USC-SIPI Report #454

Speech Recognition Error Modeling for Robust Speech Processing and Natural Language Understanding Applications

By
Prashanth Gurunath Shivakumar

May 2021

Signal and Image Processing Institute
UNIVERSITY OF SOUTHERN CALIFORNIA
USC Viterbi School of Engineering
Department of Electrical Engineering-Systems
3740 McClintock Avenue, Suite 400
Los Angeles, CA 90089-2564 U.S.A.

Speech Recognition Error Modeling for Robust Speech Processing and Natural Language Understanding Applications

by

Prashanth Gurunath Shivakumar

A Dissertation Presented to the
FACULTY OF THE USC GRADUATE SCHOOL
UNIVERSITY OF SOUTHERN CALIFORNIA
In Partial Fulfillment of the
Requirements for the Degree
DOCTOR OF PHILOSOPHY
(Electrical Engineering)

May 2021

Dedication

To Amma and Papa.

Acknowledgements

First and foremost, I would like to express my gratitude to my advisors Prof. Panayiotis Georgiou and Prof. Shrikanth Narayanan for always being supportive during my Ph.D and research, and for providing complete freedom for my research. With their immense knowledge, they have always directed me with my research with patience and motivation.

My sincere thanks also goes to my dissertation committee members Prof. Keith Jenkins and Prof. Maja Mataric for guiding me with insightful comments and profound questions.

I thank my fellow colleagues and lab-mates in the Signal Processing for Communication Understanding and Behavior Analysis (SCUBA) and Signal Analysis and Interpretation Laboratory (SAIL) for stimulating discussions.

Finally, I would like to thank my family, my parents Mrs. M. N. Banashankari and Prof. G. K. Shivakumar for always being supportive and motivating throughout my life.

Table of Contents

List Of Tables

List Of Figures

Abstract

Automatic Speech Recognition (ASR) is gaining a lot of importance in everyday life. ASR has become a core component of human computer interaction. It is a key part of many applications involving virtual assistants, voice assistants, gaming, robotics, natural language understanding, education, communication-pronunciation tutoring, call routing, interactive media entertainment, etc. The growth of such applications and their adaptations in everyday scenarios, points to the ASR becoming an ubiquitous part of our daily life in the foreseeable, near future. This has become partly possible due to high performance achieved by state-of-the-art speech recognition systems. However, the errors resulting from ASR can often have a negative impact towards the downstream applications. In this work, we focus on modeling the errors of the ASR with the hypothesis that an accurate modeling of such errors can be used to recover from the ASR errors and alleviate the negative consequences towards its downstream applications.

We model the ASR as a phrase-based noisy transformation channel and propose an error correction system that can learn from the aggregate errors of all the independent modules constituting the ASR and attempt to invert those. The proposed system can exploit long-term context and can re-introduce previously pruned or unseen phrases in addition to better choosing between existing ASR output possibilities. We show that the system can provide improvements over a range of different ASR conditions without degrading any accurate transcription. We also show that the proposed system provides consistent improvements even on out-of-domain tasks as well as over highly optimized ASR models re-scored by recurrent neural language models. Further, we propose sequence-to-sequence neural network for modeling the ASR errors by incorporating much longer contextual information. We propose different optimal architectures and feature representations, in terms of subwords, and demonstrate improvements over the phrase-based noisy channel model.

Additionally, we propose a novel word vector representation, Confusion2Vec, motivated from the human speech production and perception that encodes representational ambiguity. The representational ambiguity of acoustics, which manifests itself in word confusions, is often resolved by both humans and machines through contextual cues. We present several techniques to train an

acoustic perceptual similarity representation ambiguity and learn on unsupervised-generated data from ASR confusion networks or lattice-like structures. Appropriate evaluations are formulated for gauging acoustic similarity in addition to semantic-syntactic and word similarity evaluations. The Confusion2Vec is able to model word confusions efficiently without compromising on the semantic-syntactic word relations, thus effectively enriching the word vector space with extra task relevant ambiguity information. The proposed Confusion2Vec can also contribute and extend to a range of representational ambiguities that emerge in various domains further to acoustic perception, such as morphological transformations, word segmentation, paraphrasing for natural language processing tasks like machine translation, and visual perceptual similarity for image processing tasks like image summarization, optical character recognition etc.

This work also contributes towards efficient coupling of ASR with various downstream algorithms operating on ASR outputs. We prove the efficacy of the Confusion2Vec by proposing a recurrent neural network based spoken language intent detection to achieve state-of-the-art results under noisy ASR conditions. We demonstrate through experiments and our proposed model that ASR often makes errors relating to acoustically similar words and the confusion2vec with inherent model of acoustic relationships between words is able to compensate for the errors. Improvements are also demonstrated when training the intent detection models on noisy ASR transcripts. This work opens new possible opportunities in incorporating the confusion2vec embeddings to a whole range of full-fledged applications.

Further, we extend the previously proposed confusion2vec by encoding each word in confusion2vec vector space by its constituent subword character n-grams. We show the subword encoding helps better represent the acoustic perceptual ambiguities in human spoken language via information modeled on lattice structured ASR output. The efficacy of the subword-confusion2vec is evaluated using semantic, syntactic and acoustic analogy and word similarity tasks. We demonstrate the benefits of subword modeling for acoustic ambiguity representation on the task of spoken language intent detection. The results significantly outperform existing word vector representations as well as the non-subword confusion2vec word embeddings when evaluated on erroneous ASR outputs. We demonstrate confusion2vec subword modeling eliminates the need for retraining/adapting the natural language understanding models on ASR transcripts.

Chapter 1

Introduction

Spoken communication is the most natural way of interaction for humans. This makes spoken language communication arguably the most preferred means of human computer interaction. The spoken natural language understanding (SLU) typically comprises of two fundamental components (i) speech processing & recognition, and (ii) natural language processing & understanding. However, errors can arise throughout the system due to various inconsistencies in human speech and language, operating environments as well as intricate interconnects of machine processing and learning algorithms. In this work, we first identify the sources of errors and attempt to address potential bottlenecks present in each of the stages of the spoken language understanding framework.

1.1 Error Sources

1.1.1 Input speech signal

Several errors are induced into the SLU systems due to the complexity of speech signals. Challenges in speech signal modeling is largely attributed to the vast amount of variability present in the signal. The variability can be largely categorized into three types: (i) acoustic variability, (ii) pronunciation variability, and (iii) language variability.

One of the primary sources of acoustic variability is due to the wide range of inter-speaker variability. Speaker variability in acoustics manifests in terms of speaker age, vocal tract structures, speech articulation and expressions. For example, kids spectral characteristics is found to be vastly different to that of adults [129, 94, 53]. Children's speech is associated with shifted spectral content and formant frequencies, high within-subject and inter-subject variabilities attributed to developmental changes in vocal tract. Children's ASR were found to be 2 to 5 times worse than adults [129]. Acoustic variability can also result due to speaker's health conditions.

Speech disabilities including dysarthria, stroke, tongue cancer etc., can have adverse effects on speech modeling. Moreover paralinguistic phenomenon like emotion, sentiment also pose challenges and induce additional errors. Other than speaker related variability, speaker background and environment induced acoustic variability is a major source of speech modeling errors. Varying amount of noise present in speaking environment can have complex interaction with speech signals, often resulting in heightened errors. Spectral characteristics of noise can generate varying error conditions. For example, spoken noise such as overlapped speech generates errors that are vastly different to the ones generated due to a power line noise. Channel characteristics such as reverberation and the inherent spectral signature of the speech recording/capturing devices is an additional source of error resulting from speech signals.

Pronunciation variability refers to the differences in phonological process involved in pronunciation among different speakers, which is also a prime source for errors. Pronunciation variability manifests in terms of different dialects, accents, non-native speakers and speaker's linguistic knowledge. Non-native speakers often project phonological processes and pronunciation rules from their native language to the target non-native language. For example, native arabic speakers often confuse phoneme "ih" with "eh" leading to potential confusion and errors between words "sit" and "set". Developing linguistic knowledge in children can result in highly varying pronunciations resulting in increased errors during speech recognition.

The use of language can vary from person to person depending on speaker's nativity, origin and general linguistic knowledge. New learners can induce errors resulting from the mismatch between speaker's language constructs and the statistical language models. Developmental stages in linguistic knowledge, especially found in children, can pose serious challenges in speech modeling. On the other hand, extensive speaker vocabulary can also prove challenging.

Addressing these errors have been the main focus of researchers in the ASR community. Some of the existing ASR technologies and the research in acoustic, pronunciation and language modeling are presented in section 2.1. Prior work in error correction and their effectiveness in terms of error reduction are discussed under section 2.1.1.

1.1.2 Machine Processing and Learning Limitations

Another source of errors is the practical limitations imposed by the machine processing and learning algorithms. Two main sources of limitations are computation complexity and memory constraints. For instance, in an ASR, a decoding beam is adopted to prevent memory explosion during generation of decoding graphs. The implications are that the ASR output can itself be non-optimal due to the potential dropping off of a better hypothesis during lattice pruning. Moreover,

ASR systems often make unrecoverable errors due to subsystem pruning (acoustic, language and pronunciation models). For example, pruning words due to acoustics, prior to re-scoring with pronunciation and language model. This can lead to aggregation of errors through each module. Further, the three modules of the ASR (acoustic, language and pronunciation) typically operate with a local view on varying contextual information. Acoustic models typically make decisions using short-term context, prior to re-scoring with longer term context based on pronunciation and language. The varying context can induce unrecoverable errors, for instance sub-optimal decisions based on short term context may not be recovered at a later stage. Finally, the ASR and the NLU operate fairly independent of each other with bottlenecks associated with the flow of information between the two modules with potential for more errors.

1.1.3 Limitations of human-evolved language encoding

Human language is complex because of the vast information encoded and certain ambiguities associated with them. One such ambiguity relating to human spoken language is due to the lack of correlation of language semantics and the acoustics. For example, words such as "right" & "write" and "see" & "sea" sound identical but can have different meanings associated between them. On the contrary, words such as "blue" & "cyan" and "king" & "queen" are semantically close but have vastly different acoustic characteristics. For such reasons, for lack of correlation between semantics and acoustics, human language encoding is not optimal. The non-optimality is a potential source for errors during spoken language processing and understanding.

Prior and existing works in representation of human language for machine processing and learning is discussed under section 2.2. Some of the challenges and effects of non optimal human language encoding in application to the task of spoken language intent detection is presented in section 2.3.

1.2 Our Contribution: Error Modeling

Our study focuses on building systems that addresses the challenges in alleviating the effects of errors under each of the three categories: (i) input speech signal induced errors, (ii) machine processing and learning limitation induced errors, and (iii) errors induced due to limitations in human-evolved language encoding.

In this work we model ASR as a phrase-based noisy transformation channel and propose an error correction system that can learn from the aggregate errors of all the independent modules constituting the ASR and attempt to invert those. The proposed system can not only recover

speech signal induced errors (discussed in section 1.1.1) but also overcome the limitations imposed by machine processing and learning algorithms (discussed in section 1.1.2). Our approach is elaborated and presented in chapter 3.

On the aspect of human language encoding, in this work, we propose a novel word vector representation, Confusion2Vec, motivated from the human speech production and perception that encodes representational ambiguity. Humans employ both acoustic similarity cues and contextual cues to decode information and we focus on a model that incorporates both sources of information. We present several techniques to train an acoustic perceptual similarity representation ambiguity and learn on unsupervised-generated data from Automatic Speech Recognition confusion networks or lattice-like structures. The proposed language encoding, Confusion2Vec, is presented in chapter 4.

Next, we demonstrate the superiority of the newly proposed human language encoding on the task of spoken language intent detection under noisy conditions imposed by automatic speech recognition (ASR) systems. We demonstrate the capabilities of the proposed language encoding to compensate for the errors made by ASR and to increase the robustness of the SLU system. We hypothesize that ASR often makes errors relating to acoustically similar words, and the confusion2vec with inherent model of acoustic relationships between words is able to compensate for the errors. The study is presented in chapter 5.

Further enhancements to Confusion2vec is explored by encoding each word in confusion2vec vector space by its constituent subword character n-grams. We show the subword encoding helps better represent the acoustic perceptual ambiguities as well as in capturing language semantics and syntax by evaluating using semantic, syntactic and acoustic analogy and word similarity tasks. We demonstrate the benefits of subword modeling for acoustic ambiguity representation in application to spoken language intent detection operating on the speech recognition output. The subword modeling for Confusion2Vec and its efficacy towards spoken language intent detection is presented in chapter 6.

Chapter 2

Prior and Existing Work

2.1 Automatic Speech Recognition

Due to the complexity of human language and quality of speech signals, improving performance of automatic speech recognition (ASR) is still a challenging task. The traditional ASR comprises of three conceptually distinct modules: acoustic modeling, dictionary and language modeling. Three modules are fairly independent of each other in research and operation.

In terms of acoustic modeling, Gaussian Mixture Model (GMM) based Hidden Markov Model (HMM) systems [134, 133] were a standard for ASR for a long time and are still used in some of the current ASR systems. Lately, advances in Deep Neural Network (DNN) led to the advent of Deep Belief Networks (DBN) and Hybrid DNN-HMM [71, 34], which basically replaced the GMM with a DNN and employed a HMM for alignments. Deep Recurrent Neural Networks (RNN), particularly Long Short Term Memory (LSTM) Networks replaced the traditional DNN and DBN systems [60]. Connectionist Temporal Classification (CTC) [59] proved to be effective with the ability to compute the alignments implicitly under the DNN architecture, thereby eliminating the need of GMM-HMM systems for computing alignments.

The research efforts for developing efficient dictionaries or lexicons have been mainly in terms of pronunciation modeling. Pronunciation modeling was introduced to handle the intra-speaker variations [160, 176], non-native accent variations [160, 176], speaking rate variations found in conversational speech [176] and increased pronunciation variations found in children's speech [150]. Various linguistic knowledge and data-derived phonological rules were incorporated to augment the lexicon.

Research efforts in language modeling share those of the Natural Language Processing (NLP) community. By estimating the distribution of words, statistical language modeling (SLM), such as n-gram, decision tree models [8], linguistically motivated models [117] amount to calculating

the probability distribution of different linguistic units, such as words, phrases [88], sentences, and whole documents [137]. Recently, Deep Neural Network based language models [5, 112, 163] have also shown success in terms of both perplexity and word error rate.

Very recently, state-of-the-art ASR systems are employing end-to-end neural network models, such as sequence-to-sequence [165] in an encoder-decoder architecture. The systems are trained end-to-end from acoustic features as input to predict the phonemes or characters [7, 23]. Such systems can be viewed as an integration of acoustic and lexicon pronunciation models. The state-of-the-art performance can be attributed towards the joint training (optimization) between the acoustic model and the lexicon models (end-to-end) enabling them to overcome the short-comings of the former independently trained models.

2.1.1 Error Correction for ASR

Several research efforts were carried out for error correction using post-processing techniques. Much of the effort involves user input used as a feedback mechanism to learn the error patterns [2, 121]. Other work employs multi-modal signals to correct the ASR errors [162, 121]. Word co-occurrence information based error correction systems have proven quite successful [142]. In [135], a word-based error correction technique was proposed. The technique demonstrated the ability to model the ASR as a noisy channel. In [77], similar technique was applied to a syllable-to-syllable channel model along with maximum entropy based language modeling. In [39], a phrase-based machine translation system was used to adapt a generic ASR to a domain specific grammar and vocabulary. The system trained on words and phonemes, was used to re-rank the n-best hypotheses of the ASR. In [33], a phrase based machine translation system was used to adapt the models to the domain-specific data obtained by manual user-corrected transcriptions. In [167], an RNN was trained on various text-based features to exploit long-term context for error correction. Confusion networks from the ASR have also been used for error correction. In [191], a bi-directional LSTM based language model was used to re-score the confusion network. In [118], a two step process for error correction was proposed in which words in the confusion network are re-ranked. Errors present in the confusion network are detected by conditional random fields (CRF) trained on n-gram features and subsequently long-distance context scores are used to model the long contextual information and re-rank the words in the confusion network. [21, 52] also makes use of confusion networks along with semantic similarity information for training CRFs for error correction.

2.2 Natural Language Processing

Decoding human language is challenging for machines. It involves estimation of efficient, meaningful representation of words. Machines represent the words in the form of real vectors and the language as a vector space. Vector space representations of language have applications spanning natural language processing (NLP) and human computer interaction (HCI) fields. More specifically, word embeddings can act as features for Machine Translation, Automatic Speech Recognition, Document Topic Classification, Information Retrieval, Sentiment Classification, Emotion Recognition, Behavior Recognition, Question Answering etc.

Early work employed words as the fundamental unit of feature representation. This could be thought of as each word representing an orthogonal vector in a n-dimensional vector space of language with n-words (often referred to as one-hot representation). Such a representation, due to the inherent orthogonality, lacks crucial information regarding inter-word relationships such as similarity. Several techniques found using co-occurrence information of words to be a better feature representation (Ex: n-gram Language Modeling).

Subsequent studies introduced few matrix factorization based techniques to estimate a more efficient, reduced dimensional vector space based on word co-occurrence information. Latent Semantic Analysis (LSA) assumes an underlying vector space spanned by orthogonal set of latent variables closely associated with the semantics/meanings of the particular language. The dimension of this vector space is much smaller than the one-hot representation [35]. LSA was proposed initially for information retrieval and indexing, but soon gained popularity for other NLP tasks. [73] proposed Probabilistic LSA replacing the co-occurrence information by a statistical class based model leading to better vector space representations.

Another popular matrix factorization method, the Latent Dirichlet Allocation (LDA) assumes a generative statistical model where the documents are characterized as a mixture of latent variables representing topics which are described by word distributions [16].

Recently neural networks gained popularity. They often outperform the N-gram models [11, 112] and enable estimation of more complex models incorporating much larger data than before. Various neural network based vector space estimation of words were proposed. [11] proposed feed-forward neural network based language models which jointly learned the distributed word representation along with the probability distribution associated with the representation. Estimating a reduced dimension continuous word representation allows for efficient probability modeling, thereby resulting in much lower perplexity compared to an n-gram model. Recurrent neural network based language models, with inherent memory, allowed for the exploitation of

much longer context, providing further improvements compared to feed forward neural networks [112].

[113] proposes a new technique of estimating vector representation (popularly termed word2vec) which showed promising results in preserving the semantic and syntactic relationships between words. Two novel architectures based on simple log-linear modeling (i) continuous skip-gram and (ii) continuous bag-of-words are introduced. Both the models are trained to model local context of word occurrences. The continuous skip-gram model predicts surrounding words given the current word. Whereas, the continuous bag-of-words model predicts the current word given its context. The task evaluation is based on answering various analogy questions testing semantic and syntactic word relationships. Several training optimizations and tips were proposed to further improve estimation of the vector space by [115, 116]. Such efficient representation of words directly influences the performance of NLP tasks like sentiment classification [83], part-of-speech tagging [99], text classification [98, 79], document categorization [180] and many more.

Subsequent research efforts on extending word2vec involve expanding the word representation to phrases [115], sentences and documents [93]. Similarly, training for contexts derived from syntactic dependencies of a word is shown to produce useful representations [96]. Using morphemes for word representations can enrich the vector space and provide gains especially for unknown, rarely occurring, complex words and morphologically rich languages [104, 18, 131, 32, 155]. Likewise, incorporating sub-word representations of words for the estimation of vector space is beneficial [17]. Similar studies using characters of words have also been tried [26]. [187] explored ensemble techniques for exploiting complementary information over multiple word vector spaces. Studies by [114, 47] demonstrate that vector space representations are extremely useful in extending the model from one language to another (or multi-lingual extensions) since the semantic relations between words are invariant across languages.

Some have tried to combine the advantages from both matrix factorization based techniques and local-context word2vec models. [127] proposes global log-bilinear model for modeling global statistical information as in the case of global matrix factorization techniques along with the local context information as in the case of word2vec.

2.3 Spoken Language Understanding

Spoken Language Understanding (SLU) systems aim at extracting semantic information from human spoken utterances. Such systems play a significant role in practical applications like personal AI voice assistants (e.g. Alexa, Siri, etc.), phone-call routing, booking system and

so on. A SLU system is typically modeled as two separate components: an ASR front-end, which translates acoustic signal into text, followed by a Natural Language Understanding (NLU) module that performs inference for downstream tasks. Typical tasks include Domain classification, Intent Detection and Slot filling. In this work, we focus on the SLU system that performs Intent Detection, a task identifying speaker's intent from speech. Such task is usually treated as an utterance classification problem [186].

In the light of success of Deep Learning techniques, applying Deep Neural Networks on intent detection has been shown to be effective, often outperforming conventional classifiers, such as Support Vector Machines [63]. In recent years, the NLU community have applied various techniques to improve intent detection performance on manual transcripts. [183, 61, 189] jointly model intent detection with slot filling, simplifying the NLU task by a unified model. [65, 84] extend the joint modeling with domain knowledge, which enables information from multiple tasks to benefit the individual tasks and allow the NLU model to be applied to multiple-domain tasks. Going one step further, [101, 188] involve adapting domain-specific language model (LM) while performing intent detection and slot filling, improving the performance on both LM and language understanding task. [100] explores strategies in joint modeling intent classification and slot filling using explicit alignment information provided by slot filling using attention-based encoder-decoder structure. On the basis of attention-based model, [58] connects context information from intent detection with slot filling using a gate mechanism. [97] employs a similar intent-augmented gating mechanism to guide the learning of the slot filling task. It further incorporates character-level embedding along with word-level embedding achieving state-of-the-art results in intent detection.

However, suffering from ASR front-end errors, such as mis-recognized words, insertions and deletions, the performance of such systems degrades significantly, as shown in [69, 36, 111] and is still the bottleneck in SLU systems. On one hand, in order to make system more ASR-robust, ASR hypotheses can be incorporated into the model's training corpus. [90, 171, 107] exploit Word Confusion Networks to efficiently connect NLU models with ASR hypotheses. [154] simulated ASR errors by randomly substituting words with their linguistically and acoustically similar candidates. On the other hand, there have been works that aim to jointly perform NLU tasks and ASR error adaption. [146, 192] employ Recurrent Neural Network (RNN) based Encoder-Decoder structure to reconstruct correct utterances from ASR hypotheses while performing intent detection and slot filling. [158] makes richer feature representations by adding acoustic pitch accent flags into word embedding.

Chapter 3

Learning from Past Mistakes: Improving Automatic Speech Recognition output via Noisy-Clean Phrase Context Modeling

3.1 Introduction

Several research efforts were carried out for error correction using post-processing techniques. Much of the effort involves user input used as a feedback mechanism to learn the error patterns [2, 121]. Other work employs multi-modal signals to correct the ASR errors [162, 121]. Word co-occurrence information based error correction systems have proven quite successful [142]. In [135], a word-based error correction technique was proposed. The technique demonstrated the ability to model the ASR as a noisy channel. In [77], similar technique was applied to a syllable-to-syllable channel model along with maximum entropy based language modeling. In [39], a phrase-based machine translation system was used to adapt a generic ASR to a domain specific grammar and vocabulary. The system trained on words and phonemes, was used to re-rank the n-best hypotheses of the ASR. In [33], a phrase based machine translation system was used to adapt the models to the domain-specific data obtained by manual user-corrected transcriptions. In [167], an RNN was trained on various text-based features to exploit long-term context for error correction. Confusion networks from the ASR have also been used for error correction. In [191], a bi-directional LSTM based language model was used to re-score the confusion network. In [118], a two step process for error correction was proposed in which words in the confusion network are re-ranked. Errors present in the confusion network are detected by conditional random fields (CRF) trained on n-gram features and subsequently long-distance context scores are used to model the long contextual information and re-rank the words in the confusion network. [21, 52] also makes

use of confusion networks along with semantic similarity information for training CRFs for error correction.

The scope of this chapter is to evaluate whether subsequent transcription corrections can take place, on top of a highly optimized ASR. We hypothesize that our system can correct the errors by (i) re-scoring lattices, (ii) recovering pruned lattices, (iii) recovering unseen phrases, (iv) providing better recovery during poor recognitions, (v) providing improvements under all acoustic conditions, (vi) handling mismatched train-test conditions, (vii) exploiting longer contextual information and (viii) text regularization. We target to satisfy the above hypotheses by proposing a Noisy-Clean Phrase Context Model (NCPCM). We introduce context of past errors of an ASR system, that consider all the automated system noisy transformations. These errors may come from any of the ASR modules or even from the noise characteristics of the signal. Using these errors we learn a noisy channel model, and apply it for error correction of the ASR output.

Compared to the above efforts, our work differs in the following aspects:

- Error corrections take place on the output of a state-of-the-art *Large Vocabulary Continuous Speech Recognition* (LVCSR) system trained on matched data. This differs from adapting to constrained domains (e.g. [33, 39]) that exploit domain mismatch. This provides additional challenges both due to the larger error-correcting space (spanning larger vocabulary) and the already highly optimized ASR output.

- We evaluate on a standard LVCSR task thus establishing the effectiveness, reproducibility and generalizability of the proposed correction system. This differs from past work where speech recognition was on a large-vocabulary task but subsequent error corrections were evaluated on a much smaller vocabulary.

- We analyze and evaluate multiple type of error corrections (including but not restricted to *Out-Of-Vocabulary* (OOV) words). Most prior work is directed towards recovery of OOV words.

- In addition to evaluating a large-vocabulary correction system on in-domain (Fisher, 42k words) we evaluate on an out-of-domain, larger vocabulary task (TED-LIUM, 150k words), thus assessing the effectiveness of our system on challenging scenarios. In this case the adaptation is to an even bigger vocabulary, a much more challenging task to past work that only considered adaptation from large to small vocabulary tasks.

- We employ multiple hypotheses of ASR to train our noisy channel model.

- We employ state-of-the-art neural network based language models under the noisy-channel modeling framework which enable exploitation of longer context.

Additionally, our proposed system comes with several advantages: (1) the system could potentially be trained without an ASR by creating a phonetic model of corruption and emulating an ASR decoder on generic text corpora, (2) the system can rapidly adapt to new linguistic patterns, e.g., can adapt to unseen words during training via contextual transformations of erroneous LVCSR outputs.

Further, our work is different from discriminative training of acoustic [177] models and discriminative language models (DLM) [136], which are trained directly to optimize the word error rate using the reference transcripts. DLMs in particular involve optimizing, tuning, the weights of the language model with respect to the reference transcripts and are often utilized in re-ranking n-best ASR hypotheses [136, 141, 184, 15, 22]. The main distinction and advantage with our method is the NCPCM can potentially re-introduce unseen or pruned-out phrases. Our method can also operate when there is no access to lattices or n-best lists. The NCPCM can also operate on the output of a DLM system.

The rest of the paper is organized as follows: Section 3.2 presents various hypotheses and discusses the different types of errors we expect to model. Section 3.3 elaborates on the proposed technique and Section 3.4 describes the experimental setup and the databases employed in this work. Results and discussion are presented in Section 3.5 and we finally conclude and present future research directions in Section 6.7.

3.2 Hypotheses

In this section we analytically present cases that we hypothesize the proposed system could help with. In all of these the errors of the ASR may stem from realistic constraints of the decoding system and pruning structure, while the proposed system could exploit very long context to improve the ASR output.

Note that the vocabulary of an ASR doesn't always match the one of the error correction system. Lets consider for example, an ASR that does not have lexicon entries for "Prashanth" or "Shivakumar" but it has the entries "Shiva" and "Kumar". Lets also assume that this ASR consistently makes the error "Pression" when it hears "Prashanth". Given training data for the NCPCM, it will learn the transformation "Pression Shiva Kumar" into "Prashanth Shivakumar", thus it will have a larger vocabulary than the ASR and learn to recover such errors. This demonstrates the ability to learn out-of-vocabulary entries and to rapidly adapt to new domains.

3.2.1 Re-scoring Lattices

1. *"I was born in nineteen ninety three in Iraq"*
2. *"I was born in nineteen ninety three in eye rack"*
3. *"I was born in nineteen ninety three in I rack"*

Phonetic Transcription: "*ay . w aa z . b ao r n . ih n . n ay n t iy n . n ay n t iy . th r iy . ih n . ay . r ae k*"

Example 1

In Example Example 1, all the three samples have the same phonetic transcription. Let us assume sample 1 is the correct transcription. Since all the three examples have the same phonetic transcription, this makes them indistinguishable by the acoustic model. The language model is likely to down-score the sample 3. It is possible that sample 2 will score higher than sample 1 by a short context LM (e.g. bi-gram or 3-gram) i.e., "in" might be followed by "eye" more frequently than "Iraq" in the training corpora. This will likely result in an ASR error. Thus, although the oracle WER can be zero, the output WER is likely going to be higher due to LM choices.

Hypothesis A: An ideal error correction system can select correct options from the existing lattice.

3.2.2 Recovering Pruned Lattices

A more severe case of Example Example 1 would be that the word "Iraq" was pruned out of the output lattice during decoding. This is often the case when there are memory and complexity constraints in decoding large acoustic and language models, where the decoding beam is a restricting parameter. In such cases, the word never ends up in the output lattice. Since the ASR is constrained to pick over the only existing possible paths through the decoding lattice, an error is inevitable in the final output.

Hypothesis B: An ideal error correction system can generate words or phrases that were erroneously pruned during the decoding process.

3.2.3 Recovery of Unseen Phrases

On the other hand, an extreme case of Example Example 1 would be that the word "Iraq" was never seen in the training data (or is out-of-vocabulary), thereby not appearing in the ASR lattice. This would mean the ASR is forced to select among the other hypotheses even with a low confidence (or output an unknown, $< unk >$, symbol) resulting in a similar error as before. This is often the case due to the constant evolution of human language or in the case of a new domain. For example, names such as "Al Qaeda" or "ISIS" were non-existent in our vocabularies a few years ago.

> ***Hypothesis C:*** An ideal error correction system can generate words or phrases that are out of vocabulary (OOV) and thus not in the ASR output.

3.2.4 Better Recovery during Poor Recognitions

An ideal error correction system would provide more improvements for poor recognitions from an ASR. Such a system could potentially offset for the ASR's low performance providing consistent performance over varying audio and recognition conditions. In real-life conditions, the ASR often has to deal with varying level of "mismatched train-test" conditions, where relatively poor recognition results are commonplace.

> ***Hypothesis D:*** An ideal error correction system can provide more corrections when the ASR performs poorly, thereby offsetting ASR's performance drop (e.g. during mismatched train-test conditions).

3.2.5 Improvements under all Acoustic Conditions

An error correction system which performs well during tough recognition conditions, as per *Hypothesis* 3.2.4 is no good if it degrades good recognizer output. Thus, in addition to our *Hypothesis* 3.2.4, an ideal system would cause no degradation on good ASR output. Such a system can be hypothesized to consistently improve upon and provide benefits over any ASR system including state-of-the-art recognition systems. An ideal system would provide improvements over the entire spectrum of ASR performance (WER).

> ***Hypothesis E:*** An ideal error correction system can not only provide improvements during poor recognitions, but also preserves good speech recognition.

3.2.6 Adaptation

We hypothesize that the proposed system would help in adaptation over mismatched conditions. The mismatch could manifest in terms of acoustic conditions and lexical constructs. The adaptation can be seen as a consequence of *Hypothesis* 3.2.4 & 3.2.5. In addition, the proposed model is capable of capturing patterns of language use manifesting in specific speaker(s) and domain(s). Such a system could eliminate the need of retraining the ASR model for mismatched environments.

> ***Hypothesis F:*** An ideal error correction system can aid in mismatched train-test conditions.

3.2.7 Exploit Longer Context

- *"**Eyes** melted, when he placed his hand on her shoulders."*
- *"**Ice** melted, when he placed it on the table."*

Example 2

The complex construct of human language and understanding enables recovery of lost or corrupted information over different temporal resolutions. For instance, in the above Example Example 2, both the phrases, "Eyes melted, when he placed" and "Ice melted, when he placed" are valid when viewed within its shorter context and have identical phonetic transcriptions. The succeeding phrases, underlined, help in discerning whether the first word is "Eyes" or "Ice". We hypothesize that an error correction model capable of utilizing such longer contexts is beneficial. As new models for phrase based mapping, such as sequence to sequence models [165], become applicable this becomes even more possible and desirable.

> ***Hypothesis G:*** An ideal error correction system can exploit longer context than the ASR for better corrections.

3.2.8 Regularization

1. • *"I guess 'cause I went on a I went on a ..."*
   • *"I guess because I went on a I went on a ..."*
2. • *"i was born in nineteen ninety two"*
   • *"i was born in 1992"*
3. • *"i was born on nineteen twelve"*
   • *"i was born on 19/12"*

Example 3

As per the 3 cases shown in Example Example 3, although both the hypotheses for each of them are correct, there are some irregularities present in the language syntax. Normalization of such surface form representation can increase readability and usability of output. Unlike traditional ASR, where there is a need to explicitly program such regularizations, our system is expected to learn, given appropriate training data, and incorporate regularization into the model.

Hypothesis H: An ideal error correction system can be deployed as an automated text regularizer.

3.3 Methodology

The overview of the proposed model is shown in Figure 3.1. In our paper, the ASR is viewed as a noisy channel (with transfer function H), and we learn a model of this channel, $\widehat{H}^{-1}$ (estimate of inverse transfer function H^{-1}) by using the corrupted ASR outputs (equivalent to signal corrupted by H) and their reference transcripts. Later on, we use this model to correct the errors of the ASR.

The noisy channel modeling mainly can be divided into word-based and phrase-based channel modeling. We will first introduce previous related work, and then our proposed NCPCM.

3.3.1 Previous related work

3.3.1.1 Word-based Noisy Channel Modeling

In [135], the authors adopt word-based noisy channel model borrowing ideas from a word-based statistical machine translation developed by IBM [19]. It is used as a post-processor module to

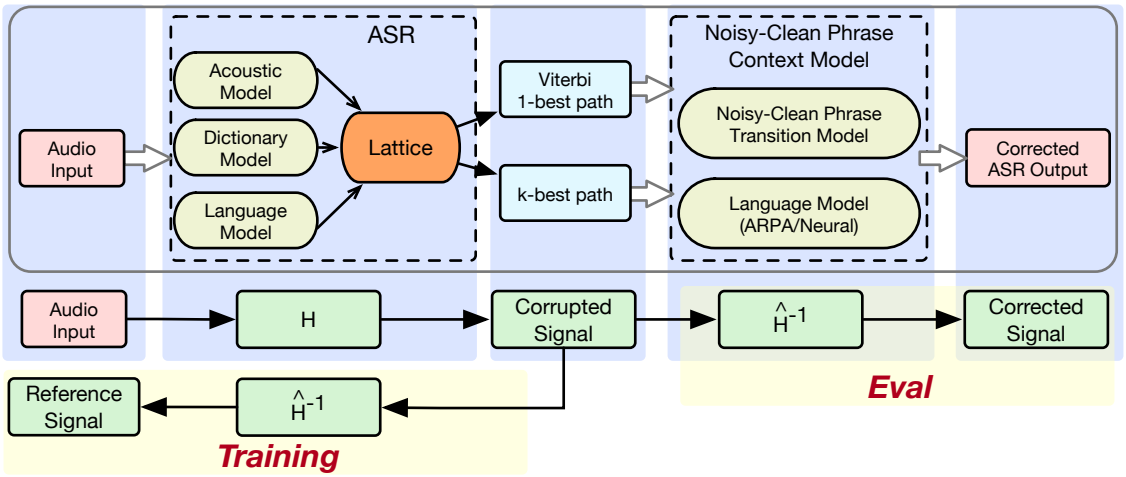


Figure 3.1: Overview of NCPCM

correct the mistakes made by the ASR. The word-based noisy channel modeling can be presented as:

$$\begin{aligned}\hat{W} &= \arg\max_{W_{\text{clean}}} P(W_{\text{clean}}|W_{\text{noisy}}) \\ &= \arg\max_{W_{\text{clean}}} P(W_{\text{noisy}}|W_{\text{clean}})P_{\text{LM}}(W_{\text{clean}})\end{aligned}$$

where $\hat{W}$ is the corrected output word sequence, $P(W_{\text{clean}}|W_{\text{noisy}})$ is the posterior probability, $P(W_{\text{noisy}}|W_{\text{clean}})$ is the channel model and $P_{\text{LM}}(W_{\text{clean}})$ is the language model. In [135], authors hypothesized that introducing many-to-one and one-to-many word-based channel modeling (referred to as fertility model) could be more effective, but was not implemented in their work.

