 4.10: Set 2: Absolute WER Reduction with 3-decoder Combination

outperformed the original ROVER in all the experiments. There is some WER reduction in all configurations, with two and three decoders' combinations. To continue our analysis, let us first present the relative WER reductions plots. Figure 4.11 reports the relative WER reduction when the output of two decoders were combined (experiments C1 to C6). The corresponding numerical values can be found in Appendix B. Figure 4.12 reports the relative WER reduction when the output of three decoders were combined (experiments C7 to C12). The corresponding numerical values can be found in Appendix B. When two decoders' outputs are combined (Figure 4.11), the highest relative WER reduction reached 7.82%, specifically in experiment C2 subplot (a), when PMI- and MSS-based error filtering were combined through the weighted aggregation scheme.
In terms of the impact of combining error detectors, the aggregation of error classifiers yielded in some cases a bigger WER reduction compared to the ROVER baseline WERs. However, in most cases, the difference between all three aggregations were not significant.
When three decoders' outputs were combined (Figure 4.12), we still see a WER reduction compared to the ROVER baseline, but it is not as substantial as the reduction recorded when two decoders were combined. The highest relative WER reduction has reached 3.39%, specifically in experiment C12 subplot (b), when PMI- and MR-based error filtering were combined through the weighted aggregation scheme. Furthermore, the three aggregation schemes of the error detectors yield mostly the same WER reduction in all experiments. This is due to the low WER reduction, which lead to a smaller margin to work with in terms of error processing.

cROVER's Summary of Findings:
Based on the findings highlighted in Figures 4.7, 4.8, 4.11, and 4.12, we can draw the following conclusions:

- The cROVER proposed approach always outperformed the original ROVER. In fact, cROVER always achieved some WER reduction in all experiments. This reduction is substantial when the baseline WER is high.

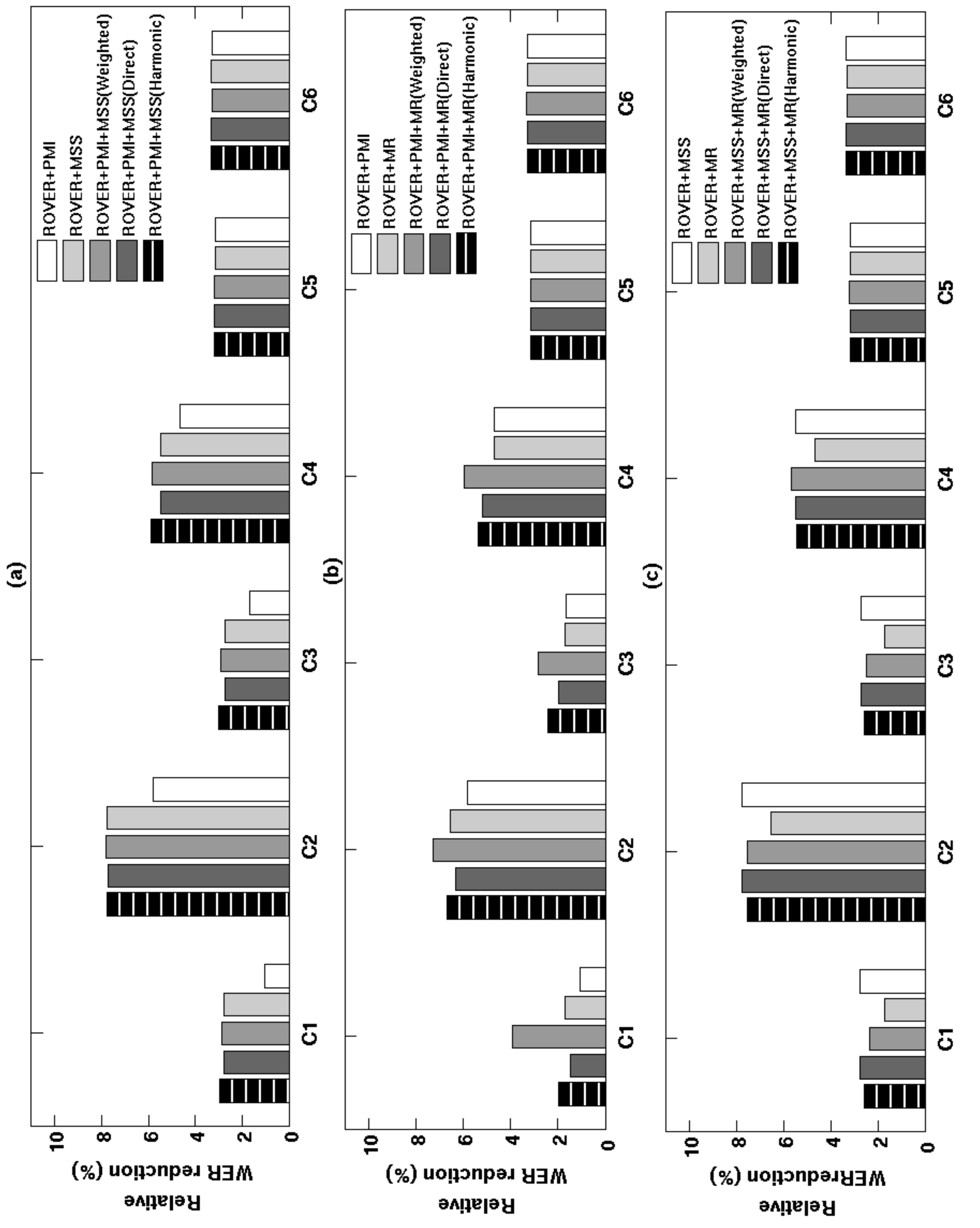

Figure 4.11: Set 2: Relative WER Reduction with 2-decoder Combination

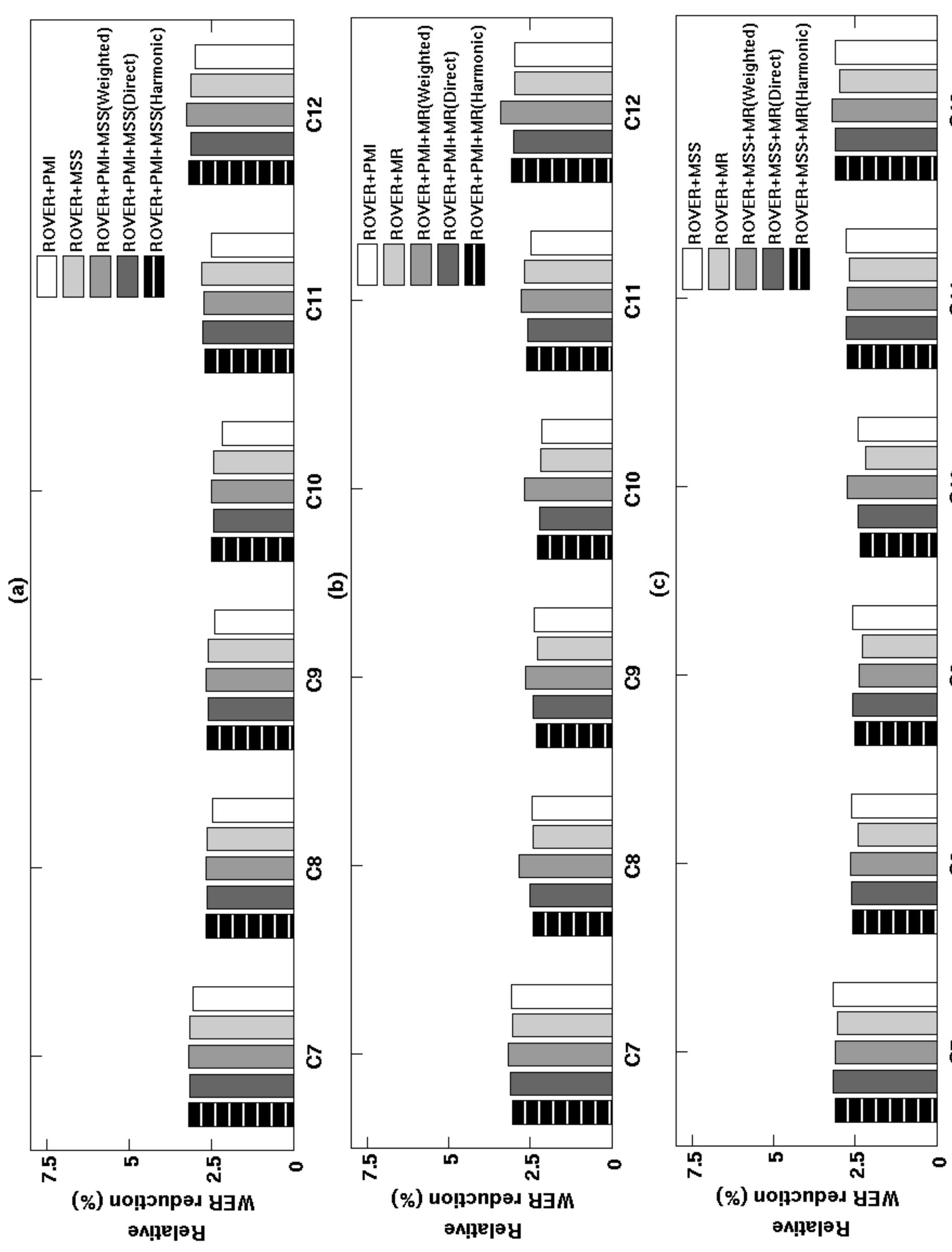

Figure 4.12: Set 2: Relative WER Reduction with 3-decoder Combination

- In some cases, the impact of automatic error detection combination yielded a larger WER reduction. This is mainly true when the WER reduction is larger. However, in general, all three aggregation schemes achieved similar performance in most cases.

4.6 *c*ROVER Computational Requirement

In this section, we describe the CPU time and memory requirements of both ROVER and *c*ROVER in order to highlight the impact of the contextual analysis integration within the original ROVER. The reported figures have been collected while testing the proposed approach with both sets, 1 and 2. Experiments were run on an Intel Core i7 930 at 2.8GHz, with 6GB of RAM (Linux Kernel 2.6.32). Each error detection classifier has its own proprietary knowledge source requirements, and therefore in this section, we will only discuss requirements of the PMI- and the LSI-based error detectors. If *c*ROVER is to be used with another error filtering approach or to be run on different hardware setting, this computational requirement study won't hold. *c*ROVER requirements highly depend on the approach used during the contextual analysis.

4.6.1 Memory Requirements

In terms of memory, there exist two cases: either load all the knowledge sources in the memory (in our case, all bigrams and unigrams from the Google corpus, as well as all the words features extracted from Wikipedia), or load only the required data based on the used language model. The second case indeed significantly reduces the memory required during the error filtering stage. However, it's not always possible to predict all the pairs of words that will be generated by the decoders. Even when we know all the words in the language model, building all the possible pairs from the set of unique words will result in a huge set, very close in size to the whole dataset in the case of large language models. In conclusion, the memory requirement of *c*ROVER is at the worst case equal to the size of the dataset composed of bigrams and unigrams in the case of PMI error filtering, or all the word feature vectors in the case of LSI filtering. These datasets can total in size several gigabytes, which might even introduce a noticeable overhead in the CPU time usage.