3.3.1.2 Phrase-based Noisy Channel Modeling

Phrase-based systems were introduced in application to phrase-based statistical translation system [85] and were shown to be superior to the word-based systems. Phrase based transformations are similar to word-based models with the exception that the fundamental unit of observation and transformation is a phrase (one or more words). It can be viewed as a super-set of the word-based [19] and the fertility [135] modeling systems.

3.3.2 Noisy-Clean Phrase Context Modeling

We extend the ideas by proposing a complete phrase-based channel modeling for error correction which incorporates the many-to-one and one-to-many as well as many-to-many words (phrase)

channel modeling for error-correction. This also allows the model to better capture errors of varying resolutions made by the ASR. As an extension, it uses a distortion modeling to capture any re-ordering of phrases during error-correction. Even though we do not expect big benefits from the distortion model (i.e., the order of the ASR output is usually in agreement with the audio representation), we include it in our study for examination. It also uses a word penalty to control the length of the output. The phrase-based noisy channel modeling can be represented as:

$$\begin{aligned} \hat{p} &= \underset{p_{\text{clean}}}{\operatorname{argmax}} P(p_{\text{clean}}|p_{\text{noisy}}) \\ &= \underset{p_{\text{clean}}}{\operatorname{argmax}} P(p_{\text{noisy}}|p_{\text{clean}}) P_{\text{LM}}(p_{\text{clean}}) w_{\text{length}}(p_{\text{clean}}) \end{aligned} \tag{3.1}$$

where $\hat{p}$ is the corrected sentence, p_{clean} and p_{noisy} are the reference and noisy sentence respectively. $w_{\text{length}}(p_{\text{clean}})$ is the output word sequence length penalty, used to control the output sentence length, and $P(p_{\text{noisy}}|p_{\text{clean}})$ is decomposed into:

$$P(p^I_{\text{noisy}}|p^I_{\text{clean}}) = \prod_{i=1}^{I} \phi(p^i_{\text{noisy}}|p^i_{\text{clean}}) D(start_i - end_{i-1}) \tag{3.2}$$

where $\phi(p^i_{\text{noisy}}|p^i_{\text{clean}})$ is the phrase channel model or phrase translation table, p^I_{noisy} and p^I_{clean} are the sequences of I phrases in noisy and reference sentences respectively and i refers to the i^{th} phrase in the sequence. $D(start_i - end_{i-1})$ is the distortion model. $start_i$ is the start position of the noisy phrase that was corrected to the i^{th} clean phrase, and end_{i-1} is the end position of the noisy phrase corrected to be the $i-1^{th}$ clean phrase.

3.3.3 Our Other Enhancements

In order to effectively demonstrate our idea, we employ (i) neural language models, to introduce long term context and justify that the longer contextual information is beneficial for error corrections; (ii) minimum error rate training (MERT) to tune and optimize the model parameters using development data.

3.3.3.1 Neural Language Models

Neural network based language models have been shown to be able to model higher order n-grams more efficiently [5, 112, 163]. In [77], a more efficient language modeling using maximum entropy was shown to help in noisy-channel modeling of a syllable-based ASR error correction system.

Incorporating such language models would aid the error-correction by exploiting the longer context information. Hence, we adopt two types of neural network language models in this work. (i) Feed-forward neural network which is trained using a sequence of one-hot word representation along with the specified context [172]. (ii) Neural network joint model (NNJM) language model [37]. This is trained in a similar way as in (i), but the context is augmented with noisy ASR observations with a specified context window. Both the models employed are feed-forward neural networks since they can be incorporated directly into the noisy channel modeling. The recurrent neural network LM could potentially be used during phrase-based decoding by employing certain caching and approximation tricks [3]. Noise Contrastive Estimation was used to handle the large vocabulary size output.

3.3.3.2 Minimum Error Rate Training (MERT)

One of the downsides of the noisy channel modeling is that the model is trained to maximize the likelihood of the seen data and there is no direct optimization to the end criteria of WER. MERT optimizes the model parameters (in our case weights for language, phrase, length and distortion models) with respect to the desired end evaluation criterion. MERT was first introduced in application to statistical machine translation providing significantly better results [122]. We apply MERT to tune the model on a small set of development data.

3.4 Experimental Setup

3.4.1 Database

For training, development, and evaluation, we employ Fisher English Training Part 1, Speech (LDC2004S13) and Fisher English Training Part 2, Speech (LDC2005S13) corpora [29]. The Fisher English Training Part 1, is a collection of conversation telephone speech with 5850 speech samples of up to 10 minutes, approximately 900 hours of speech data. The Fisher English Training Part 2, contains an addition of 5849 speech samples, approximately 900 hours of telephone conversational speech. The corpora is split into training, development and test sets for experimental purposes as shown in Table 3.1. The splits of the data-sets are consistent over both the ASR and

Database	Train			Development			Test		
	Hours	Utterances	Words	Hours	Utterances	Words	Hours	Utterances	Words
Fisher English	1,890.5	1,833,088	20,724,957	4.7	4906	50,245	4.7	4914	51,230
TED-LIUM	-	-	-	1.6	507	17,792	2.6	1155	27,512

Table 3.1: Database split and statistics

the subsequent noisy-clean phrase context model. The development dataset was used for tuning the phrase-based system using MERT.

We also test the system under mismatched training-usage conditions on TED-LIUM. TED-LIUM is a dedicated ASR corpus consisting of 207 hours of TED talks [140]. The data-set was chosen as it is significantly different to Fisher Corpus. Mismatch conditions include: (i) variations in channel characteristics, Fisher, being a telephone conversations corpus, is sampled at 8kHz where-as the TED-LIUM is originally 16kHz, (ii) noise conditions, the Fisher recordings are significantly noisier, (iii) utterance lengths, TED-LIUM has longer conversations since they are extracted from TED talks, (iv) lexicon sizes, vocabulary size of TED-LIUM is much larger with 150,000 words where-as Fisher has 42,150 unique words, (v) speaking intonation, Fisher being telephone conversations is spontaneous speech, whereas the TED talks are more organized and well articulated. Factors (i) and (ii) mostly affect the performance of ASR due to acoustic differences while (iii) and (iv) affect the language aspects, (v) affects both the acoustic and linguistic aspects of the ASR.

3.4.2 System Setup

3.4.2.1 Automatic Speech Recognition System

We used the Kaldi Speech Recognition Toolkit [130] to train the ASR system. In this paper, the acoustic model was trained as a DNN-HMM hybrid system. A tri-gram maximum likelihood estimation (MLE) language model was trained on the transcripts of the training dataset. The CMU pronunciation dictionary [175] was adopted as the lexicon. The resulting ASR is state-of-the-art both in architecture and performance and as such additional gains on top of this ASR are challenging.

3.4.2.2 Pre-processing

The reference outputs of ASR corpus contain non-verbal signs, such as [laughter], [noise] etc. These event signs might corrupt the phrase context model since there is little contextual information between them. Thus, in this paper, we cleaned our data by removing all these non-verbal signs from dataset. The text data is subjected to traditional tokenization to handle special symbols. Also, to prevent data sparsity issues, we restricted all of the sample sequences to a maximum length of 100 tokens (given that the database consisted of only 3 sentences having more than the limit). The NCPCM has two distinct vocabularies, one associated with the ASR transcripts and the other one pertaining to the ground-truth transcripts. The ASR dictionary is often smaller than

the ground-truth transcript mainly because of not having a pronunciation-phonetic transcriptions for certain words, which usually is the case for names, proper-nouns, out-of-language words, broken words etc.

3.4.2.3 NCPCM

We use the Moses toolkit [86] for phrase based noisy channel modeling and MERT optimization. The first step in the training process of NCPCM is the estimation of the word alignments. IBM models are used to obtain the word alignments in both the directions (reference-ASR and ASR-reference). The final alignments are obtained using heuristics (starting with the intersection of the two alignments and then adding the additional alignment points from the union of two alignments). For computing the alignments "mgiza", a multi-threaded version of GIZA++ toolkit [123] was employed. Once the alignments are obtained, the lexical translation table is estimated in the maximum likelihood sense. Then on, all the possible phrases along with their word alignments are generated . A max phrase length of 7 was set for this work. The generated phrases are scored to obtain a phrase translation table with estimates of phrase translation probabilities. Along with the phrase translation probabilities, word penalty scores (to control the translation length) and re-ordering/distortion costs (to account for possible re-ordering) are estimated. Finally, the NCPCM model is obtained as in the equation 3.2. During decoding equation 3.1 is utilized.

For training the MLE n-gram models, SRILM toolkit [159] was adopted. Further we employ the Neural Probabilistic Language Model Toolkit [172] to train the neural language models. The neural network was trained for 10 epochs with an input embedding dimension of 150 and output embedding dimension of 750, with a single hidden layer. The weighted average of all input embeddings was computed for padding the lower-order estimates as suggested in [172].

The NCPCM is an ensemble of phrase translation model, language model, translation length penalty, re-ordering models. Thus the tuning of the weights associated with each model is crucial in the case of proposed phrase based model. We adopt the line-search based method of MERT [13]. We try two optimization criteria with MERT, i.e., using BLEU(B) and WER(W).

3.4.3 Baseline Systems

We adopt four different baseline systems because of their relevance to this work:

Baseline-1: *ASR Output*: The raw performance of the ASR system, because of its relevance to the application of the proposed model.

Baseline-2: *Re-scoring lattices using RNN-LM*: In order to evaluate the performance of the

system with more recent re-scoring techniques, we train a recurrent-neural network with an embedding dimension of 400 and sigmoid activation units. Noise contrastive estimation is used for training the network and is optimized on the development data set which is used as a stop criterion. Faster-RNNLM [1] toolkit is used to train the recurrent-neural network. For re-scoring, 1000-best ASR hypotheses are decoded and the old LM (MLE) scores are removed. The RNN-LM scores are computed from the trained model and interpolated with the old LM. Finally, the 1000-best hypotheses are re-constructed into lattices, scored with new interpolated LM and decoded to get the new best path hypothesis.

Baseline-3: *Word-based noisy channel model*: In order to compare to a prior work described in Section 3.3.1.1 which is based on [135]. The word-based noisy channel model is created in a similar way as the NCPCM model with three specific exceptions: (i) the max-phrase length is set to 1, which essentially converts the phrase based model into word based, (ii) a bi-gram LM is used instead of a tri-gram or neural language model, as suggested in [135], (iii) no re-ordering/distortion model and word penalties are used.

Baseline-4: *Discriminative Language Modeling (DLM)*: Similar to the proposed work, DLM makes use of the reference transcripts to tune language model weights based on specified feature sets in order to re-rank the n-best hypothesis. Specifically, we employ the perceptron algorithm [136] for training DLMs. The baseline system is trained using unigrams, bigrams and trigrams (as in [15, 184, 141]) for a fair comparison with the proposed NCPCM model. We also provide results with an extended feature set comprising of rank-based features and ASR LM and AM scores. Refr (Reranker framework) is used for training the DLMs [14] following most recommendations from [15]. 100-best ASR hypotheses are used for training and re-ranking purposes.

3.4.4 Evaluation Criteria

The final goal of our work is to show improvements in terms of the transcription accuracy of the overall system. Thus, we provide word error rate as it is a standard in the ASR community. Moreover, Bilingual Evaluation Understudy (BLEU) score [126] is used for evaluating our work, since our model can be also treated as a transfer-function ("translation") system from ASR output to NCPCM output.

[1] https://github.com/yandex/faster-rnnlm

1.		
	REF:	oysters clams and mushrooms i think
	ASR:	wasters clams and mushrooms they think
	ORACLE:	wasters clams and mushrooms i think
	NCPCM:	oysters clams and mushrooms they think
		Example of hypotheses B
2.		
	REF:	yeah we had this awful month this winter where it was like a good day if it got up to thirty it was ridiculously cold
	ASR:	yeah we had this awful month uh this winter where it was like a good day if i got up to thirty was ridiculous lee cold
	ORACLE:	yeah we had this awful month this winter where it was like a good day if it got up to thirty it was ridiculous the cold
	NCPCM:	yeah we had this awful month uh this winter where it was like a good day if i got up to thirty it was ridiculously cold
		Example of hypotheses A, B, G
3.		
	REF:	oh well it depends on whether you agree that al qaeda came right out of afghanistan
	ASR:	oh well it depends on whether you agree that al <unk> to came right out of afghanistan
	ORACLE:	oh well it depends on whether you agree that al <unk> to came right out of afghanistan
	NCPCM:	oh well it depends on whether you agree that al qaeda to came right out of afghanistan
		Example of hypotheses C
4.		
	REF:	they laugh because everybody else is laughing and not because it's really funny
	ASR:	they laughed because everybody else is laughing and not because it's really funny
	ORACLE:	they laugh because everybody else is laughing and not because it's really funny
	NCPCM:	they laugh because everybody else is laughing and not because it's really funny
		Example of hypotheses A, G
5.		
	REF:	yeah especially like if you go out for ice cream or something
	ASR:	yeah it specially like if you go out for ice cream or something
	ORACLE:	yeah it's especially like if you go out for ice cream or something
	NCPCM:	yeah especially like if you go out for ice cream or something
		Example of hypotheses A
6.		
	REF:	we don't have a lot of that around we kind of live in a nicer area
	ASR:	we don't have a lot of that around we kinda live in a nicer area
	ORACLE:	we don't have a lot of that around we kind of live in a nicer area
	NCPCM:	we don't have a lot of that around we kind of live in a nicer area
		Example of hypotheses A, H

Table 3.2: Analysis of selected sentences.
REF: Reference ground-truth transcripts; ASR: Output ASR transcriptions;
ORACLE: Best path through output lattice given the ground-truth transcript; NCPCM: Transcripts after NCPCM error-correction
Green color highlights correct phrases. Orange color highlights incorrect phrases.

3.5 Results and Discussion

In this section we demonstrate the ability of our proposed NCPCM in validating our hypotheses A-H from Section 3.2 along with the experimental results. The experimental results are presented in three different tasks: (i) overall WER experiments, highlighting the improvements of the proposed system, presented in Tables 6.1, 3.4 & 3.5, (ii) detailed analysis of WERs over subsets of data, presented in Figures 3.3 & 3.2, and (iii) analysis of the error corrections, presented in Table 3.2. The assessment and discussions of each task is structured similar to Section 3.2 to support their respective claims.

3.5.1 Re-scoring Lattices

Table 3.2 shows selected samples through the process of the proposed error correction system. In addition to the reference, ASR output and the proposed system output, we provide the ORACLE transcripts to assess the presence of the correct phrase in the lattice. Cases 4-6 from Table 3.2 have the correct phrase in the lattice, but get down-scored in the ASR final output which is then recovered by our system as hypothesized in *Hypothesis* 3.2.1.

3.5.2 Recovering Pruned Lattices

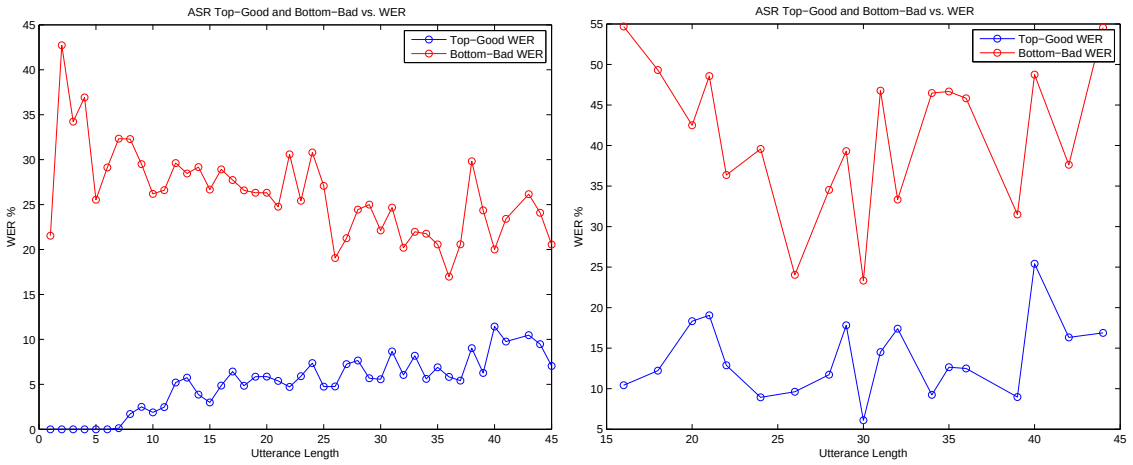
In the cases 1 and 2 from Table 3.2, we see the correct phrases are not present in the ASR lattice, although they were seen in the training and are present in the vocabulary. However, the proposed system manages to recover the phrases as discussed in *Hypothesis* 3.2.2. Moreover, Case 2 also demonstrates an instance where the confusion occurs due to same phonetic transcriptions ("ridiculously" versus "ridiculous lee") again supporting *Hypothesis* 3.2.1.

3.5.3 Recovery of Unseen Phrases

Case 3 of Table 3.2, demonstrates an instance where the word "qaeda" is absent from the ASR lexicon (vocabulary) and hence absent in the decoding lattice. This forces the ASR to output an unknown-word token ($< unk >$). We see that the system recovers an out-of-vocabulary word "qaeda" successfully as claimed in *Hypothesis* 3.2.3.

3.5.4 Better Recovery during Poor Recognitions

To justify the claim that our system can offset for the performance deficit of the ASR at tougher conditions (as per *Hypothesis* 3.2.4), we formulate a sub-problem as follows:



(a) In Domain (Fisher)

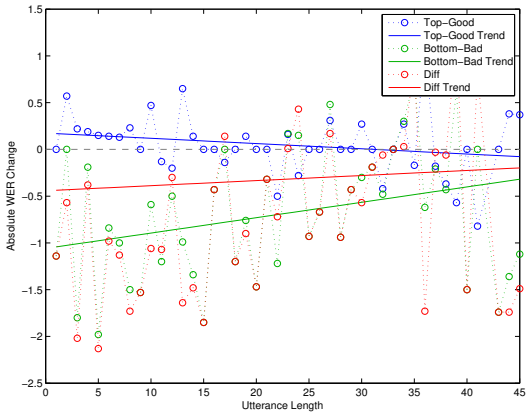
(b) Out of Domain (TEDLIUM)

Figure 3.2: Top-Good, Bottom-Bad WER Splits. As we can see the WER for top-good is often 0%, which leaves no margin for improvement. We will see the impact of this later, as in Fig. 3.3

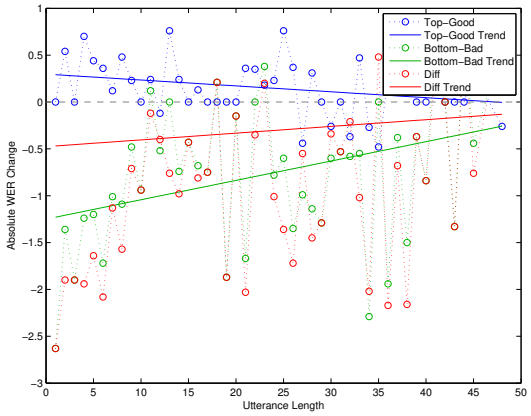
Problem Formulation: We divide equally, per sentence length, our development and test datasets into good recognition results (top-good) and poor recognition results (bottom-bad) subsets based on the WER of the ASR and analyze the improvements and any degradation caused by our system.

Figure 3.3 shows the plots of the above mentioned analysis for different systems as captioned. The blue lines are representative of the improvements provided by our system for top-good subset over different utterance lengths, i.e., it indicates the difference between our system and the original WER of the ASR (negative values indicate improvement and positive values indicate degradation resulting from our system). The green lines indicate the same for bottom-bad subset of the database. The red indicates the difference between the bottom-bad WERs and the top-good WERs, i.e., negative values of red indicate that the system provides more improvements to the bottom-bad subset relative to the top-good subset. The solid lines represent their respective trends which is obtained by a simple linear regression (line-fitting).

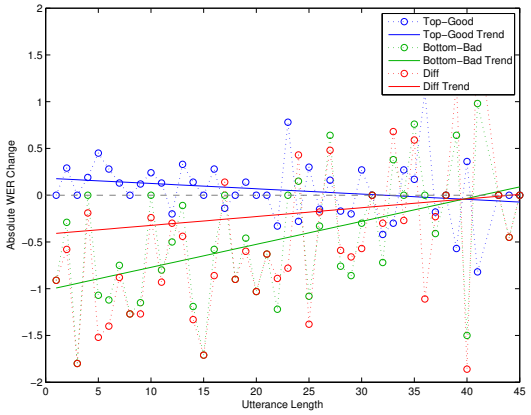
For poor recognitions, we are concerned about the bottom-bad subset, i.e., the green lines in Figure 3.3. Firstly, we see that the solid green line is always below zero, which indicates there is always improvements for bottom-bad i.e., poor recognition results. Second, we observe that the solid red line usually stays below zero, indicating that the performance gains made by the system add more for the bottom-bad poor recognition results compared to the top-good subset (good recognitions). Further, more justifications are provided later in the context of out-of-domain task (Section 3.5 3.5.6) where high mismatch results in tougher recognition task are discussed.



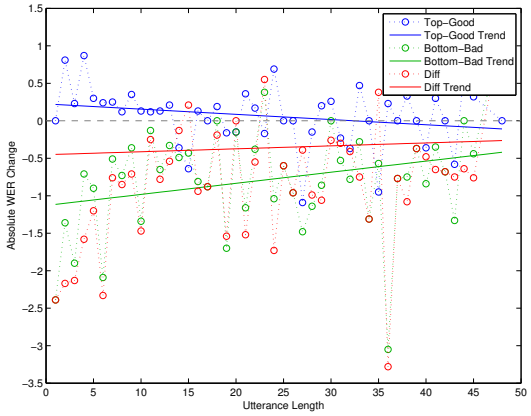
(a) Dev: NCPCM + MERT(W)



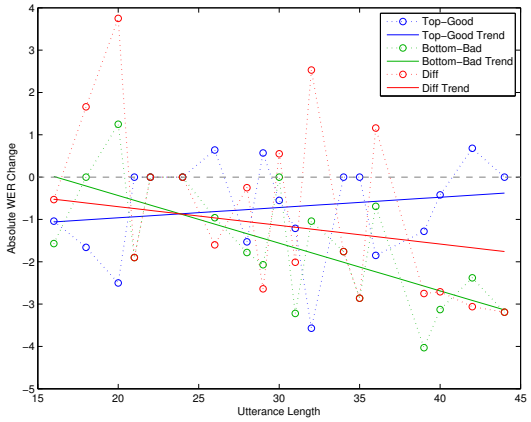
(b) Test: NCPCM + MERT(W)



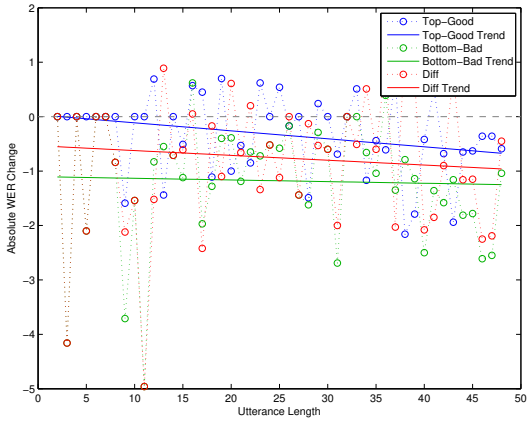
(c) Dev: NCPCM + 5gram NNLM + MERT(W)



(d) Test: NCPCM + 5gram NNLM + MERT(W)



(e) Out-of-Domain Dev: NCPCM + generic LM + MERT(W)



(f) Out-of-Domain Test: NCPCM + generic LM + MERT(W)

Figure 3.3: Length of ASR hypotheses vs. absolute WER change (NCPCM).
Blue & Green lines represent difference between WER of our system and the baseline ASR, for top-good and bottom-bad hypotheses, respectively. In an ideal scenario, all these lines would be below 0, thus all providing a change in WER towards improving the system. However we see in some cases that the WER increases, especially when the hypotheses length is short and when the performance is good. This is as expected since from Fig. 3.2 some cases are at 0% WER due to the already highly-optimized nature of our ASR.
The red line represents the aggregate error over all data for each word length and as we can see in **all** cases the trend is one of **improving** the WER, justifying Hypotheses D, E, F, G.

In domain testing on Fisher Data				
Method	Dev		Test	
	WER	BLEU	WER	BLEU
ASR output (Baseline-1)	15.46%	75.71	17.41%	72.99
ASR + RNNLM re-scoring (Baseline-2)	16.17%	74.39	18.39%	71.24
Word based + bigram LM (Baseline-3)	16.23%	74.28	18.10%	71.76
Word based + bigram LM + MERT(B)	15.46%	75.70	**17.40%**	72.99
Word based + bigram LM + MERT(W)	**15.39%**	75.65	**17.40%**	72.77
Word based + trigram LM + MERT(B)	15.48%	75.59	17.47%	72.81
Word based + trigram LM + MERT(W)	15.46%	75.46	17.52%	72.46
DLM (Baseline-4)	23.65%	63.35	25.36%	61.19
DLM w/ extended feats	24.48%	62.92	26.12%	60.98
Proposed NCPCM	20.33%	66.70	22.32%	63.81
NCPCM + MERT(B)	**15.11%**	**76.06**	**17.18%**	**73.00**
NCPCM + MERT(W)	**15.10%**	**76.08**	**17.15%**	**73.05**
NCPCM + MERT(B) w/o re-ordering	**15.27%**	**76.02**	**17.11%**	**73.33**
NCPCM + MERT(W) w/o re-ordering	**15.19%**	**75.90**	**17.18%**	**73.04**
NCPCM + 10best + MERT(B)	**15.19%**	**76.12**	**17.17%**	**73.22**
NCPCM + 10best + MERT(W)	**15.16%**	**75.91**	**17.21%**	**73.03**

Table 3.3: Noisy-Clean Phrase Context Model (NCPCM) results (uses exactly same LM as ASR)

3.5.5 Improvements under all Acoustic Conditions

To justify the claim that our system can consistently provide benefits over any ASR system (*Hypothesis* 3.2.5), we need to show that the proposed system: (i) does not degrade the performance of the good recognition, (ii) provides improvements to poor recognition instances, of the ASR. The latter has been discussed and confirmed in the previous Section 3.5 3.5.4. For the former, we provide evaluations from two point of views: (1) assessment of WER trends of top-good and bottom-bad subsets (as in the previous Section 3.5 3.5.4), and (2) overall absolute WER of the proposed systems.

Firstly, examining Figure 3.3, we are mainly concerned about the top-good subset pertaining to degradation/improvement of good recognition instances. We observe that the solid blue line is close to zero in all the cases, which implies that the degradation of good recognition is extremely minimal. Moreover, we observe that the slope of the line is almost zero in all the cases, which indicates that the degradation is minimal and mostly consistent over different utterance lengths. Moreover, assessing the degradation from the absolute WER perspective, Figure 3.2a shows the WER over utterance lengths for the top-good and bottom-bad subsets for the in-domain case. The top-good WER is small, at times even 0% (perfect recognition) thereby allowing very small

margin for improvement. In such a case, we see minimal degradation. Although we lose a bit on very good recognitions which is extremely minimal, we gain significantly in the case of 'bad' recognitions. Thus to summarize, the damage that this system can make, under the best ASR conditions, is minimal and offset by the potential significant gains present when the ASR hits some tough recognition conditions.

WER experiments:

Secondly, examining the overall WER, Table 6.1 gives the results of the baseline systems and the proposed technique. Note that we use the same language model as the ASR. This helps us evaluate a system that does not include additional information. We provide the performance measures on both the development and held out test data. The development data is used for MERT tuning.

Baseline results: The output of the ASR (Baseline-1) suggests that the development data is less complex compared to the held out test set. In our case, the RNN-LM based lattice re-scoring (Baseline-2) doesn't help. This results shows that even with a higher order context, the RNN-LM is unable to recover the errors present in the lattice, suggesting that the errors stem from pruning during decoding. We note that the word-based system (Baseline-3) doesn't provide any improvements. Even when we increase context (trigram LM) and use MERT optimization, the performance is just on par with the original ASR output. Further, DLM re-ranking (Baseline-4) fails to provide any improvements in our case. This result is in conjunction with the finding in [15], where the DLM provides improvements only when used in combination with ASR baseline scores. However, we believe introduction of ASR scores into NCPCM can be beneficial as would be in the case of DLMs. Thus, to demonstrate the independent contribution of NCPCM vs DLM's, rather than investigate fusion methods, we don't utilize baseline ASR scores for either of the two methods. We plan to investigate the benefits of multi-method fusion in our future work. When using the extended feature set for training the DLM, we don't observe improvements. With our setup, none of the baseline systems provide noticeable significant improvements over the ASR output. We believe this is due to the highly optimized ASR setup, and the nature of the database itself being noisy telephone conversational speech. Overall, the results of baseline highlights: (i) the difficulty of the problem for our setup, (ii) re-scoring is insufficient and emphasizes the need for recovering pruned out words in the output lattice.

NCPCM results: The NCPCM is an ensemble of phrase translation model, language model, word penalty model and re-ordering models. Thus the tuning of the weights associated with each model is crucial in the case of the phrase based models [119]. The NCPCM without tuning, i.e., assigning random weights to the various models, performs very poorly as expected. The

Cross domain testing on TED-LIUM Data						
Method	Dev		Test			
	WER	BLEU	WER	Δ_1	Δ_2	BLEU
Baseline-1 (ASR)	26.92%	62.00	23.04%	0%	-10.9%	65.71
ASR + RNNLM re-scoring (Baseline-2)	**24.05%**	**64.74**	**20.78%**	**9.8%**	0%	**67.93**
Baseline-3 (Word-based)	29.86%	57.55	25.51%	-10.7%	-22.8%	61.79
Baseline-4 (DLM)	33.34%	53.12	28.02%	-21.6%	-34.8%	58.50
DLM w/ extended feats	30.51%	57.14	29.33%	-27.3%	-41.1%	57.60
NCPCM + MERT(B)	**26.06%**	**63.30**	**22.51%**	**2.3%**	-8.3%	**66.67**
NCPCM + MERT(W)	**26.15%**	**63.10**	**22.74%**	**1.3%**	-9.4%	**66.36**
NCPCM + generic LM + MERT(B)	**25.57%**	**63.98**	**22.38%**	**2.9%**	-7.7%	**66.97**
NCPCM + generic LM + MERT(W)	**25.56%**	**63.83**	**22.33%**	**3.1%**	-7.5%	**66.96**
RNNLM re-scoring + NCPCM + MERT(B)	**23.36%**	**65.88**	**20.40%**	**11.5%**	**1.8%**	**68.39**
RNNLM re-scoring + NCPCM + MERT(W)	**23.32%**	**65.76**	**20.57**	**10.7%**	**1%**	**68.07**
RNNLM re-scoring + NCPCM + generic LM + MERT(B)	**23.00%**	**66.48**	**20.31%**	**11.8%**	**2.3%**	**68.52**
RNNLM re-scoring + NCPCM + generic LM + MERT(W)	**22.80%**	**66.19**	**20.23%**	**12.2%**	**2.6%**	**68.49**

Table 3.4: Results for out-of-domain adaptation using Noisy-Clean Phrase Context Models (NCPCM)
Δ_1:Relative % improvement w.r.t baseline-1; Δ_2:Relative % improvement w.r.t baseline-2;

word-based model lacks re-ordering/distortion modeling and word penalty models and hence are less sensitive to weight tuning. Thus it is unfair to compare the un-tuned phrase based models with the baseline or word-based counterpart. Hence, for all our subsequent experiments, we only include results with MERT. When employing MERT, all of the proposed NCPCM systems significantly outperform the baseline (statistically significant with $p < 0.001$ for both word error and sentence error rates [57] with 51,230 word tokens and 4,914 sentences as part of the test data). We find that MERT optimized for WER consistently outperforms that with optimization criteria of BLEU score. We also perform trials by disabling the distortion modeling and see that results remain relatively unchanged. This is as expected since the ASR preserves the sequence of words with respect to the audio and there is no reordering effect over the errors. The phrase based context modeling provides a relative improvement of 1.72% (See Table 6.1) over the baseline-3 and the ASR output. Using multiple hypotheses (10-best) from the ASR, we hope to capture more relevant error patterns of the ASR model, thereby enriching the noisy channel modeling capabilities. However, we find that the 10-best gives about the same performance as the 1-best. In this case we considered 10 best as 10 separate training pairs for training the system. In the future we want to exploit the inter-dependency of this ambiguity (the fact that all the 10-best hypotheses represent a single utterance) for training and error correction at test time.

3.5.6 Adaptation

WER experiments:

To assess the adaptation capabilities, we evaluate the performance of the proposed noisy-clean phrase context model on an out-of-domain task, TED-LIUM data-base, shown in Table 3.4.

Baseline Results: The baseline-1 (ASR performance) confirms of the heightened mismatched conditions between the training Fisher Corpus and the TED-LIUM data-base. Unlike in matched in-domain evaluation, the RNNLM re-scoring provides drastic improvements (9.8% relative improvement with WER) when tuned with out-of-domain development data set. The mismatch in cross domain evaluation reflects in considerably worse performance for the word-based and DLM baselines (compared to matched conditions).

NCPCM Results: However, we see that the phrase context modeling provides modest improvements over the baseline-1 of approximately 2.3% (See Table 3.4) relative on the held-out test set. We note that the improvements are consistent compared to the earlier in-domain experiments in Table 6.1. Moreover, since the previous LM was trained on Fisher Corpus, we adopt a more generic English LM which provides further improvements of up to 3.1% (See Table 3.4).

We also experiment with NCPCM over the re-scored RNNLM output. We find the NCPCM to always yield consistent improvements over the RNNLM output (See Δ_1 & Δ_2 in Table 3.4). An overall gains of 2.6% relative is obtained over the RNNLM re-scored output (baseline-2) i.e., 12.2% over ASR (baseline-1) is observed. This confirms that the NCPCM is able to provide improvements parallel, in conjunction to the RNNLM or any other system that may improve ASR performance and therefore supports the *Hypothesis* 3.2.5 in yielding improvements in the highly optimized ASR environments. This also confirms the robustness of the proposed approach and its application to the out-of-domain data. More importantly, the result confirms *Hypothesis* 3.2.6, i.e., our claim of rapid adaptability of the system to varying mismatched acoustic and linguistic conditions. The extreme mismatched conditions involved in our experiments supports the possibility of going one step further and training our system on artificially generated data of noisy transformations of phrases as in [168, 141, 22, 89, 40, 184]. Thus possibly eliminating the need for an ASR for training purposes.

Further, comparing the WER trends from the in-domain task (Figure 3.3b) to the out-of-domain task (Figure 3.3f), we firstly find that the improvements in the out-of-domain task are obtained for both top-good (good recognition) and bottom-bad (bad recognition), i.e., both the solid blue line and the solid green line are always below zero. Secondly, we observe that the improvements are more consistent throughout all the utterance lengths, i.e., all the lines have near zero slopes compared to the in-domain task results. Third, comparing Figure 3.2a with Figure 3.2b, we observe more room for improvement, both for top-good portion as well as the bottom-bad WER subset of data set. The three findings are fairly meaningful considering the high mismatch of the out-of-domain data.