4.6.2 CPU Time

Each experiment has been repeated fifty times, and CPU times were then averaged to compensate for the fluctuations in CPU time readings. Furthermore, only the required bigram counts and word features have been loaded into the memory. Table 4.5 reports the CPU time of the original ROVER with both sets, when two and three decoders' outputs are combined. Table 4.6 reports the CPU usage of

Table 4.5: Original ROVER CPU time in seconds

	Set 1	Set 2
2 Decoders	0.25	1.48
3 Decoders	0.56	2.72

ROVER when augmented with the PMI-based error filtering. Experiments were run on Set 2, and the context window size ranged from one single word to 50 words. The CPU usage was averaged on the six different combinations of two decoders, as well as on the six different combinations of three decoders. We can observe that the CPU usage slightly increases with the size of the context window. This can be explained by the fact that more PMI score computation is required at each slot of the WTN when the left and right context is larger. Tables 4.7

Table 4.6: *c*ROVER-PMI: Set 2 Average CPU time in seconds

$N(w)$	1	5	10	15	20	30	40	50
2 dec.	7.622	7.672	7.710	7.739	7.760	7.830	7.870	7.870
3 dec.	10.261	10.298	10.402	10.504	10.701	10.782	10.797	11.793

and 4.8 report the CPU usage of ROVER when augmented with the LSI-based error filtering, with both aggregations (MSS and MR). The word feature vectors dimension ranged between 50 and 500 features. The CPU usage was averaged on the six different combinations of two decoders, as well as on the six different combinations of three decoders. We notice here that the CPU time increases with

the increase of the feature vector length, especially due to the cosine similarity score computations.

Table 4.7: *c*ROVER-MSS: Set 1 Average CPU time in seconds

Dimensions	50	100	200	300	400	500
2 decoders	0.6880	0.6910	0.7366	0.7603	0.7976	0.8103
3 decoders	1.4450	1.4510	1.5516	1.5743	1.5990	1.6266

Table 4.8: *c*ROVER-MR: Set 1 Average CPU time in seconds

Dimensions	50	100	200	300	400	500
2 decoders	0.4936	0.4980	0.5000	0.5060	0.5100	0.5206
3 decoders	0.9330	0.9400	0.9513	0.9633	0.9613	0.9743

Figure 4.13 shows the difference in CPU time between the original ROVER and the context augmented ROVER. It can be concluded that the PMI-based error filtering required much more CPU time than the LSI filtering.
The analysis above shows that there is indeed an overhead when augmenting ROVER with any error filtering stage. However, the research's main goal was not to optimize the *c*ROVER, but rather to propose and assess the approach in terms of WER reduction. Therefore, during the implementation of the system, issues related to optimization and speed were not tackled because they have not been considered as part of the work scope. Obviously, there are several approaches that can be used to optimize the CPU usage as well as the memory requirements. For instance, caching could be used to store probabilities or PMIs of bigrams. The system should first look into the cache for the already-seen word pairs before querying the large knowledge repositories (unigram and bigram counts, or word feature vectors). Furthermore, multi-threading design could guarantee, at some level, to a faster parallel processing of several slots in the word transition network. This approach may even be used to speed up the original ROVER as well.

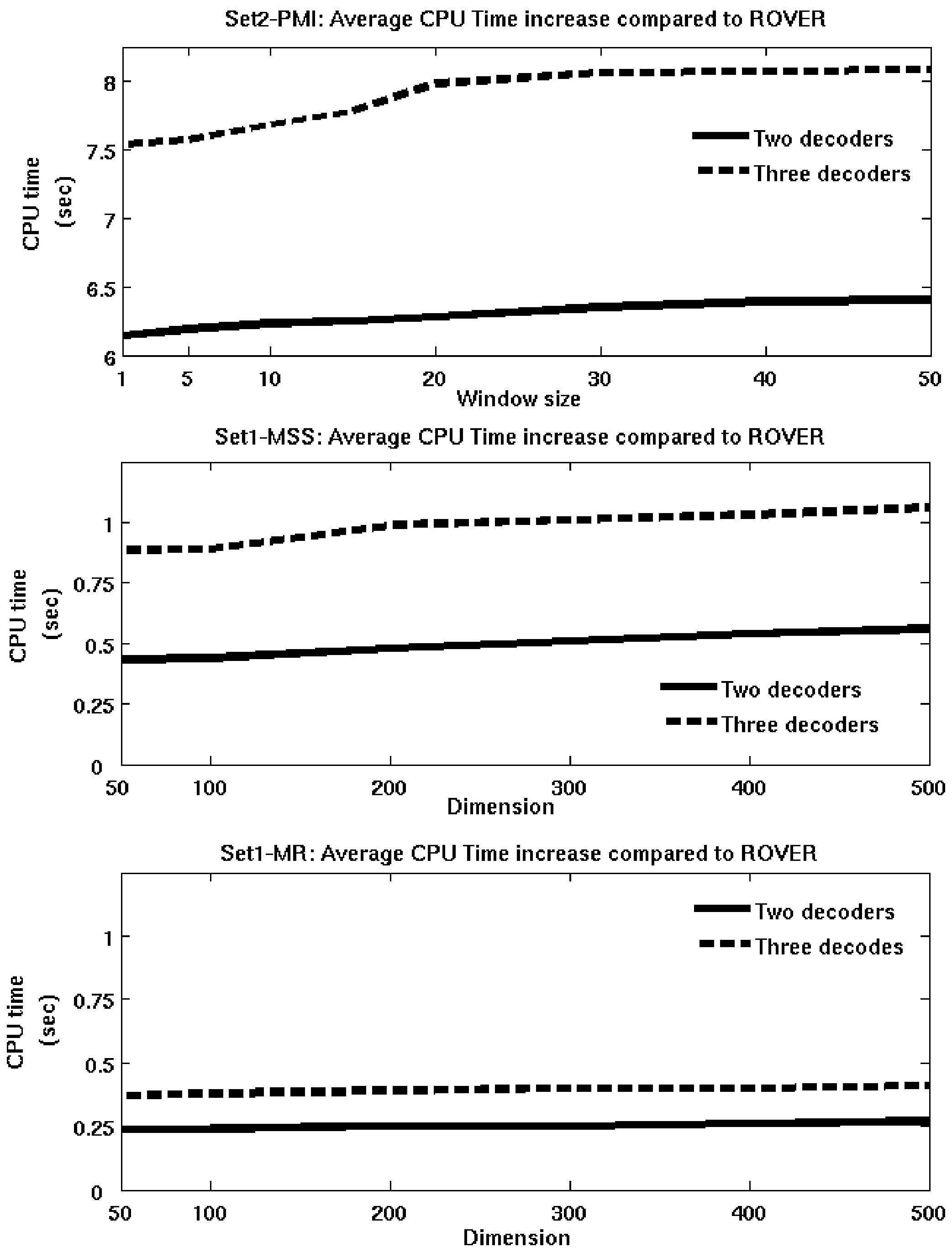

Figure 4.13: CPU Time increase compared to ROVER

4.7 New Voting Mechanisms for ROVER

In this section, we study the performance of the proposed confidence-based voting scheme compared to ROVER's original voting schemes. It is worth mentioning that only the frequency based voting mechanism was used as a baseline to assess the new voting algorithm. This is due to the same reasons explained earlier, specifically the lack of reliable decoder confidence scores. Similarly to *c*ROVER, the novel voting scheme has been evaluated using LSI- and PMI-based error detection techniques.

4.7.1 Voting Assessment with Set 1

A subset of the first testing set has been used to optimize the β parameter in Equation 3.13. The PMI context window has been set to twenty words. For the LSI- filtering, the dimensionality of the feature vectors has been set to 300. The baseline WER for the original ROVER's frequency-based voting is shown in Table 4.3. The combination IDs for Set 1 have been previously defined in Table 4.2. Figure 4.14 reports the absolute WER reduction achieved with ROVER when the new confidence-based scoring is used, and when only two decoders were combined (experiments C1 to C6). The corresponding numerical values can be found in Appendix C. For each experiment, we have reported the WER reduction when scores from single error filters, as well as a weighted combination, and a harmonic combination of these confidence scores, were used in Equation 3.13. Figure 4.15 reports the absolute WER reduction achieved with ROVER when the new confidence-based scoring is used, and when three decoders were combined (experiments C7 to C12). The corresponding numerical values can be found in Appendix C. The first conclusion we can draw so far, is that the new voting scheme is not able to outperform the original ROVER voting scheme in all experiments. In Figure 4.14 for example, the ROVER voting outperformed the new voting scheme in experiments C2 and C4. As discussed earlier, the analysis of the absolute WER plots is not straightforward because we have to relate the WER reduction percentages to the WER baseline reported in Table 4.3; we have therefore plotted the relative WER reduction instead. Figure 4.16 reports the relative WER reduction achieved by the new voting scheme when outputs of two

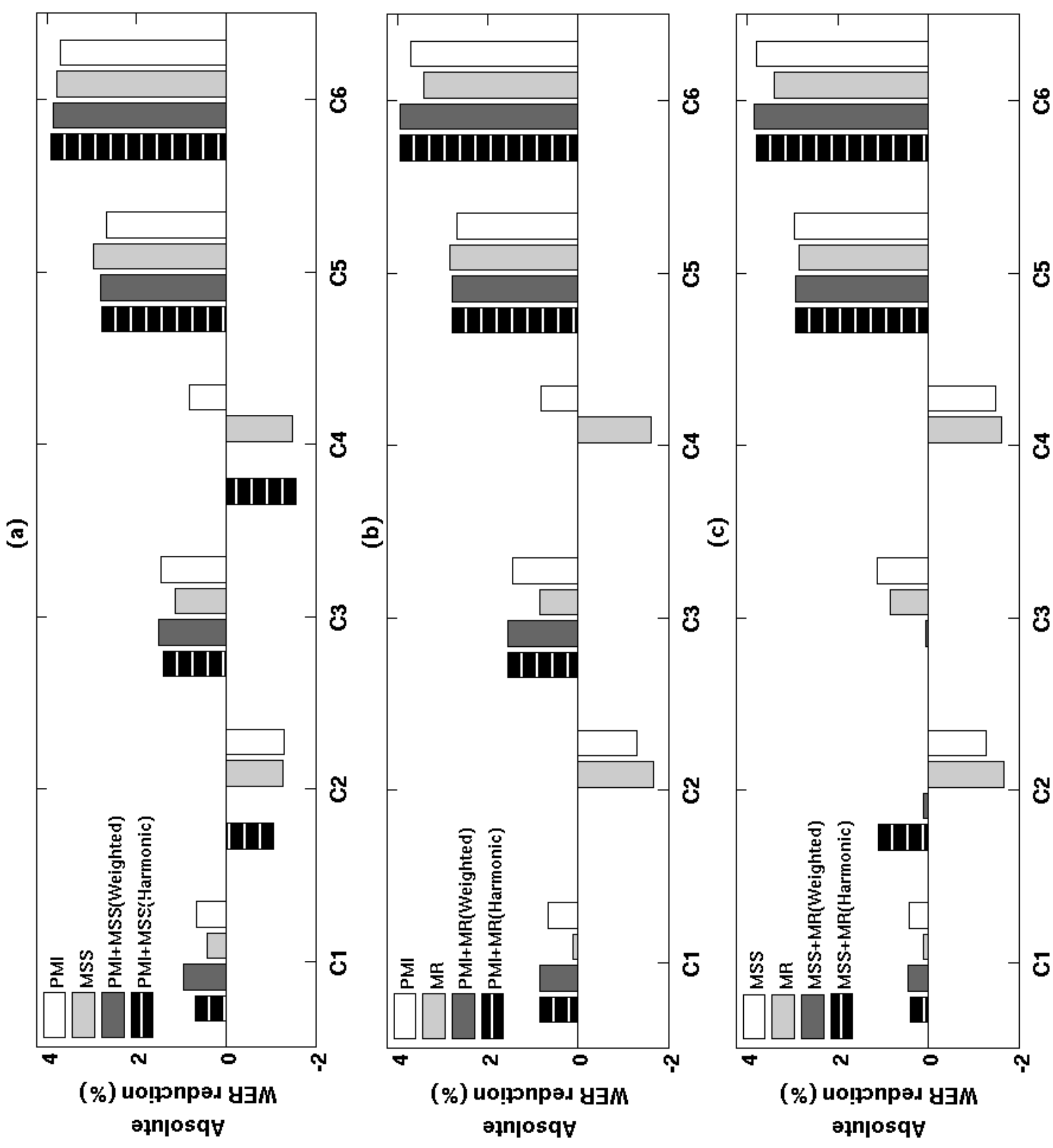

Figure 4.14: Set 1: Absolute WER Reduction with 2-decoder Combination

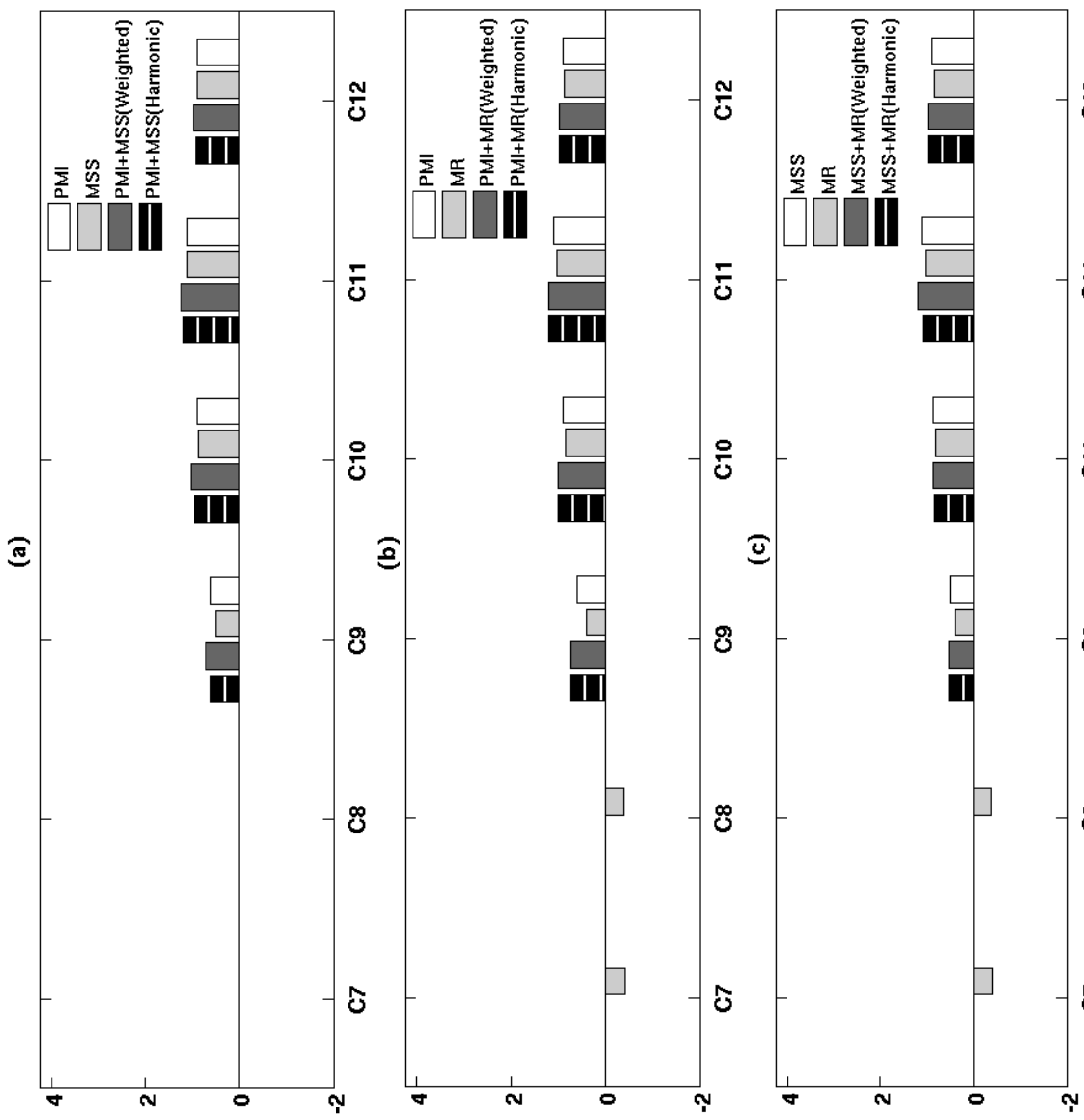

Figure 4.15: Set 1: Absolute WER Reduction with 3-decoder Combination

decoders are combined. The corresponding numerical values can be found in Appendix C. Figure 4.17 reports the relative WER reduction with the new voting scheme when outputs from the three decoders are combined. The highest relative WER achieved for two decoders is 16.16%, specifically with experiment C6, when PMI and MR scores were combined using the weighted aggregation scheme (subplot (b)). However, in some cases, the new voting scheme increased the relative WER by almost 8%. When three decoders were combined together, the WER reduction didn't exceed 7%. In some experiments, there has been no change at all compared to the original ROVER scheme, and in some others, a slight increase in WER has been recorded. It is also worth mentioning that the combination of error detection techniques, through weighted and harmonic aggregations, has not achieved a significant WER reduction compared to individual error filters. The new voting scheme outperformed the cROVER in some cases, but this WER reduction is not consistent throughout the whole set of experiments. In other words, in some experiments, the achieved WER reduction exceeds cROVER reductions, but in several cases, the WER increased. The cROVER approach has always outperformed the original ROVER, in all experiments and in both testing sets. This can be explained by the fact that confidence scores from the different error filters are not reliable. In other words, correct words can have very low confidence scores, while, erroneous outputs might get high scores from the different error detectors. This is in concordance with the fact that the current error filtering procedures suffer from low recall and precision rates.

The fact that the cROVER approach is relying on these confidence scores, and is still able to outperform the new voting algorithm can be explained by the usage of the threshold. In fact, the pruning threshold K, is optimized during a training stage. Therefore, the resulting threshold takes into account this fluctuation in the confidence scores. The choice of K encapsulates the information related to the confidence scores coming from different error filters. This information is missing in the new voting scheme because of the direct use of raw scores from error classifiers.