In domain testing on Fisher Data				
Method	Dev		Test	
	WER	BLEU	WER	BLEU
Baseline-1 (ASR output)	15.46%	75.71	17.41%	72.99
Baseline-2 (ASR + RNNLM re-scoring)	16.17%	74.39	18.39%	71.24
Baseline-3 (Word based + 5gram NNLM)	15.47%	75.63	17.41%	72.92
Word based + 5gram NNLM + MERT(B)	15.46%	75.69	**17.40%**	72.99
Word based + 5gram NNLM + MERT(W)	**15.42%**	75.58	**17.38%**	72.75
NCPCM + 3gram NNLM + MERT(B)	**15.46%**	**75.91**	**17.37%**	**73.24**
NCPCM + 3gram NNLM + MERT(W)	**15.28%**	**75.94**	**17.11%**	**73.31**
NCPCM + 5gram NNLM + MERT(B)	**15.35%**	**75.99**	**17.20%**	**73.34**
NCPCM + 5gram NNLM + MERT(W)	**15.20%**	**75.96**	**17.08%**	**73.25**
NCPCM + NNJM-LM (5,4) + MERT(B)	**15.29%**	**75.93**	**17.13%**	**73.26**
NCPCM + NNJM-LM (5,4) + MERT(W)	**15.28%**	**75.94**	**17.13%**	**73.29**

Table 3.5: Results for Noisy-Clean Phrase Context Models (NCPCM) with Neural Network Language Models (NNLM) and Neural Network Joint Models (NNJM)

3.5.7 Exploit Longer Context

Firstly, inspecting the error correction results from Table 3.2, cases 2 and 4 hint at the ability of the system to select appropriate word-suffixes using long term context information.

Second, from detailed WER analysis in Figure 3.3, we see that the bottom-bad (solid green line) improvements decrease with increase in length in most cases, hinting at potential improvements to be found by using higher contextual information for error correction system as future research directions. Moreover, closer inspection across different models, comparing the trigram MLE model (Figure 3.3b) with the 5gram NNLM (Figure 3.3d), we find that the NNLM provides minimal degradation and better improvements especially for longer utterances by exploiting more context (the blue solid line for NNLM has smaller intercept value as well as higher negative slope). We also find that for the bottom-bad poor recognition results (green solid-line), the NNLM gives consistent (smaller positive slope) and better improvements especially for the higher length utterances (smaller intercept value). Thus emphasizing the gains provided by higher context NNLM.

WER experiments: Third, Table 3.5 shows the results obtained using a neural network language model of higher orders (also trained only on the in-domain data). For a fair comparison, we adopt a higher order (5gram) NNLM for the baseline-3 word based noise channel modeling system. Even with a higher order NNLM, the baseline-3 fails to improve upon the ASR. We don't include the baseline-4 results under this section, since DLM doesn't include a neural network model. Comparing results from Table 6.1 with Table 3.5, we note the benefits of higher order LMs, with

the 5-gram neural network language model giving the best results (a relative improvement of 1.9% over the baseline-1), outperforming the earlier MLE n-gram models as per *Hypothesis* 3.2.7.

Moreover, experimental comparisons with baseline-3 (word-based) and NCPCM models, both incorporating identical 5-gram neural network language models confirms the advantages of NCPCM (a relative improvement of 1.7%). However, the neural network joint model LM with target context of 5 and source context of 4 did not show significant improvements over the traditional neural LMs. We expect the neural network models to provide further improvements with more training data.

3.5.8 Regularization

Finally, the last case in Table 3.2 is of text regularization as described in Section 3.2, *Hypothesis* 3.2.8. Overall, in our experiments, we found that approximately 20% were cases of text regularization and the rest were a case of the former hypotheses.

3.6 Conclusions & Future Work

In this work, we proposed a noisy channel model for error correction based on phrases. The system post-processes the output of an automated speech recognition system and as such any contributions in improving ASR are in conjunction of NCPCM. We presented and validated a range of hypotheses. Later on, we supported our claims with apt problem formulation and their respective results. We showed that our system can improve the performance of the ASR by (i) re-scoring the lattices (*Hypothesis* 3.2.1), (ii) recovering words pruned from the lattices (*Hypothesis* 3.2.2), (iii) recovering words never seen in the vocabulary and training data (*Hypothesis* 3.2.3), (iv) exploiting longer context information (*Hypothesis* 3.2.7), and (v) by regularization of language syntax (*Hypothesis* 3.2.8). Moreover, we also claimed and justified that our system can provide more improvement in low-performing ASR cases (*Hypothesis* 3.2.4), while keeping the degradation to minimum in cases when the ASR performs well (*Hypothesis* 3.2.5). In doing so, our system could effectively adapt (*Hypothesis* 3.2.6) to changing recognition environments and provide improvements over any ASR systems.

In our future work, the output of the noisy-clean phrase context model will be fused with the ASR beliefs to obtain a new hypothesis. We also intend to introduce ASR confidence scores and signal SNR estimates, to improve the channel model. We are investigating introducing the probabilistic ambiguity of the ASR in the form of lattice or confusion networks as inputs to the channel-inversion model.

Further, we will utilize sequence-to-sequence (Seq2seq) translation modeling [165] to map ASR outputs to reference transcripts. The Seq2seq model has been shown to have benefits especially in cases where training sequences are of variable length [27]. We intend to employ Seq2seq model to encode ASR output to a fixed-size embedding and decode this embedding to generate the corrected transcripts.

Chapter 4

Confusion2Vec: Towards Enriching Vector Space Word Representations with Representational Ambiguities

4.1 Introduction

The goal of this study is to come up with a new vector space representation for words which incorporates the uncertainty information in the form of word confusions present in lattice like structures (e.g. confusion networks). Here, the word confusions are any word level ambiguities resultant of any algorithms such as machine translation, ASR etc., or can be knowledge-based like word segmentation information or data driven. For example, acoustic confusable words in ASR lattices: "two" and "to" (see Figure 4.1). A word lattice is a compact representation (directed acyclic weighted graphs) of different word sequences that are likely possible. A confusion network is a special type of lattice, where each word sequence is made to pass through each node of the graph. The lattices and confusion networks embed word confusion information. The study takes motivation from human perception, i.e., the ability of humans to decode information based on two fairly independent information streams (see Section 4.2.1 for examples): (i) linguistic context (modeled by word2vec like word vector representations), and (ii) acoustic confusability (relating to phonology). However, the present word vector representations like word2vec only incorporate the contextual confusability during modeling. Hence, in order to handle confusability and to decode human language/speech successfully, there is a need to model both the dimensions. Although, primarily, the motivation is derived from human speech and perception, the confusions are not constrained to acoustics and can be extended to any confusions parallel to the linguistic contexts, for example, confusions present in lattices. Most of the machine learning algorithms output predictions as a probability measure. This uncertainty information stream can be expressed in the form of a lattice or a confusion network temporally, and is often found to contain useful

information for subsequent processing and analysis. The scope of this work is to introduce a complementary (ideally orthogonal) subspace in addition to the underlying word vector space representation captured by word2vec. This new subspace captures the word confusions orthogonal to the syntactic and semantics of the language. We propose Confusion2Vec vector space operating on lattice like structures, specifically word confusion networks. We introduce several training configurations and evaluate their effectiveness. We also formulate appropriate evaluation criterion to assess the performance of each orthogonal subspaces, first independently and then jointly. Analysis of the proposed word vector space representation is carried out.

The rest of the paper is organized as follows. Motivation for Confusion2vec, i.e., the need to model word-confusions for word embeddings, is provided through means of human speech & perception, machine learning, and through potential applications in section 4.2. A particular case study is chosen and the problem is formulated in section 4.3. In section 4.4, different training configurations for efficient estimation of word embeddings are proposed. Additional tuning schemes for the proposed Confusion2vec models are presented in section 4.5. Evaluation criterion formulation and evaluation database creation is presented in section 4.6. Experimental setup and baseline system is described in section 4.7. Results are tabulated and discussed in section 6.5. Word vector space analysis is performed and findings are presented in section 4.9. Section 4.10 discusses with the help of few toy examples, the benefits of the Confusion2vec embeddings for the task of ASR error correction. Section 6.7 draws the conclusion of the study and finally the future research directions are discussed in Section 6.8.

4.2 Motivation

One efficient way to represent words as vectors is to represent them in a space that preserves the semantic and syntactic relations between the words in the language. Word2vec describes a technique to achieve such a representation by trying to predict the current word from its local context (or vice-versa) over a large text corpora. The estimated word vectors are shown to encode efficient syntactic-semantic language information. In this work we propose a new vector space for word representation which incorporates various forms of word confusion information in addition to the semantic & syntactic information. The new vector space is inspired and motivated from the following factors from human speech production & perception and machine learning.

4.2.1 Human speech production, perception and hearing

In our every day interactions, confusability can often result in the need for context to decode the underlying words.

"Please ____ a seat." (Example 1)

In Example 1, the missing word could be guessed from its context and narrowed down to either "have" or "take". This context information is modeled through language models. More complex models such as word2vec also use the contextual information to model word vector representations.

On the other hand, confusability can also originate from other sources such as acoustic representations.

"I want to seat" (Example 2)

In Example 2, the underlined word is mispronounced/misheard, and grammatically incorrect. In this case, considering the context there exists a lot of possible correct substitutions for the word "seat" and hence the context is less useful. The acoustic construct of the word "seat" can present additional information in terms of acoustic alternatives/similarity, such as "sit" and "seed".

"I want to s—" (Example 3)

Similarly in Example 3, the underlined word is incomplete. The acoustic confusability information can be useful in the above case of broken words. Thus, since the confusability is acoustic, purely lexical vector representations like word2vec fail to encode or capture it. In this work, we propose to additionally encode the word (acoustic) confusability information to learn a better word embedding. Although the motivation is specific to acoustics in this case, it could be extended to other inherent sources of word-confusions spanning various machine learning applications.

4.2.2 Machine Learning Algorithms

Most of the machine learning algorithms output hypothesis as a probability measure. Such a hypothesis could be represented in the form of a lattice, confusion network or n-best lists. It is often useful to consider the uncertainty associated with the hypothesis for subsequent processing and analysis (see Section 4.11 for potential applications). The uncertainty information is often, orthogonal to the contextual dimension and is specific to the task attempted by the machine learning algorithms.

Along this direction, recently, there have been several efforts concentrated on introducing lattice information into the neural network architecture. Initially, Tree-LSTM was proposed enabling

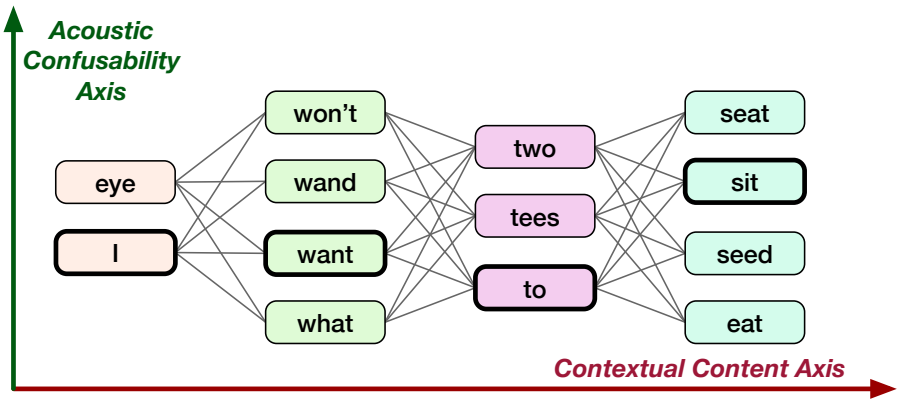


Figure 4.1: **An example confusion network for ground-truth utterance "I want to sit."**

tree-structured network topologies to be inputted to the RNNs [166], which could be adapted and applied to lattices [156]. LatticeRNN was proposed for processing word level lattices for ASR [91]. Lattice based Gated Recurrent Units (GRUs) [161] and lattice-to-sequence models [169] were proposed for reading word lattice as input, specifically a lattice with tokenization alternatives for machine translation models. LatticeLSTM was adopted for lattice-to-sequence model incorporating lattice scores for the task of speech translation by [156]. [20] proposed Neural lattice language models which enables to incorporate many possible meanings for words and phrases (paraphrase alternatives).

Thus, a vector space representation capable of embedding relevant uncertainty information in the form of word confusions present in lattice-like structures or confusion networks along with the Semantic & Syntactic can be potentially superior to word2vec space.

4.3 Case Study: Application to Automatic Speech Recognition

In this work, we consider the ASR task as a case study to demonstrate the effectiveness of the proposed Confusion2vec model in modeling acoustic word-confusability. However, the technique can be adopted for a lattice or confusion network output from potentially any algorithm to capture various patterns as discussed in section 4.11, in which case the confusion-subspace (vertical ambiguity in figure 4.1), is no longer constrained to acoustic word-confusions.

An ASR lattice contains multiple paths over acoustically similar words. A lattice could be transformed and represented as a linear graph forcing every path to pass through all the nodes [185, 105]. Such a linear graph is referred to as a confusion network. Figure 4.1 shows a sample confusion network output by ASR for the ground truth "I want to sit". The confusion network could be viewed along two fundamental dimensions of information (see figure 4.1): (i) Contextual

axis - sequential structure of a sentence, (ii) Acoustic axis - similarly sounding word alternatives. Traditional word vector representations such as word2vec only model the contextual information (the horizontal (red) direction in Figure 4.1). The word confusions, for example, the acoustic contextualization as in Figure 4.1 (the vertical (green) direction in Figure 4.1) is not encoded. We propose to additionally capture the co-occurrence information along the acoustic axis orthogonal to the word2vec. This is the main focus of our work, i.e., to jointly learn the vertical, word-confusion context and the horizontal, semantic and syntactic context. In other words, we hypothesize to derive relationships between the semantics and syntaxes of language and the word-confusions (acoustic-confusion).

4.3.1 Related Work

[10] trained a continuous word embedding of acoustically alike words (using n-gram feature representation of words) to replace the state space models (HMMs), decision trees and lexicons of an ASR. Through the use of such an embedding and lattice re-scoring technique demonstrated improvements in word error rates of ASR. The embeddings are also shown to be useful in application to the task of ASR error detection by [56]. A few evaluation strategies are also devised to evaluate phonetic and orthographic similarity of words. Additionally, there have been studies concentrating on estimating word embeddings from acoustics [80, 28, 95, 68] with evaluations based on acoustic similarity measures. Parallely, word2vec like word embeddings have been used successfully to improve ASR Error detection performance [54, 55]. We believe the proposed exploitation of both information sources, i.e., acoustic relations and linguistic relations (semantics and syntaxes) will be beneficial in ASR and error detection, correction tasks. The proposed confusion2vec operates on the lattice output of the ASR in contrast to the work on acoustic word embeddings [80, 28, 95, 68] which is directly trained on audio. The proposed Confusion2vec differs to the works by [10] and [56], which also utilizes audio data with the hypothesis that the layer right below softmax layer of a deep end-to-end ASR contains acoustic similarity information of words. Confusion2vec can also be potentially trained without an ASR, on artificially generated data, emulating an ASR [168, 141, 22, 89, 40, 184]. Thus, Confusion2vec can potentially be trained in a completely unsupervised manner and with appropriate model parameterization incorporate various degrees of acoustic confusability, e.g. stemming from noise or speaker conditions.

Further, in contrast to the prior works on lattice encoding RNNs [166, 156, 91, 161, 169, 20], which concentrate on incorporating the uncertainty information embedded in the word lattices by modifying the input architecture for recurrent neural network, we propose to introduce the ambiguity information from the lattices to the word embedding explicitly. We expect similar

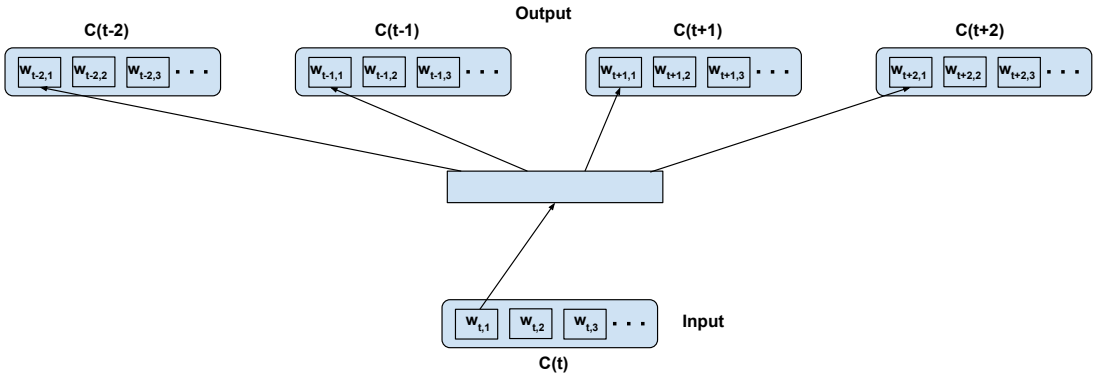


Figure 4.2: **Baseline Word2Vec Training scheme for Confusion networks.**
$c(t)$ is a unit word confusion in the confusion network at a time-stamp t, i.e., $c(t)$ represents a set of arcs between two adjacent nodes of a confusion network, representing a set of confusable words.
$w_{t,i}$ is the i^{th} most probable word in the confusion $c(t)$.
Word confusions are sorted in decreasing order of their posterior probability:
$P(w_{t,1}) > P(w_{t,2}) > P(w_{t,3})...$

advantages as with lattice encoding RNNs in using the pre-trained confusion2vec embedding towards various tasks like ASR, Machine translation etc. Moreover, our architecture doesn't require memory which has significant advantages in terms of training complexity. We propose to train the embedding in a similar way to word2vec models [113]. All the well studied previous efforts towards optimization of training such models [115, 116], should apply to our proposed model.

4.4 Proposed Models

4.4.1 Baseline Word2Vec Model

The popular word2vec work [113] proposed log-linear models, i.e., neural network consisting of a single linear layer (projection matrix) without non-linearity. These models have significant advantages in training complexity. [113] found the skip-gram model to be superior to the bag-of-word model in a semantic-syntactic analogy task. Hence, we only employ the skip-gram configuration in this work. Appropriately, the skip-gram word2vec model is also adopted as the baseline for this work. However, we strongly believe the proposed concept (introducing word ambiguity information) is independent of the modeling technique itself and should translate to relatively newer techniques like GloVe [127] and fastText [17].

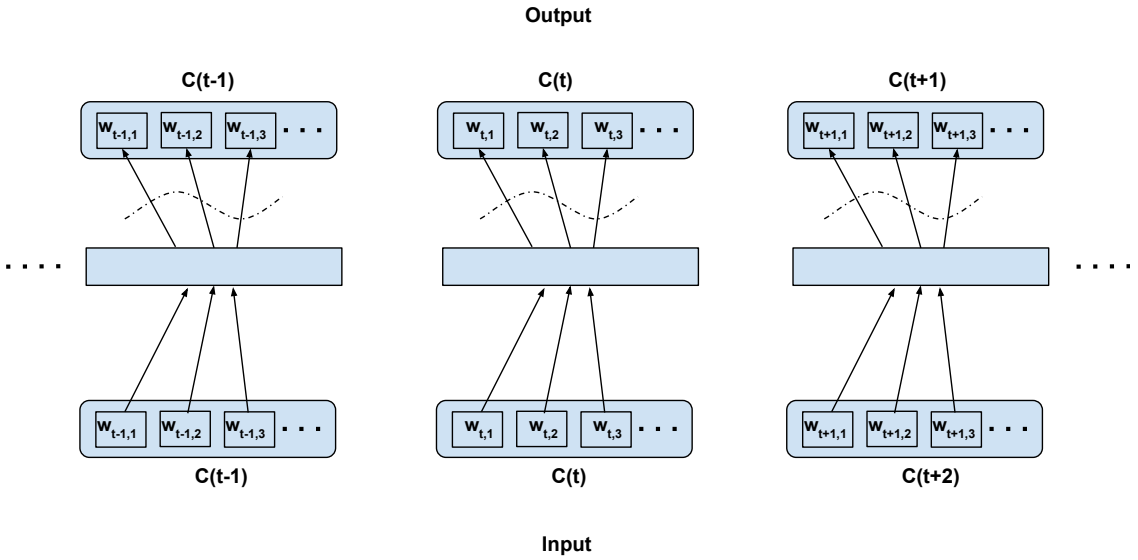


Figure 4.3: **Proposed Intra-Confusion Training Scheme for Confusion networks.**
$c(t)$ is a unit word confusion in the confusion network at a time-stamp t, i.e., $c(t)$ represents a set of arcs between two adjacent nodes of a confusion network, representing a set of confusable words.
$w_{t,i}$ is the i^{th} most probable word in the confusion $c(t)$.
Word confusions are sorted in decreasing order of their posterior probability:
$P(w_{t,1}) > P(w_{t,2}) > P(w_{t,3})...$
The dotted curved lines denote that the self-mapping is disallowed.

We adapt the word2vec contextual modeling to operate on the confusion network (in our case confusion network of an ASR). Figure 4.2 shows the training configuration of the skip-gram word2vec model on the confusion network. The baseline model (traditional skip-gram) only considers the context of the top hypothesis of the confusion network (single path) for training. The words $w_{t-2,1}$, $w_{t-1,1}$, $w_{t+1,1}$ and $w_{t+2,1}$ (i.e., the most probable words in the confusions $C(t-2)$, $C(t-1)$, $C(t+1)$ and $C(t+2)$ respectively) are predicted from $w_{t,1}$ (i.e., the most probable word in $C(t)$) for a skip-window of 2 as depicted in Figure 4.2.

4.4.2 Intra-Confusion Training

Next, we explore the direct adaptation of the skip-gram modeling but on the confusion dimension (i.e., considering word confusions as contexts) rather than the traditional sequential context. Figure 4.3 shows the training configuration over a confusion network. In short every word is linked with every other alternate word in the confusion dimension (i.e., between set of confusable words) through the desired network (as opposed to the temporal context dimension in the word2vec training). Note, we disallow any word being predicted from itself (this constrain is indicated with curved dotted lines in the figure). As depicted in the Figure 4.3, the word $w_{t,i}$ (confusion

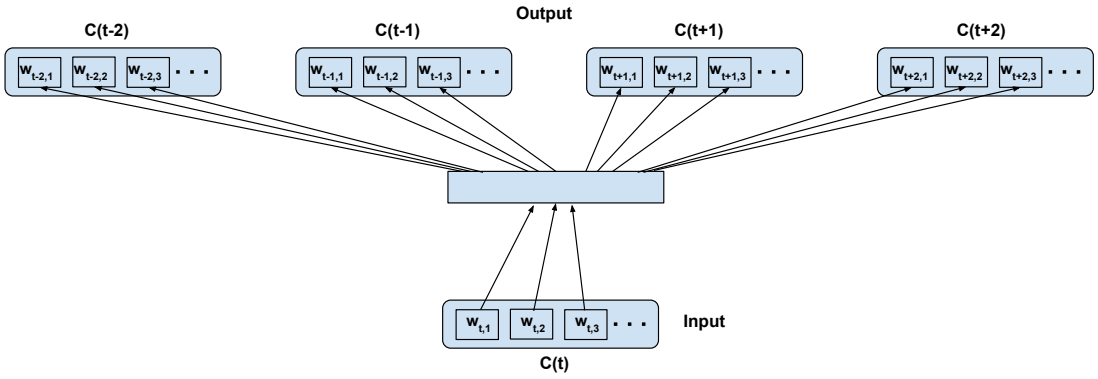


Figure 4.4: **Proposed Inter-Confusion Training Scheme for Confusion networks.**
$c(t)$ is a unit word confusion in the confusion network at a time-stamp t, i.e., $c(t)$ represents a set of arcs between two adjacent nodes of a confusion network, representing a set of confusable words.
$w_{t,i}$ is the i^{th} most probable word in the confusion $c(t)$.
Word confusions are sorted in decreasing order of their posterior probability:
$P(w_{t,1}) > P(w_{t,2}) > P(w_{t,3})...$

context) is predicted from $w_{t,j}$ (current word), where $i = 1, 2, 3 \ldots \text{length}(C(t))$ and $j \neq i$, for each $j = 1, 2, 3 \ldots \text{length}(C(t))$ for confusion $C(t)\ \forall t$. We expect such a model to capture inherent relations over the different word confusions. In the context of an ASR lattice, we expect it to capture intrinsic relations between similarly sounding words (acoustically similar). However, the model would fail to capture any semantic and syntactic relations associated with the language. The embedding obtained from this configuration can be fused (concatenated) with the traditional skip-gram word2vec embedding to form a new subspace representing both the independently trained subspaces. The number of training samples generated with this configuration is:

$$\#\text{Samples} = \sum_{i=1}^{n} D_i \times (D_i - 1) \tag{4.1}$$

where n is the number of time steps, D_i is the number of confusions at the i^{th} time step.

4.4.3 Inter-Confusion Training

In this configuration, we propose to model both the linguistic contexts and the word confusion contexts simultaneously. Figure 4.4 illustrates the training configuration. Each word in the current confusion is predicted from each word from the succeeding and preceding confusions over a predefined local context. To elaborate, the words $w_{t-t',i}$ (context) are predicted from $w_{t,j}$ (current word) for $i = 1, 2, 3 \ldots \text{length}(C(t - t'))$, $j = 1, 2, 3 \ldots \text{length}(C(t))$, $t' \in 1, 2, -1, -2$ for skip-window of 2 for current confusion $C(t) \forall t$ as per Figure 4.4. Since we assume the acoustic

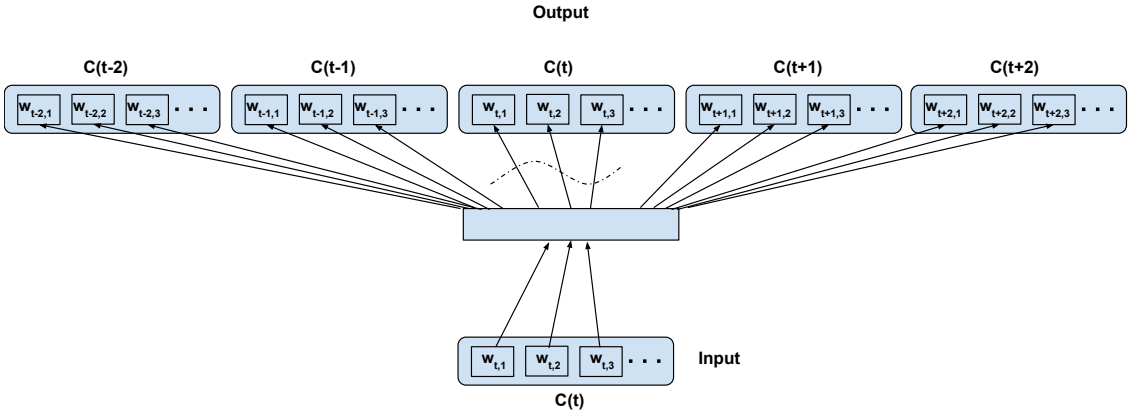


Figure 4.5: **Proposed Hybrid-Confusion Training Scheme for Confusion networks.**
$c(t)$ is a unit word confusion in the confusion network at a time-stamp t, i.e., $c(t)$ represents a set of arcs between two adjacent nodes of a confusion network, representing a set of confusable words.
$w_{t,i}$ is the i^{th} most probable word in the confusion $c(t)$.
Word confusions are sorted in decreasing order of their posterior probability:
$P(w_{t,1}) > P(w_{t,2}) > P(w_{t,3})...$
The dotted curved lines denote that the self-mapping is disallowed.

similarities for a word to be co-occurring, we expect to jointly model the co-occurrence of both the context and confusions. This also has the additional benefit of generating more training samples than the intra-confusion training. The number of training samples generated is given by:

$$\#\text{Samples} = \sum_{i=1}^{n} \sum_{\substack{j=i-S_w \\ j \neq i}}^{i+S_w} D_i \times D_j \tag{4.2}$$

where n is the total number of time steps, D_i is the number of word confusions at the i^{th} time step, S_w is the skip-window size (i.e., sample S_w words from history and S_w words from the future context of current word).

4.4.4 Hybrid Intra-Inter Confusion Training

Finally, we merge both the intra-confusion and inter-confusion training. This can be seen as a super-set of word2vec, inter-confusion and intra-confusion training configurations. Figure 4.5 illustrates the training configuration. The words $w_{t-t',i}$ (context) are predicted from $w_{t,j}$ (current word) for $i = 1, 2, 3 \ldots \text{length}(C(t-t'))$, $j = 1, 2, 3 \ldots \text{length}(C(t))$, $t' \in 1, 2, 0, -1, -2$ such that if $t' = 0$ then $i \neq j$; for skip-window of 2 for current confusion $C(t) \forall t$ as depicted in Figure 4.5. We simply add the combination of training samples from the above two proposed techniques (i.e., the number of samples is the sum of equation 4.1 and equation 4.2).

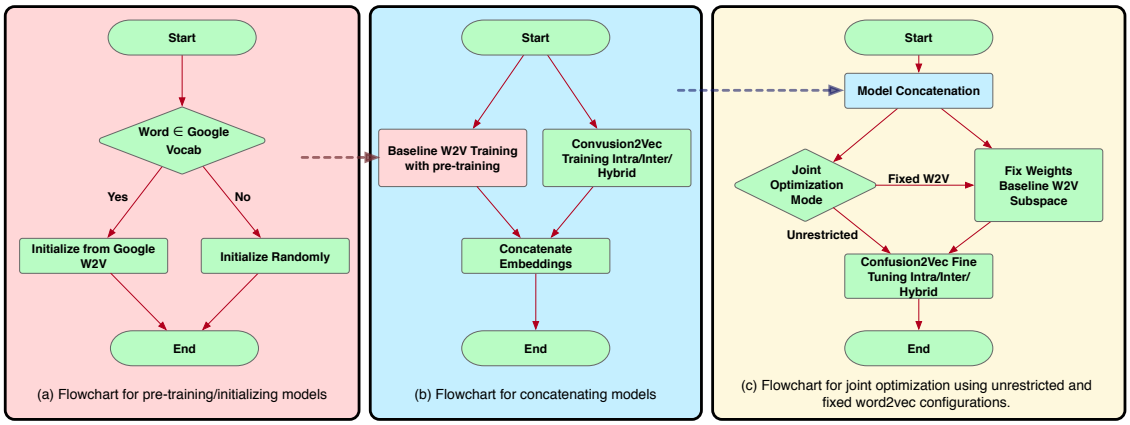


Figure 4.6: **Flowcharts for proposed training schemes**

4.5 Training Schemes

4.5.1 Model Initialization/Pre-training

Very often, it has been found that better model initializations lead to better model convergence [46]. This is more significant in the case of under-represented words. Moreover, for training the word confusion mappings, it would benefit to build upon the contextual word embeddings, since our final goal is in conjunction with both contextual and confusion information. Hence, we experiment initializing all our models with the original Google's word2vec model[1] trained on Google News dataset with 100 billion words as described by [115]. Pre-training rules are explained in the flowchart in Figure 4.6(a). For the words present in the Google's word2vec vocabulary, we directly initialize the embeddings with word2vec. The embeddings for rest of the words are randomly initialized following uniform distribution.

4.5.2 Model Concatenation

The hypothesis with model concatenation is that the two subspaces, one representing the contextual subspace (word2vec), and the other capturing the confusion subspace can be both trained independently and concatenated to give a new vector space which manifests both the information and hence a potentially useful vector word representation. Flowchart for model concatenation is shown in Figure 4.6(b). The model concatenation can be mathematically represented as:

$$NEW_{n\times e_1+e_2} = \begin{bmatrix} W2V_{n\times e_1} & C2V_{n\times e_2} \end{bmatrix} \tag{4.3}$$

[1] https://code.google.com/archive/p/word2vec/

where NEW is the new concatenated vector space of dimensions $n \times e_1 + e_2$, n is the vocabulary size, e_1 and e_2 are the embedding sizes of $W2V$ and $C2V$ subspaces respectively.

4.5.3 Joint Optimization

Further to the model concatenation scheme, one could fine-tune the new vector space representation to better optimize to the task criterion (fine-tuning involves re-training end-to-end with a relatively lower learning rate than usual). This could be viewed as a case of relaxing the strict independence between two subspaces as in the case of model concatenation. The fine-tuning itself could be either of the aforementioned proposed techniques. We specifically try two configurations of joint optimization:

4.5.3.1 Unrestricted

In this configuration, we optimize both the subspaces, i.e., the contextual (word2vec) and the confusion subspaces. The hypothesis is the fine-tuning allows the two subspaces to interact to achieve the best possible representation. The flowchart for the unrestricted joint optimization is displayed in Figure 4.6(c).

4.5.3.2 Fixed Word2Vec

In this configuration, we fix the contextual (word2vec) subspace and fine-tune only the confusion subspace. Since the word2vec already provides robust contextual representation, any fine-tuning on contextual space could possibly lead to sub-optimal state. Keeping the word2vec subspace fixed also allows the model to concentrate more specifically towards the confusion since the fixed subspace compensates for all the contextual mappings during training. This allows us to constrain the updatable parameters during joint optimization. It also allows for the possibility to directly use available word2vec models without modifications. The flowchart for the fixed Word2Vec joint optimization is displayed in Figure 4.6(c).

4.6 Evaluation Methods

Prior literature suggests, there are two prominent ways for evaluating the vector space representation of words. One is based on Semantic&Syntactic analogy task as introduced by [113]. The other common approach has been to assess the word similarities by computing the rank-correlation (Spearman's correlation) on human annotated word similarity databases [143] like WordSim-353 [49]. Although, the two evaluations can judge the vector representations of words efficiently for

Word Pair 1		Word Pair 2	
i'd	eyed	phi	fie
seedar	cedar	rued	rude
air	aire	spade	spayed
scent	cent	vile	vial
cirrus	cirrous	sold	soled
curser	cursor	pendant	pendent
sensor	censor	straight	strait

Table 4.1: **Few examples from Acoustic Analogy Task Test-set**

semantics and syntax of a language, we need to device an evaluation criteria for the word confusions, specifically for our case scenario - the acoustic confusions of words. For this, we formulate evaluations for acoustic confusions parallel to the analogy task and the word similarity task.

4.6.1 Analogy Tasks

4.6.1.1 Semantic&Syntactic Analogy Task

[113] introduced an analogy task for evaluating the vector space representation of words. The task was based on the intuition that the words, say "king" is similar to "man" in the same sense as the "queen" is to "woman" and thus relies on answering questions relating to such analogies by performing algebraic operations on word representations. For example, the analogy is correct if the vector("woman") is most similar to vector("king")-vector("man")+vector("queen"). The analogy question test set is designed to test both syntactic and semantic word relationships. The test set contains five types of semantic questions (8869 questions) and nine types of syntactic questions (10675 questions). Finally, the efficiency of the vector representation is measured using the accuracy achieved on the analogy test set. We employ this for testing the Semantic & Syntactic (contextual axis as in terms of Figure 4.1) relationship inherent in the vector space.

4.6.1.2 Acoustic Analogy Task

The primary purpose of the acoustic analogy task is to independently gauge the acoustic similarity information captured by the embedding model irrespective of the inherent Semantic and Syntactic linguistic information. Adopting similar idea and extending the same for evaluation of word confusions, we formulate the acoustic confusion analogy task (vertical context test as in terms of Figure 4.1) as follows. For similar sounding word pairs, "see" & "sea" and "red" & "read", the word vector "see" is similar to "sea" in the same sense as the word "red" is to "read". We set up an acoustic analogy question set on acoustically similar sounding words, more specifically

homophones. Table 4.1 lists a few examples from our data set. Detailed description of the creation of dataset is presented in section 4.7.3.1.