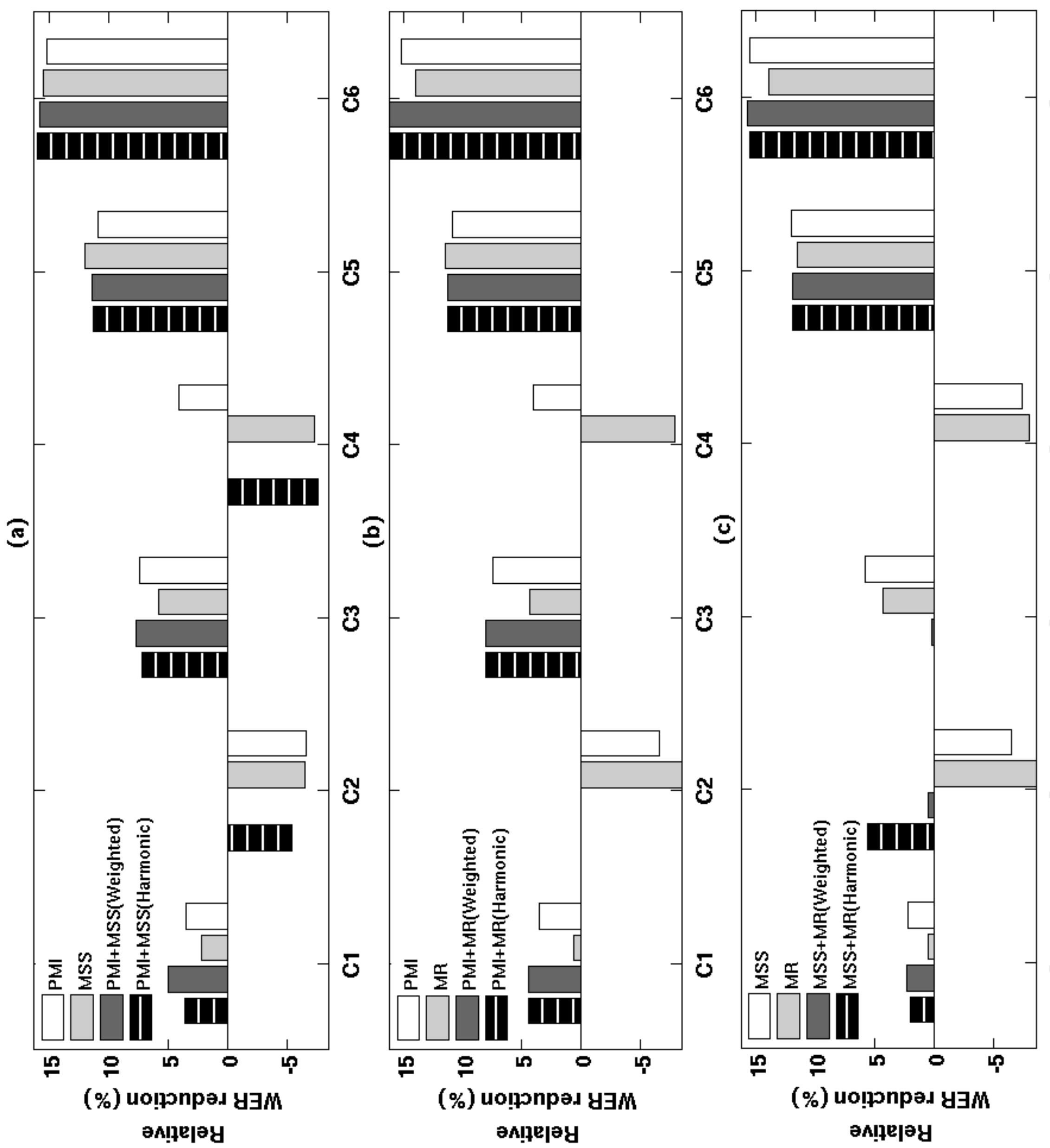

Figure 4.16: Set 1: Relative WER Reduction with 2-decoder Combination

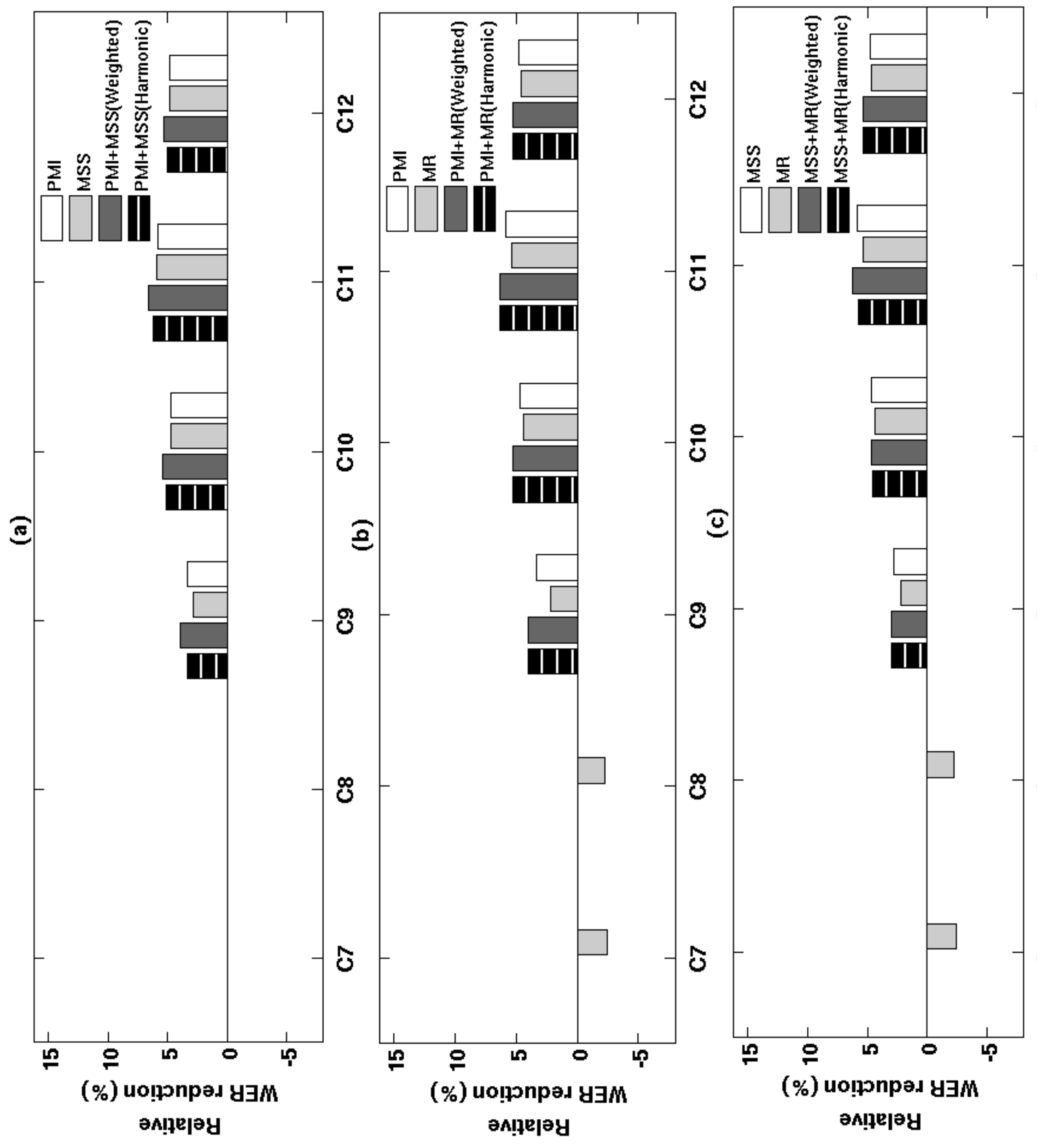

Figure 4.17: Set 1: Relative WER Reduction with 3-decoder Combination

4.7.2 Voting Assessment with Set 2

The confidence-based voting scheme has also been assessed on the second test set. 10% of the set has been used to optimize the β weighting factor of Equation 3.13. The same configurations as for set 1 have been used for the PMI and LSI error detectors. The baseline WER for the ROVER system with the frequency-based voting scheme was reported in Table 4.4. The experiment IDs for Set 2 are defined in Table 4.2. Figure 4.18 reports the absolute WER reduction when ROVER is used with the new voting scheme, and when two decoders' outputs are combined. The corresponding numerical values can be found in Appendix D. Figure 4.19 reports the absolute WER reduction achieved with ROVER when the new confidence-based scoring is used, and when three decoders were combined (experiments C7 to C12). The corresponding numerical values can be found in Appendix D. Similarly to our findings with Set 1, the new scoring mechanism does not always outperform the original ROVER scoring system. In some of the experiments (specifically combinations C5 and C6, in Figure 4.18), the original ROVER voting outperforms our proposed one. In Figure 4.19, the new voting procedure outperforms the original ROVER, but in most of the experiments, the reduction in WER is small. To continue with the analysis of the results, we report the relative WER reduction. Figure 4.20 reports the relative WER reduction achieved by the new voting scheme when outputs of two decoders are combined. The corresponding numerical values can be found in Appendix D. Figure 4.21 reports the relative WER reduction with the new voting scheme when outputs from the three decoders are combined. The highest relative WER reduction has been achieved when two decoders were combined, Figure 4.20, and the MSS and PMI scores were aggregated using harmonic combination scheme, subplot (a), experiment C4. The reduction reached 9.77%. Similarly to Set 1, the difference in terms of WER, between using scores from individual error detectors, and their aggregations, is not significant. WER reduction when three decoders were combined is small.

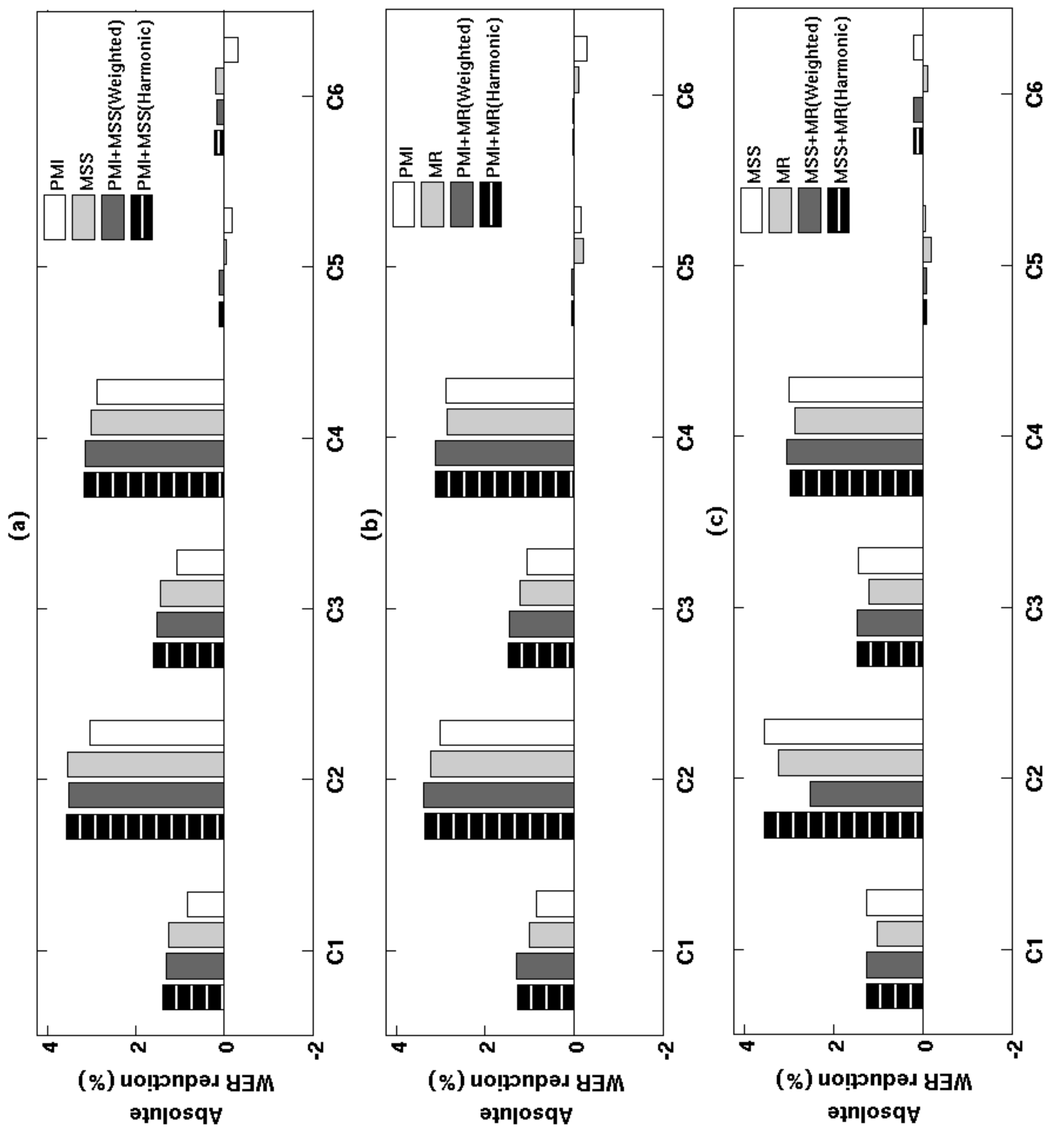

Figure 4.18: Set 2: Absolute WER Reduction with 2-decoder Combination

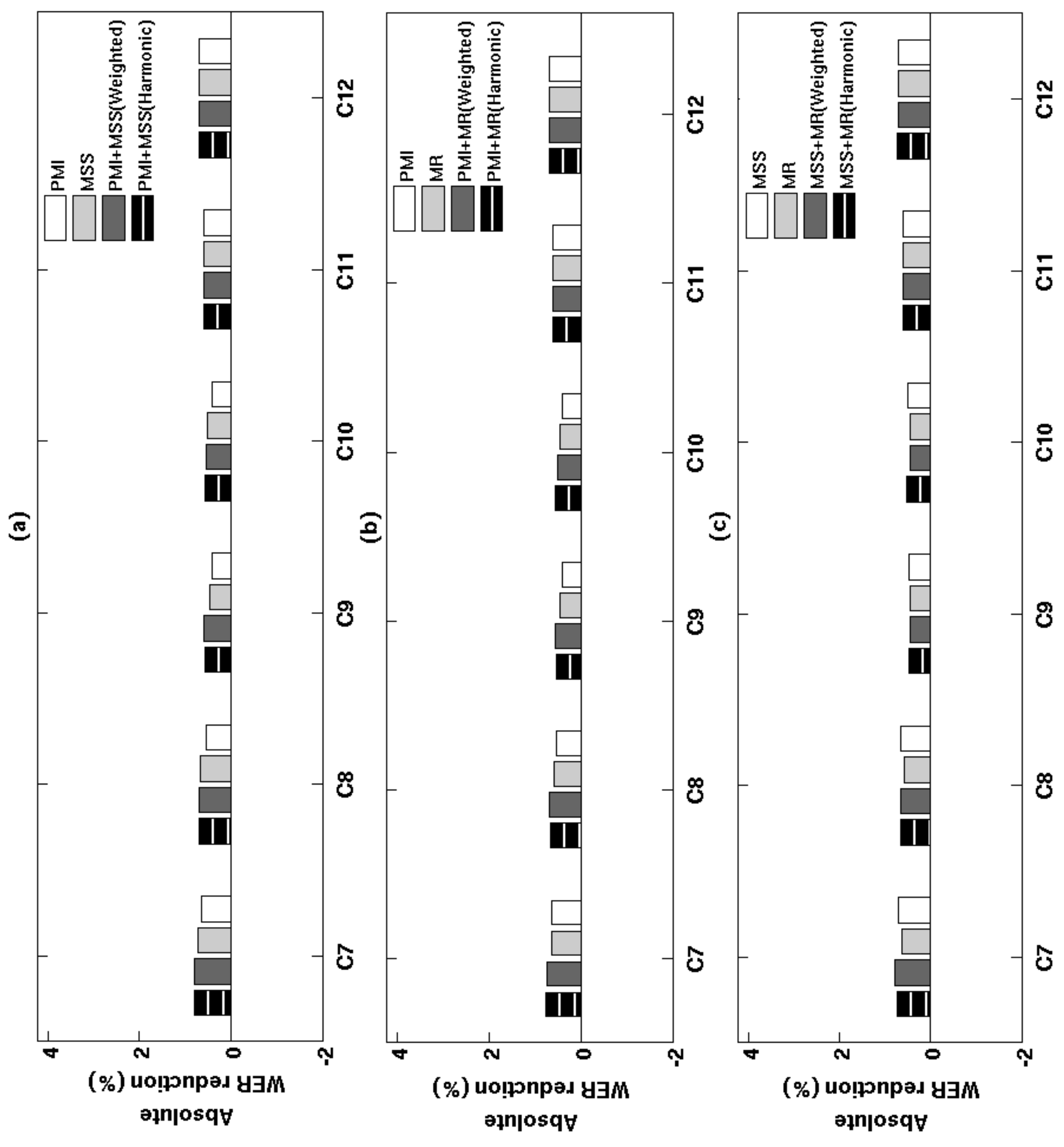

Figure 4.19: Set 2: Absolute WER Reduction with 3-decoder Combination

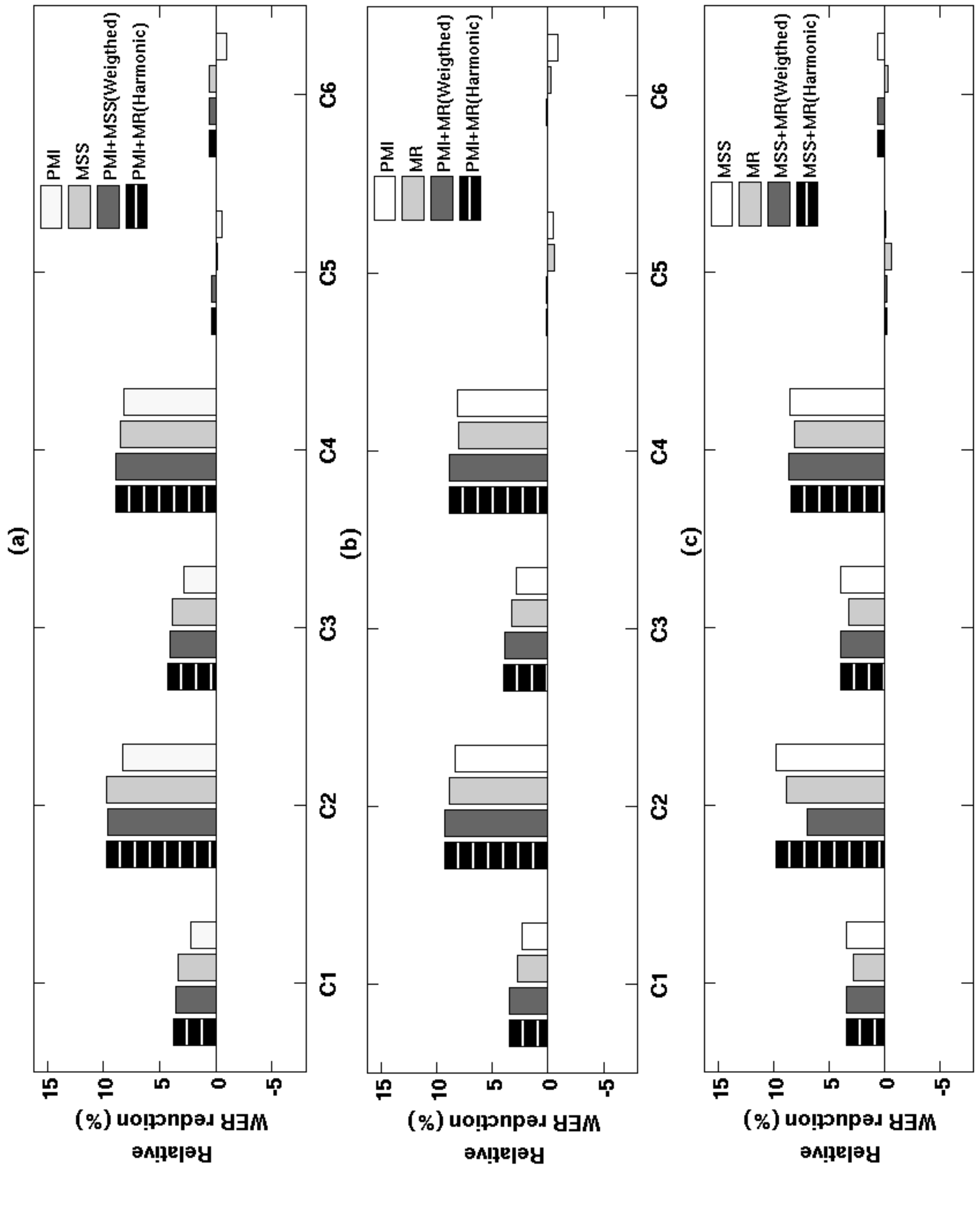

Figure 4.20: Set 2: Relative WER Reduction with 2-decoder Combination

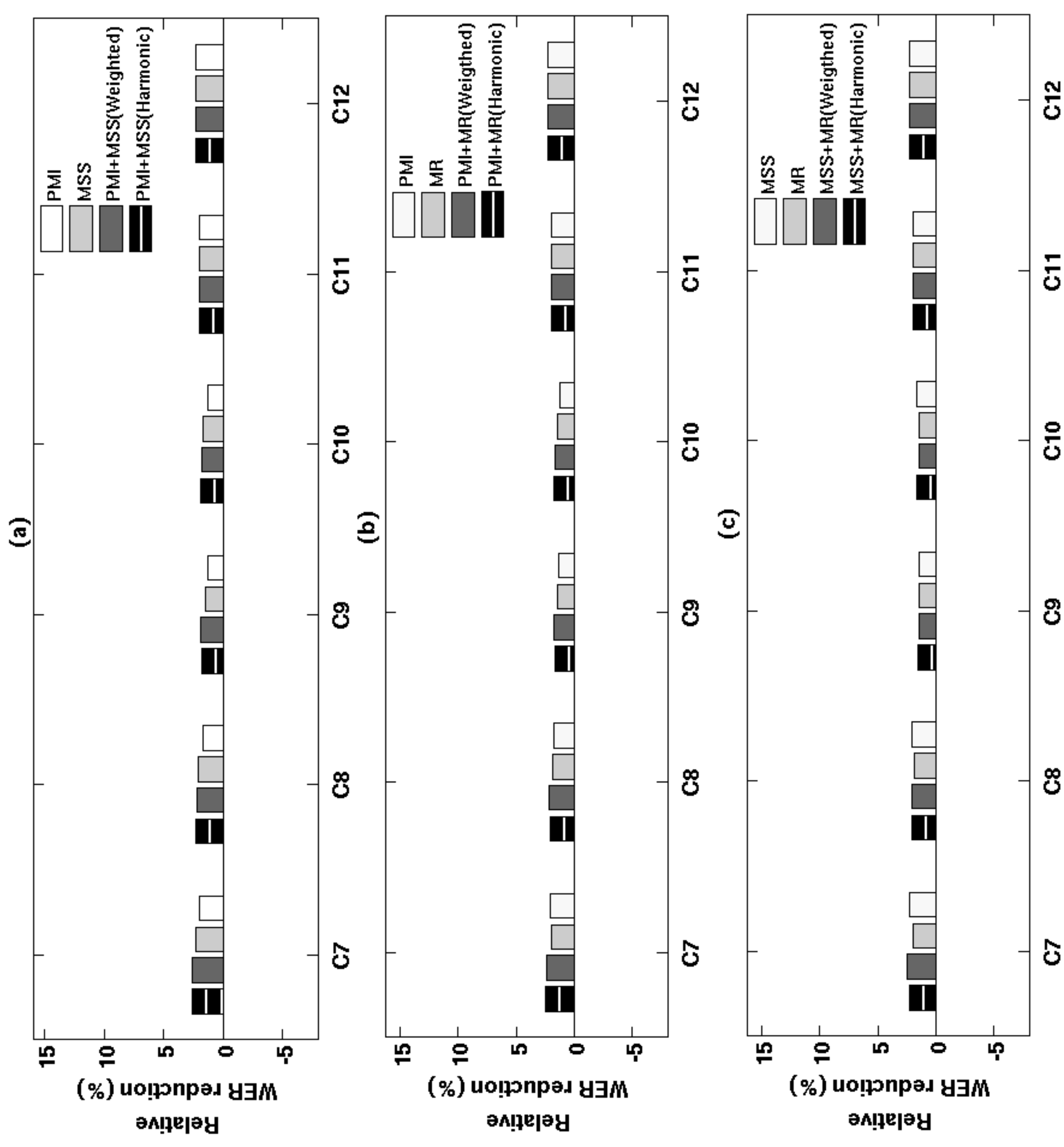

Figure 4.21: Set 2: Relative WER Reduction with 3-decoder Combination

New Voting Scheme Summary of Findings:

- Unlike the *c*ROVER approach, the new voting scheme does not guarantee a lower WER in all the experiments. In some combinations, the original ROVER voting outperformed our confidence-based voting scheme. The *c*ROVER approach achieves consistent WER reduction through all experiments with both testing sets.
- With the proposed confidence-based voting, the impact of aggregating confidence scores from several error filters is not significant.
- In some experiments, the new voting scheme achieved a lower WER than *c*ROVER.
- Similar to *c*ROVER, reducing the WER when three decoders are combined is more difficult.
- The voting scheme is vulnerable due to the use of unreliable confidence scores from error classifiers. *c*ROVER seems to be more robust, because of the use of a decision threshold, optimized through a training stage. The choice of this parameter was revealed to be crucial, because it encapsulates the fluctuations in confidence scores from several error decoders. These scores' fluctuations are certainly caused by the low recall and precision rates of the classifiers, where correct words have very low confidences, whereas erroneous tokens have high scores.

4.8 Conclusion

This chapter has presented a set of experiments carried out to evaluate the *c*ROVER proposed approach, as well as the confidence-based voting scheme. The error detection techniques were first evaluated, followed by the assessment of *c*ROVER on two datasets. The proposed approach achieved nearly 10% in relative WER reduction, when two decoders were combined, and up to a 5% reduction when three decoders were combined. *c*ROVER consistently outperformed

the original ROVER technique, in all the experiments and with both datasets. In some cases, the combination of several error detection techniques has led to an even lower WER.
An analysis of the hardware requirement of *c*ROVER has been presented. Augmenting ROVER with automatic error detection mechanisms was revealed to add an overhead in processing and memory requirements. It is worth repeating that the research work's main goals did not involve optimization of the proposed approach, in terms of CPU and memory consumption.
The chapter concluded with the assessment of the proposed scoring scheme. Unlike the *c*ROVER, the improvment in terms of WER reduction was not consistent. In some cases, the reduction reached up to 16% in terms of relative WER, beating the *c*ROVER's lowest WER, but in other cases, the WER has not changed and even increased in a few experiments.

Chapter 5

Conclusion

Several objectives were set up for the research presented in this thesis.

Objective 1 A novel framework to improve on the performance of the original ROVER system. The framework is generic in terms of recognizers, and is application domain independent.