4.6.1.3 Semantic&Syntactic-Acoustic Analogy Task

Further, rather than evaluating the Semantic-Syntactic tasks and the acoustic analogy tasks independently, we could test for both together. Intuitively, the word vectors in each of the two subspaces should interact together. We would expect for an analogy, "see"-"saw":"take"-"took", the word "see" has a homophone alternative in "sea", thus there is a possibility of the word "see" being confused with "sea" in the new vector space. Thus an algebraic operation such as $vector(\text{“}see\text{”}) - vector(\text{“}saw\text{”}) + vector(\text{“}take\text{”})$ should be similar to $vector(\text{“}took\text{”})$ as before. Moreover the $vector(\text{“}sea\text{”}) - vector(\text{“}saw\text{”}) + vector(\text{“}take\text{”})$ should also be similar to $vector(\text{“}took\text{”})$. This is because we expect the $vector(\text{“}sea\text{”})$ to be similar to $vector(\text{“}see\text{”})$ under the acoustic subspace. We also take into account the more challenging possibility of more than one homophone word substitution. For example, $vector(\text{“}see\text{”}) - vector(\text{“}saw\text{”}) + vector(\text{“}allow\text{”})$ is similar to $vector(\text{“}allowed\text{”})$, $vector(\text{“}aloud\text{”})$ and $vector(\text{“}sea\text{”}) - vector(\text{“}saw\text{”}) + vector(\text{“}allow\text{”})$. The hypothesis is that to come up with such a representation the system should jointly model both the language semantic-syntactic relations and the acoustic word similarity relations between words. The task is designed to test Semantic-Acoustic relations and the Syntactic-Acoustic relationships. In other words, in terms of Figure 4.1, the task evaluates both the horizontal & vertical context

Type of Relationship	Word Pair 1		Word Pair 2	
Currency	India	Rupee	Korea	One (Won)
	Canada	Dollar	Denmark	Krona (Krone)
	Japan	Yen	Sweden	Krone (Krona)
Family	Buoy (Boy)	Girl	Brother	Sister
	Boy	Girl	King	Quean (Queen)
	Boy	Girl	Sun (Son)	Daughter
Adjective-to-Adverb	Calm	Calmly	Sloe (Slow)	Slowly
Opposite	Aware	Unaware	Possible	Impassible (Impossible)
Comparative	Bad	Worse	High	Hire (Higher)
Superlative	Bad	Worst	Grate (Great)	Greatest
Present Participle	Dance	Dancing	Rite (Write)	Writing
Past Tense	Dancing	Danced	Flying	Flu (Flew)
Plural	Banana	Bananas	Burred (Bird)	Birds
Plural Verbs	Decrease	Decreases	Fined (Find)	Finds
Multiple Homophone Substitutions	Wright (Write)	Writes	Sea (See)	Sees
	Rowed (Road)	Roads	I (Eye)	Ayes (Eyes)
	Si (See)	Seize (Sees)	Right (Write)	Writes

Table 4.2: **Few examples from Semantic & Syntactic - Acoustic Analogy Task Test Set**
The words in the parenthesis are the original ones as in the analogy test set [113] which have been replaced by their homophone alternatives.

Word1	Word2	Acoustic Rating	WordSim353
I	Eye	1.0	-
Adolescence	Adolescents	0.9	-
Allusion	Illusion	0.83	-
Sewer	Sower	0.66	-
Fighting	Defeating	0.57	7.41
Day	Dawn	0.33	7.53
Weather	Forecast	0.0	8.34

Table 4.3: **Examples of Acoustic Similarity Ratings**
Acoustic Rating: 1.0 = Identically sounding, 0.0 = Highly acoustically dissimilar
WordSim353: 10.0 = High word similarity, 0.0 = Low word similarity
Word pairs not present in WordSim353 is denoted by '-'

together. A few examples of this task is listed in Table 4.2. Section 4.7.3.2 details the creation of the database.

4.6.2 Similarity Ratings

4.6.2.1 Word Similarity Ratings

Along with the analogy task the word similarity task [49] has been popular to evaluate the quality of word vector representations in the NLP community [127, 104, 76, 143]. In this work we employ the WordSim-353 dataset [49] for the word similarity task. The dataset has a set of 353 word pairs with diverse range of human annotated scores relating to the similarity/dissimilarity of the two words. The rank-order correlation (Spearman correlation) between the human annotated scores and the cosine similarity of word vectors is computed. Higher correlation corresponds to better preservation of word similarity order represented by the word vectors, and hence better quality of the embedding vector space.

4.6.2.2 Acoustic Similarity Ratings

Employing a similar analogous idea to word similarity ratings and extending it to reflect the quality of word confusions, we formulate an acoustic word similarity task. The attempt is to have word pairs scored similar to as in WordSim-353 database, but with the scores reflecting the acoustic similarity. Table 4.3 lists a few randomly picked examples from our dataset. The dataset generation is described in section 4.7.3.3.

4.7 Data & Experimental Setup

4.7.1 Database

We employ Fisher English Training Part 1, Speech (LDC2004S13) and Fisher English Training Part 2, Speech (LDC2005S13) corpora [29] for training the ASR. The corpora consists of approximately 1915 hours of telephone conversational speech data sampled at 8kHz. A total of 11972 speakers were involved in the recordings. The speech corpora is split into three speaker disjoint subsets for training, development and testing for ASR modeling purposes. A subset of the speech data containing approximately 1905 hours were segmented into 1871731 utterances to train the ASR. Both the development set and the test set consists of 5000 utterance worth 5 hours of speech data each. The transcripts contain approximately 20.8 million word tokens with 42150 unique entries.

4.7.2 Experimental Setup

4.7.2.1 Automatic Speech Recognition

KALDI toolkit is employed for training the ASR [130]. A hybrid DNN-HMM based acoustic model is trained on high resolution (40 dimensional) Mel Frequency Cepstral Coefficients (MFCC) along with i-vector features to provide speaker and channel information for robust modeling. The CMU pronunciation dictionary [175] is pruned to corpora's vocabulary and is used as a lexicon for the ASR. A trigram language model is trained on the transcripts of the training subset data. The ASR system achieves a word error rates (WER) of 16.57% on the development and 18.12% on the test datasets. The decoded lattice is used to generate confusion network based on minimum bayes risk criterion [182]. The ASR output transcriptions resulted in a vocabulary size of 41274 unique word tokens.

4.7.2.2 Confusion2Vec

For training the Confusion2Vec, the training subset of the Fisher corpora is used. The total number of tokens resulting from the multiple paths over the confusion network is approximately 69.5 million words, i.e., an average of 3.34 alternative word confusions present for each word in the confusion network. A minimum frequency threshold of 5 is set to prune the rarely occurring tokens from the vocabulary, which resulted in the reduction of the vocabulary size from 41274 to 32848. Further, we also subsample the word tokens as suggested by [115] which was shown to be helpful. Both the frequency thresholding and the downsampling resulted in a reduction of

Task	Total Samples	Retained
Semantic&Syntactic Analogy	19544	11409
Acoustic Analogy	20000	2678
Semantic&Syntactic-Acoustic Analogy	7534	3860
WordSim-353	353	330
Acoustic Confusion Ratings	1372	943

Table 4.4: **Statistics of Evaluation Datasets**

word tokens from 69.5 million words to approximately 33.9 million words. The Confusion2Vec and Word2Vec are trained using the Tensorflow toolkit [1]. Negative Sampling objective is used for training as suggested for better efficiency [115]. For the skip-gram training, the batch-size of 256 was chosen and 64 negative samples were used for computing the negative sampling loss. The skip-window was set to 4 and was trained for a total of 15 epochs, since it provided optimal performance with traditional word2vec embeddings, evaluating for word analogy task, for the size of our database. During fine-tuning, the model was trained with a reduced learning rate and with other parameters unchanged. All the above parameters were fixed for consistent and fair comparison.

4.7.3 Creation of Evaluation Datasets

4.7.3.1 Acoustic Analogy Task

We collected a list of homophones in English [2], and created all possible combinations of pairs of acoustic confusion analogies. For homophones with more than 2 words, we list all possible confusion pairs. Few examples from the dataset are listed in Table 4.1. We emphasize that the consideration of only homophones in the creation of the dataset is a strict and a difficult task to solve, since the ASR lattice contains more relaxed word confusions.

4.7.3.2 Semantic&Syntactic-Acoustic Analogy Task

We construct an analogy question test set by substituting the words in the original analogy question test set from [113] with their respective homophones. Considering all the 5 types of semantic questions and 9 types of syntactic questions, for any words in the analogies with homophone alternatives, we swap with the homophone. We prune all the original analogy questions having no words with homophone alternatives. For analogies having more than one words with homophone alternatives, we list all permutations. We found that the number of questions generating by the

[2] http://homophonelist.com/homophones-list/ (Accessed: 2018-04-30)

above method, being exhaustive, was large and hence we randomly sample from the list to retain 948 semantic questions and 6586 syntactic questions. Table 4.2 lists a few examples with single and multiple homophone substitutions for Semantic&Syntactic-Acoustic Analogy Task from our data set.

4.7.3.3 Acoustic Similarity Task

To create a set of word pairs scored by their acoustic similarity, we add all the homophone word pairs with an acoustic similarity score of 1.0. To get a more diverse range of acoustic similarity scores, we also utilize all the 353 word pairs from the WordSim-353 dataset and compute the normalized phone edit distance using the CMU Pronunciation Dictionary [175]. The normalized phone edit distance is of the range between 0 and 1. The edit distance of 1 refers to the word pair having almost 0 overlap between their respective phonetic transcriptions and thus being completely acoustically dissimilar and vice-versa. We use $1 -$ phone-edit-distance as the acoustic similarity score between the word pair. Thus a score of 1.0 signifies that the two words are identically sounding, whereas as 0 refers to words sounding drastically dissimilar. In the case of a word having more than one phonetic transcriptions (pronunciation alternatives), we use the minimum normalized edit distance. Table 4.3 shows a few randomly picked examples from the generated dataset.

Finally, for evaluation the respective corpora are pruned to match the in-domain training dataset vocabulary. Table 4.4 lists the samples in each evaluation dataset before and after pruning.

4.7.4 Performance Evaluation Criterion

In the original work by [113], the efficiency of the vector representation is measured using the accuracy achieved on the analogy test set. But, in our case, note that the Semantic&Syntactic analogy task and the Semantic&Syntactic-Acoustic analogy task are mutually exclusive of each other. In other words, the model can get only one, either one of the analogies correct, meaning any increments with one task will result in decrements over the other task. Moreover, while jointly modeling two orthogonal information streams (i) contextual co-occurrences, and (ii) acoustic word confusions, finding the nearest word vector nearest to the specific analogy is no longer an optimal evaluation strategy. This is because the word vector nearest to the analogy operation can either be along the contextual axis or the confusion axis, i.e., each analogy could possibly have two correct answers. For example, the analogy "write"-"wrote" : "read" can be right when the nearest word vector is either "read" (contextual dimension) or "red" (confusion dimension). To incorporate this,

Model	Analogy Tasks				Similarity Tasks	
	S&S	Acoustic	S&S-Acoustic	Average Accuracy	Word Similarity	Acoustic Similarity
Google Word2Vec	**61.42%**	0.9%	16.99%	26.44%	**0.6893**	-0.3489
Word2Vec GroundTruth	35.15%	0.3%	7.86%	14.44%	0.5794	-0.2444
Baseline Word2Vec	34.27%	0.7%	11.27%	15.41%	0.4992	0.1944
Intra-Confusion	22.03%	52.58%	14.61%	29.74%	0.105*	**0.8138**
Inter-Confusion	36.15%	**60.57%**	20.44%	**39.05%**	0.2937	0.8055
Hybrid Intra-Inter	30.53%	53.55%	**29.35%**	37.81%	0.0963*	0.7858

Table 4.5: **Results: Different proposed models**
For the analogy tasks: the accuracies of baseline word2vec models are for top-1 evaluations, whereas of the other models are for top-2 evaluations (as discussed in Section 4.6.1). Detailed semantic analogy and syntactic analogy accuracies, the top-1 evaluations and top-2 evaluations for all the models are available under Appendix in Table A.1.
For the similarity tasks: all the correlations (Spearman's) are statistically significant with $p < 0.001$ except the ones with the asterisks. Detailed $p - values$ for the correlations are presented under Appendix in Table A.2.
S&S: Semantic & Syntactic Analogy.

we provide the accuracy over top-2 nearest vectors, i.e., we count the analogy question as correct if any of the top-2 nearest vector satisfies the analogy. This also holds for the acoustic confusion analogy tasks, especially for relations involving triplet homophones. For example, the analogy "write" - "right" : "road" can be right when the nearest word vector is either "rode" or "rowed" (for triplet homophones "road"/"rode"/"rowed"). Thus, we present evaluations by comparing the top-1 (nearest vector) evaluation with baseline word2vec against the top-2 evaluation for the proposed confusion2vec models. To maintain consistency, we also provide the top-2 evaluations for the baseline word2vec models in the appendix.

Moreover, since we have 3 different analogy tasks, we provide the average accuracy among the 3 tasks in order to have an easy assessment of the performance of various proposed models.

4.8 Results

Table 6.1 lists the results for various models. We provide evaluations on three different analogy tasks and two similarity tasks as discussed in Section 4.6. Further, more thorough results with the Semantic and Syntactic accuracy splits are provided under the appendix to gain deeper insights.

4.8.1 Baseline Word2Vec Model

We consider 3 variations of Word2Vec baseline model. First, we provide results with the Google's Word2Vec model [3] which is trained with orders more training data, and is thus a high upper bound on the Semantic&Syntactic task. The Google's Word2Vec model was pruned to match

[3] https://code.google.com/archive/p/word2vec

the vocabulary of our corpora to make the evaluation comparable. Second, we consider the Word2Vec model trained on the in-domain ground truth transcripts. Third, for a more fair comparison with the other proposed models, we provide evaluations on Word2Vec model trained on the noisy ASR output transcripts. All the three baseline models result in good performance on Semantic&Syntactic analogy tasks and word similarity task as expected. The Google's model achieves an accuracy of 61.42% on the Semantic&Syntactic analogy task. We note that the Syntactic accuracy (70.79%) is much higher than the Semantic accuracy (28.98%) (see Appendix Table A.1). This could be due to our pruned evaluation test set (see Table 4.4). Both the in-domain models improve on the Semantic accuracy while losing on the syntactic accuracy over the Google model (see Appendix Table A.1). The shortcomings of the in-domain models compared to the Google Word2Vec on the Semantic&Syntactic analogy task can be attributed towards the amount of training data and its extensive vocabulary. The in-domain models are trained on 20.8 million words versus the 100 billion of Google's News dataset. Moreover the vocabulary of in-domain models are approximately 42,150 versus the 3 million of Google [115] and thus unfair to compare with rest of the models. Comparing the two in-domain models, we observe the model trained on clean data performs better than the one trained on ASR transcripts as expected. However, the performance difference is minimal which is encouraging. We see the noisy transcripts negatively affect the semantic accuracies while the syntactic accuracy remains identical which makes sense. Further, evaluating the Acoustic analogy and Semantic&Syntactic-Acoustic analogy tasks, all the three baseline models perform poorly. An unusual thing we note is that the Google W2V model performs better comparatively to the other baseline models in the Semantic&Syntactic-Acoustic analogy task. A deeper examination revealed that the model compensates well for homophone substitutions on Semantic&Syntactic analogies which have very similar spellings. This suggests that the typographical errors present in the training data of the Google model results in a small peak in performance for the Semantic&Syntactic-Acoustic analogy task.

On the evaluations of similarity tasks, all the baseline models perform well on the word similarity tasks as expected. However, they exhibit poor results on the acoustic similarity task. One interesting observation is Google Word2Vec and the in-domain Word2Vec model trained on clean transcript show negative correlation, whereas the model trained on noisy transcript shows a small positive correlation. One of the possible reasons behind this is due to the influence of the ASR language model on the word confusions in the lattice which enforces contextual constraints during ASR decoding and hence results in a positive correlation.

Overall, the results indicate that the baseline models are largely inept of capturing any relationships over the acoustic word confusions present in a confusion network or a lattice. In our specific case, the baseline models are poor in capturing relationships between acoustically similar words.

4.8.2 Intra-Confusion

With intra-confusion training, we expect the model to capture acoustically similar word relationships, while completely ignoring any contextual relations. Hence, we expect the model to perform well on acoustic analogies and acoustic similarity tasks and to perform poorly on Semantic&Syntactic analogies and word similarity tasks. The Table 6.1 lists the results obtained using intra-confusion training. The results are in conjunction with our expectations. The model gives the worst results in Semantic&Syntactic analogy task. However, we observe that the syntactic analogy accuracy to be a fair amount higher than the semantic accuracy (see Appendix Table A.1). We think this is mainly because of syntactically similar words appearing along the word confusion dimension in the confusion networks, resultant of the constraints enforced on the confusion network by the (ASR) language model - which are known to perform better for syntactic tasks [113]. The model also gives the highest correlation on the acoustic similarity task, while performing poorly on the word similarity task.

4.8.3 Inter-Confusion

With inter-confusion training, we hypothesized that the model is capable of jointly modeling both the contextual statistics as well as the word confusion statistics. Hence, we expect the model to perform well on both the Semantic&Syntactic analogy and Acoustic analogy tasks and in doing so result in better performance with Semantic&Syntactic-Acoustic analogy task. We also expect the model to give high correlations for both word similarity and acoustic similarity tasks. From Table 6.1, we observe that as hypothesized the inter-confusion training shows improvements in the Semantic&Syntactic analogy task. Quite surprisingly, the inter-confusion training shows better performance than the intra-confusion training for the Acoustic analogy task, hinting that having good contextual representation could mutually be beneficial for the confusion representation. However, we don't observe any improvements in the Semantic&Syntactic-Acoustic analogy task. Evaluating on the similarity tasks, the results support the observations drawn from analogy tasks, i.e., the model fares relatively well in both word similarity and acoustic similarity.

Model	Analogy Tasks				Similarity Tasks	
	S&S	Acoustic	S&S-Acoustic	Average Accuracy	Word Similarity	Acoustic Similarity
Baseline Word2Vec	61.13%	0.9%	16.66%	26.23%	**0.6036**	-0.4327
Intra-Confusion	63.97%	16.92%	**43.34%**	41.41%	0.5228	0.62
Inter-Confusion	**65.45%**	**27.33%**	38.29%	**43.69%**	0.5798	0.5825
Hybrid Intra-Inter	65.19%	20.35%	42.18%	42.57%	0.5341	**0.6237**

Table 4.6: **Results with pre-training/initialization**
For the analogy tasks: the accuracies of baseline word2vec models are for top-1 evaluations, whereas of the other models are for top-2 evaluations (as discussed in Section 4.6.1). Detailed semantic analogy and syntactic analogy accuracies, the top-1 evaluations and top-2 evaluations for all the models are available under Appendix in Table A.3.
For the similarity tasks: all the correlations (Spearman's) are statistically significant. Detailed $p-values$ for the correlations are presented under Appendix in Table A.4.
S&S: Semantic & Syntactic Analogy.

4.8.4 Hybrid Intra-Inter Confusion

The hybrid intra-inter confusion training shows comparable performance in jointly modeling on both the Semantic&Syntactic and Acoustic analogy tasks. One crucial observation is that it gives significantly better performance with the Semantic&Syntactic-Acoustic analogy task. This suggests that jointly modeling both the intra-confusion and inter-confusion word mappings are useful. However, it achieves better results by compromising on the semantic analogy (see Appendix Table A.1) accuracy and hence also negatively affecting the word similarity task. The model achieves good correlation on the acoustic similarity task.

Overall, our proposed Confusion2Vec models capture significantly more useful information compared to the baseline models judging by the average accuracy over the analogy tasks. One particular observation we see across all the proposed models is that the performance remains fairly poor for the Semantic&Syntactic-Acoustic analogy task. This suggests that the Semantic&Syntactic-Acoustic analogy task is inherently hard to solve. We believe that to achieve better results with Semantic&Syntactic-Acoustic analogies, it is necessary to have robust performance on one of the tasks (Semantic&Syntactic analogies or Acoustic analogies) to begin with, i.e., better model initialization could help. Next, we experiment with model initializations/pre-training.

4.8.5 Model Initialization/Pre-training

Table 4.6 lists the results with model initialization/pre-training. The baseline model is initialized from the Google Word2Vec model. Rest of the models are initialized from the baseline word2vec model (i.e., the baseline model initialized from Google Word2Vec), since this would enable full

compatibility with the vocabulary. Since the Google Word2Vec model is 300 dimensional, this forces all the pre-trained models (in Table 4.6) to be 300, opposed to 256 dimensions (in Table 6.1).

Pre-training the baseline model provides improvements with Semantic&Syntactic analogy results to be close and comparable to that of the Google's Word2Vec model. For intra-confusion model, the pre-training provides drastic improvements on Semantic&Syntactic analogy task at the expense of the Acoustic analogy task. Even-though the accuracy of Acoustic analogy task decreases comparatively to without pre-training, it remains significantly better than the baseline model. More importantly, the Semantic&Syntactic-Acoustic analogy task accuracy doubles. Inter-Confusion model does not compromise on the Semantic&Syntactic analogy tasks, in doing so gives comparable performances to the baseline model. Additionally it also does well on the Acoustic and Semantic&Syntactic-Acoustic analogy task as was the case without pre-training. In the case of hybrid intra-inter confusion model, similar trends are observed as was with no pre-training, but with considerable improvements in accuracies. Pre-training also helps in boosting the correlations for the word similarity tasks for all the models. Overall, we find the pre-training to be extremely useful.

4.8.6 Model Concatenation

The first 4 rows of Table 4.7 show the results with model concatenation. We concatenate each of the three proposed models (from Table 6.1) with the pre-trained baseline word2vec. Thus the resulting vector space is 556 dimensional (300 (pre-trained baseline word2vec) + 256 (proposed models from Table 6.1) = 556). In our case, we believe the dimension expansion of the vector space is insignificant in terms of performance considering the relatively low amount of training data compared to Google's word2vec model. To be completely fair in judgment, we create a new baseline model with 556 dimensional embedding space for comparison. To train the new baseline model, the 556 dimension embedding was initialized with 300 dimensional Google's word2vec embedding and the rest of the dimensions as zeros (null space). Comparing the 556 dimension baseline from Table 4.7 with the previous 300 dimensional baseline from Table 4.6, the results are almost identical which confirms the dimension expansion is insignificant with respect to performance.

With model concatenation, we see slightly better results (average analogy accuracy) comparing with the pre-trained models from Table 4.6, an absolute increase of up-to approximately 5% among the best results. The correlations with similarity tasks are similar and comparable with the earlier results with the pre-trained models.

	Model	Fine-tuning Scheme	Analogy Tasks				Similarity Tasks	
			S&S	Acoustic	S&S-Acoustic	Average	Word	Acoustic
1	Baseline Word2Vec (556 dim.)	-	61.13%	0.93%	16.53%	26.2%	**0.5973**	-0.4341
	Model Concatenation							
2	Word2Vec (F) + Intra-Confusion (F)	-	67.03%	25.43%	40.36%	44.27%	0.5102	0.7231
3	Word2Vec (F) + Inter-Confusion (F)	-	70.84%	35.25%	35.18%	47.09%	0.5609	0.6345
4	Word2Vec (F) + Hybrid Intra-Inter (F)	-	68.08%	11.39%	41.3%	40.26%	0.4142	0.5285
	Fixed Word2Vec Joint Optimization							
5	Word2Vec (F) + Intra-Confusion (L)	inter	71.65%	20.54%	33.76%	41.98%	0.5676	0.4437
6	Word2Vec (F) + Intra-Confusion (L)	intra	67.37%	28.64%	39.09%	45.03%	0.5211	0.6967
7	Word2Vec (F) + Intra-Confusion (L)	hybrid	70.02%	25.84%	37.18%	44.35%	0.5384	0.6287
8	Word2Vec (F) + Inter-Confusion (L)	inter	72.01%	35.25%	33.58%	46.95%	0.5266	0.5818
9	Word2Vec (F) + Inter-Confusion (L)	intra	69.7%	39.32%	39.07%	49.36%	0.5156	0.7021
10	Word2Vec (F) + Inter-Confusion (L)	hybrid	**72.38%**	37.75%	37.95%	49.36%	0.5220	0.6674
11	Word2Vec (F) + Hybrid Intra-Inter (L)	inter	71.36%	8.55%	33.21%	37.71%	0.5587	0.302
12	Word2Vec (F) + Hybrid Intra-Inter (L)	intra	66.85%	13.33%	40.1%	40.09%	0.4996	0.5691
13	Word2Vec (F) + Hybrid Intra-Inter (L)	hybrid	68.32%	11.61%	38.19%	39.37%	0.5254	0.4945
	Unrestricted Joint Optimization							
14	Word2Vec (L) + Intra-Confusion (L)	inter	62.12%	46.42%	36.4%	48.31%	0.5513	0.7926
15	Word2Vec (L) + Intra-Confusion (L)	intra	64.85%	40.55%	42.38%	49.26%	0.5033	0.7949
16	Word2Vec (L) + Intra-Confusion (L)	hybrid	31.65%	61.91%	23.55%	39.04%	0.1067*	**0.8309**
17	Word2Vec (L) + Inter-Confusion (L)	inter	64.98%	52.99%	34.79%	50.92%	0.5763	0.7725
18	Word2Vec (L) + Inter-Confusion (L)	intra	65.88%	49.4%	41.51%	**52.26%**	0.5379	0.7717
19	Word2Vec (L) + Inter-Confusion (L)	hybrid	37.86%	**67.21%**	25.96%	43.68%	0.2295	0.8294
20	Word2Vec (L) + Hybrid Intra-Inter (L)	inter	65.54%	27.97%	36.87%	43.46%	0.5338	0.6953
21	Word2Vec (L) + Hybrid Intra-Inter (L)	intra	64.42%	20.05%	**42.56%**	42.34%	0.4920	0.6942
22	Word2Vec (L) + Hybrid Intra-Inter (L)	hybrid	65.79%	22.63%	41.3%	43.24%	0.4967	0.6986

Table 4.7: **Model concatenation and joint optimization results**
Acronyms: (F):Fixed embedding, (L):Learn embedding during joint training, S&S: Semantic & Syntactic Analogy.
For the analogy tasks: the accuracies of baseline word2vec models are for top-1 evaluations, whereas of the other models are for top-2 evaluations (as discussed in Section 4.6.1). Detailed semantic analogy and syntactic analogy accuracies, the top-1 evaluations and top-2 evaluations for all the models are available under Appendix in Table A.5.
For the similarity tasks: all the correlations (Spearman's) are statistically significant with $p < 0.001$ except the ones with the asterisks. Detailed $p - values$ for the correlations are presented under Appendix in Table A.6.

4.8.7 Joint Optimization

4.8.7.1 Fixed Word2Vec

Rows 5-13 of Table 4.7 display the results of joint optimization with concatenated, fixed Word2Vec embeddings and learn-able confusion2vec embeddings. As hypothesized with fixed Word2Vec subspace, the results indicate better accuracies for the Semantic&Syntactic analogy task. Thereby, the improvements also reflect on the overall average accuracy of the analogy tasks. This also confirms the need for joint optimization which boosts the average accuracy up-to approximately 2% absolute over the unoptimized concatenated model.

4.8.7.2 Unrestricted Optimization

The last 9 rows of Table 4.7 display the results obtained by jointly optimizing the concatenated models without constraints. Both the subspaces are fine tuned to convergence with various proposed training criteria. We consistently observe improvements with the unrestricted optimization over the unoptimized model concatenations. In terms of average accuracy we observe a increase in average accuracy by up-to 5% absolute approximate over the unoptimized concatenated models. Moreover, we obtain improvements over the Fixed Word2Vec joint-optimized models, up-to 2-3% (absolute) in average accuracies. The best overall model in terms of average accuracies is obtained by unrestricted joint optimization on the concatenated baseline word2vec and inter-confusion models by fine-tuning with the intra-confusion training scheme.

4.8.8 Results Summary

Firstly, comparing among the different training schemes (see Table 6.1), the inter-confusion training consistently gives the best Acoustic analogy accuracies, whereas the hybrid training scheme often gives the best Semantic&Syntactic-Acoustic analogy accuracies. As far as the Semantic&Syntactic analogy task is concerned, the intra-confusion is often found to give preference to syntactic relations, while the inter-confusion boosts the semantic relations and the hybrid scheme balances both relations (see Appendix Table A.1). Next, pre-training/initializing the model gives drastic improvements in overall average accuracy of analogy tasks. Concatenating the baseline word2vec with the confusion2vec model gives slightly better results. More optimizations and fine-tuning over the concatenated model gives considerably the best results.

Overall, the best results are obtained with unrestricted joint optimization of baseline word2vec and inter-confusion model, i.e., fine-tuning using intra-confusion training mode. In terms of average analogy accuracies the confusion2vec model outperforms the baseline by an absolute 26.06%. The best performing confusion2vec model outperforms the word2vec model even on the Semantic&Syntactic analogy tasks (by a relative 7.8%). Moreover, even the comparison between the top-2 evaluations of both the word2vec and confusion2vec suggests very similar performance on Semantic&Syntactic-analogy tasks (see Appendix Table A.5). This confirms and emphasizes that the confusion2vec doesn't compromise on the information captured by word2vec but succeeds in augmenting the space with word confusions. Another highlight observation is that modeling the word confusions boost the semantic and syntactic scores of the Semantic&Syntactic analogy task (compared to word2vec), suggesting inherent information in word confusions which could be exploited for better semantic-syntactic word relation modeling.

4.9 Vector Space Analysis

In this section, we compare the vector space plots of the typical word2vec space and the proposed confusion2vec vector space for specifically chosen set of words. We choose a subset of words representing three categories to reflect semantic relationships, syntactic relationships and acoustic relationships. The vector space representations of the words are then subjected to dimension reduction using principle component analysis (PCA) to obtain 2D vectors which are used for plotting.

4.9.1 Semantic Relationships

For analyzing the semantic relationships, we compile random word pairs (constrained by the availability of these in our training data) representing Country-Cities relationships. The 2D plot for baseline pre-trained word2vec model is shown in Figure 4.9 and for the proposed confusion2vec model, specifically for the randomly selected, jointly-optimized word2vec + intra-confusion model (corresponding to row 6 in Table 4.7) is displayed in Figure 4.10. The following observations can be made comparing the two PCA plots:

- Examining the baseline word2vec model, we find the Cities are clustered over the upper half of the plot (highlighted with blue hue in Figure 4.9) and Countries are clustered together at the bottom half (highlighted with red hue in Figure 4.9).

- Similar trends are observed with the proposed confusion2vec model, where the cities are clustered together over the right half of the plot (highlighted with blue hue in Figure 4.10) and the countries are grouped together towards the left half (highlighted with red hue in Figure 4.10).

- In the Word2Vec space, the vectors of Country-City word pairs are roughly parallel, pointing north-east (i.e., vectors are approximately similar).

- Similar to the word2vec space, with the Confusion2Vec, we observe the vectors of Country-City word pairs are fairly parallel and point to the east (i.e., vectors are highly similar).

The four observations indicate that the Confusion2Vec preserves the Semantic relationships between the words (similar to the Word2Vec space).

4.9.2 Syntactic Relationships

To analyze the syntactic relationships, we create 30 pairs of words composed of Adjective-Adverb, Opposites, Comparative, Superlative, Present-Participle, Past-tense, Plurals. The PCA 2D plots for baseline pre-trained word2vec model and the proposed confusion2vec model are illustrated in Figure 4.11 and Figure 4.12 respectively. The following inferences can be made from the two plots:

- Inspecting the baseline word2vec model, we observe that the word pairs depicting syntactic relations occur often close-by (highlighted with red ellipses in Figure 4.11).
- Few semantic relations are also apparent and are highlighted with blue ellipses in Figure 4.11. For example, animals are clustered together.
- Similarly, with the Confusion2Vec model, we observe syntactic clusters of words highlighted with red ellipses in Figure 4.12.
- Semantic relations apparent in the case of word2vec is also evident with the Confusion2Vec, which are highlighted with blue ellipses in Figure 4.12.
- Additionally with the Confusion2Vec model, we find clusters of acoustically similar words (with similar phonetic transcriptions). These are highlighted using a green ellipse in Figure 4.12.

The above findings confirm that the confusion2vec models preserve the syntactic relationships similar to word2vec models, supporting our hypothesis.

4.9.3 Acoustic Relationships

In order to analyze the relationships of similarly sounding words in the word vector spaces under consideration, we compose 20 pairs of acoustically similar sounding words, with similar phonetic transcriptions. The 2D plot obtained after PCA for the baseline word2vec model is shown in Figure 4.13 and the proposed confusion2vec model is shown in Figure 4.14. We make the following observations from the two figures:

- Observing the baseline Word2vec model, no apparent trends are found between the acoustically similar words. For example, there is no trivial relationships apparent from the plot in Figure 4.13 between the word “no” and “know”, “try” and “tri” etc.

Example	Ground-truth	ASR output	W2V Similarity	C2V Similarity
1.1	"yes right answer"	"yes [right/write] answer"	0.96190	0.96218
1.2	"yes right answer"	"yes write answer"	0.93122	0.93194
1.3	"yes write answer"	"yes [right/write] answer"	0.99538	0.99548
1.4	"yes rite answer"	"yes [right/write] answer"	0.84216	0.88206
1.5	"yes rite answer"	"yes right answer"	0.86003	0.87085
1.6	"yes rite answer"	"yes write answer"	0.82073	0.87034
2.1	"she likes sea"	"[she/shea] likes [see/sea]"	0.91086	0.92130
2.2	"she likes sea"	"shea likes see"	0.73295	0.77137
2.3	"shea likes see"	"[she/shea] likes [see/sea]"	0.94807	0.95787
2.4	"shea likes see"	"[she/shea] likes [see/rocket]"	0.93560	0.93080
2.5	"she likes sea"	"[she/shea] likes [see/rocket]"	0.85853	0.85757

Table 4.8: **Cosine Similarity between the ASR Ground-truth and ASR output in application to ASR error correction for baseline pre-trained word2vec and the proposed confusion2vec: jointly optimized intra-confusion + baseline word2vec models**

Example 1.1-1.6 inherits structure as in Figure 4.7a, i.e., "yes [right/write] answer" assigns weight of 1.0 to yes and answer, 0.75 to right, 0.25 to write. Similarly Example 2.1-2.5 inherits structure as in Figure 4.7b

(a) Example 1

(b) Example 2

Figure 4.7: **Confusion Network Examples**

- However, inspecting the proposed confusion2vec model, there is an obvious trend apparent, the acoustically similar words are grouped together in pairs and occur roughly in similar distances. The word pairs are highlighted with blue ellipses in Figure 4.14.
- Additionally, in the Figure 4.14, as highlighted in green ellipse, we observe the 4 words "no", "not", "knot" and "know" occur in close proximity. The word pair "no" and "not" portray Semantic/Syntactic relation whereas the pairs "knot" & "not" and "no" & "know" are acoustically related.

The above findings suggest that the word2vec baseline model fails to capture any acoustic relationships whereas the proposed confusion2vec successfully models the confusions present in the lattices, in our specific case the acoustic confusions from the ASR lattices.

4.10 Discussion

In this section, we demonstrate why the proposed embedding space is superior for modeling word lattices with the support of toy examples. Lets consider a simple task of ASR error correction.