Objective 2 Scalable framework in terms of the number of decoders to be combined together.

Objective 3 Implementation and assessment of the proposed framework against existing approaches.

Objective 4 A novel voting scheme for the ROVER procedure

The research work presented in this thesis fully observes these objectives. This thesis presented an approach, called *c*ROVER, where the *c* stands for context. The idea is to embed a contextual analysis within the ROVER procedure in order to eliminate as many errors as possible. The augmented ROVER with error filtering works as follows: first, the WTN is built from the outputs of different decoders. Second, an error filtering stage is introduced on the newly built WTN to eliminate erroneous words. Finally, the voting algorithm browses the newly updated WTN to select the winner word at each slot of the WTN. The error filtering has not been applied on the whole transcription to minimize falsely tagging correctly transcribed words as errors. Therefore, upon building the WTN, the error detector is only applied on to those slots with discrepancies between the different speech decoders. To tackle the problem of the error detection's poor performance in terms of recall and precision, the authors combined different error detection approaches, in the hope that the new technique achieves higher recall, without degrading the precision rate. Three different combination schemes were presented, namely direct combination, and score aggregation through harmonic and weighted averaging.

The *c*ROVER approach is a novel approach, and to the author's knowledge there has been no similar work. Furthermore, the proposed method is recognizer-independent, and most importantly domain-independent. This is a crucial characteristic that we worked hard to preserve. In fact, in the speech recognition field, it is always difficult to propose a novel framework that is generic, especially in terms of the domain. Usually, there are constraints and requirements related to the domain or to the nature of the application. During our research, we tried several approaches and tools to tune and further improve the proposed approach, but this has come at the cost of sacrificing the recognizer and domain-independent feature. The author made this as one of the fundamental objectives of the thesis, to preserve this feature. As far as the scalability is concerned, there has been no alteration to the original WTN building stage, and therefore *c*ROVER is as scalable as the original ROVER. Obviously there is the cost of the error filtering stage, which has been discussed at the end of the thesis. The assessment of the proposed framework, has been carried out through the use of the widely-used probabilistic error detection approaches, namely PMI- and LSI-based techniques. Both techniques were combined and integrated within ROVER. Exhaustive experimentation showed that *c*ROVER was able to achieve in some scenarios up to 9.57% in terms of relative WER reduction, when two decoders are combined, and up to 5.04% when three decoders are combined together. Experiments also showed that it is possible to reduce the WER even further when error detection techniques are combined. However, based on the experiments carried out on two separate testing sets, there has been no significant difference between the different aggregation schemes.

An analysis of the computational requirements of the *c*ROVER approach has been presented. Results showed that the integration of a contextual analysis

within the ROVER process caused an overhead in terms of CPU and memory usage. However, since the main goal of this work did not involve any numerical or implementation optimization of the *c*ROVER approach, the authors are convinced that this overhead can be reduced drastically once factors pertaining to reducing the hardware requirements, are taken into consideration.
A novel voting scheme has been proposed. The new confidence-based scoring involves confidences from the contextual analysis, and aggregates them in a weighted manner with the original ROVER scoring. Experiments have shown that it is possible to outperform the original ROVER's voting schemes, reaching up to 16% of relative WER reduction. However, this improvment has not been consistent in all the experiments, unlike the *c*ROVER approach, which outperformed the ROVER technique in all configurations and with both testing sets. The author concludes that the new voting scheme is vulnerable because it relies on the contextual analysis' confidence scores.

5.1 Future Directions

The work in this dissertation paves the way for the usage of more error detection techniques in speech transcription. We have achieved notable performance improvment when integrating an error filtering stage within the original ROVER procedure. However, the current state of the art in this field is far from commercial usefulness. Having said that, we feel that combining several error classifiers is a promising direction towards improving the recall and precision rates. Three simple schemes have been attempted in this thesis. Experiments showed it is possible to outperform individual error detectors by aggregating them. Therefore, a more thorough study is needed to identify different aggregation schemes. In this same direction of thought, it might be more adequate to rely on machine learning tools towards a more intelligent combination of error classifiers. Neural networks or support vectors machines can be used to learn which error detector is efficient for different types of tokens (stopwords, verbs, nouns, etc).
In this thesis, we did not make suggestions as to how the proposed approach can generalize to non-confidence-based error detectors, such as pattern matching. The main focus was instead on finding techniques which rely on the thresholding

of a confidence score to decide whether or not a given token is an error. The simplest way of doing this is to rely on the direct combination scheme we proposed in this work. If at least one of the classifiers tags the given token as an error, the composite decision would be then to classify the word as erroneous output. However, it is crucial to investigate novel ways to do hybrid combination between probabilistic and non-probabilistic techniques. This can be interesting especially in the limited domain and vocabulary applications, where non-probabilistic approaches (mainly pattern matching techniques) perform quite well given the fact that all possible errors can be exhaustively recorded due to grammar constraints and limited vocabulary size.

As far as the ROVER voting mechanisms are concerned, we also see it is important to investigate more voting algorithms. The fact that the original ROVER voting schemes rely on the speech recognizers' confidence score makes the whole procedure vulnerable. Issues in the confidence measures of speech recognizers are far from being solved. In our experiments, for example, we were quite unable to use them as they turned out to be all equal to one (especially the scores coming from the most widely-deployed and trusted commercial engine, Nuance N9). Our proposed voting scheme that relied on confidence scores collected from error detection techniques, proved to be not as robust as the original ROVER voting, even though in most cases, it outperformed it and yielded a lower WER. A more intelligent voting might need to be investigated. The use of machine learning tools might be beneficial to decide, at each slot of the WTN, the winner word. The idea is to come up with a set of features that are reliable enough to lead to a decent selection of the winner token. A sample set of features, can for example involve the frequency of occurrence, the word type (stopword, verb, noums, etc), decoders' and error classifiers' confidence scores, etc.

As discussed in the beginning of this thesis, researchers are tackling the problem of LVCSR from several aspects, including the front end signal processing, the language and acoustic modeling, the search, and the post-processing field, including decoders' output combination and automatic error spotting and correction. The current status of LVCSR performance seems to have reached a plateau in terms of WER, and the next big thing in this area would be to achieve a WER reduction between 10% and 20%. We strongly believe that this can only be possible if all the advances at different levels (front end, post-processing, language and acoustic modeling) are combined together in a cooperative manner, in order to take advantage of all the power of the individual techniques and compensate for their shortcomings.

References

[1] N. M. I. Group, “The history of automatic speech recognition evaluations at NIST,” May 2009. [Online]. Available: http://www.itl.nist.gov/iad/mig/publications/ASRhistory/index.html

[2] G. Zweig and M. Picheny, “Advances in large vocabulary continuous speech recognition,” IBM Watson Research Center, Yorktown Heights, NY, Tech. Rep., 2004.

[3] Y. Gao, B. Ramabhadran, J. Chen, and M. Picheny, “Innovative approaches for large vocabulary name recognition,” in *International Conference on Accoustics, Speech and Signal Processing (ICASSP)*, 2001, pp. 53–56.

[4] F. Bechet, R. De Mori, and G. Subsol, “Very large vocabulary proper name recognition for directory assistance,” in *IEEE Automatic Speech Recognition and Understanding Workshop*, December 2001.

[5] N. Wang, P. Huang, and S. Jia-Lin, “Chinese large-vocabulary name recognition system using character description and syllable spelling recognition,” in *International Symposium on Chinese Spoken Language Processing*, 2005, pp. 17–20.

[6] E. Hakan, “Speech recognition for a travel reservation system,” in *International Conference on Aritificial Intelligence*, 2001.

[7] B. Pellom, W. Ward, R. Cole, J. Hansen, and K. Hacioglu, “University of colorado dialog systems for travel and navigation,” in *Workshop on Human Language Technology and Knowledge Managment*, 2001.

[8] Y. Dong, Y. Ju, Y. Wang., G. Zweig, and A. Alex, "Automated directory assistance system - from theory to practice," in *Interspeech*, 2007, pp. 2709–2712.

[9] J. Olsen, C. Yang, X. Yang, and G. Ding, "A decoder for large vocabulary continuous short message dictation on embedded devices," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2008, pp. 4337–4340.

[10] H. Xuechen, "A web-based intelligent tutoring system for english dictation," in *International Conference on Artificial Intelligence and Computational Intelligence (AICI)*, 2009, pp. 583–586.

[11] I. Varga, S. Aalburg, B. Andrassy, S. Astrov, J. Bauer, C. Beaugeant, C. Geissler, and H. Hoge, "ASR in mobile phones - an industrial approach," *IEEE Transaction on Speech and Audio Processing*, vol. 10, no. 8, p. 562, Feb. 2003.

[12] L. M. Lee and Y. U. Kalsom, "Hands-free messaging application (iSay-SMS): A proposed framework," in *International Conference on Science and Social Research (CSSR)*, December 2010, pp. 1126–1131.

[13] J. Cohen, "Embedded speech recognition applications in mobile phones: Status, trends, and challenges," in *IEEE Conference on Acoustics, Speech and Signal Processing*, april 2008, pp. 5352–5355.

[14] S. Chen and Y. Wei, "A study on speech control interface for vehicle on-board diagnostic system," in *Fourth International Conference on Genetic and Evolutionary Computing (ICGEC)*, December 2010, pp. 614–617.

[15] P. Angkititrakul, M. Petracca, A. Sathyanarayana, and J. Hansen, "UT-Drive: Driver behavior and speech interactive systems for in-vehicle environments," in *IEEE Intelligent Vehicles Symposium*, June 2007, pp. 566–569.

[16] T. Hain, V. Wan, L. Burget, M. Karafiat, J. Dines, J. Vepa, G. Garau, and M. Lincoln, "The AMI system for the transcription of speech in meetings," in

IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), vol. 4, april 2007, pp. 357–360.

[17] D. Kim, G. Evermann, T. Hain, D. Mrva, S. Tranter, L. Wang, and P. Woodland, “Recent advances in broadcast news transcription,” in *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, November 2003, pp. 105–110.

[18] C. Martins, A. Teixeira, and J. Neto, “Dynamic language modeling for a daily broadcast news transcription system,” in *IEEE Workshop on Automatic Speech Recognition Understanding (ASRU)*, December 2007, pp. 165–170.

[19] R. Schluter and H. Ney, “Using phase spectrum information for improved speech recognitionperformance,” in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 1, 2001.

[20] A. Zolnay, D. Kocharov, R. Schluter, and H. Ney, “Acoustic feature combination for robust speech recognition,” in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 1, 18-23 2005, pp. 457–460.

[21] ——, “Using multiple acoustic feature sets for speech recognition,” *Speech Communication*, vol. 49, no. 6, pp. 514–525, 2007.

[22] H. Qiang and S. Cox, “Hierarchical language modeling for audio events detection in a sports game,” in *IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)*, March 2010, pp. 2286–2289.

[23] H. Songfang and S. Renals, “Hierarchical bayesian language models for conversational speech recognition,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 8, pp. 1941–1954, November 2010.

[24] Y. Rhee, J. S. Yoon, and H. Kim, “Acoustic model adaptation based on pronunciation variability analysis for non-native speech recognition,” in *IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)*, May 2006, pp. 1–6.

[25] J. Fiscus, “A post-processing system to yield reduced word error rates: Recogniser output voting error reduction (rover),” in *Proceedings 1997 IEEE Workshop on Automatic Speech Recognition and Understanding*, Santa Barbara, CA, 1997, pp. 347–352.

[26] D. Jurafsky and J. Martin, *An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition.* Prentice-Hall Inc,, 2000.

[27] A. Acero, L. Deng, and M. Seltzer, “Acoustic modeling,” Online, 2009, http://research.microsoft.com/en-us/projects/acoustic-modeling/.

[28] M. I. Group, “NIST benchmark tests,” Online, 2009. [Online]. Available: http://www.itl.nist.gov/iad/mig//tests/

[29] L. Lamel, “Large vocabulary continous speech recognition: from laboratory systems towards real-world applications,” *IEEE Transactions on Spoken Language Systems*, vol. 200, no. 1, 1996.

[30] K. Abida and F. Karray, “Systems combination in large vocabulary continuous speech recognition,” in *IEEE International Conference on Autonomous and Intelligent Systems (AIS)*, June 2010, pp. 1–6.

[31] N. M. I. Group, “1997 HUB5 english evaluation,” LDC, Philadelphia, 1997. [Online]. Available: http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2002S23

[32] B. Hoffmeister, D. Hillard, S. Hahn, R. Schluter, M. Ostendorf, and H. Ney, “Cross-site and intra-site ASR system combination: comparisons on lattice and 1-best methods,” in *IEEE International Conference on Acoustics, Speech and Signal Processing, 2007. ICASSP 2007*, vol. 4, 2007, pp. 1145–1148.

[33] B. Hoffmeister, T. Klein, R. Schlüter, and H. Ney, “Frame based system combination and a comparison with weighted ROVER and CNC,” in *Ninth International Conference on Spoken Language Processing.* ISCA, 2006.

[34] L. Mangu, E. Brill, and A. Stolcke, "Finding consensus in speech recognition: word error minimization and other applications of confusion networks." *Computer Speech and Language*, vol. 14, no. 4, pp. 373–400, 2000.

[35] G. Evermann and P. Woodland, "Posterior probability decoding, confidence estimation and system combination," in *Proc. Speech Transcription Workshop*, vol. 27, 2000.

[36] B. Hoffmeister, R. Schluter, and H. Ney, "icnc and irover: The limits of improving system combination with classification?" in *International Speech Communication Association - Interspeech08*, Sep. 2008.

[37] F. Wessel, R. Schlüter, and H. Ney, "Explicit word error minimization using word hypothesis posterior probabilities," in *IEEE International Conference on Acoustics, Speech and Signal Processing, 2001. ICASSP 2001*, vol. 1, May 2001, pp. 33–36.

[38] R. Zhang and A. I. Rudnicky, "Investigations of issues for using multiple acoustic models to improve continuous speech recognition," in *Ninth International Conference on Spoken Language Processing, INTERSPEECH 2006 - ICSLP*, Sep. 2006, pp. 529–533.

[39] D. Hillard, B. Hoffmeister, M. Ostendorf, R. Schlüter, and H. Ney, "iROVER: Improving system combination with classification." in *HLT-NAACL (Short Papers)*, C. L. Sidner, T. Schultz, M. Stone, and C. Zhai, Eds. The Association for Computational Linguistics, 2007, pp. 65–68.

[40] D. Litman, J. Hirschberg, and M. Swertz, "Predicting automatic speech recognition performance using prosodic cues," in *The North American Chapter of the Association for Computational Linguisitics*, 2000, pp. 218–225.

[41] B. Kim, M. Jeong, and G. Geunbae Lee, "Using higher-level linguistic knowledge for speech recognition error in a spoken Q/A dialog," in *The HLT-NAACL special workshop on Higher-Level Linguistic Information for Speech Processing*, 2004, pp. 48–55.

[42] S. Cox and S. Dasmahapatra, "A semantically-based confidence measure for speech recognition," in *International Conference on Spoken Language Processing*, vol. 4, 2000, pp. 206–209.

[43] Y. Shi, "An investigation of linguistic information for speech recognition error detection," Ph.D. dissertation, University of Maryland, 2008.

[44] S. Kaki, E. Sumita, and H. Iida, "A method for correcting errors in speech recognition using the statistical features of character co-occurrence," in *ACLCOLING*, 1998, pp. 653–657.

[45] J. E. Caviedes and J. J. Cimino, "Towards the development of a conceptual distance metric for the UMLS," *Journal of Biomedical Informatics*, vol. 37, pp. 77–85, 2004.

[46] C. Bousquet, M. Jaulent, G. Chatellier, and P. Degoulet, "Using semantic distancefor the efficient coding of medical concepts," in *Annual Symposium of the American Medical Informatics Association*, 2000, pp. 96–100.

[47] P. Resnik, "Using information content to evaluate semantic similarity in a taxonomy," in *International Joint Conference on Artificial Intelligence*, 1995, pp. 448–453.

[48] S. Patwardhan and T. Pedersen, "Using wordnet-based context vectors to estimate the semantic relatedness of concepts," in *EACL 2006 Workshop Making Sense of Sense - Bringing Computational Linguistics and Psycholinguistics Together*, April 2006, pp. 1–8.

[49] S. Cox and S. Dasmahapatra, "High-level approaches to confidence estimation in speech recognition," *IEEE Transactions on Speech and Audio Processing*, vol. 10, no. 7, pp. 460–471, October 2002.

[50] D. Inkpen and A. Desilets, "Semantic similarity for detecting recognition errors in automatic speech transcripts." in *HLT/EMNLP*, 2005, pp. 49–56.

[51] A. Sarma and D. D. Palmer, "Context-based speech recognition error detection and correction," in *Proceedings of HLT-NAACL 2004*. Association for Computational Linguistics, 2004, pp. 85–88.

[52] K. Voll, "A methodology of error detection: Improving speech recognition in radiology," Ph.D. dissertation, Simon Fraser University, 2006.

[53] K. D. Voll, M. S. Atkins, and B. Forster, "Improving the utility of speech recognition through error detection." *J. Digital Imaging*, vol. 21, no. 4, pp. 371–377, 2008.

[54] M. Hunt, "Figures of merit for assessing connected word recognisers," *Speech Communication*, vol. 9, pp. 239–336, 1990.

[55] N. M. I. Group, "Speech recognition scoring toolkit (SCTK) version 2.4.0," nov 2009. [Online]. Available: http://www.itl.nist.gov/iad/mig/tools/

[56] J. Fiscus, J. Garofolo, D. Pallett, M. Przybocki, W. Fisher *et al.*, "1997 english broadcast news speech (HUB4)," 1998. [Online]. Available: http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC98S71

[57] D. Huggins-Daines, "CMU Sphinx open source models," 2008. [Online]. Available: http://www.speech.cs.cmu.edu/sphinx/models/

[58] D. Graff and C. Cieri, "Egnlish Gigaword," 2003.

[59] T. Brants and A. Franz, "Web 1T 5-gram version 1," LDC 2006T13, 2006.

[60] D. Rhode, "SVDLIBC doug rohde's SVD C library," 2010. [Online]. Available: http://tedlab.mit.edu/~dr/SVDLIBC/

[61] K. Abida, F. Karray, and W. Abida, "cROVER: The context-augmented ROVER," *Journal of Control and Intelligent Systems*, vol. 39, no. 4, pp. 223–233, 2011.

[62] ——, "Combination of error detection techniques in automatic speech transcription," in *Springer Lecture Notes in Aritificial Intelligence (LNAI)*, Kamel, Ed., 2011, vol. 6752, pp. 231–240.

[63] K. Abida, W. Abida, and F. Karray, "cROVER: Improving ROVER using automatic error detection," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, May 2011, pp. 1753–1756.

[64] K. Abida and F. Karray, “ROVER enhancement with automatic error detection,” in *Proceedings of 12th Annual conference on the International Speech Communication Association (INTERSPEECH)*, August 2011, pp. 2885–2888.

Appendix A

*c*ROVER WER Numerical Values for Set 1

This appendix reports the numerical values for the relative and absolute WER reduction percentage for Set 1, in table A.1 and A.2 respectively. Each column holds the WER reduction when ROVER is augmented with an error detector. When error detectors are combined, the aggregation scheme is first specified, then the WER reduction percentages for all combinations pairs are reported.

Table A.1: Set 1: *c*ROVER Relative WER reduction (%)

				ROVER + Weighted Combination			ROVER + Harmonic Combination			ROVER + Direct Combination		
ID	ROVER + PMI	ROVER + MSS	ROVER + MR	MR + MSS	MR + PMI	MSS + PMI	MR + MSS	MR + PMI	MSS + PMI	MR + MSS	MR + PMI	MSS + PMI
C1	4.33	3.48	3.38	3.38	4.59	4.86	3.43	4.17	4.70	3.59	3.06	3.59
C2	3.07	2.05	2.05	2.20	2.40	2.46	2.15	2.30	2.15	2.05	2.40	2.20
C3	5.01	4.29	3.78	4.55	5.52	5.93	4.55	5.57	5.32	4.65	5.37	4.65
C4	1.92	1.96	1.96	2.06	2.01	2.01	2.06	1.96	1.96	2.06	2.01	2.01
C5	5.27	6.36	8.07	8.47	8.47	7.82	8.39	7.05	8.31	7.99	6.85	7.99
C6	5.24	7.36	7.69	9.24	9.24	9.33	9.41	8.71	9.57	9.33	7.32	9.41
C7	2.62	2.38	2.56	2.62	2.62	2.74	2.62	2.62	2.68	2.62	2.56	2.62
C8	2.48	2.36	2.36	2.48	2.54	2.54	2.48	2.54	2.48	2.48	2.60	2.48
C9	2.36	3.51	3.46	3.62	3.95	4.17	3.62	3.95	4.34	3.35	3.90	3.40
C10	1.60	3.62	3.57	4.21	4.05	4.37	4.31	3.99	4.79	4.15	3.83	4.21
C11	1.58	4.31	4.42	4.94	4.42	4.94	4.99	4.36	4.94	4.94	4.42	5.05
C12	1.62	3.56	3.88	4.15	3.88	3.83	3.77	3.02	4.26	3.83	2.64	3.88

Table A.2: Set 1: *c*ROVER Absolute WER reduction (%)

				ROVER + Weighted Combination			ROVER + Harmonic Combination			ROVER + Direct Combination		
ID	ROVER + PMI	ROVER + MSS	ROVER + MR	MR + MSS	MR + PMI	MSS + PMI	MR + MSS	MR + PMI	MSS + PMI	MR + MSS	MR + PMI	MSS + PMI
C1	0.82	0.66	0.64	0.64	0.87	0.92	0.65	0.79	0.89	0.68	0.58	0.68
C2	0.60	0.40	0.40	0.43	0.47	0.48	0.42	0.45	0.42	0.40	0.47	0.43
C3	0.98	0.84	0.74	0.89	1.08	1.16	0.89	1.09	1.04	0.91	1.05	0.91
C4	0.39	0.40	0.40	0.42	0.41	0.41	0.42	0.40	0.40	0.42	0.41	0.41
C5	1.30	1.57	1.99	2.09	2.09	1.93	2.07	1.74	2.05	1.97	1.69	1.97
C6	1.28	1.80	1.88	2.26	2.26	2.28	2.30	2.13	2.34	2.28	1.79	2.30
C7	0.44	0.40	0.43	0.44	0.44	0.46	0.44	0.44	0.45	0.44	0.43	0.44
C8	0.42	0.40	0.40	0.42	0.43	0.43	0.42	0.43	0.42	0.42	0.44	0.42
C9	0.43	0.64	0.63	0.66	0.72	0.76	0.66	0.72	0.79	0.61	0.71	0.62
C10	0.30	0.68	0.67	0.79	0.76	0.82	0.81	0.75	0.90	0.78	0.72	0.79
C11	0.30	0.82	0.84	0.94	0.84	0.94	0.95	0.83	0.94	0.94	0.84	0.96
C12	0.30	0.66	0.72	0.77	0.72	0.71	0.70	0.56	0.79	0.71	0.49	0.72

Appendix B

*c*ROVER WER Numerical Values for Set 2

This appendix reports the numerical values for the relative and absolute WER reduction percentage for Set 2, in table B.1 and B.2 respectively. Each column holds the WER reduction when ROVER is augmented with an error detector. When error detectors are combined, the aggregation scheme is first specified, then the WER reduction percentages for all combinations pairs are reported.