As shown by [4, 124, 151], often, the information needed to correct the errors are embedded in the lattices. The toy examples in Figure 4.7a & 4.7b depict the real scenarios encountered in ASR. The lattice feature representation is a weighted vector sum of all words in the confusion and its context present in the lattice (see Figure 4.8). We compare the proposed confusion2vec embeddings with the popular word2vec using cosine similarity as the evaluation measure. Table 4.8 lists the evaluation for the following cases: (i) ASR output is correct, (ii) ASR output is wrong and the correct candidate is present in the lattice, (iii) ASR output is wrong and the correct candidate is absent from the lattice, and (iv) ASR output is wrong and with no lattice available. The following observations are drawn from the results:

1. Confusion2vec shows higher similarity with the correct answers when the ASR output is correct (see Table 4.8 example 1.1, 2.1).

2. Confusion2vec exhibits higher similarity with the correct answers when the ASR output is wrong - meaning the representation is closer to the correct candidate and therefore more likely to correct the errors (see Table 4.8 example 1.2, 2.2, 1.3, 2.3).

3. Confusion2vec yields high similarity even when the correct word candidate is not present in the lattice - meaning confusion2vec leverages inherent word representation knowledge to aid re-introduction of pruned or unseen words during error correction (see Table 4.8 example 1.4, 1.5, 1.6).

4. The confusion2vec shows low similarity in the case of fake lattices with highly unlikely word alternatives (see Table 4.8 example 2.4, 2.5).

All the above observations are supportive of the proposed confusion2vec word representation and is in line with the expectations for the task of ASR error correction.

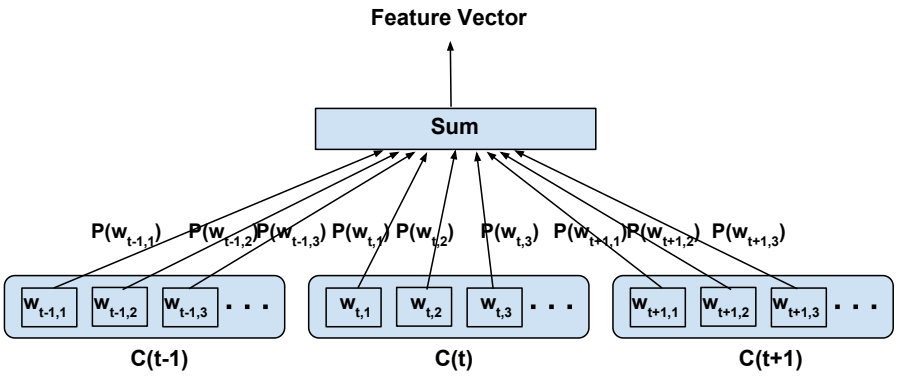


Figure 4.8: **Computation of lattice feature vector.**

4.11 Potential Applications

In addition to the above discussed ASR error correction task, other potential application include:

Machine Translation: In Machine Translation, word lattices are used to provide multiple sources for generating a single translation [144, 44]. Word lattices derived from reordered hypotheses [31, 120, 66], morphological transformations [43, 66], word segmentations [41], paraphrases [125] are used to introduce ambiguity and alternatives for training machine translation systems [178, 42, 44]. Source language alternatives can also be exploited by introducing ambiguity derived from the combination of multiple machine translation systems [110, 139, 138]. *In the case of Machine Translation, the word-confusion subspace is associated with morphological transformations, word segmentations, paraphrases, part-of-speech information, etc., or a combination of them.* Although the word-confusion subspace is not orthogonal, the explicit modeling of such ambiguity relationships is beneficial.

NLP: Other NLP based applications like paraphrase generation [132], word segmentation [87], part-of-speech tagging [87] also operate on lattices. As discussed in section 4.2.2, confusion2vec can exploit the ambiguity present in the lattices for betterment of the tasks.

ASR: In ASR systems, word lattices and confusion networks are often re-scored using various algorithms to improve their performances by exploiting ambiguity [164, 105, 181, 102]. *In the case of ASR, the word-confusion subspace is associated with acoustic similarity of words which is often orthogonal to the semantic-syntactic subspace as discussed in section 4.2.1.* Example 1, Example 2 and Example 3 are prime cases supporting the need for jointly modeling acoustic word confusions and semantic-syntactic subspace.

Spoken Language Understanding: Similarly, as in the case of ASR, Confusion2Vec could exploit the inherent acoustic word-confusion information for keyword spotting [105], confidence score estimation [105, 147, 81, 78], domain adaptation [151], lattice compression [105], spoken content retrieval [24, 74], system combinations [105, 72] and other spoken language understanding tasks [64, 170, 106] which operate on lattices.

Speech Translation: In speech translation systems, incorporating the word lattices and confusion networks (instead of the single top hypothesis) is beneficial in better integrating speech recognition system to the machine translation systems [12, 108, 109, 145]. Similarly, exploiting uncertainty information between the "ASR - Machine Translation - Speech synthesis" systems for Speech-to-speech translation is useful [92, 174]. Since speech translation involves combination of ASR and the Machine Translation systems, the word-confusion subspace is associated with a

combination of acoustic word-similarity (for ASR) and morphological-segmentation-paraphrases ambiguities (for Machine Translation).

"See son winter is here" $\longrightarrow$ "voir fils hiver est ici" (Example 4)

"Season winter is here" $\longrightarrow$ "saison hiver est ici" (Example 5)

Example 4 and Example 5 demonstrate a case of speech translation of identically sounding English phrases to French. Words "See son" and "Season" demonstrate ambiguity in terms of word segmentation. Whereas the phrases "See son" and "Season" also exhibit ambiguity in terms of acoustic similarity. By modeling both word-segmentation and acoustic-confusion through word vector representations, the confusion2vec can provide crucial information that the french words "voir" and "saison" are confusable under speech translation framework.

Optical Character Recognition: In optical character recognition (OCR) systems, the confusion axis is related to pictorial structures of the characters/words. For example, say the characters "a" and "o" are easily confusable thus leading to similar character vectors in the embedding space. In the case of word level confusions leading to words "ward" and "word" being similar with confusion2vec (word2vec would have the words "word" and "ward" fairly dissimilar). Having this crucial optical confusion information is useful during OCR decoding on sequence of words when used in conjunction with the linguistic contextual information.

Image/Video Scene Summarization: The task of scene summarization involves generating descriptive text summarizing the content in one or more images. Intuitively, the task would benefit from linguistic contextual knowledge during the text generation. However, with the confusion2vec, one can model and expect to capture two additional information streams (i) pictorial confusion of image/object recognizer, and (ii) pictorial context, i.e., modeling objects occurring together (e.g. we can expect oven to often appear nearby a stove or other kitchen based appliances). The additional streams of valuable information embedded in the lattices can contribute for better decoding. In other words, for example, word2vec can exhibit high dissimilarity between the words "lifebuoy" and "donuts", however the confusion2vec can capture their pictorial similarity in a better word space representation and thus aiding in their end application of scene summarization.

4.12 Conclusion

In this work, we proposed a new word vector representation motivated from human speech & perception and aspects of machine learning for incorporating word confusions from lattice like

structures. The proposed confusion2vec model is meant to capture additional word-confusion information and improve upon the popular word2vec models without compromising the inherent information captured by the word2vec models. Although the word confusions could be domain/task specific, we present a case study on ASR lattices where the confusions are based on acoustic similarity of words. Specifically, with respect to ASR related applications, the aim is to capture the contextual statistics, as with word2vec, and additionally also capture the acoustic word confusion statistics. Several training configurations are proposed for confusion2vec model, each differing in the utilization of the embedded information present in the lattice or confusion network for modeling the word vector space. Further, techniques like pre-training/initializations, model concatenation and joint optimization are proposed and evaluated for the confusion2vec models. Appropriate evaluation schemes are formulated for the domain specific application. The evaluation schemes are inspired from the popular analogy based question test set and word similarity tasks. A new analogy task and word similarity tasks are designed for the acoustic confusion/similarity scenario. A detailed tabulation of results are presented for the confusion2vec model and compared to the baseline word2vec models.

The results show that the confusion2vec can augment additional task-specific word confusion information without compromising on the semantic and syntactic relationships captured by the word2vec models. Next, detailed analysis is conducted on the confusion2vec vector space through PCA reduced 2-dimensional plots for three independent word relations: (i) Semantic relations, (ii) Syntactic relations, and (iii) Acoustic relations. The analysis further supports our aforementioned experimental inferences. Few toy examples are presented towards the task of ASR error correction to support the adequacy of the Confusion2vec over the word2vec word representations. The study validates through various hypotheses and test results, the potential benefits of the confusion2vec model.

4.13 Future Work

In future, we plan to work on improving the confusion2vec model by incorporating the sub-word and phonemic transcription of words during training. Sub-words and character transcription information is shown to improve the word vector representation [17, 26]. We believe the sub-words and phoneme transcriptions of words are even more relevant to confusion2vec than characters. In addition to the improvements expected towards the semantic and syntactic representations (word2vec), since the sub-words and phoneme transcriptions of acoustically similar words are similar, it should help in modeling the confusion2vec to a much greater extent.

Apart from concentrating on improving the confusion2vec model, this work opens new possible opportunities in incorporating the confusion2vec embeddings to a whole range of full-fledged applications such as ASR error correction, Speech translation tasks, Machine translation, Discriminative language models, Optical character recognition, Image/Video scene summarization etc.

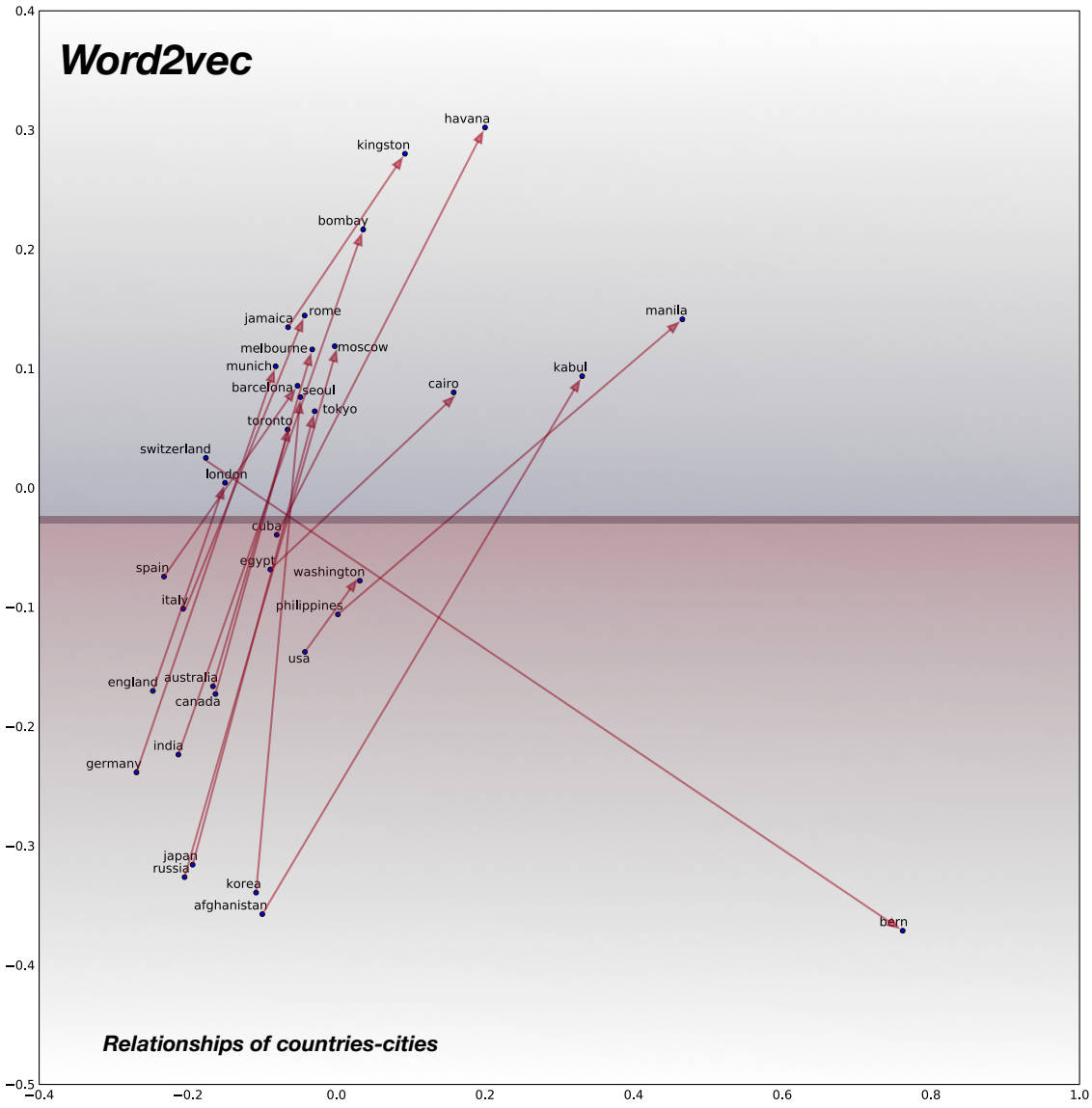


Figure 4.9: **2D plot after PCA of word vector representation on baseline pre-trained word2vec**
Demonstration of Semantic Relationship on Randomly chosen pairs of Countries and Cities
Country-City vectors are almost parallel/similar. Countries are clustered together on the bottom half (highlighted with red hue) and the cities on upper half (highlighted with blue hue).

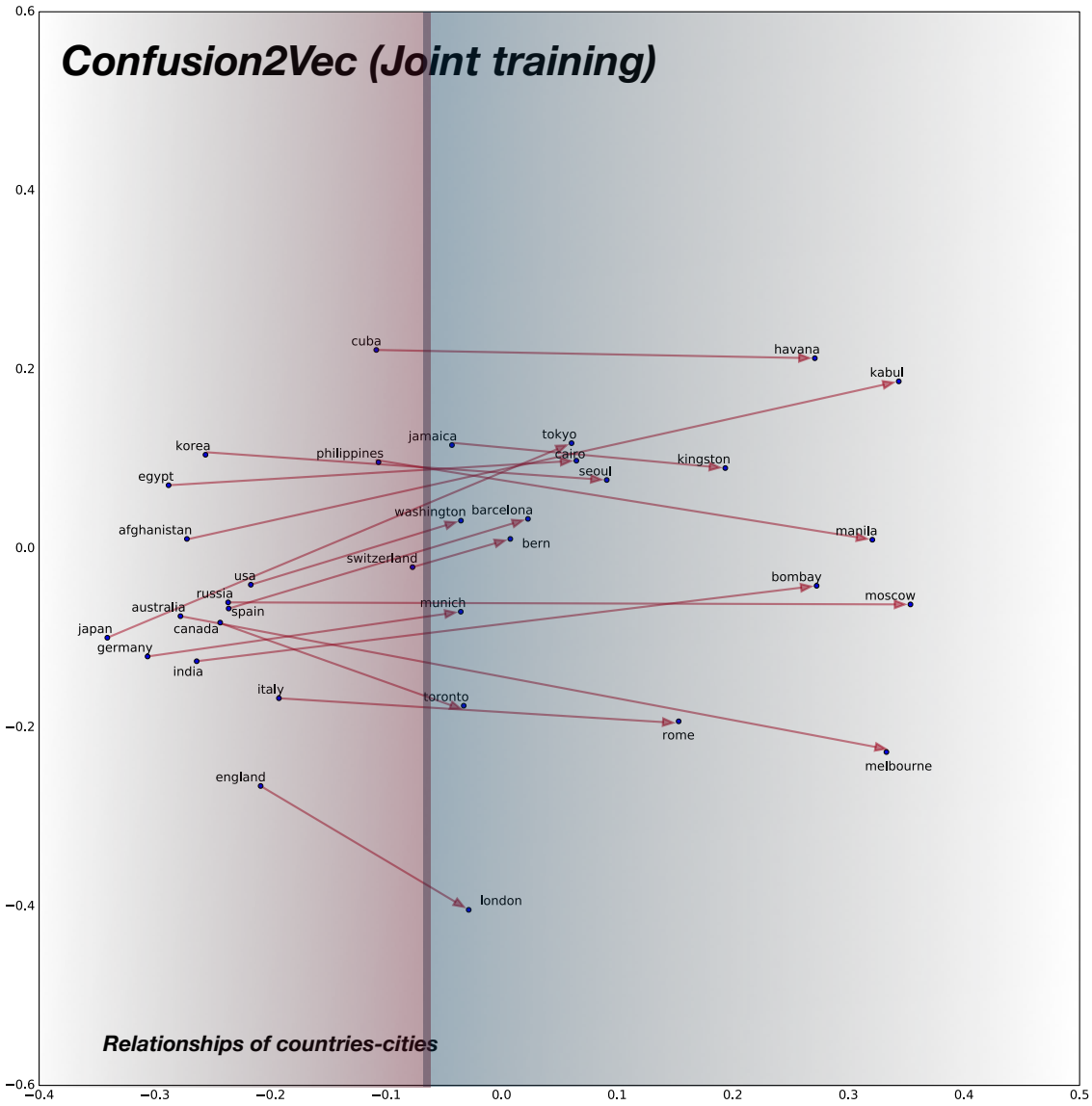


Figure 4.10: **2D plot after PCA of word vector representation on jointly optimized pre-trained word2vec + intra-confusion models**
Demonstration of Semantic Relationship on Randomly chosen pairs of Countries and Cities
Observe the semantic relationships are preserved as in the case of word2vec model: Country-City vectors are almost parallel/similar. Countries are clustered together on the left half (highlighted with red hue) and the cities on right half (highlighted with blue hue).

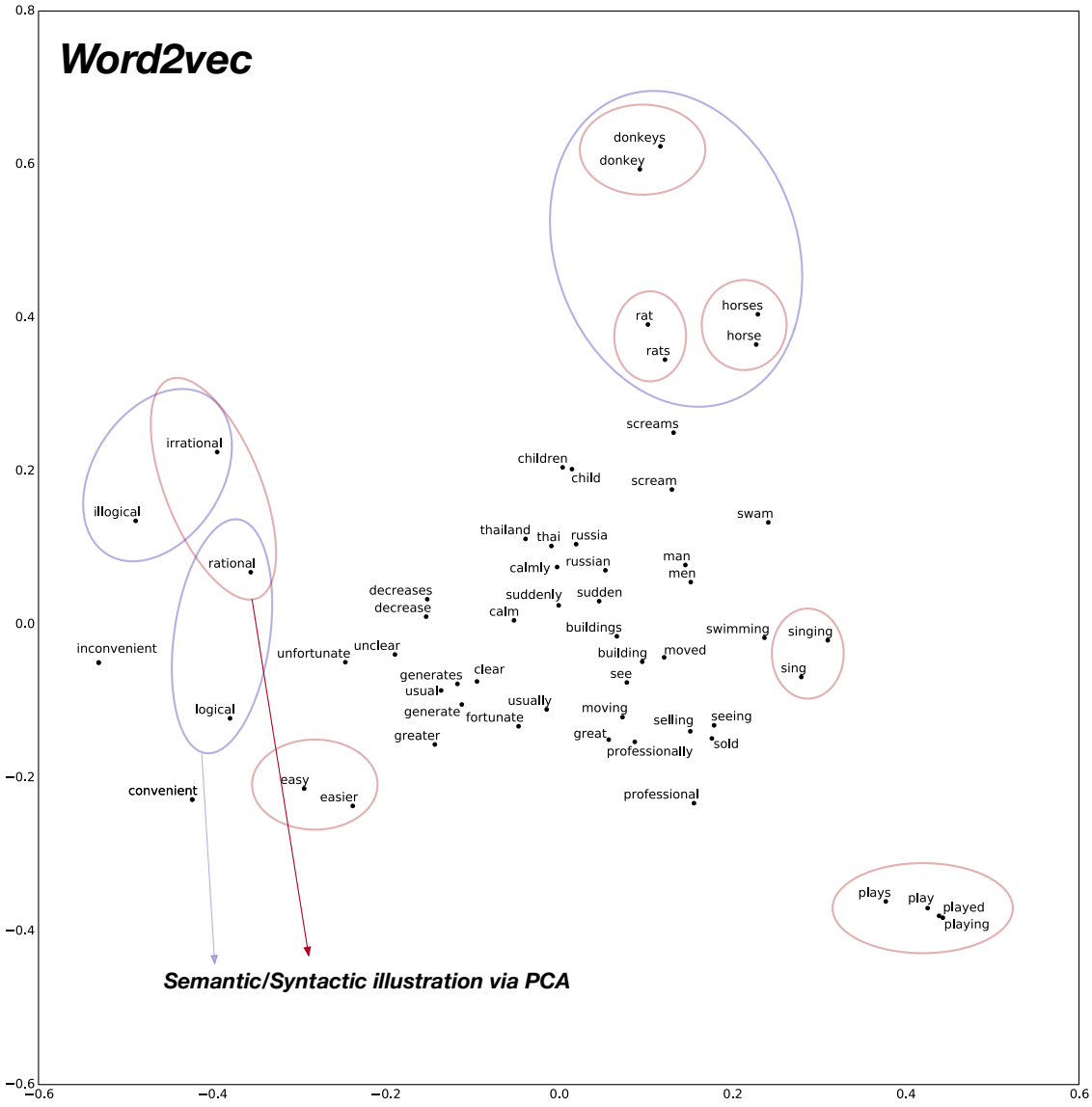


Figure 4.11: **2D plot after PCA of word vector representation on baseline pre-trained word2vec**
Demonstration of Syntactic Relationship on Randomly chosen 30 pairs of Adjective-Adverb, Opposites, Comparative, Superlative, Present-Participle, Past-tense, Plurals
Observe the clustering of syntactically related words (Ex: highlighted with red ellipses). Few semantically related words are highlighted with blue ellipses (Ex: animals)

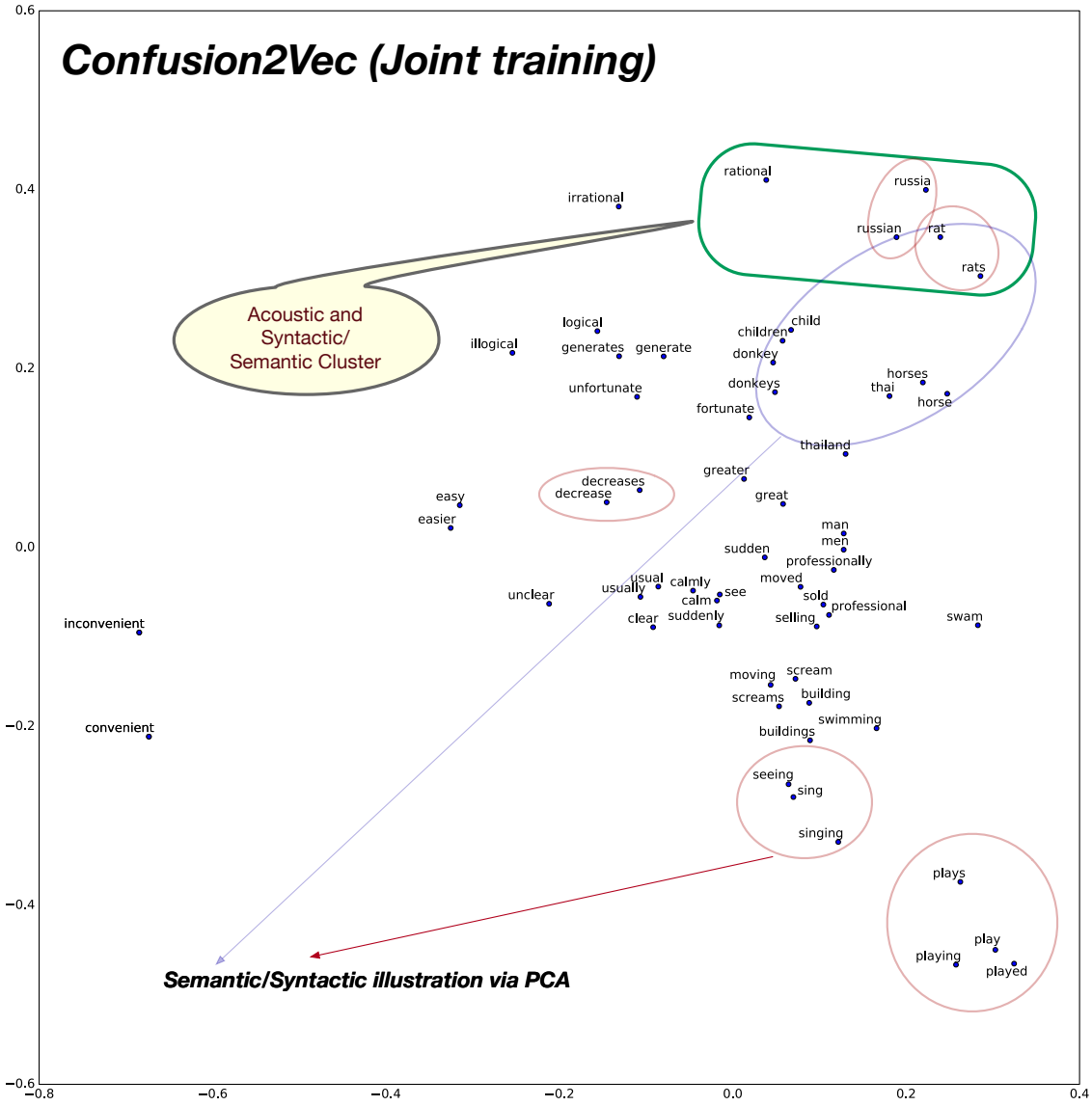


Figure 4.12: **2D plot after PCA of word vector representation on jointly optimized pre-trained word2vec + intra-confusion models**
Demonstration of Syntactic Relationship on Randomly chosen 30 pairs of Adjective-Adverb, Opposites, Comparative, Superlative, Present-Participle, Past-tense, Plurals
Syntactic clustering is preserved by Confusion2Vec similar to Word2Vec. Red ellipses highlight few examples of syntactically related words. Similar to Word2Vec, semantically related words (Ex: animals), highlighted with blue ellipses, are also clustered together. Additionally Confusion2Vec clusters acoustically similar words together (indicated with green ellipse).

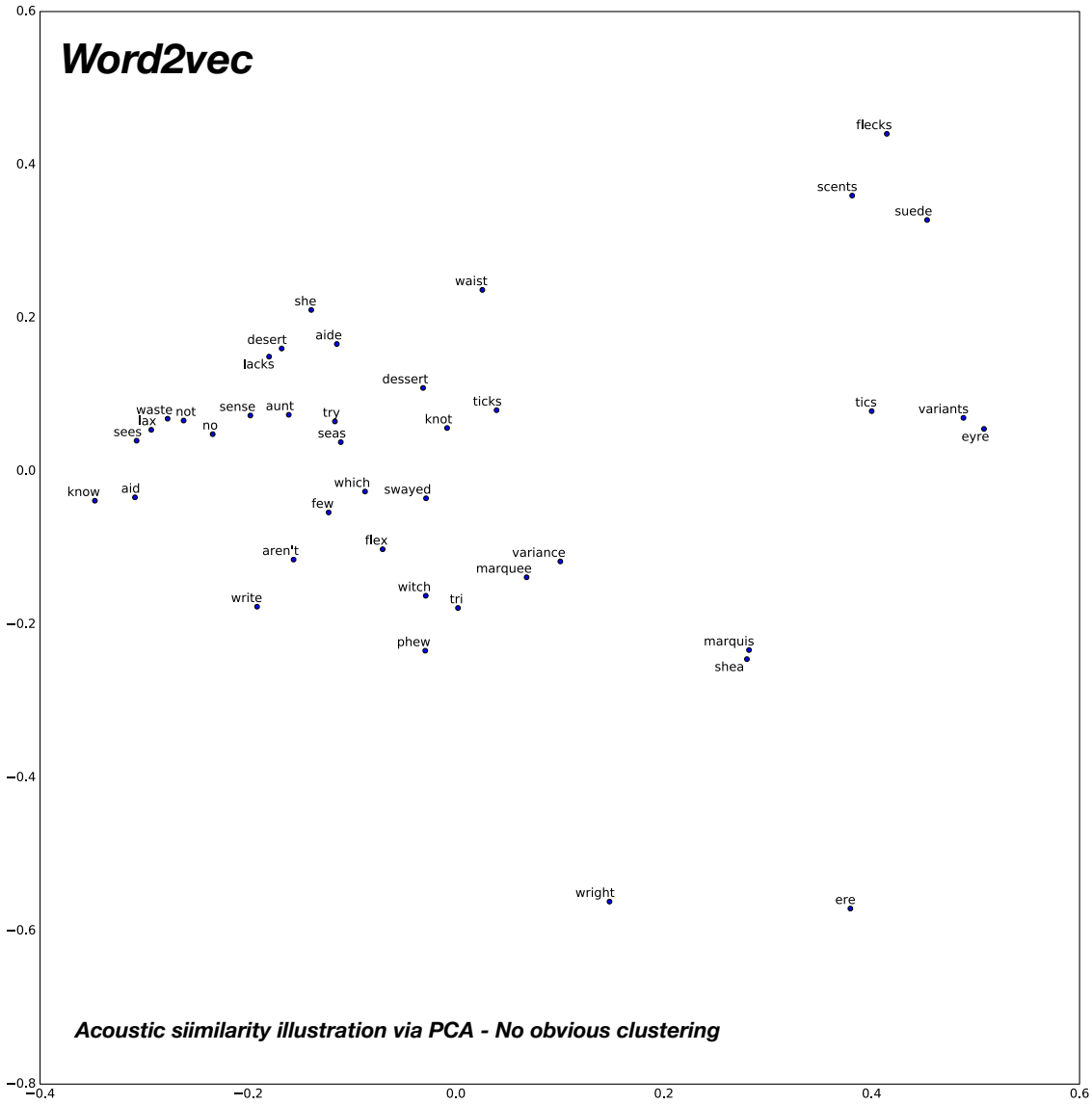


Figure 4.13: **2D plot after PCA of word vector representation on baseline pre-trained word2vec**
Demonstration of Vector Relationship on Randomly chosen 20 pairs of Acoustically Similar Sounding Words
No apparent relations between acoustically similar words are evident.

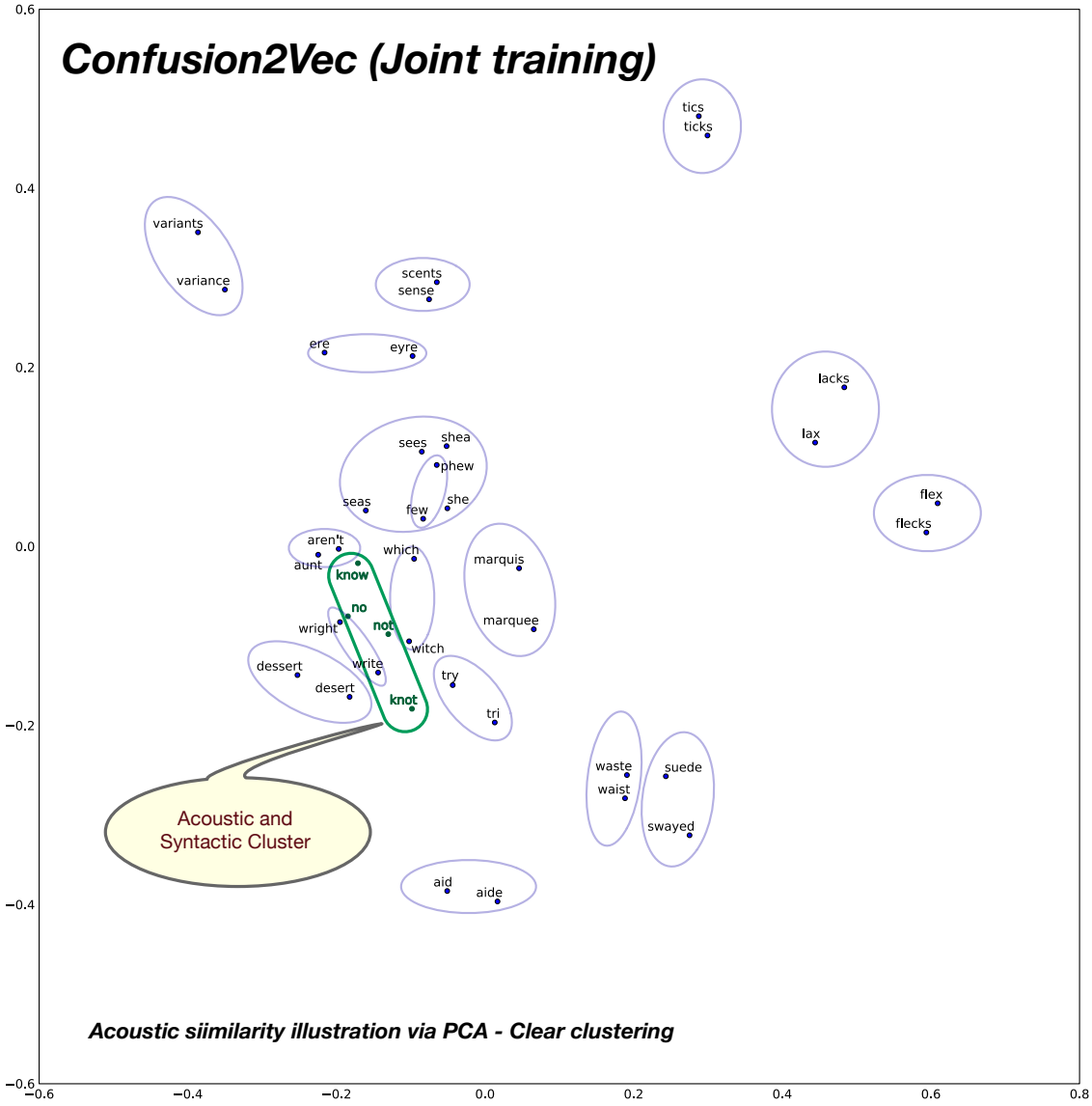


Figure 4.14: **2D plot after PCA of word vector representation on jointly optimized pre-trained word2vec + intra-confusion models**
Demonstration of Vector Relationship on Randomly chosen 20 pairs of Acoustically Similar Sounding Words
Confusion2Vec clusters acoustically similar words together (highlighted with blue ellipses). Additionally, inter-relations between syntactically related words and acoustically related words are also evident (highlighted with green ellipse).

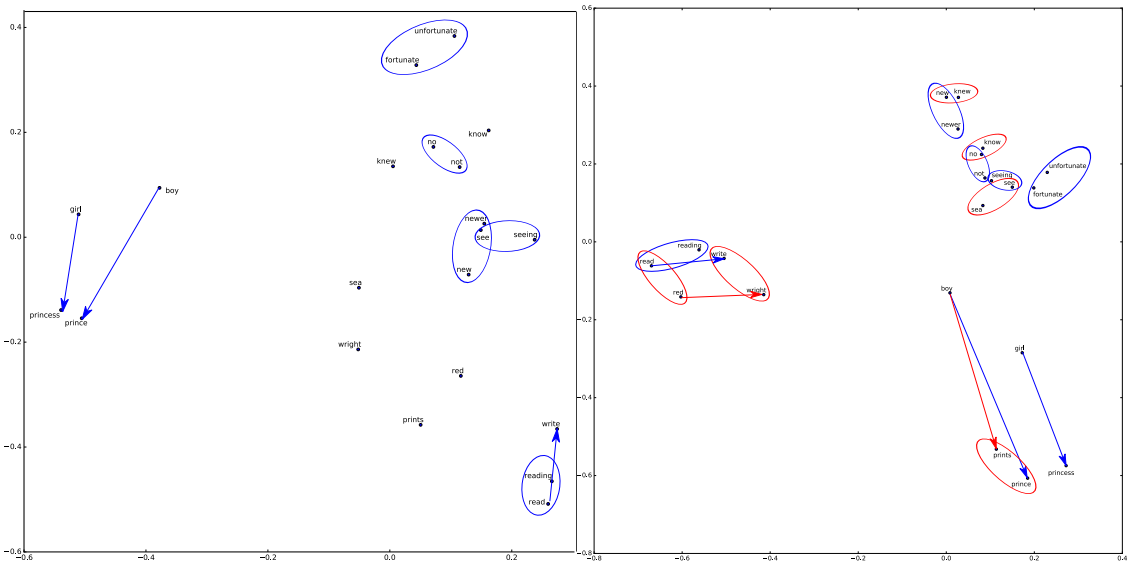
Chapter 5

Spoken Language Intent Detection using Confusion2Vec

5.1 Introduction

In this chapter, we specifically target the task of spoken language intent detection on noisy ASR transcripts. In contrast to the majority of the works which mostly deal with the innovation of classification models [101, 100, 58, 97], in our study, we concentrate on robust word feature representations. We propose to employ the confusion2vec [149] word vector representation to compensate for the errors made by an ASR and to provide enhanced and robust performance for the task of spoken language intent detection. Confusion2vec captures acoustic similarity information of words in addition to the semantic-syntactic relations and is trained in a completely unsupervised manner on ASR lattices decoded on an out-of-domain corpora. Moreover, unlike the studies which adapt the ASR to the target datasets and tasks [146, 101], we treat the ASR as a generic independent module, but contribute towards bridging the gap between the ASR and the NLU model. We demonstrate with our experiments, on the benchmark ATIS dataset [70], the vital role of the confusion2vec to the robustness of the intent classification.

The rest of the chapter is structured as follows. In section 5.2, we present the proposed methodology, provide brief description of the confusion2vec word embedding and the intent classification model. Section 5.3 describes the databases employed, our experimental setup and the baseline systems. In section 5.4, we present and discuss the experimental results. Finally, section 5.5 concludes the study and discusses the future work.



(a) Word2vec Space (b) Confusion2vec Space

Figure 5.1: 2D Vector space illustration after PCA dimension reduction for Word2vec and Confusion2vec

The blue ellipses indicate syntactic word relations. The red ellipses indicate acoustic similarity relations. The blue arrows illustrate the semantic relationships. The red arrows illustrate the interaction of acoustic similarity with semantic relationships.

The word2vec space is rich in semantic and syntactic word relations, however no trivial acoustic similarity is evident.

The confusion2vec space preserves the semantic and syntactic word relations, moreover captures additional acoustic similarity information.

5.2 Proposed Technique

In this section, we first describe the confusion2vec word vector representation for the task of spoken language intent detection and then introduce the recurrent neural network intent classification model.

5.2.1 Confusion2vec Word Embedding

The role of word vector representations is crucial for NLP [30]. Efficient and information rich word embeddings like word2vec [115], glove [127] are shown to capture semantics and syntactics of the language. Using such efficient word representations have proven to be beneficial in the NLU tasks like named entity detection [153], intent detection [67]. The SLU tasks like intent detection [82, 103, 51, 9], slot-filling [50], spoken dialogue systems [48] have also benefited from using information rich word embeddings. However, they are less than optimal in the cases of erroneous transcriptions, for example ASR transcriptions [146, 101], since the errors corrupt the semantic-syntactic space over local context of occurrence and thereby introduce noise in the model.

In this work, we propose to employ recently proposed confusion2vec word vector representation [149] for the task of intent detection to counter for errors present in the spoken transcriptions. Motivated from human speech production and perception, the confusion2vec models the acoustic relations of words in addition to the semantic and syntactic relationships of words [149]. The confusion2vec uses unsupervised training techniques similar to skip-gram of word2vec, but operates on lattice-like structures or confusion networks output by the ASR. Since the confusion networks of a typical ASR exhibits confusions between words on two principle axes (i) contextual, and (ii) acoustic similarity, the confusion2vec is devised to operate on both the axes, thereby modeling local context information (like word2vec) as well as acoustic similarity information. Figure 5.1 illustrates the 2-dimensional word vector space for word2vec and confusion2vec after dimension reduction using principal component analysis (PCA). From the figure (and from extensive analysis done in [149]), it is evident that confusion2vec space captures acoustic similarity between words without compromising the information captured by the word2vec. Complex meaningful, useful interactions between the acoustic subspace and the semantic-syntactic subspaces are also observed. For more information we would like to point the interested readers towards [149], which in detail describes and analyzes the confusion2vec embedding.

In application to the spoken language intent detection task, the nature of ASR errors are often acoustically related. Confusion2vec incorporates real, unsupervised, ASR output as its training corpus, thus the feature representation incorporates confusions (errors) nearby in its embedding space. In other words, we hypothesize that the embedded acoustic similarity information in confusion2vec limits the impact of errors made by the ASR, and thus allows subsequent NLP tasks to be minimally affected. We expect the following with respect to the intent detection task:

- We expect our model to be less affected from ASR errors and thus achieve better performance in the case of noisy ASR transcriptions.

- We expect our model to be at-least on par with word2vec under clean conditions.

5.2.2 Intent Classification Model

Since the contribution of this work is towards word feature representations, we employ a fairly simple recurrent neural network model for the classification task. However, we believe the contributions on feature representations are orthogonal to the classification model and thus expect even better performance for more complex models like in [101, 100, 58, 97]. In this work, we use Bi-directional Long Short-Term Memory (LSTM) units, as shown in Figure 5.2. Given an

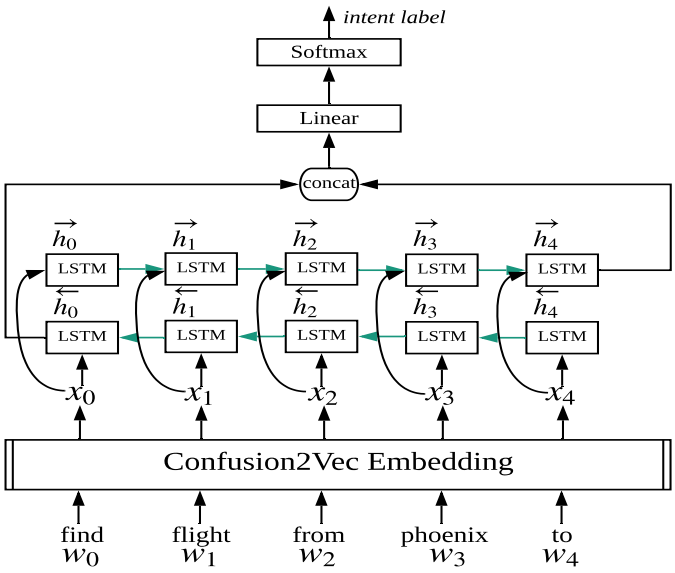


Figure 5.2: Intent Classification RNN Model

input utterance $w_0, w_1, ..., w_T$, each word in the input sequence is mapped to its word vector representation $x_0, x_1, ..., x_T$ by embedding look up. We formulate the model outputs as

$$\overrightarrow{h_t} = \overrightarrow{LSTM}(\overrightarrow{h_{t-1}}, x_t; \overrightarrow{\Theta}) \qquad \overleftarrow{h_t} = \overleftarrow{LSTM}(\overleftarrow{h_{t+1}}, x_t; \overleftarrow{\Theta}) \tag{5.1}$$

$$\hat{P}_{\text{intent}} = Softmax(W\ [\overrightarrow{h_T}, \overleftarrow{h_0}] + b) \tag{5.2}$$

where h_t is the LSTM output of each direction at each time step t, Θ is the parameter of the LSTM. We feed the concatenation of two directional LSTM outputs at the last time step into the linear output layer (with weights W and bias b) which projects it into the intent label space. Finally, the intent label is predicted from the Softmax-normalized probability distribution over all intent classes.

5.3 Database & Experimental Setup

5.3.1 Database

The ASR is trained on the Fisher English Training (LDC2004S13 and LDC2005S13) Speech corpora [29]. The confusion2vec is trained on the output of the ASR, i.e., the confusion networks generated via Fisher English Corpora. The database setup for the ASR and confusion2vec is identical and explained in detail in [149].

We trained the intent detection model on ATIS (Airline Travel Information Systems) dataset [70], which comes with audio recordings and corresponding manual transcripts about humans asking for flight information. Following [65, 58], we apply the same train, development, test split setup. The setup contains 4478, 500 and 893 intent-labeled reference utterances in train, development and test set respectively. In order to evaluate our model's robustness to ASR outputs, we also construct our ASR output set by decoding the corresponding audio recording for each of the data splits using the ASR. In cases where an utterance is labeled with multiple intent labels, the top intent was selected as the true label, yielding 18 intents in total.

5.3.2 Experimental Setup

The training setup for the ASR and the confusion2vec is identical to our previous work [149]. For decoding the ATIS dataset, through our ASR, the audio samples were down-sampled from 16kHz to 8kHz. The ASR achieves a WER rate of 18.54% on the ATIS test set. We choose the confusion2vec model yielding the best performance in [149], i.e., independently trained C2V-1 and C2V-c models are concatenated and jointly optimized with intra-Confusion2vec scheme (556 dimensions).

For intent detection, we train models on the 4478 utterances in training set, and tune hyper-parameters based on the classification accuracy on the 500 reference utterances in development set. The model with the best performance on the development set is chosen and evaluated on both reference test set and ASR test set. The hyper-parameter space we experimented with is as follows: Batch size is set to 1, i.e. each sentence is viewed as an independent sample. Hidden dimension of LSTM unit is tuned over $\{256, 128, 64, 32\}$, and dropout is tuned over $\{0.1, 0.2, 0.25\}$. We select Adam optimizer, with learning rate set to be among $\{0.001, 0.0005\}$. The maximum number of epochs is set to 50 with early-stopping strategy.

5.3.3 Baseline Systems

The first set of baselines compare different conventional word embeddings. They include: (i) random initialization (556 dimensions) sampled from a uniform distribution, (ii) vanilla GloVe[1] (300 dimensions) as in [127], and (iii) skip-gram Word2Vec[2] (556 dimensions) fine-tuned on Fisher English corpus reference transcripts (for fair comparison with confusion2vec). We also tried the vanilla Google word2vec[2]. However, the performance was found to be consistently lower than the

[1] https://nlp.stanford.edu/projects/glove/
[2] https://code.google.com/archive/p/word2vec

fine-tuned version, thus, we don't include it in the comparisons. Note, only the randomly initialized word embedding is trainable, while all other embeddings are fixed throughout the training. All the above baselines use identical RNN architecture for intent classification as described in section 5.2.2.

The second set of baselines compare our proposed model with the recent state-of-the-art models, including: (i) a joint intent detection, slot filling & LM model [101], (ii) an attention-based joint model that incorporates alignment information provided by slot filling task [100], and (iii) an intent-augmented gating mechanism based model which further incorporates character-level embedding along with word-level embedding [97]. The baselines are trained under our experiment setup using the same hyper-parameters reported in their original papers. We also reproduce the result of each model under their original experiment settings[3], and report the obtained scores in parentheses for reference. For the adapted model trained on ASR, we consider a joint intent slot filling and intent detection model which performs sentence reconstruction from ASR hypotheses [146] as the baseline, and report the scores on ASR outputs and corresponding ASR WER from original paper.

5.4 Results and Discussion

5.4.1 Training on Reference Clean Transcripts

Table 5.1 and Figure 5.3 illustrate the results obtained training on clean transcripts. First, we compare the results between different word feature embedding (refer to upper half of Table 5.1). On clean reference transcripts, GloVe embeddings provides the best performance (as found in [51]). Both word2vec and random initialization provide identical results. The proposed confusion2vec gives considerably lower CER compared to the Word2Vec and random initialization. Although GloVe outperforms confusion2vec, we believe the comparison of confusion2vec is more fair with that of word2vec, since both use skip-gram modeling. With the proposed Confusion2vec system, we don't expect improvements on clean reference transcripts, since the acoustic similarity/confusion is less relevant. As expected, we observe no degradation in performance with confusion2vec and is on par with the popular, leading word vector representations for the task of intent detection on clean transcriptions.

On noisy ASR transcripts, we see an increase in CER with all models. Although, random initialization performed identical to Word2Vec on clean transcriptions, we see Word2Vec performs relatively better on ASR transcriptions. This observation confirms that better word feature representations exhibit higher robustness to errors. Similar trend is apparent with GloVe embeddings in

Model	Reference	ASR	Δ_{diff}
Random	2.69	10.75	8.06
GloVe [127]	1.90	8.17	6.27
Word2Vec [115]	2.69	8.06	6.16
C2V (proposed)	2.46	**6.38**	**3.92**
Liu and Lane [101]	1.90 (1.57)	9.41 (8.29)[4]	7.51 (6.72)
Liu and Lane [100]	**1.79** (1.90)	8.06 (8.40)	6.27 (6.50)
Li et al. [97]	2.02 (**1.34**)	9.18 (9.07)	7.16 (7.73)

Table 5.1: Results with Training on Reference: Classification Error Rates (CER) for Reference and ASR Transcripts.
Δ_{diff} is the absolute degradation of model from clean to ASR.
The numbers inside parenthesis indicate CER obtained reproducing the result of each model under their original experiment settings[3].

comparison with random initialization, although we observe slightly higher CER and degradation (between clean and noisy transcripts) compared to word2vec. The proposed confusion2vec gives the least CER among all the models (a relative improvement of 20.84% over word2vec, 21.9% over GloVe and 40.65% over random initialization). Moreover, C2V displays higher robustness going from clean to noisy ASR transcriptions (degradation, Δ_{diff}, is minimal). A relative improvement in robustness of 37.48%, 36.36% and 51.36% compared to GloVe, Word2vec and random initialization respectively is observed with C2V (in terms of Δ_{diff}). The confusion2vec word feature representation is able to use the embedded acoustic similarity information to recover from errors resulting from acoustically confusable words in the ASR output transcriptions.

Further, we compare our proposed system with the recent state-of-the-art works on SLU (see bottom part of Table 5.1). Note, the recent works employ much more complex modeling techniques compared to ours. Thus, as expected the recent works outperform our simple RNN architecture testing on clean transcriptions. However, on noisy ASR transcriptions, even with a much simpler model, the proposed confusion2vec achieves significantly lower CER (a relative improvement of at-least 20.84%) compared to state-of-the-art models. Moreover, again, the degradation with confusion2vec is the least among all the models, a relative 37.48% lesser degradation compared to the recent works. The results highlight the potent robustness of the confusion2vec word feature representation. In addition, we believe that the gains from the complex classification modeling are orthogonal to gains from confusion2vec word feature representations and thus should result in additional gains incorporating more complex models with confusion2vec.

[3]Original settings of [101, 100, 97], make use of train + dev data for training. They pre-process data by substituting the digits with a token.

Model	WER %	CER %
Random	18.54	5.15
GloVe [127]	18.54	6.94
Word2Vec [115]	18.54	5.49
C2V (proposed)	18.54	**4.70**
Schumann and Angkititrakul [146]	10.55	5.04[4]

Table 5.2: Results with Training and Testing on ASR transcripts.

5.4.2 Training on ASR

Further, we also perform additional experiments by training the intent classification models on noisy ASR transcripts. A more robust feature representation should theoretically help in reducing the noise in the model translating to better performance. From Table 5.2, it is evident that all the models improve with the matched noisy train and test conditions. The proposed confusion2vec

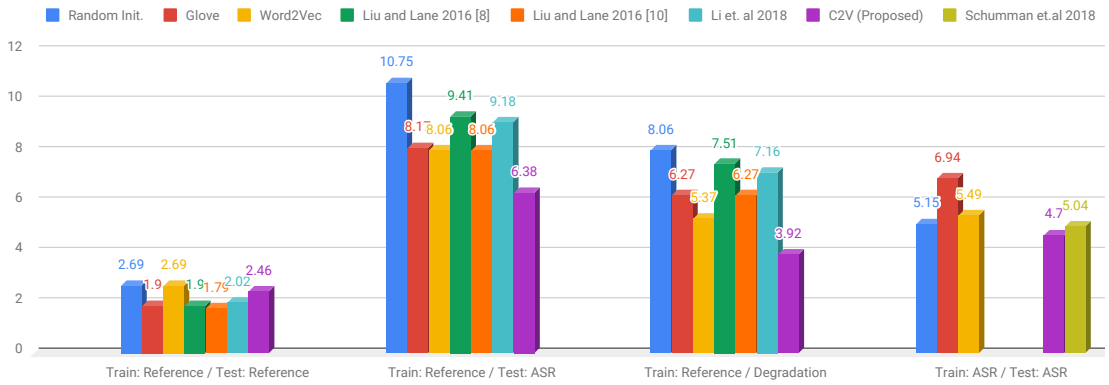


Figure 5.3: Comparison of CER for different systems

based model gives the least CER among all the models. The confusion2vec feature representation is better able to explain the (acoustic) errors and doing so reduces confusion and noise in the intent classification model, thereby resulting in a better and robust performance. Moreover, comparing it with the recent study by Schumann and Angkititrakul [146], although the results are not directly comparable due to differences in the WER of the ASRs, our proposed method achieves a lower CER in-spite of much worse WER conditions[4]. This suggests that explicitly modeling in-domain ASR errors as in [146] is of lesser effect compared to modeling the general acoustic signatures between words in a language as in the case with confusion2vec.

[4]We don't domain-constrain, optimize or rescore our ASR, as in [146, 101]. We treat ASR as an independent module for fair comparison with other models and for domain-generalization and portability of our system and conclusions.

Finally, comparing the results from Table 5.1 and Table 5.2, it is encouraging to see that the proposed confusion2vec model trained on clean transcripts is able to inherit enough robustness to achieve lower CER (than GloVe) and comparable CER to the models (word2vec) trained on ASR output, possibly reducing the need for adaptation on ASR and allowing for more generalizable systems.

5.5 Conclusion and Future Work

In this paper, we proposed an intent detection model based on confusion2vec word vector representation targeting noisy ASR transcriptions. The proposed word embeddings significantly outperform the popular leading word vector representations like word2vec and GloVe in the cases of noisy ASR output. Comparisons are made with various recent state-of-the-art studies, and we find the proposed method improves over them by a considerable margin despite using relatively simple RNN architectures for classification. The robustness of confusion2vec also extends to models trained on noisy ASR, achieving the least CER among the conventional word embedding as well as the recent studies. Encouraging results suggest confusion2vec robustness to errors eliminates the need for adapting the intent classification models on noisy ASR outputs.

In future, we plan to apply and evaluate the proposed confusion2vec on additional SLU tasks like slot-filling, domain classification and named entity recognition. We believe the proposed model should provide similar advantages, especially under noisy conditions. Addressing multiple SLU tasks also allows us to use more complex joint-modelling systems with confusion2vec. The better, more complex, models should provide improvements orthogonal to confusion2vec feature representations, and we thus expect to see further improvements. We also plan to conduct more in-depth analysis on how the signal conditions and ASR performance affect each model; we expect confusion2vec to provide more gains as the ASR performance deteriorates. Representation of multiple path outputs from the ASR with confusion2vec instead of only the best path is also a possible future direction.

Chapter 6

Confusion2Vec 2.0: Enriching Ambiguity Representations with Subwords

6.1 Introduction

Decoding human language is a core component for spoken language understanding. Although it comes very naturally to humans, it is a challenging task for machines. Human language is a complex construct involving multiple dimensions of information involving semantics, syntax and often contain ambiguities which make it difficult for machine inference. Several word vector representations have been proposed for effectively describing the human language in the natural language processing community. The neural networks have been proven to be an effective tool for estimation of such encoding. Contextual modeling techniques like language modeling, i.e., predicting the next word in the sentence given a window of preceding context have been shown to model meaningful word representations [11, 112]. Bag-of-word based contextual modeling, where the current word is predicted given both its left and right (local) contexts has shown to capture language semantics and syntax [113]. Similarly, predicting local context from the current word, referred to as skip-gram modeling, is shown to better represent semantic and syntactic distances between words [115]. In [127] log bi-linear models combining global word co-occurrence information and local context information, termed as global vectors (GloVe), is shown to produce meaningful structured vector space. Bi-directional language models is proposed in [128], where internal states of deep neural networks are combined to model complex characteristics of word use and its variance over linguistic contexts. The advantages of bi-directional modeling are further exploited along with self-attention using transformer networks [173] to estimate a representation, termed as BERT (Bidirectional Encoder Representations from Transformers), that has proved its efficacy on a multitude of natural language understanding tasks [38]. Models such as BERT,

ELMo estimate word representations that vary depending on the context, whereas the context-free representations including GloVe and Word2Vec generate a single representation irrespective of the context.

However, most of the word vector representations infer the knowledge through contextual modeling and many of the ambiguities present in human language is often unrecognized or ignored. For instance, from the perspective of spoken language, the ambiguities can be associated with how similar the words sound, i.e., for example, the words "see" and "sea" sound acoustically identical but have different meanings. The ambiguities can also be associated with the underlying speech signal itself due to wide range of acoustic environments involving noise, overlapped speech and channel, room characteristics. These ambiguities often project themselves as errors through ASR systems. Most of the existing word vector representations such as word2vec [115, 113], fasttext [17], GloVe [127], BERT [38], ELMo [128] don't account for the ambiguities present in speech signals and thus degrade while processing on top of ASR transcripts.

Confusion2vec was recently proposed to handle ambiguity information present in human language from the aspects of human speech production and perception [148]. Application to domain of speech and acoustics, confusion2vec is estimated by unsupervised skip-gram training on the ASR output lattices and confusion networks. The analysis of inherent acoustic ambiguity information of the embeddings displayed meaningful interactions between the semantic-syntactic subspace and acoustic similarity subspaces. In [152], the efficacy of the confusion2vec is confirmed on the task of spoken language intent detection. The confusion2vec significantly outperformed typical word embeddings including word2vec, GloVe when evaluated on top of ASR transcripts by reducing the classification error rate by approximately 20% relative.

Although, there have been few attempts in leveraging information present in word lattices and word confusion networks for several tasks [166, 91, 169, 179, 157, 75], the main downside with these works is that the word representation estimated by such techniques are task dependent and are restricted to a particular domain and dataset. Moreover, availability of most of the task specific datasets are limited and task specific speech data is expensive to collect. The advantage with the Confusion2Vec is that it estimates a task independent word vector representations by unsupervised learning on lattices or confusion networks generated by an ASR on random speech conversations.

In this chapter, we incorporate subwords to represent each word for modeling both the acoustic ambiguity information and the contextual information. Each word is modeled as a sum of constituent n-gram characters. Our motivation behind the use of subwords are the following: (i) it incorporates morphological information of the words by encoding internal structure of words [17],

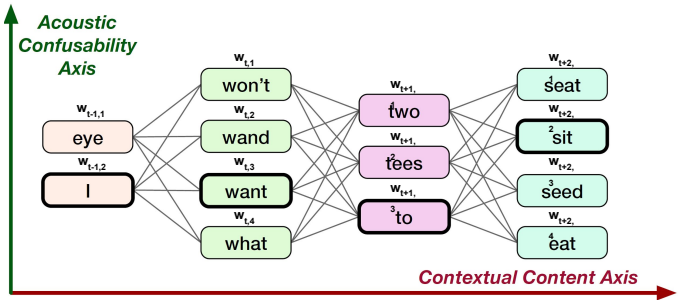


Figure 6.1: Example Confusion Network Output by ASR

(ii) the bag of character n-grams often have a high overlap between acoustically ambiguous words, (iii) subwords help model under-represented words more efficiently, thereby leading to more robust estimation with limited available data, which is the case since training Confusion2Vec is restricted to ASR lattice outputs, (iv) subwords enable representations for out-of-vocabulary words which are common-place with end-to-end ASR systems outputting characters.

The rest of the chapter is organized as follows: Confusion2vec is reviewed in Section 6.2. The proposed subword modeling is presented in Section 6.3. Section 6.4 gives details of the datasets employed, the experimentation setup and the evaluation methodology. The results are presented in section 6.5. Section 6.6 demonstrates the efficacy of the proposed word embedding model to the application of spoken language intent detection task. Finally, the chapter is concluded in section 6.7 and future work discussed in section 6.8.

6.2 Confusion2Vec

In the field of psycho-acoustics, it is established that humans also relate words with how they sound [6] in addition to semantics and syntax. Inspired by psycho-acoustics, human speech and perception, we previously proposed confusion2vec [148]. The core idea is to estimate a hyper-space that not only captures the semantics and syntax of human language, but also augments the vector space with acoustic ambiguity information, i.e., word acoustic similarity information. In other words, word2vec, GloVe can be viewed as a subspace of the confusion2vec vector space.

Several different methodologies are proposed for capturing the ambiguity information. The methodologies are an adaptation of the skip-gram modeling for word confusion networks or lattice-like structures. The word lattices are directed acyclic weighted graphs of all the word sequences that are likely possible. A confusion network is a specific type of lattice with constraints that each word sequence passes through each node of graph. Such lattice-like structures can be derived from machine learning algorithms that output probability measures, for example, an ASR. Figure 6.1,

illustrates a confusion network that can possibly result from a speech recognition system. Unlike typical sentences, which are used for training word embeddings like word2vec, GloVe, BERT, ELMo etc., the information in the confusion network can be viewed along two dimensions: (i) contextual dimension, and (ii) acoustic ambiguity dimension.

More specifically, 4 configuration of skip-gram modeling algorithms are proposed in our recent work [148], namely: (i) top-confusion, (ii) intra-confusion, (iii) inter-confusion, and (iv) hybrid model. The top-confusion version considers only the most-probable path of the ASR confusion network and applies the typical skip-gram model on it. The intra-confusion version applies the skip-gram modeling on the acoustic ambiguity dimension of the confusion network and ignores the contextual information, i.e., each ambiguous word alternative is predicted by the other over a pre-defined local context. The inter-confusion version applies the skip-gram modeling on the contextual dimension but over each of the acoustic ambiguous words. The hybrid model is a combination of both the intra and inter-confusion configurations. More information on the training configuration is available in [148].

6.3 Confusion2Vec 2.0 subword model

Subword encoding of words have been popular in modeling semantics and syntax of language using word vector representations [17, 38, 128]. The use of subwords are mainly motivated by the fact that the subwords incorporate morphological information which can be helpful, for example, in relating the prefixes, suffixes and the word root. In this work, we apply subword representation for encoding the word ambiguity information in the human language. We believe we have a much stronger case for the use of subwords for representing the acoustic similarities (ambiguities) between the words in the language since more similarly sounding words often have highly overlapping subword representations. This helps model ascertain the level of overlap and in doing so estimate the magnitude of acoustic similarity robustly. Moreover, use of subword should help in efficient encoding of under-represented words in the language. This is crucial in the case of confusion2vec because we are restricted to ASR lattices for training data limiting word-word co-occurrence in contrast to typical word vector representation which can be trained on large amounts of easily available plain text data. Another important aspect is the ability to represent out-of-vocabulary words which are common place occurrence with end-to-end ASR systems outputting character sequences.

In the proposed model, each word w is represented as a sum of its constituent n-gram character subwords. This enables the model to infer the internal structure of each word. For example, a word "`want`" is represented with the vector sum of the following subwords:

<wa, wan, ant, nt>, <wan, want, ant>, <want, want>, <want>

Symbols < and > are used to represent the beginning and end of the word. The n-grams are generated for n=3 upto n=6. It is apparent that an acoustically ambiguous, similar sounding word "`wand`" has a high degree of overlap with the set of n-gram characters.

In this chapter, we consider two modeling variations: (i) inter-confusion, and (ii) intra-confusion versions of confusion2vec with the subword encoding.

6.3.1 Intra-Confusion Model

The goal of the intra-confusion model is to estimate the inter-word relations between the acoustically ambiguous words that appear in the ASR lattices. For this, we perform skip-gram modeling over the acoustic similarity dimension (see Figure 6.1) and ignore the contextual dimension of the utterance. The objective of the intra-confusion model is to maximize the following log-likelihood:

$$\sum_{t=1}^{T} \sum_{\hat{a} \in \hat{A}_t} \sum_{a \in A_t} log\ p(w_{t,a}|w_{t,\hat{a}}) \tag{6.1}$$

where T is the length of the utterance (confusion network) in terms of number of words, $w_{i,j}$ is the word in the confusion network output by the ASR at time-step i and j is the index of the word among the ambiguous alternatives. $\hat{A}_t$ is the set of indices of all ambiguous words at time-step t, $\hat{a}$ is the index of the current word along the acoustic ambiguity dimension, $A_t \subseteq \hat{A}_t - \hat{a}$ is the subset of ambiguous words barring $\hat{a}$ at the current word t, i.e., for example from Figure 6.1, for the current word, $w_{t,\hat{a}}$, "`want`", $A_t \subseteq$ {`wand, won't, what`}. Additionally, for subword encoding, each word input is represented as:

$$w_{i,j} = \sum_{s \in S_w} x_s \tag{6.2}$$

where S_w is the set of all character n-grams ranging from n=3 to n=6 and the word itself and x_s is the vector representation for n-gram subword s. Few training samples (input, target) generated for this configuration pertaining to input confusion network in Figure 6.1 are `(I, eye), (eye, I), (want, wand), (want, won't), (won't, what), (wand, what)` etc.

6.3.2 Inter-Confusion Model

The aim of the inter-confusion model is to jointly model the contextual co-occurrence information and the acoustic ambiguity co-occurrence information along both the axis depicted in the confusion network. Here, the skip-gram modeling is performed over time context and over all the possible acoustic ambiguities. The objective of the inter-confusion model is to maximize the following log-likelihood:

$$\sum_{t=1}^{T} \sum_{\hat{a} \in \hat{A}_t} \sum_{c \in C_t} \sum_{a \in A_c} log\ p(w_{c,a}|w_{t,\hat{a}}) \tag{6.3}$$

where C_t corresponds to set of indices of nodes of confusion network, i.e., words around the current word t along the time-axis and c is the current context index. A_c is the set of indices of acoustically ambiguous words at a context c. For example, for the current word, $w_{t,\hat{a}}$, "`want`" in Figure 6.1, $A_c \subseteq$ {`I, eye, two, tees, to, seat, sit, seed, eat`} and $A_t \subseteq$ {`wand, won't, what, want`}. Note, each word input is subword encoded as in equation 6.2. Few training samples (input, target) generated for this configuration are `(want, I), (want, eye), (want, two), (want, to), (want, tees), (what, I), (what, eye), (what, to), (what, tees), (what, two), (won't, eye)` etc.

6.3.3 Training Loss and Objective

Negative sampling is employed for training the embedding model. Negative sampling was first introduced for training word2vec representation [115]. It is a simplification of the Noise Contrastive Estimation objective [62]. The negative sampling for training the embedding can be posed as a set of binary classification problems which operates on two classes: presence of signal or absence (noise). In the context of word embeddings the presence of the context words are treated as positive class and the negative class is randomly sampled from the unigram distribution of the vocabulary. The negative sampling for subword model can be expressed using binary logistic loss as:

$$log\ \sigma(\sum_{s \in S_{w_i}} x_s^T o_{w_t}) + \sum_{k=1}^{K} \mathbb{E}_{w_k \sim P_n(w)} log\ \sigma(-\sum_{s \in S_{w_i}} x_s^T o_{w_k}) \tag{6.4}$$

where $\sigma(x) = \frac{1}{1+e^{-x}}$, w_i is the input word, w_t is the output word, S_{w_i} is the set of n-gram character subwords for the word w_i, x_s is the vector representation for the character n-gram subword s and o_{w_t} is the output vector representation of target word w_t. K is the number of negative samples to be drawn from the negative sample, noise distribution $P_n(w)$. The noise distribution $P_n(w)$ is chosen to be the unigram distribution of words in the vocabulary raised to the $3/4^{\text{th}}$ power

as suggested in [115]. Note, for confusion2vec the input word w_i and target word w_t are derived according to equations 6.1 and 6.3 for implementing the respective training configurations

6.4 Data and Experimental Setup

6.4.1 Database

Fisher English Training Part 1, Speech (LDC2004S13) and Fisher English Training Part 2, Speech (LDC2005S13) corpora [29] are used for both training the ASR and the confusion2vec 2.0 embeddings. The choice of database is based on [148] for direct comparison purposes. The corpus consists of spontaneous telephonic conversations between 11,972 native English speakers. The speech data amounts to approximately 1,915 hours sampled at 8 kHz. The corpus is divided into 3 parts for training (1,905 hours, 1,871,731 utterances), development (5 hours, 5000 utterances) and test (5 hours, 5000 utterances). Overall, the transcripts contain approximately 20.8 million word tokens and vocabulary size of 42,150.

6.4.2 Experimental Setup

The experimental setup is maintained identical to [148] for direct comparison. Brief detail of the setup is as follows:

6.4.2.1 Automatic speech recognition

A hybrid HMM-DNN based acoustic model is trained on the train subset of the speech corpus using the KALDI speech recognition toolkit [130]. 40 dimensional mel frequency cepstral coefficients (MFCC) features are extracted along with the i-vector features for training the acoustic model. The i-vector features are used to provide speaker and channel characteristics to aid acoustic modeling. The acoustic model, DNN, comprises of 7 layers with P-norm non-linearity (p=2) each with 350 units [190]. The DNN is trained using 5 MFCC frame splices with left and right context of 2 to classify among 7979 Gaussian mixtures with stochastic gradient descent optimizer. CMU pronunciation dictionary [175] is utilized as the word-pronunciation transcription lexicon. Tri-gram language model is trained on the training subset of the Fisher English Speech Corpus. The ASR yields word error rates (WER) of 16.57% and 18.12% on the development and the test datasets. Lattices are derived during the ASR decoding with a decoding beam size of 11 and lattice beam size of 6. The lattices are converted to confusion networks with the minimum Bayes risk criterion [182] for training the confusion2vec embeddings. The resulting confusion networks

have a vocabulary size of 41,274 and 69.5 million words, with an average of 3.34 alternative (ambiguous) words for each edge in the graph.

6.4.2.2 Confusion2Vec 2.0

In order to train the embedding, most frequent words are sub-sampled as suggested in [115], with the rejection threshold set to 10^{-4}. Also, a minimum frequency threshold of 5 is set and the rarely occurring words are pruned from the vocabulary. The context window size for both the acoustic ambiguity and contextual dimensions are uniformly sampled between 1 and 5. The dimension of the word vectors are set to 300. The number of negative samples for negative sampling is chosen to be 64. The learning rate is set to 0.01 and trained for a total of 15 epochs using stochastic gradient descent. All the hyper-parameters are empirically chosen for optimal performance. We implemented the confusion2vec 2.0 by modifying the source code from fastText[1] [17]. We make our source code and trained models available.

6.4.3 Evaluation Metrics

For evaluating the inherent semantic and syntactic knowledge of the word embeddings, we employ two tasks: (i) semantic-syntactic analogy task, and (ii) word similarity task. The word analogy task was first proposed in [113] which comprises of word pair analogy questions of the form W_1 is to W_2 as W_3 is to W_4. The analogy is answered correct if $vec(W_1) - vec(W_2) + vec(W_3)$ is most similar to $vec(W_4)$. Another prominent approach is the word similarity task, where rank-correlation between cosine similarity of set of pair of word vectors and human annotated word similarity scores are assessed [143]. For word similarity task, we use the WordSim-353 database [49] consisting of 353 pairs of words annotated over a score of 1 to 10 depending on the magnitude of word similarity as perceived by humans.