Table B.1: Set 2: *c*ROVER Relative WER reduction (%)

				ROVER + Weighted Combination			ROVER + Harmonic Combination			ROVER + Direct Combination		
ID	ROVER + PMI	ROVER + MSS	ROVER + MR	MR + MSS	MR + PMI	MSS + PMI	MR + MSS	MR + PMI	MSS + PMI	MR + MSS	MR + PMI	MSS + PMI
C1	0.39	1.03	0.63	0.86	1.46	1.06	0.96	0.73	1.10	1.03	0.54	1.03
C2	2.12	2.83	2.38	2.75	2.65	2.85	2.75	2.44	2.83	2.83	2.30	2.82
C3	0.62	1.01	0.64	0.92	1.05	1.09	0.96	0.90	1.12	1.01	0.74	1.01
C4	1.66	1.95	1.66	2.01	2.11	2.07	1.93	1.90	2.09	1.95	1.84	1.95
C5	1.02	1.02	1.02	1.04	1.02	1.04	1.03	1.02	1.03	1.02	1.02	1.03
C6	1.09	1.10	1.09	1.09	1.10	1.09	1.10	1.09	1.10	1.10	1.09	1.10
C7	0.97	1.00	0.96	0.99	1.00	1.01	0.98	0.96	1.01	1.00	0.99	1.00
C8	0.77	0.82	0.76	0.83	0.89	0.83	0.81	0.76	0.83	0.82	0.79	0.82
C9	0.75	0.81	0.72	0.75	0.83	0.83	0.79	0.73	0.82	0.81	0.76	0.81
C10	0.67	0.75	0.68	0.85	0.83	0.78	0.73	0.71	0.77	0.75	0.69	0.75
C11	0.77	0.86	0.83	0.85	0.86	0.84	0.85	0.81	0.83	0.86	0.80	0.85
C12	0.93	0.97	0.93	1.00	1.06	1.01	0.97	0.96	1.00	0.97	0.94	0.97

Table B.2: Set 2: *c*ROVER Absolute WER reduction (%)

				ROVER + Weighted Combination			ROVER + Harmonic Combination			ROVER + Direct Combination		
ID	ROVER + PMI	ROVER + MSS	ROVER + MR	MR + MSS	MR + PMI	MSS + PMI	MR + MSS	MR + PMI	MSS + PMI	MR + MSS	MR + PMI	MSS + PMI
C1	1.05	2.77	1.70	2.32	3.93	2.86	2.59	1.97	2.96	2.77	1.45	2.77
C2	5.82	7.76	6.53	7.54	7.27	7.82	7.54	6.69	7.76	7.76	6.31	7.74
C3	1.66	2.71	1.71	2.47	2.81	2.92	2.57	2.41	3.00	2.71	1.98	2.71
C4	4.67	5.49	4.67	5.66	5.94	5.83	5.44	5.35	5.89	5.49	5.18	5.49
C5	3.14	3.14	3.14	3.20	3.14	3.20	3.17	3.14	3.17	3.14	3.14	3.17
C6	3.29	3.32	3.29	3.29	3.32	3.29	3.32	3.29	3.32	3.32	3.29	3.32
C7	3.06	3.16	3.03	3.12	3.16	3.19	3.09	3.03	3.19	3.16	3.12	3.16
C8	2.45	2.61	2.42	2.64	2.84	2.64	2.58	2.42	2.64	2.61	2.52	2.61
C9	2.38	2.58	2.29	2.38	2.64	2.64	2.51	2.32	2.61	2.58	2.42	2.58
C10	2.15	2.41	2.18	2.73	2.66	2.50	2.34	2.28	2.47	2.41	2.22	2.41
C11	2.48	2.77	2.67	2.74	2.77	2.70	2.74	2.61	2.67	2.77	2.58	2.74
C12	2.97	3.10	2.97	3.19	3.39	3.23	3.10	3.07	3.19	3.10	3.00	3.10

Appendix C

Confidence-based Voting WER Numerical Values for Set 1

This appendix reports the numerical values for the relative and absolute WER reduction percentage for Set 1, in table C.1 and C.2 respectively. Each column holds the WER reduction when the confidence based voting algorithm is used. When error detectors are combined, the aggregation scheme is first specified, then the WER reduction percentages for all combinations pairs are reported.

Table C.1: Set 1: Relative WER Reduction using Confidence-based Voting (%)

				ROVER + Weighted Combination			ROVER + Harmonic Combination		
ID	ROVER + PMI	ROVER + MSS	ROVER + MR	MR + MSS	MR + PMI	MSS + PMI	MR + MSS	MR + PMI	MSS + PMI
C1	3.48	2.22	0.58	2.32	4.44	4.96	2.06	4.44	3.59
C2	-6.65	-6.50	-8.54	0.51	0.00	0.00	5.63	0.00	-5.47
C3	7.41	5.78	4.35	0.20	8.03	7.67	0.05	8.03	7.16
C4	4.03	-7.32	-7.91	0.00	0.00	0.00	0.00	0.00	-7.61
C5	10.86	12.0	11.51	11.92	11.27	11.35	11.92	11.27	11.27
C6	15.21	15.46	13.95	15.75	16.16	15.83	15.46	16.16	16.03
C7	0.00	0.00	-2.44	0.00	0.00	0.00	0.00	0.00	0.00
C8	0.00	0.00	-2.25	0.00	0.00	0.00	0.00	0.00	0.00
C9	3.35	2.80	2.20	2.97	4.06	3.90	2.97	4.06	3.35
C10	4.74	4.69	4.42	4.69	5.27	5.43	4.53	5.27	5.11
C11	5.84	5.89	5.36	6.26	6.31	6.57	5.73	6.31	6.20
C12	4.80	4.80	4.63	5.33	5.28	5.28	5.33	5.28	5.06

Table C.2: Set 1: Absolute WER Reduction Voting using Confidence-based Voting(%)

				ROVER + Weighted Combination			ROVER + Harmonic Combination		
ID	ROVER + PMI	ROVER + MSS	ROVER + MR	MR + MSS	MR + PMI	MSS + PMI	MR + MSS	MR + PMI	MSS + PMI
C1	0.66	0.42	0.11	0.44	0.84	0.94	0.39	0.84	0.68
C2	-1.30	-1.27	-1.67	0.10	0.00	0.00	1.10	0.00	-1.07
C3	1.45	1.13	0.85	0.04	1.57	1.50	0.01	1.57	1.40
C4	0.82	-1.49	-1.61	0.00	0.00	0.00	0.00	0.00	-1.55
C5	2.68	2.96	2.84	2.94	2.78	2.80	2.94	2.78	2.78
C6	3.72	3.78	3.41	3.85	3.95	3.87	3.78	3.95	3.92
C7	0.00	0.00	-0.41	0.00	0.00	0.00	0.00	0.00	0.00
C8	0.00	0.00	-0.38	0.00	0.00	0.00	0.00	0.00	0.00
C9	0.61	0.51	0.40	0.54	0.74	0.71	0.54	0.74	0.61
C10	0.89	0.88	0.83	0.88	0.99	1.02	0.85	0.99	0.96
C11	1.11	1.12	1.02	1.19	1.20	1.25	1.09	1.20	1.18
C12	0.89	0.89	0.86	0.99	0.98	0.98	0.99	0.98	0.94

Appendix D

Confidence-based Voting WER Numerical Values for Set 2

This appendix reports the numerical values for the relative and absolute WER reduction percentage for Set 2, in table D.1 and D.2 respectively. Each column holds the WER reduction when the confidence based voting algorithm is used. When error detectors are combined, the aggregation scheme is first specified, then the WER reduction percentages for all combinations pairs are reported.

Table D.1: Set 2: Relative WER Reduction using Confidence-based Voting (%)

				ROVER + Weighted Combination			ROVER + Harmonic Combination		
ID	ROVER + PMI	ROVER + MSS	ROVER + MR	MR + MSS	MR + PMI	MSS + PMI	MR + MSS	MR + PMI	MSS + PMI
C1	2.26	3.37	2.72	3.42	3.48	3.56	3.42	3.42	3.77
C2	8.31	9.71	8.86	6.94	9.27	9.66	9.71	9.25	9.77
C3	2.84	3.89	3.24	3.94	3.91	4.10	3.91	3.97	4.31
C4	8.14	8.48	8.05	8.62	8.81	8.87	8.39	8.81	8.93
C5	-0.52	-0.15	-0.62	-0.28	0.15	0.34	-0.25	0.18	0.34
C6	-0.91	0.60	-0.33	0.60	0.09	0.54	0.60	0.06	0.63
C7	2.02	2.27	1.99	2.49	2.37	2.56	2.30	2.43	2.56
C8	1.72	2.10	1.85	2.10	2.20	2.20	2.10	2.07	2.23
C9	1.30	1.49	1.43	1.43	1.75	1.88	1.53	1.69	1.78
C10	1.28	1.64	1.44	1.44	1.67	1.77	1.67	1.77	1.83
C11	1.93	1.93	1.93	1.93	1.93	1.93	1.93	1.93	1.93
C12	2.24	2.24	2.24	2.24	2.24	2.24	2.24	2.24	2.24

Table D.2: Set 2: Absolute WER Reduction using Confidence-based Voting (%)

				ROVER + Weighted Combination			ROVER + Harmonic Combination		
ID	ROVER + PMI	ROVER + MSS	ROVER + MR	MR + MSS	MR + PMI	MSS + PMI	MR + MSS	MR + PMI	MSS + PMI
C1	0.84	1.25	1.01	1.27	1.29	1.32	1.27	1.27	1.40
C2	3.03	3.54	3.23	2.53	3.38	3.52	3.54	3.37	3.56
C3	1.06	1.45	1.21	1.47	1.46	1.53	1.46	1.48	1.61
C4	2.89	3.01	2.86	3.06	3.13	3.15	2.98	3.13	3.17
C5	-0.17	-0.05	-0.20	-0.09	0.05	0.11	-0.08	0.06	0.11
C6	-0.30	0.20	-0.11	0.20	0.03	0.18	0.20	0.02	0.21
C7	0.64	0.72	0.63	0.79	0.75	0.81	0.73	0.77	0.81
C8	0.54	0.66	0.58	0.66	0.69	0.69	0.66	0.65	0.70
C9	0.41	0.47	0.45	0.45	0.55	0.59	0.48	0.53	0.56
C10	0.40	0.51	0.45	0.45	0.52	0.55	0.52	0.55	0.57
C11	0.60	0.60	0.60	0.60	0.60	0.60	0.60	0.60	0.60
C12	0.70	0.70	0.70	0.70	0.70	0.70	0.70	0.70	0.70