For assessing the word acoustic ambiguity (similarity) information, we conduct the acoustic analogy task, Semantic&syntactic–acoustic analogy task and Acoustic similarity tasks proposed in [148]. Acoustic analogy task comprises of word pair analogies compiled using homophones which answer questions of the form: W_1 sounds similar to W_2 as W_3 sounds similar to W_4. The acoustic analogy task is designed to assess the ambiguity information embedded in the word vector space [148]. The semantic&syntactic-acoustic analogy task is designed to assess both semantic, syntactic and acoustic ambiguity information simultaneously. The analogies are formed by replacing certain words by their homophone alternatives in the original semantic and syntactic analogy task [148]. The acoustic word similarity task is analogous to the word similarity task,

[1] https://github.com/facebookresearch/fastText

	Model	Analogy Tasks				Similarity Tasks	
		S&S	Acoustic	S&S-Acoustic	Average Accuracy	Word Similarity	Acoustic Similarity
	Google W2V [115]	61.42%	0.9%	16.99%	26.44%	**0.6893**	-0.3489
	In-domain W2V	59.17%	0.6%	8.15%	22.64%	0.4417	-0.4377
	fastText [17]	**75.93%**	0.46%	17.40%	31.26%	0.7361	-0.3659
Confusion2Vec 1.0 (word) [148]	C2V-a	63.97%	16.92%	43.34%	41.41%	0.5228	0.6200
	C2V-c	**65.45%**	27.33%	38.29%	43.69%	0.5798	0.5825
Confusion2Vec 2.0 (subword)	C2V-a	56.74%	**50.79%**	**44.67%**	**50.73%**	0.3181	**0.8108**
	C2V-c	56.87%	**51.00%**	**44.98%**	**50.95%**	0.2893	**0.8106**

Table 6.1: **Results: Different proposed models**
C2V-a: Intra-Confusion, C2V-c: Inter-Confusion, S&S: Semantic & Syntactic Analogy.
For the analogy tasks: the accuracies of baseline word2vec models are for top-1 evaluations, whereas of the other models are for top-2 evaluations (as discussed in [148]). For the similarity tasks: all the correlations (Spearman's) are statistically significant with $p < 0.001$.

i.e., it contains of word pairs which are rated on their acoustic similarity based on the normalized phone edit distances. A value of 1.0 refers to two words sounding identical and 0.0 refers to the word pairs being acoustically dissimilar. More details regarding the evaluation methodologies are available in [148]. The evaluation datasets are made available.

6.5 Results

Table 6.1 lists the results in terms of accuracies for analogy tasks and rank-correlations for similarity tasks. The first two rows correspond to results with the original word2vec. Google W2V model is the open source model released by Google[2], trained on 100 billion word Google News dataset. We also train an in-domain version of original word2vec on the Fisher English corpus for fair comparison with the confusion2vec models, referred to as "In-domain W2V" in Table 6.1. The fastText model employed is the open source model trained on Wikipedia dumps with a vocabulary size of more than 2.5 million words released by Facebook[3]. The middle two rows of the table correspond to confusion2vec embeddings without subword encoding and they are taken directly from [148]. The bottom two rows correspond to the results obtained with subword encoding. Note, the confusion2vec 1.0 is initialized on the Google word2vec model for better convergence. The confusion2vec 2.0 model is initialized on the fastText model to maintain compatibility with subword encodings. We normalize the vocabulary for all the experiments, meaning the same vocabulary is used to evaluate the analogy and similarity tasks to allow for fair comparisons.

Comparing the baseline word2vec and fastText embeddings to the confusion2vec, we observe the baseline embeddings perform well on the semantic&syntactic analogy task and provide good

[2]https://code.google.com/archive/p/word2vec/
[3]https://fasttext.cc/docs/en/pretrained-vectors.html

	Model	Analogy Tasks				Similarity Tasks	
		S&S	Acoustic	S&S-Acoustic	Average Accuracy	Word Similarity	Acoustic Similarity
	Google W2V [115]	61.42%	0.9%	16.99%	26.44%	**0.6893**	-0.3489
	In-domain W2V	59.17%	0.6%	8.15%	22.64%	0.4417	-0.4377
	fastText [17]	75.93%	0.46%	17.40%	31.26%	0.7361	-0.3659
Confusion2Vec 1.0 (word) [148]	C2V-1 + C2V-a	67.03%	25.43%	40.36%	44.27%	0.5102	0.7231
	C2V-1 + C2V-c	70.84%	35.25%	35.18%	47.09%	0.5609	0.6345
	C2V-1 + C2V-c (JT)	65.88%	**49.4%**	41.51%	**52.26%**	0.5379	**0.7717**
Confusion2Vec 2.0 (subword)	fastText + C2V-a	**76.10%**	22.67%	**49.15%**	**49.31%**	0.5744	**0.7577**
	fastText + C2V-c	**76.16%**	22.56%	**49.12%**	**49.12%**	0.5732	**0.7573**

Table 6.2: **Results: Different proposed models**
C2V-a: Intra-Confusion, C2V-c: Inter-Confusion, S&S: Semantic & Syntactic Analogy.
For the analogy tasks: the accuracies of baseline word2vec models are for top-1 evaluations, whereas of the other models are for top-2 evaluations (as discussed in [148]). For the similarity tasks: all the correlations (Spearman's) are statistically significant with $p < 0.001$.

positive correlation on the word similarity task as expected. However, they perform poorly on the acoustic analogy task, semantic&syntactic-acoustic analogy task and give small negative correlation on the acoustic analogy task. All the confusion2vec models perform relatively good on semantic&syntactic analogy task and word similarity task, but more importantly give high accuracies on acoustic analogy task and semantic&syntactic-acoustic analogy tasks and provide high positive correlation with the acoustic similarity task.

Specifically with Confusion2Vec 2.0, among the analogy tasks, we observe the subword encoding enhances the acoustic ambiguity modeling. For the acoustic analogy task we find relative improvement of upto 46.41% over its non-subword counterpart. Moreover, even for the semantic&syntactic-acoustic analogy task, we observe improvements with subword encoding. However, we find a small reduction in performance for the original semantic and syntactic analogy task. Regardless of the small dip in the performance, the accuracies remain acceptable in comparison to the in-domain word2vec model. Overall, taking the average accuracy of all the analogy tasks, we obtain an increase of approximately 16.62% relative over the non-subword confusion2vec models.

Investigating the results for the similarity tasks, we find significant and high correlation of 0.81 for acoustic similarity task with the subword encoding. Again, a small degradation is observed with the word similarity task obtaining a correlation of 0.3181 against the 0.4417 of the in-domain baseline word2vec model. Overall, the results of the analogy and the similarity tasks suggest the subword encoding greatly enhances the ambiguity modeling of confusion2vec.

6.5.1 Model Concatenation

Further, the confusion2vec model can be concatenated with the other word embedding models to produce a new word vector space that can result in better representations as seen in [148]. Table 6.2 lists the results of the concatenated models. For the previous, non-subword version of the confusion2vec, the vector models are concatenated with the word2vec model trained on the ASR output transcripts (C2V-1). The choice of using the C2V-1 instead of the Google W2V for concatenation was based on empirical findings. Where as to maintain compatibility of subword encoding, the confusion2vec 2.0 models are concatenated with fastText models.

First, comparisons between the non-concatenated versions in Table 6.1 and the concatenated version in Table 6.2, of the non-subword models, we observe a decent improvement of approximately 7.22% relative in average analogy accuracy after concatenation. We don't observe significant improvement with subword based models after concatenation in terms of average analogy accuracy. However, we observe different dynamics between the acoustic ambiguity and the semantic and syntactic subspaces. Concatenation results in improved semantic and syntactic evaluations with the expense of drop in accuracies of acoustic analogy task. We also note improvements (9.27% relative) in semantic&syntactic-acoustic analogy task after concatenation confirming meaningful existence of both ambiguity and semantic-syntactic relations. Moreover, the word similarity task also yields better correlation after concatenation.

Next, comparisons of the confusion2vec 1.0 (non-subword) and the subword version, we observe significant improvements in semantic&symantic analogy task (7.51% relative) as well as the semantic&syntactic-acoustic analogy tasks (21.78% relative). Moreover, the subword models outperform the non-subword version in both of the similarity tasks. The subword models slightly under-perform in the acoustic analogy task, but more crucially outperform the Google W2V and FastText baselines significantly.

Further, the concatenated models can be fine-tuned and optimized to exploit additional gains as found in [148]. The row corresponding to Confusion2Vec 1.0 - C2V + C2V-c (JT) is the best result obtained in [148] which involves 2-passes. The Confusion2Vec 2.0 with the subword modeling with a single pass training gives comparable performance to the 2-pass approach. Thus we skip the 2-pass approach with the subword model in favor of ease of training and reproducibility.

6.6 Spoken Language Intent Detection with Confusion2Vec 2.0

In this section, we apply the proposed word vector embedding to the task of spoken language intent detection to assess the practicality in application to real word scenarios. Spoken language intent detection is the process of decoding the speaker's intent in contexts involving voice commands, call routing and any human computer interactions. Most of the spoken language technologies comprises of an ASR to convert the speech signal to text. This process introduces errors into the pipeline via ASR conditioned on the varying speaker and noise environments. However, popular approaches to spoken intent detection in the natural language processing community assume clean text as input to the intent classification systems. The erroneous ASR outputs result in degradation of the intent detection classification process. Few efforts have focused on handling the errors of the ASR to make the subsequent intent detection process more robust to errors. These efforts often involve training the intent classification systems on noisy ASR transcripts. The downsides of training the intent classifiers on the ASR is that the systems are limited with the amount of speech data available. Moreover, varying speech signal conditions and use of different ASR models make such classifiers non-optimal and less practical. In many scenarios, speech data is not available to enable adaptation on ASR transcripts.

In our previous work [152], we applied the non-subword version of the confusion2vec to the task of spoken language intent detection. We demonstrated the confusion2vec is able to perform as efficiently as the popular word embeddings like word2vec and GloVe on clean manual transcripts giving comparable classification error rates. More importantly, we were able to illustrate the robustness of the confusion2vec embeddings when evaluated on the noisy ASR transcripts. The confusion2vec gives significantly better accuracies (upto relative 20% improvements) when evaluated on ASR transcripts compared to the word2vec, GloVe embeddings and state-of-the-art models involving more complex neural network intent classification architectures. Moreover, we also illustrated the confusion2vec undergoes the least degradation between the clean and ASR transcripts. We also found that the confusion2vec consistently provides the least classification error rates even when the intent classifier is trained on ASR transcripts. The experiments indicated that the difference in accuracies between training the intent classifier on clean versus the ASR transcripts is reduced to 0.89% from 2.57% absolute. Overall, the results illustrate the confusion2vec has inherent knowledge of the acoustic ambiguity (similarity) word relations which correlates with the ASR errors using which the classifier is able to recover from certain errors more efficiently.

In this chapter, we incorporate the confusion2vec 2.0 embeddings with inherent knowledge of acoustic ambiguity to allow robust intent classification. The enhanced effects of the subword modeling in capturing acoustic ambiguity, verified by the previous evaluations, we believe the proposed model could further improve the spoken utterance classification. In doing so, we aim to eliminate the need for re-training the classifiers on the ASR outputs.

6.6.1 Intent classification

For intent classification we adopt a simple RNN architecture identical to [152] for direct comparison. The architecture of the neural network is intentionally kept simple for effective inference of the efficacy of the proposed embedding word features. The classifier comprises of an embedding layer followed by a single layer of bi-directional recurrent neural network (RNN) with long short-term memory (LSTM) units which is followed by a linear dense layer with softmax function to output a probability distribution across all the intent categories. The embedding layer is fixed throughout the training except for the randomly initialized embeddings where the embedding is estimated on the in-domain data specific to the task of intent detection.

6.6.2 Database and Experimental Setup

6.6.2.1 Database

We conduct experiments on the Airline Travel Information Systems (ATIS) benchmark dataset [70]. The dataset comprises of humans making flight related inquiries with an automated answering machine with audio recorded and its transcripts manually annotated. ATIS consists of 18 intent categories. The dataset is divided into train (4478 samples), development (500 samples) and test (893 samples) consistent with previous works [152, 65, 58]. For ASR evaluations, the audio recordings are down-sampled from 16kHz to 8kHz and then decoded using the ASR setup described in section 6.4.2.1 using the audio mappings[4]. The ASR achieves a WER of 18.54% on the ATIS test set.

6.6.2.2 Experimental Setup

The intent classification models are trained on the 4478 samples of training subset and the hyper-parameters are tuned on the development set. We choose the best set of hyper-parameters yielding the best results on the development set and then apply it on the unseen held-out test subset of both the manual clean transcripts and the ASR transcripts and report the results. For training

[4] https://github.com/pgurunath/slu_confusion2vec

we treat each utterance as a single sample (batch size = 1). The hyper-parameter space we experiment are as follows: the hidden dimension size of the LSTM is tuned over $\{32, 64, 128, 256\}$, the learning rate over $\{0.0005, 0.001\}$, the dropout is tuned over $\{0.1, 0.15, 0.2, 0.25\}$. The Adam optimizer is employed for optimization and trained for a total of 50 epochs with early stopping when the loss on the development set doesn't improve for 5 consecutive epochs.

6.6.2.3 Baselines

We include results from several baseline systems for providing comparisons of Confusion2Vec 2.0 with the popular context-free word embeddings, contextual embeddings, popular established NLU systems and the current state-of-the-art.

1. **Context-Free Embeddings**: GloVe[5] [127], skip-gram word2vec[6] [115] and fastText[7] [17] word representations are employed. They are referred to as context-free embeddings since the word representations are static irrespective of the context.

2. **ELMo**: Peters et al. [128] proposed deep contextualized word representation based on character based deep bidirectional language model trained on large text corpus. The models effectively model syntax and semantics of the language along varying linguistic contexts. Unlike context-free embeddings, ELMo embeddings have varying representations for each word depending on the word's context. We employ the original model trained on 1 Billion Word Benchmark with 93.6 million parameters[8]. For intent-classification we add a single bi-directional LSTM layer with attention for multi-task joint intent and slot predictions.

3. **BERT**: Devlin et al. [38] introduced BERT bidirectional contextual word representations based on self attention mechanism of Transformer models. BERT models make use of masked language modeling and next sentence prediction to model language. Similar to ELMo, the word embeddings are contextual, i.e., vary according to the context. We employ "bert-base-uncased" model[9] with 12 layers of 768 dimensions each trained on BookCorpus and English Wikipedia corpus. For intent-classification we add a single bi-directional LSTM layer with attention for multi-task joint intent and slot predictions.

4. **Joint SLU-LM**: Liu and Lane [101] employed joint modeling of the next word prediction along with intent and slot labeling. The unidirectional RNN model updates intent states for each word input and uses it as context for slot labeling and language modeling.

[5] https://nlp.stanford.edu/projects/glove/
[6] https://code.google.com/archive/p/word2vec/
[7] https://fasttext.cc/docs/en/pretrained-vectors.html
[8] https://allennlp.org/elmo
[9] https://github.com/google-research/bert

5. **Attn. RNN Joint SLU**: Liu and Lane [100] proposed attention based encoder-decoder bidirectional RNN model in a multi-task model for joint intent and slot-filling tasks. A weighted average of the encoder bidirectional LSTM hidden states provides information from parts of the input word sequence which is used together with time aligned encoder hidden state for the decoder to predict the slot labels and intent.

6. **Slot-Gated Attn.**: Goo et al. [58] introduced a slot-gated mechanism which introduces additional gate to improve slot and intent prediction performance by leveraging intent context vector for slot filling task.

7. **Self Attn. SLU**: Li et al. [97] proposed self-attention model with gate mechanism for joint learning of intent classification and slot filling by utilizing the semantic correlation between slots and intents. The model estimates embeddings augmented with intent information using self attention mechanism which is utilized as a gate for slot filling task.

8. **Joint BERT**: Chen et al. [25] proposed to use BERT embeddings for joint modeling of intent and slot-filling. The pre-trained BERT embeddings are fine tuned for (i) sentence prediction task - intent detection, and (ii) sequence prediction task - slot filling. The Joint BERT model lacks the bi-directional LSTM layer in comparison to the earlier baseline *BERT* based model.

9. **SF-ID Network**: E et al. [45] introduced a bi-directional interrelated model for joint modeling of intent detection and slot-filling. An iteration mechanism is proposed where the SF subnet introduces the intent information to slot-filling task while the ID-subnet applies the slot information to intent detection task. For the task of slot-filling a conditional random field layer is used to derive the final output.

10. **ASR Robust ELMo**: Huang and Chen [75] proposed ASR robust contextualized embeddings for intent detection. ELMo embeddings are fine-tuned with a novel loss function which minimizes the cosine distance between the acoustically confused words found in ASR confusion networks. Two techniques based on supervised and unsupervised extraction of word confusions are explored. The fine-tuned contextualized embeddings are then utilized for spoken language intent detection.

6.6.3 Results

Table 6.3 lists the results of the intent detection in terms of classification error rates (CER). The "Reference" column corresponds to results on manually annotated transcripts of ATIS and

Model	Reference	ASR	Δ_{diff}
Random	2.69	10.75	8.06
GloVe [127]	1.90	8.17	6.27
Word2Vec [115]	2.69	8.06	5.37
fastText [17]	1.90	8.40	6.50
ELMo [128] †*	1.46	7.05	5.59
BERT [38] †*	**1.12**	6.16	5.04
Joint SLU-LM [101] †	1.90	9.41	7.51
Attn. RNN Joint SLU [100] †	1.79	8.06	6.27
Slot-Gated Attn. [58] †	3.92	10.64	6.72
Self Attn. SLU [97] †	2.02	9.18	7.16
Joint BERT [25] †*	2.46	7.73	5.27
SF-ID Network [45] †	3.14	10.53	7.39
ASR Robust ELMo (unsup.) [75] *	3.24	5.26	2.02
ASR Robust ELMo (sup.) [75] *	3.46	5.03	**1.57**
C2V 1.0 [148]	2.46	6.38	3.92
C2V-c 2.0	3.36	5.82	**2.46**
C2V-a 2.0	2.46	**4.37**	**1.91**
fastText + C2V-c 2.0	1.79	**4.70**	**2.91**
fastText + C2V-a 2.0	1.90	**5.04**	**3.14**

Table 6.3: Results: Model trained on clean Reference: Classification Error Rates (CER) for Reference and ASR Transcripts
Δ_{diff} is the absolute degradation of model from clean to ASR. C2V 1.0 corresponds to C2V-1 + C2V-c (JT) in Table 6.1 and 6.2.
† indicates joint modeling of intent and slot-filling.
∗ indicates contextual embeddings.

the "ASR" corresponds to the evaluations on the noisy speech recognition transcripts. Firstly, evaluating on the Reference clean transcripts, we observe the confusion2vec 2.0 with subword encoding is able to achieve similar performance to the popular word embedding models and the state-of-the-art. The best performing confusion2vec 2.0 achieves a CER of 1.79%. Among the different versions of the proposed subword based confusion2vec, we find that the concatenated versions are slightly better. We believe this is because the concatenated models exhibit better semantic and syntactic relations (see Table 6.1 and 6.2) compared to the non-concatenated ones. Among the baseline models, the contextual embedding like BERT and ELMo gives the best CER. Note, the proposed confusion2vec embeddings are context-free and is able to outperform other context-free embedding models such as GloVe, word2vec and fastText.

Secondly, evaluating the performance on the erroneous ASR transcripts, we find that all the subword based confusion2vec 2.0 models outperform the popular word vector embeddings by a big margin. The subword-confusion2vec gives drastic improvement of approximately 45.78% relative to the best performing context-free word embeddings. The proposed embeddings also improve

Model	WER %	CER %
Random	18.54	5.15
GloVe [127]	18.54	6.94
Word2Vec [115]	18.54	5.49
Schumann and Angkititrakul [146]	10.55	5.04[10]
C2V 1.0	18.54	4.70
C2V-c 2.0	18.54	4.82
C2V-a 2.0	18.54	**4.26**
fastText + C2V-c 2.0	18.54	**3.70**
fastText + C2V-a 2.0	18.54	**4.26**

Table 6.4: Results: Model trained and evaluated on ASR transcripts.
C2V 1.0 corresponds to C2V-1 + C2V-c (JT) in Table 6.1 and 6.2

over the contextual embeddings including BERT and ELMo (relative improvements of 29.06%). Moreover, the results are also a good improvement over the non-subword confusion2vec word vectors (31.50% improvement). This confirms our initial hypothesis that the subword encoding is better able to represent the acoustic ambiguities in the human language. Comparisons between the different versions of the proposed confusion2vec, the intra-confusion configuration yields the least CER. Inspecting the degradation, Δ_{diff} (drop in performance between the clean and ASR evaluations), we find that all the confusion2vec 2.0 with subword information undergo the least degradation, thereby re-affirming the robustness to the noise in the transcripts.

Table 6.4 presents the results obtained by training models on the ASR transcripts and evaluated on the ASR transcripts. Here we omit all the joint intent-slot filling baseline models, since training on ASR transcripts need aligned set of slot labels due to insertion, substitution and deletion errors which is out-of-scope of this study. We note that the confusion2vec models give significantly lower CER. The subword based confusion2vec models also provide improvements over the non-subword based confusion2vec model (21.28% improvement). Comparing the results in Table 6.3 and Table 6.4, we would like to highlight the subword-confusion2vec model gives a minimum CER of 4.37% on model trained on clean transcripts which is much better than the CER obtained by popular word embeddings like word2vec, GloVe, fastText even when trained on the ASR transcripts (15.15% better relatively). These results prove the subword-confusion2vec models can eliminate the need for re-training any natural language understanding and processing algorithms on ASR transcripts for robust performance.

6.7 Conclusion

In this chapter, we propose to use subword encoding for modeling the acoustic ambiguity information into the word vector representations along with the semantic and syntax of the language. Each word in the language is represented as a sum of its constituent character n-gram subwords. The advantages of the subwords are confirmed by evaluating the proposed models on various word analogy tasks and word similarity tasks designed to assess the effective acoustic ambiguity, semantic and syntactic knowledge inherent in the models. Finally, the proposed subword models are applied to the task of spoken language intent detection system. The results of intent classification system suggest the proposed subword confusion2vec models greatly enhance the classification performance when evaluated on the noisy ASR transcripts. The highlight of the results is that the subword-confusion2vec models totally eliminate the need for re-training the classifier on the ASR transcripts.

6.8 Future Work

In the future, we plan to model ambiguity information using deep contextual modeling techniques such as BERT. We believe bidirectional information modeling with attention can further enhance ambiguity modeling. On the application side, we plan to implement and assess the effect of using confusion2vec models for a wide range of natural language understanding and processing applications such as speech translation, dialogue tracking etc. On the analysis front, we would like to apply the proposed embeddings and evaluate the effects of WER on the performance of spoken language understanding task and the improvements provided by confusion2vec. Assessing the benefits of confusion2vec over wide range of underlying speech signal environments including type of noise, amount of noise, transferability over different ASR systems can be very useful for the domain of spoken language understanding.

Conclusion

The thesis addresses an important problem of handling speech recognition errors for robust speech processing and spoken language understanding. The problem is tackled from the perspective of three basic underlying error sources which eventually results in speech recognition errors: (i) variations in underlying speech signal, (ii) limitations of several machine learning algorithms, and (iii) ambiguities present in human language. In this thesis, we proposed a noisy channel model for error correction based on phrases termed "Noisy-Clean Phrase Context Modeling". The system post-processes the output of an automated speech recognition system effective in correcting and recovering from several different types of speech recognition errors. The NCPCM is able to provide improvements over wide range of underlying input signal variations, and wide spectrum of ASR word error rates. The NCPCM is also able to adapt over certain limitations and restrictions induced to maintain computational complexity and memory tractable. The NCPCM system is shown to improve the speech recognition even over a highly optimized ASR. Further, we introduced a new human language encoding in the form of word vector representation, termed "Confusion2Vec", that takes into account several ambiguity associated in human spoken language, more specifically, we model the acoustic ambiguity (similarity) in human language. The acoustic ambiguity is introduced into the word vector representation by unsupervised modeling of confusions present in the speech recognition output lattices. The acoustic ambiguity is shown to co-exist with the semantics and syntax of human language and in doing results in a robust word vector representation that is robust to ASR errors. The efficacy of the word vector representation is confirmed on the task of spoken language intent classification. The Confusion2Vec achieves significantly lower classification error rates when evaluated on noisy ASR transcripts compared to popular word vector representations. Confusion2Vec with the inherent knowledge of acoustic confusable words which correlates with the ASR error is able to recover from them. Finally, to enhance the ambiguity modeling capacity of the Confusion2Vec, we propose to encode each word by their constituent character n-grams. This results in an increased ability of the model to capture the acoustic ambiguity information, which reflects in the improved performance on the acoustic,

semantic and syntactic analogy tasks and word similarity tasks. The enhancements with the subword modeling is ensured with spoken language intent classification, which results in improved classification error rates compared with the non-subword version of Confusion2Vec. The subword Confusion2Vec model completely eliminates the need for retraining spoken utterancce classifier on ASR transcripts.

References

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: a system for large-scale machine learning. In *OSDI*, volume 16, pages 265–283, 2016.

[2] William A Ainsworth and SR Pratt. Feedback strategies for error correction in speech recognition systems. *International Journal of Man-Machine Studies*, 36(6):833–842, 1992.

[3] Tamer Alkhouli, Felix Rietig, and Hermann Ney. Investigations on phrase-based decoding with recurrent neural network language and translation models. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 294–303, 2015.

[4] Alexandre Allauzen. Error detection in confusion network. In *Eighth Annual Conference of the International Speech Communication Association*, 2007.

[5] Ebru Arisoy, Tara N Sainath, Brian Kingsbury, and Bhuvana Ramabhadran. Deep neural network language models. In *Proceedings of the NAACL-HLT 2012 Workshop: Will We Ever Really Replace the N-gram Model? On the Future of Language Modeling for HLT*, pages 20–28. Association for Computational Linguistics, 2012.

[6] Jennifer Aydelott and Elizabeth Bates. Effects of acoustic distortion and semantic context on lexical access. *Language and cognitive processes*, 19(1):29–56, 2004.

[7] Dzmitry Bahdanau, Jan Chorowski, Dmitriy Serdyuk, Philemon Brakel, and Yoshua Bengio. End-to-end attention-based large vocabulary speech recognition. In *Proceedings of Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*, pages 4945–4949. IEEE, 2016.

[8] Lalit R Bahl, Peter F Brown, Peter V de Souza, and Robert L Mercer. A tree-based statistical language model for natural language speech recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 37(7):1001–1008, 1989.

[9] Kaspars Balodis and Daiga Deksne. Intent detection system based on word embeddings. In *International Conference on Artificial Intelligence: Methodology, Systems, and Applications*, pages 25–35. Springer, 2018.

[10] Samy Bengio and Georg Heigold. Word embeddings for speech recognition. In *Proceedings of the 15th Conference of the International Speech Communication Association, Interspeech*, 2014.

[11] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155, 2003.

[12] Nicola Bertoldi, Richard Zens, and Marcello Federico. Speech translation by confusion network decoding. In *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, volume 4, pages IV–1297. IEEE, 2007.

[13] Nicola Bertoldi, Barry Haddow, and Jean-Baptiste Fouet. Improved minimum error rate training in Moses. *The Prague Bulletin of Mathematical Linguistics*, 91:7–16, 2009.

[14] Daniel M Bikel and Keith B Hall. Refr: an open-source reranker framework. In *INTERSPEECH*, pages 756–758, 2013.

[15] Dan Bikelf, Chris Callison-Burchc, Yuan Caoc, Nathan Glennd, Keith Hallf, Eva Haslerg, Damianos Karakosc, Sanjeev Khudanpurc, Philipp Koehng, Maider Lehrb, et al. Confusion-based statistical language modeling for machine translation and speech recognition. 2012.

[16] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.

[17] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146, 2017. ISSN 2307-387X.

[18] Jan Botha and Phil Blunsom. Compositional morphology for word representations and language modelling. In *International Conference on Machine Learning*, pages 1899–1907, 2014.

[19] Peter F Brown, John Cocke, Stephen A Della Pietra, Vincent J Della Pietra, Fredrick Jelinek, John D Lafferty, Robert L Mercer, and Paul S Roossin. A statistical approach to machine translation. *Computational linguistics*, 16(2):79–85, 1990.

[20] Jacob Buckman and Graham Neubig. Neural lattice language models. *arXiv preprint arXiv:1803.05071*, 2018.

[21] E Byambakhishig, Katsuyuki Tanaka, Ryo Aihara, Toru Nakashika, Tetsuya Takiguchi, and Yasuo Ariki. Error correction of automatic speech recognition based on normalized web distance. In *Fifteenth Annual Conference of the International Speech Communication Association*, 2014.

[22] Arda Celebi, Hasim Sak, Erinç Dikici, Murat Saraçlar, Maider Lehr, E Prud'hommeaux, Puyang Xu, Nathan Glenn, Damianos Karakos, Sanjeev Khudanpur, et al. Semi-supervised discriminative language modeling for Turkish ASR. In *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*, pages 5025–5028. IEEE, 2012.

[23] William Chan, Navdeep Jaitly, Quoc Le, and Oriol Vinyals. Listen, attend and spell: A neural network for large vocabulary conversational speech recognition. In *Proceedings of Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*, pages 4960–4964. IEEE, 2016.

[24] Ciprian Chelba, Timothy J Hazen, and Murat Saraclar. Retrieval and browsing of spoken content. *IEEE Signal Processing Magazine*, 25(3), 2008.

[25] Qian Chen, Zhu Zhuo, and Wen Wang. Bert for joint intent classification and slot filling. *arXiv preprint arXiv:1902.10909*, 2019.

[26] Xinxiong Chen, Lei Xu, Zhiyuan Liu, Maosong Sun, and Huan-Bo Luan. Joint learning of character and word embeddings. In *IJCAI*, pages 1236–1242, 2015.

[27] Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*, 2014.

[28] Yu-An Chung, Chao-Chung Wu, Chia-Hao Shen, Hung-Yi Lee, and Lin-Shan Lee. Audio word2vec: Unsupervised learning of audio segment representations using sequence-to-sequence autoencoder. *arXiv preprint arXiv:1603.00982*, 2016.

[29] Christopher Cieri, David Miller, and Kevin Walker. The Fisher Corpus: a resource for the next generations of speech-to-text. In *International Conference on Language Resources and Evaluation*, volume 4, pages 69–71. LREC, 2004.

[30] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *Journal of machine learning research*, 12(Aug):2493–2537, 2011.

[31] Marta R Costa-jussà and José AR Fonollosa. Analysis of statistical and morphological classes to generate weighted reordering hypotheses on a statistical machine translation system. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 171–176. Association for Computational Linguistics, 2007.

[32] Ryan Cotterell and Hinrich Schütze. Morphological word-embeddings. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1287–1292, 2015.

[33] Horia Cucu, Andi Buzo, Laurent Besacier, and Corneliu Burileanu. Statistical error correction methods for domain-specific ASR systems. In *International Conference on Statistical Language and Speech Processing*, pages 83–92. Springer, 2013.

[34] George E Dahl, Dong Yu, Li Deng, and Alex Acero. Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(1):30–42, 2012.

[35] Scott Deerwester, Susan T Dumais, George W Furnas, Thomas K Landauer, and Richard Harshman. Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391, 1990.

[36] Anoop Deoras and Ruhi Sarikaya. Deep belief network based semantic taggers for spoken language understanding. In *Interspeech*, pages 2713–2717, 2013.

[37] Jacob Devlin, Rabih Zbib, Zhongqiang Huang, Thomas Lamar, Richard M Schwartz, and John Makhoul. Fast and robust neural network joint models for statistical machine translation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics,*, pages 1370–1380. ACL, 2014.

[38] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[39] Luis Fernando D'Haro and Rafael E Banchs. Automatic correction of ASR outputs by using machine translation. Interspeech, 2016.

[40] Erinç Dikici, Arda Celebi, and Murat Saraçlar. Performance comparison of training algorithms for semi-supervised discriminative language modeling. In *Thirteenth Annual Conference of the International Speech Communication Association*, 2012.

[41] Chris Dyer. Using a maximum entropy model to build segmentation lattices for mt. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 406–414. Association for Computational Linguistics, 2009.

[42] Christopher Dyer, Smaranda Muresan, and Philip Resnik. Generalizing word lattice translation. Technical report, MARYLAND UNIV COLLEGE PARK INST FOR ADVANCED COMPUTER STUDIES, 2008.

[43] Christopher J Dyer. The'noisier channel': translation from morphologically complex languages. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 207–211. Association for Computational Linguistics, 2007.

[44] Christopher James Dyer. *A formal model of ambiguity and its applications in machine translation*. University of Maryland, College Park, 2010.

[45] Haihong E, Peiqing Niu, Zhongfu Chen, and Meina Song. A novel bi-directional interrelated model for joint intent detection and slot filling, 2019.

[46] Dumitru Erhan, Yoshua Bengio, Aaron Courville, Pierre-Antoine Manzagol, Pascal Vincent, and Samy Bengio. Why does unsupervised pre-training help deep learning? *Journal of Machine Learning Research*, 11(Feb):625–660, 2010.

[47] Manaal Faruqui and Chris Dyer. Improving vector space word representations using multilingual correlation. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 462–471, 2014.

[48] Emmanuel Ferreira, Bassam Jabaian, and Fabrice Lefèvre. Zero-shot semantic parser for spoken language understanding. In *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[49] Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. Placing search in context: The concept revisited. In *Proceedings of the 10th international conference on World Wide Web*, pages 406–414. ACM, 2001.

[50] Mauajama Firdaus, Shobhit Bhatnagar, Asif Ekbal, and Pushpak Bhattacharyya. A deep learning based multi-task ensemble model for intent detection and slot filling in spoken language understanding. In *International Conference on Neural Information Processing*, pages 647–658. Springer, 2018.

[51] Mauajama Firdaus, Shobhit Bhatnagar, Asif Ekbal, and Pushpak Bhattacharyya. Intent detection for spoken language understanding using a deep ensemble model. In *Pacific Rim International Conference on Artificial Intelligence*, pages 629–642. Springer, 2018.

[52] Yohei Fusayasu, Katsuyuki Tanaka, Tetsuya Takiguchi, and Yasuo Ariki. Word-error correction of continuous speech recognition based on normalized relevance distance. In *IJCAI*, pages 1257–1262, 2015.

[53] Matteo Gerosa, Diego Giuliani, and Shrikanth Narayanan. Acoustic analysis and automatic recognition of spontaneous children's speech. In *Ninth International Conference on Spoken Language Processing*, 2006.

[54] S. Ghannay, Y. Estève, and N. Camelin. Word embeddings combination and neural networks for robustness in asr error detection. In *2015 23rd European Signal Processing Conference (EUSIPCO)*, pages 1671–1675, Aug 2015. doi: 10.1109/EUSIPCO.2015.7362668.

[55] Sahar Ghannay, Yannick Estève, Nathalie Camelin, Camille Dutrey, Fabian Santiago, and Martine Adda-Decker. Combining continuous word representation and prosodic features for asr error prediction. In *Proceedings of the Third International Conference on Statistical Language and Speech Processing - Volume 9449*, SLSP 2015, pages 84–95, New York, NY, USA, 2015. Springer-Verlag New York, Inc. ISBN 978-3-319-25788-4. doi: 10.1007/978-3-319-25789-1_9. URL `http://dx.doi.org/10.1007/978-3-319-25789-1_9`.

[56] Sahar Ghannay, Yannick Estève, Nathalie Camelin, and Paul deléglise. Acoustic word embeddings for asr error detection. In *Interspeech 2016*, pages 1330–1334, 2016. doi: 10.21437/Interspeech.2016-784. URL `http://dx.doi.org/10.21437/Interspeech.2016-784`.

[57] Laurence Gillick and Stephen J Cox. Some statistical issues in the comparison of speech recognition algorithms. In *Proceedings of Acoustics, Speech, and Signal Processing,. ICASSP-89., 1989 International Conference on*, pages 532–535. IEEE, 1989.

[58] Chih-Wen Goo, Guang Gao, Yun-Kai Hsu, Chih-Li Huo, Tsung-Chieh Chen, Keng-Wei Hsu, and Yun-Nung Chen. Slot-gated modeling for joint slot filling and intent prediction. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, volume 2, pages 753–757, 2018.

[59] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376. ACM, 2006.

[60] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *Proceedings of Acoustics, speech and signal processing (ICASSP), 2013 IEEE international conference on*, pages 6645–6649. IEEE, 2013.

[61] Daniel Guo, Gokhan Tur, Wen-tau Yih, and Geoffrey Zweig. Joint semantic utterance classification and slot filling with recursive neural networks. In *2014 IEEE Spoken Language Technology Workshop (SLT)*, pages 554–559. IEEE, 2014.

[62] Michael U Gutmann and Aapo Hyvärinen. Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics. *The journal of machine learning research*, 13(1):307–361, 2012.

[63] Patrick Haffner, Gokhan Tur, and Jerry H Wright. Optimizing svms for complex call classification. In *2003 IEEE International Conference on Acoustics, Speech, and Signal Processing, 2003. Proceedings.(ICASSP'03).*, volume 1, pages I–I. IEEE, 2003.

[64] Dilek Hakkani-Tür, Frédéric Béchet, Giuseppe Riccardi, and Gokhan Tur. Beyond asr 1-best: Using word confusion networks in spoken language understanding. *Computer Speech & Language*, 20(4):495–514, 2006.

[65] Dilek Hakkani-Tür, Gökhan Tür, Asli Celikyilmaz, Yun-Nung Chen, Jianfeng Gao, Li Deng, and Ye-Yi Wang. Multi-domain joint semantic frame parsing using bi-directional rnn-lstm. In *Interspeech*, pages 715–719, 2016.

[66] Christian Hardmeier, Arianna Bisazza, and Marcello Federico. Word lattices for morphological reduction and chunk-based reordering. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pages 88–92. Association for Computational Linguistics, 2010.

[67] Homa B Hashemi, Amir Asiaee, and Reiner Kraft. Query intent detection using convolutional neural networks. In *International Conference on Web Search and Data Mining, Workshop on Query Understanding*, 2016.

[68] Wanjia He, Weiran Wang, and Karen Livescu. Multi-view recurrent neural acoustic word embeddings. *arXiv preprint arXiv:1611.04496*, 2016.

[69] Yulan He and Steve Young. A data-driven spoken language understanding system. In *2003 IEEE Workshop on Automatic Speech Recognition and Understanding (IEEE Cat. No. 03EX721)*, pages 583–588. IEEE, 2003.

[70] Charles T Hemphill, John J Godfrey, and George R Doddington. The atis spoken language systems pilot corpus. In *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27, 1990*, 1990.

[71] Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6):82–97, 2012.

[72] Björn Hoffmeister, Dustin Hillard, Stefan Hahn, Ralf Schluter, M Ostendor, and Hermann Ney. Cross-site and intra-site asr system combination: Comparisons on lattice and 1-best methods. In *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, volume 4, pages IV–1145. IEEE, 2007.

[73] Thomas Hofmann. Probabilistic latent semantic analysis. In *Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence*, pages 289–296. Morgan Kaufmann Publishers Inc., 1999.

[74] Takaaki Hori, I Lee Hetherington, Timothy J Hazen, and James R Glass. Open-vocabulary spoken utterance retrieval using confusion networks. In *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, volume 4, pages IV–73. IEEE, 2007.

[75] Chao-Wei Huang and Yun-Nung Chen. Learning asr-robust contextualized embeddings for spoken language understanding. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8009–8013. IEEE, 2020.

[76] Eric H Huang, Richard Socher, Christopher D Manning, and Andrew Y Ng. Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 873–882. Association for Computational Linguistics, 2012.

[77] Minwoo Jeong, Sangkeun Jung, and Gary Geunbae Lee. Speech recognition error correction using maximum entropy language model. In *Proc. of INTERSPEECH*, pages 2137–2140, 2004.

[78] Hui Jiang. Confidence measures for speech recognition: A survey. *Speech communication*, 45(4):455–470, 2005.

[79] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*, 2016.

[80] Herman Kamper, Weiran Wang, and Karen Livescu. Deep convolutional acoustic word embeddings using word-pair side information. In *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*, pages 4950–4954. IEEE, 2016.

[81] Thomas Kemp and Thomas Schaaf. Estimating confidence using word lattices. In *Fifth European Conference on Speech Communication and Technology*, 1997.

[82] Joo-Kyung Kim, Gokhan Tur, Asli Celikyilmaz, Bin Cao, and Ye-Yi Wang. Intent detection using semantically enriched word embeddings. In *2016 IEEE Spoken Language Technology Workshop (SLT)*, pages 414–419. IEEE, 2016.

[83] Yoon Kim. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*, 2014.

[84] Young-Bum Kim, Sungjin Lee, and Karl Stratos. Onenet: Joint domain, intent, slot prediction for spoken language understanding. In *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 547–553. IEEE, 2017.

[85] Philipp Koehn, Franz Josef Och, and Daniel Marcu. Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 48–54. ACL, 2003.

[86] Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, et al. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th annual meeting of the ACL on interactive poster and demonstration sessions*, pages 177–180. Association for Computational Linguistics, 2007.

[87] Canasai Kruengkrai, Kiyotaka Uchimoto, Jun'ichi Kazama, Yiou Wang, Kentaro Torisawa, and Hitoshi Isahara. An error-driven word-character hybrid model for joint chinese word segmentation and pos tagging. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*, pages 513–521. Association for Computational Linguistics, 2009.

[88] Hong-Kwang Jeff Kuo and Wolfgang Reichl. Phrase-based language models for speech recognition. In *Sixth European Conference on Speech Communication and Technology*, 1999.

[89] Gakuto Kurata, Nobuyasu Itoh, and Masafumi Nishimura. Training of error-corrective model for ASR without using audio data. In *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*, pages 5576–5579. IEEE, 2011.

[90] Gakuto Kurata, Nobuyasu Itoh, Masafumi Nishimura, Abhinav Sethy, and Bhuvana Ramabhadran. Leveraging word confusion networks for named entity modeling and detection from conversational telephone speech. *Speech Communication*, 54(3):491–502, 2012.

[91] Faisal Ladhak, Ankur Gandhe, Markus Dreyer, Lambert Mathias, Ariya Rastrow, and Björn Hoffmeister. Latticernn: Recurrent neural networks over lattices. In *INTERSPEECH*, pages 695–699, 2016.

[92] Alon Lavie, Alex Waibel, Lori Levin, Michael Finke, Donna Gates, Marsal Gavalda, Torsten Zeppenfeld, and Puming Zhan. Janus-iii: Speech-to-speech translation in multiple languages. In *Acoustics, Speech, and Signal Processing, 1997. ICASSP-97., 1997 IEEE International Conference on*, volume 1, pages 99–102. IEEE, 1997.

[93] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *International Conference on Machine Learning*, pages 1188–1196, 2014.

[94] Sungbok Lee, Alexandros Potamianos, and Shrikanth Narayanan. Acoustics of children's speech: Developmental changes of temporal and spectral parameters. *The Journal of the Acoustical Society of America*, 105(3):1455–1468, 1999.

[95] Keith Levin, Katharine Henry, Aren Jansen, and Karen Livescu. Fixed-dimensional acoustic embeddings of variable-length segments in low-resource settings. In *Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on*, pages 410–415. IEEE, 2013.

[96] Omer Levy and Yoav Goldberg. Dependency-based word embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 302–308, 2014.

[97] Changliang Li, Liang Li, and Ji Qi. A self-attentive model with gate mechanism for spoken language understanding. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3824–3833, 2018.

[98] Joseph Lilleberg, Yun Zhu, and Yanqing Zhang. Support vector machines and word2vec for text classification with semantic features. In *Cognitive Informatics & Cognitive Computing (ICCI* CC), 2015 IEEE 14th International Conference on*, pages 136–140. IEEE, 2015.

[99] Wang Ling, Chris Dyer, Alan W Black, and Isabel Trancoso. Two/too simple adaptations of word2vec for syntax problems. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1299–1304, 2015.

[100] Bing Liu and Ian Lane. Attention-based recurrent neural network models for joint intent detection and slot filling. In *Interspeech 2016*, pages 685–689, 2016. doi: 10.21437/Interspeech.2016-1352.

[101] Bing Liu and Ian Lane. Joint online spoken language understanding and language modeling with recurrent neural networks. *arXiv preprint arXiv:1609.01462*, 2016.

[102] Xunying Liu, Yongqiang Wang, Xie Chen, Mark JF Gales, and Philip C Woodland. Efficient lattice rescoring using recurrent neural network language models. In *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, pages 4908–4912. IEEE, 2014.

[103] Yi Luan, Shinji Watanabe, and Bret Harsham. Efficient learning for spoken language understanding tasks with word embedding based pre-training. In *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[104] Thang Luong, Richard Socher, and Christopher Manning. Better word representations with recursive neural networks for morphology. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 104–113, 2013.

[105] Lidia Mangu, Eric Brill, and Andreas Stolcke. Finding consensus in speech recognition: word error minimization and other applications of confusion networks. *Computer Speech & Language*, 14(4):373–400, 2000.

[106] Alex Marin, Tom Kwiatkowski, Mari Ostendorf, and Luke Zettlemoyer. Using syntactic and confusion network structure for out-of-vocabulary word detection. In *Spoken Language Technology Workshop (SLT), 2012 IEEE*, pages 159–164. IEEE, 2012.

[107] Ryo Masumura, Yusuke Ijima, Taichi Asami, Hirokazu Masataki, and Ryuichiro Higashinaka. Neural confnet classification: Fully neural network based spoken utterance classification using word confusion networks. *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6039–6043, 2018.

[108] Lambert Mathias and William Byrne. Statistical phrase-based speech translation. In *Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on*, volume 1, pages I–I. IEEE, 2006.

[109] Evgeny Matusov, Stephan Kanthak, and Hermann Ney. On the integration of speech recognition and statistical machine translation. In *Ninth European Conference on Speech Communication and Technology*, 2005.

[110] Evgeny Matusov, Nicola Ueffing, and Hermann Ney. Computing consensus translation for multiple machine translation systems using enhanced hypothesis alignment. In *11th Conference of the European Chapter of the Association for Computational Linguistics*, 2006.

[111] Grégoire Mesnil, Yann Dauphin, Kaisheng Yao, Yoshua Bengio, Li Deng, Dilek Hakkani-Tur, Xiaodong He, Larry Heck, Gokhan Tur, Dong Yu, et al. Using recurrent neural networks for slot filling in spoken language understanding. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 23(3):530–539, 2015.

[112] Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernockỳ, and Sanjeev Khudanpur. Recurrent neural network based language model. In *Proceedings of Interspeech*, volume 2, page 3, 2010.

[113] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[114] Tomas Mikolov, Quoc V Le, and Ilya Sutskever. Exploiting similarities among languages for machine translation. *arXiv preprint arXiv:1309.4168*, 2013.

[115] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[116] Andriy Mnih and Koray Kavukcuoglu. Learning word embeddings efficiently with noise-contrastive estimation. In *Advances in neural information processing systems*, pages 2265–2273, 2013.

[117] Robert Moore, Douglas Appelt, John Dowding, J Mark Gawron, and Douglas Moran. Combining linguistic and statistical knowledge sources in natural-language processing for ATIS. In *Proc. ARPA Spoken Language Systems Technology Workshop*, 1995.

[118] Ryohei Nakatani, Tetsuya Takiguchi, and Yasuo Ariki. Two-step correction of speech recognition errors based on n-gram and long contextual information. In *INTERSPEECH*, pages 3747–3750, 2013.

[119] Graham Neubig and Taro Watanabe. Optimization for statistical machine translation: A survey. *Computational Linguistics*, 42(1):1–54, 2016.

[120] Jan Niehues and Muntsin Kolss. A pos-based model for long-range reorderings in smt. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 206–214. Association for Computational Linguistics, 2009.

[121] JM Noyes and CR Frankish. Errors and error correction in automatic speech recognition systems. *Ergonomics*, 37(11):1943–1957, 1994.

[122] Franz Josef Och. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 160–167. ACL, 2003.

[123] Franz Josef Och and Hermann Ney. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51, 2003.

[124] Jun Ogata and Masataka Goto. Speech repair: Quick error correction just by using selection operation for speech input interfaces. In *Ninth European Conference on Speech Communication and Technology*, 2005.

[125] Takashi Onishi, Masao Utiyama, and Eiichiro Sumita. Paraphrase lattice for statistical machine translation. *IEICE TRANSACTIONS on Information and Systems*, 94(6):1299–1305, 2011.

[126] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. ACL, 2002.

[127] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[128] Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*, 2018.

[129] Alexandros Potamianos and Shrikanth Narayanan. Robust recognition of children's speech. *IEEE Transactions on speech and audio processing*, 11(6):603–616, 2003.

[130] Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, et al. The Kaldi speech recognition toolkit. In *IEEE 2011 workshop on automatic speech recognition and understanding*, number EPFL-CONF-192584. IEEE Signal Processing Society, 2011.

[131] Siyu Qiu, Qing Cui, Jiang Bian, Bin Gao, and Tie-Yan Liu. Co-learning of word representations and morpheme representations. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 141–150, 2014.

[132] Chris Quirk, Chris Brockett, and William Dolan. Monolingual machine translation for paraphrase generation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, 2004. URL `http://aclweb.org/anthology/W04-3219`.

[133] Lawrence Rabiner and Biing-Hwang Juang. *Fundamentals of Speech Recognition*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1993. ISBN 0-13-015157-2.

[134] Lawrence R Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[135] Eric K Ringger and James F Allen. Error correction via a post-processor for continuous speech recognition. In *Acoustics, Speech, and Signal Processing, 1996. ICASSP-96. Conference Proceedings., 1996 IEEE International Conference on*, volume 1, pages 427–430. IEEE, 1996.

[136] Brian Roark, Murat Saraclar, and Michael Collins. Discriminative n-gram language modeling. *Computer Speech & Language*, 21(2):373–392, 2007.

[137] Ronald Rosenfeld. Two decades of statistical language modeling: Where do we go from here? *Proceedings of the IEEE*, 88(8):1270–1278, 2000.

[138] Antti-Veikko Rosti, Necip Fazil Ayan, Bing Xiang, Spyros Matsoukas, Richard Schwartz, and Bonnie Dorr. Combining outputs from multiple machine translation systems. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 228–235, 2007.

[139] Antti-Veikko Rosti, Spyros Matsoukas, and Richard Schwartz. Improved word-level system combination for machine translation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 312–319, 2007.

[140] Anthony Rousseau, Paul Deléglise, and Yannick Estève. Enhancing the ted-lium corpus with selected data for language modeling and more ted talks. In *LREC*, pages 3935–3939, 2014.

[141] Kenji Sagae, Maider Lehr, E Prud'hommeaux, Puyang Xu, Nathan Glenn, Damianos Karakos, Sanjeev Khudanpur, Brian Roark, Murat Saraclar, Izhak Shafran, et al. Hallucinated n-best lists for discriminative language modeling. In *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*, pages 5001–5004. IEEE, 2012.

[142] Arup Sarma and David D Palmer. Context-based speech recognition error detection and correction. In *Proceedings of HLT-NAACL 2004: Short Papers*, pages 85–88. Association for Computational Linguistics, 2004.

[143] Tobias Schnabel, Igor Labutov, David Mimno, and Thorsten Joachims. Evaluation methods for unsupervised word embeddings. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 298–307, 2015.

[144] Josh Schroeder, Trevor Cohn, and Philipp Koehn. Word lattices for multi-source translation. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 719–727. Association for Computational Linguistics, 2009.

[145] Tanja Schultz, Szu-Chen Jou, Stephan Vogel, and Shirin Saleem. Using word latice information for a tighter coupling in speech translation systems. In *Eighth International Conference on Spoken Language Processing*, 2004.

[146] Raphael Schumann and Pongtep Angkititrakul. Incorporating asr errors with attention-based, jointly trained rnn for intent detection and slot filling. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6059–6063. IEEE, 2018.

[147] Matthew Stephen Seigel and Philip C Woodland. Combining information sources for confidence estimation with crf models. In *Twelfth Annual Conference of the International Speech Communication Association*, 2011.

[148] Prashanth Gurunath Shivakumar and Panayiotis Georgiou. Confusion2vec: Towards enriching vector space word representations with representational ambiguities. *PeerJ Computer Science*, 5:e195, 2019.

[149] Prashanth Gurunath Shivakumar and Panayiotis G. Georgiou. Confusion2vec: Towards enriching vector space word representations with representational ambiguities. *CoRR*, abs/1811.03199, 2018.

[150] Prashanth Gurunath Shivakumar, Alexandros Potamianos, Sungbok Lee, and Shrikanth Narayanan. Improving speech recognition for children using acoustic adaptation and pronunciation modeling. In *Proc. Workshop on Child, Computer and Interaction (WOCCI)*, 2014.

[151] Prashanth Gurunath Shivakumar, Haoqi Li, Kevin Knight, and Panayiotis Georgiou. Learning from past mistakes: Improving automatic speech recognition output via noisy-clean phrase context modeling. *arXiv preprint arXiv:1802.02607*, 2018.

[152] Prashanth Gurunath Shivakumar, Mu Yang, and Panayiotis Georgiou. Spoken language intent detection using confusion2vec. *arXiv preprint arXiv:1904.03576*, 2019.

[153] Scharolta Katharina Sienčnik. Adapting word2vec to named entity recognition. In *Proceedings of the 20th nordic conference of computational linguistics, nodalida 2015, may 11-13, vilnius, lithuania*, number 109, pages 239–243. Linköping University Electronic Press, 2015.

[154] Edwin Simonnet, Sahar Ghannay, Nathalie Camelin, and Yannick Estève. Simulating asr errors for training slu systems. In *Proceedings of the 11th Language Resources and Evaluation Conference*, Miyazaki, Japan, May 2018. European Language Resource Association.

[155] Radu Soricut and Franz Och. Unsupervised morphology induction using word embeddings. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1627–1637, 2015.

[156] Matthias Sperber, Graham Neubig, Jan Niehues, and Alex Waibel. Neural lattice-to-sequence models for uncertain inputs. *arXiv preprint arXiv:1704.00559*, 2017.

[157] Matthias Sperber, Graham Neubig, Ngoc-Quan Pham, and Alex Waibel. Self-attentional models for lattice inputs. *arXiv preprint arXiv:1906.01617*, 2019.

[158] Sabrina Stehwien and Ngoc Thang Vu. First step towards enhancing word embeddings with pitch accent features for dnn-based slot filling on recognized text. In *Konferenz Elektronische Sprachsignalverarbeitung 2017, Saarbrücken*, 2017.

[159] Andreas Stolcke. SRILM-an extensible language modeling toolkit. In *Seventh international conference on spoken language processing*, 2002.

[160] Helmer Strik and Catia Cucchiarini. Modeling pronunciation variation for ASR: A survey of the literature. *Speech Communication*, 29(2):225–246, 1999.

[161] Jinsong Su, Zhixing Tan, Deyi Xiong, Rongrong Ji, Xiaodong Shi, and Yang Liu. Lattice-based recurrent neural network encoders for neural machine translation. In *AAAI*, pages 3302–3308, 2017.

[162] Bernhard Suhm, Brad Myers, and Alex Waibel. Multimodal error correction for speech user interfaces. *ACM transactions on computer-human interaction (TOCHI)*, 8(1):60–98, 2001.

[163] Martin Sundermeyer, Ralf Schlüter, and Hermann Ney. LSTM neural networks for language modeling. In *Proceedings of Interspeech*, pages 194–197, 2012.

[164] Martin Sundermeyer, Zoltán Tüske, Ralf Schlüter, and Hermann Ney. Lattice decoding and rescoring with long-span neural network language models. In *Fifteenth Annual Conference of the International Speech Communication Association*, 2014.

[165] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

[166] Kai Sheng Tai, Richard Socher, and Christopher D Manning. Improved semantic representations from tree-structured long short-term memory networks. *arXiv preprint arXiv:1503.00075*, 2015.

[167] Yik-Cheung Tam, Yun Lei, Jing Zheng, and Wen Wang. Asr error detection using recurrent neural network language model and complementary ASR. In *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, pages 2312–2316. IEEE, 2014.

[168] Qun Feng Tan, Kartik Audhkhasi, Panayiotis G Georgiou, Emil Ettelaie, and Shrikanth S Narayanan. Automatic speech recognition system channel modeling. In *Eleventh Annual Conference of the International Speech Communication Association*, 2010.

[169] Zhixing Tan, Jinsong Su, Boli Wang, Yidong Chen, and Xiaodong Shi. Lattice-to-sequence attentional neural machine translation models. *Neurocomputing*, 284:138–147, 2018.

[170] Gokhan Tur, Jerry Wright, Allen Gorin, Giuseppe Riccardi, and Dilek Hakkani-Tür. Improving spoken language understanding using word confusion networks. In *Seventh International Conference on Spoken Language Processing*, 2002.

[171] Gökhan Tür, Anoop Deoras, and Dilek Hakkani-Tür. Semantic parsing using word confusion networks with conditional random fields. In *INTERSPEECH*, pages 2579–2583. Citeseer, 2013.

[172] Ashish Vaswani, Yinggong Zhao, Victoria Fossum, and David Chiang. Decoding with large-scale neural language models improves translation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1387–1392. EMNLP, 2013.

[173] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[174] Wolfgang Wahlster. *Verbmobil: foundations of speech-to-speech translation*. Springer Science & Business Media, 2013.

[175] R Weide. The cmu pronunciation dictionary, release 0.6, 1998.

[176] Mirjam Wester. Pronunciation modeling for ASR–knowledge-based and data-derived methods. *Computer Speech & Language*, 17(1):69–85, 2003.

[177] P.C. Woodland and D. Povey. Large scale discriminative training of hidden Markov models for speech recognition. *Computer Speech & Language*, 16(1):25 – 47, 2002. ISSN 0885-2308. doi: http://dx.doi.org/10.1006/csla.2001.0182. URL `http://www.sciencedirect.com/science/article/pii/S0885230801901822`.

[178] Joern Wuebker and Hermann Ney. Phrase model training for statistical machine translation with word lattices of preprocessing alternatives. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 450–459. Association for Computational Linguistics, 2012.

[179] Fengshun Xiao, Jiangtong Li, Hai Zhao, Rui Wang, and Kehai Chen. Lattice-based transformer encoder for neural machine translation. *arXiv preprint arXiv:1906.01282*, 2019.

[180] Chao Xing, Dong Wang, Xuewei Zhang, and Chao Liu. Document classification with distributions of word vectors. In *Signal and Information Processing Association Annual Summit and Conference (APSIPA), 2014 Asia-Pacific*, pages 1–5. IEEE, 2014.

[181] Wayne Xiong, Jasha Droppo, Xuedong Huang, Frank Seide, Mike Seltzer, Andreas Stolcke, Dong Yu, and Geoffrey Zweig. Achieving human parity in conversational speech recognition. *arXiv preprint arXiv:1610.05256*, 2016.

[182] Haihua Xu, Daniel Povey, Lidia Mangu, and Jie Zhu. Minimum bayes risk decoding and system combination based on a recursion for edit distance. *Computer Speech & Language*, 25(4):802–828, 2011.

[183] Puyang Xu and Ruhi Sarikaya. Convolutional neural network based triangular crf for joint intent detection and slot filling. In *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*, pages 78–83. IEEE, 2013.

[184] Puyang Xu, Brian Roark, and Sanjeev Khudanpur. Phrasal cohort based unsupervised discriminative language modeling. In *Thirteenth Annual Conference of the International Speech Communication Association*, 2012.

[185] Jian Xue and Yunxin Zhao. Improved confusion network algorithm and shortest path search from word lattice. In *Acoustics, Speech, and Signal Processing, 2005. Proceedings.(ICASSP'05). IEEE International Conference on*, volume 1, pages I–853. IEEE, 2005.

[186] Sibel Yaman, Li Deng, Dong Yu, Ye-Yi Wang, and Alex Acero. An integrative and discriminative technique for spoken utterance classification. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(6):1207–1214, 2008.

[187] Wenpeng Yin and Hinrich Schütze. Learning word meta-embeddings. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*, 2016. URL `http://aclweb.org/anthology/P/P16/P16-1128.pdf`.

[188] Huifeng Zhang, Su Zhu, Shuai Fan, and Kai Yu. Joint spoken language understanding and domain adaptive language modeling. In *International Conference on Intelligent Science and Big Data Engineering*, pages 311–324. Springer, 2018.

[189] Xiaodong Zhang and Houfeng Wang. A joint model of intent determination and slot filling for spoken language understanding. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, IJCAI'16, pages 2993–2999. AAAI Press, 2016. ISBN 978-1-57735-770-4.

[190] Xiaohui Zhang, Jan Trmal, Daniel Povey, and Sanjeev Khudanpur. Improving deep neural network acoustic models using generalized maxout networks. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 215–219. IEEE, 2014.

[191] D. Zheng, Z. Chen, Y. Wu, and K. Yu. Directed automatic speech transcription error correction using bidirectional LSTM. In *2016 10th International Symposium on Chinese Spoken Language Processing (ISCSLP)*, pages 1–5, Oct 2016. doi: 10.1109/ISCSLP.2016.7918446.

[192] Su Zhu, Ouyu Lan, and Kai Yu. Robust spoken language understanding with unsupervised asr-error adaptation. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6179–6183. IEEE, 2018.

Appendix A

Confusion2Vec

Model	Analogy Tasks							
	Semantic&Syntactic Analogy			Acoustic Analogy	Semantic&Syntactic-Acoustic Analogy			Average Accuracy
	Semantic	Syntactic	Semantic&Syntactic		Semantic-Acoustic	Syntactic-Acoustic	Semantic&Syntactic-Acoustic	
Google Word2Vec	28.98% (35.75%)	70.79% (78.74%)	61.42% (69.1%)	0.9% (1.42%)	6.54% (14.38%)	17.9% (27.46%)	16.99% (26.42%)	26.44% (32.31%)
Word2Vec GroundTruth	42.39% (51.57%)	33.14% (43.14%)	35.15% (44.98%)	0.3% (0.6%)	5.17% (10.69%)	8.13% (11.93%)	7.86% (11.82%)	14.44% (19.13%)
Baseline Word2Vec	38.33% (46.7%)	33.1% (42.36%)	34.27% (43.33%)	0.7% (1.16%)	11.76% (14.38%)	11.23% (15.11%)	11.27% (15.05%)	15.41% (19.85%)
Intra-Confusion	0.51% (0.78%)	18.59% (28.17%)	14.54% (22.03%)	41.93% (52.58%)	0.98% (2.29%)	9.62% (15.67%)	8.94% (14.61%)	21.8% (29.74%)
Inter-Confusion	16.15% (23.7%)	26.14% (39.74%)	23.9% (36.15%)	48.58% (60.57%)	3.27% (6.86%)	12.13% (21.61%)	11.42% (20.44%)	27.97% (39.05%)
Hybrid Intra-Inter	2.07% (2.58%)	28.91% (38.6%)	22.89% (30.53%)	40.78% (53.55%)	1.96% (2.94%)	20.99% (31.63%)	19.48% (29.35%)	27.72% (37.81%)

Table A.1: **Analogy Task Results with Semantic & Syntactic splits: Different proposed models**

Numbers inside parenthesis indicate top-2 evaluation accuracy;
Numbers outside parenthesis represent top-1 evaluation accuracy.
Google Word2Vec, Word2Vec Groundtruth (trained on in-domain) and Baseline Word2Vec (trained on ASR transcriptions) perform better with the Semantic&Syntactic tasks, but fares poorly with Acoustic analogy task.
Intra-Confusion performs well on Acoustic analogy task while compromising on Semantic&Syntactic task.
Inter-Confusion performs well on both the Acoustic analogy and Semantic&Syntactic tasks.
Hybrid Intra-Inter training performs fairly well on all the three analogy tasks (Acoustic, Semantic&Syntactic and Semantic&Syntactic-Acoustic).

Model	Similarity Tasks	
	Word Similarity	Acoustic Similarity
Google Word2Vec	0.6893 (7.9e-48)	-0.3489 (2.2e-28)
Word2Vec GroundTruth	0.5794 (4.2e-29)	-0.2444 (1e-10)
Baseline Word2Vec	0.4992 (3.3e-22)	0.1944 (1.7e-9)
Intra-Confusion	0.105 (0.056)	0.8138 (5.1e-224)
Inter-Confusion	0.2937 (5.4e-8)	0.8055 (5.1e-216)
Hybrid Intra-Inter	0.0963 (0.08)	0.7858 (1.5e-198)

Table A.2: **Similarity Task Results: Different proposed models**
Similarity in terms of Spearman's correlation.
Numbers inside parenthesis indicate correlation $p - value$ for similarity tasks
Google Word2Vec, Baseline Word2Vec and Word2Vec Groundtruth, all show high correlations with word similarity, while showing poor correlations on acoustic similarity. Google Word2Vec and Word2Vec Groundtruth models trained on clean data exhibit negative acoustic similarity correlation. Baseline Word2Vec trained on noisy ASR shows a small positive acoustic similarity correlation.
Intra-Confusion, Inter-Confusion and Hybrid Intra-Inter training show higher correlations on Acoustic similarity.

Model	Analogy Tasks							
	Semantic&Syntactic Analogy			Acoustic Analogy	Semantic&Syntactic-Acoustic Analogy			Average Accuracy
	Semantic	Syntactic	Semantic&Syntactic		Semantic-Acoustic	Syntactic-Acoustic	Semantic&Syntactic-Acoustic	
Baseline Word2Vec	34.92% (41.96%)	68.7% (78.82%)	61.13% (70.56%)	0.9% (1.46%)	14.38% (19.28%)	16.85% (24.25%)	16.66% (23.86%)	26.23% (31.96%)
Intra-Confusion	11.5% (15.53%)	67.56% (77.96%)	54.99% (63.97%)	9.04% (16.92%)	7.84% (10.46%)	36.92% (46.17%)	34.61% (43.34%)	32.88% (41.41%)
Inter-Confusion	25.77% (33.12%)	60.1% (74.79%)	52.4% (65.45%)	16.54% (27.33%)	10.78% (14.05%)	28.9% (40.38%)	27.46% (38.29%)	32.13% (43.69%)
Hybrid Intra-Inter	15.64% (21.94%)	66.73% (77.68%)	55.28% (65.19%)	10.49% (20.35%)	6.86% (11.11%)	35.4% (44.85%)	33.13% (42.18%)	36.27% (42.57%)

Table A.3: **Analogy Task Results with Semantic & Syntactic splits: Model pre-training/initialization**
Numbers inside parenthesis indicate top-2 evaluation accuracy;
Numbers outside parenthesis represent top-1 evaluation accuracy.
Pre-training is helpful in all the cases. Pre-training boosts the Semantic&Syntactic Analogy accuracy for all.
For Intra-Confusion, Inter-Confusion and Hybrid Intra-Inter models, pre-training boosts the Semantic&Syntactic-Acoustic Analogy accuracies. A small dip in Acoustic Analogy accuracies is observed. However, overall average accuracy is improved.

Model	Similarity Tasks	
	Word Similarity	Acoustic Similarity
Baseline Word2Vec	0.6036 (3.8e-34)	-0.4327 (2.5e-44)
Intra-Confusion	0.5228 (1.4e-24)	0.62 (2.95e-101)
Inter-Confusion	0.5798 (4.9e-31)	0.5825 (9.1e-87)
Hybrid Intra-Inter	0.5341 (9.8e-26)	0.6237 (8.8e-103)

Table A.4: **Similarity Task Results: Model pre-training/initialization**
Similarity in terms of Spearman's correlation.
Numbers inside parenthesis indicate correlation $p - value$ for similarity tasks.
Pre-training boosts the Word Similarity correlation for all the models. The correlation is improved considerably in the case of Intra-Confusion, Inter-Confusion and Hybrid Intra-Inter models while maintaining good correlation on acoustic similarity.

Model	Fine-tuning Scheme	Analogy Tasks							
		Semantic&Syntactic Analogy			Acoustic Analogy	Semantic&Syntactic-Acoustic Analogy			Average Accuracy
		Semantic	Syntactic	Semantic&Syntactic		Semantic-Acoustic	Syntactic-Acoustic	Semantic&Syntactic-Acoustic	
Baseline Word2Vec (556 dim.)	-	34.49% (41.53%)	68.7% (78.82%)	61.13% (70.25%)	0.93% (1.46%)	15.36% (19.93%)	16.63% (23.89%)	16.53% (23.57%)	26.2% (31.76%)
Model Concatenation									
Word2Vec (F) + Intra-Confusion (F)	-	6.22% (9.5%)	71.03% (83.65%)	56.51% (67.03%)	13.59% (25.43%)	6.54% (11.76%)	33.91% (42.8%)	31.74% (40.36%)	33.95% (44.27%)
Word2Vec (F) + Inter-Confusion (F)	-	36.53% (47.01%)	57.94% (77.72%)	53.14% (70.84%)	20.99% (35.25%)	10.46% (16.01%)	26.31% (36.83%)	25.05% (35.18%)	33.06% (47.09%)
Word2Vec (F) + Hybrid Intra-Inter (F)	-	11.85% (17.32%)	71.85% (82.74%)	58.4% (68.08%)	6.35% (11.39%)	7.84% (12.18%)	34.38% (43.78%)	32.28% (41.3%)	32.34% (40.26%)
Fixed Word2Vec Joint Optimization									
Word2Vec (F) + Intra-Confusion (L)	inter	22.96% (32.42%)	66.19% (82.98%)	56.5% (71.65%)	12.73% (20.54%)	13.4% (18.3%)	26.22% (35.09%)	25.21% (33.76%)	31.48% (41.98%)
Word2Vec (F) + Intra-Confusion (L)	intra	6.69% (11.58%)	69.79% (83.48%)	55.65% (67.37%)	17.03% (28.64%)	8.17% (13.73%)	31.85% (47.64%)	29.97% (39.09%)	34.22% (45.03%)
Word2Vec (F) + Intra-Confusion (L)	hybrid	11.69% (19.79%)	69.31% (84.53%)	56.39% (70.02%)	14.86% (25.84%)	9.8% (16.67%)	30.02% (38.94%)	28.42% (37.18%)	33.22% (44.35%)
Word2Vec (F) + Inter-Confusion (L)	inter	39.19% (50.57%)	58.35% (78.21%)	54.05% (72.01%)	23.33% (35.25%)	12.42% (18.3%)	24.45% (34.89%)	23.5% (33.58%)	33.63% (46.95%)
Word2Vec (F) + Inter-Confusion (L)	intra	22.76% (32.85%)	62.07% (80.34%)	53.26% (69.7%)	24.76% (39.32%)	7.52% (11.11%)	29.97% (41.47%)	28.19% (39.07%)	35.40% (49.36%)
Word2Vec (F) + Inter-Confusion (L)	hybrid	30.54% (43.21%)	61.56% (80.81%)	54.61% (72.38%)	23.6% (37.75%)	8.5% (14.71%)	28.25% (39.95%)	26.68% (37.95%)	34.96% (49.36%)
Word2Vec (F) + Hybrid Intra-Inter (L)	inter	27.02% (35.9%)	67.52% (81.6%)	58.45% (71.36%)	5.04% (8.55%)	11.76% (16.67%)	26.28% (34.64%)	25.13% (33.21%)	29.54% (37.71%)
Word2Vec (F) + Hybrid Intra-Inter (L)	intra	10.48% (15.84%)	70.44% (81.57%)	57.00% (66.85%)	7.21% (13.33%)	6.21% (12.09%)	34.07% (42.52%)	31.87% (40.1%)	32.03% (40.09)
Word2Vec (F) + Hybrid Intra-Inter (L)	hybrid	15.41% (23.31%)	70.56% (82.61%)	58.2% (68.32%)	6.39% (11.61%)	8.17% (12.09%)	32.36% (40.43%)	30.44% (38.19%)	31.68% (39.37%)
Unrestricted Joint Optimization									
Word2Vec (L) + Intra-Confusion (L)	inter	8.6% (14.74%)	57.96% (75.8%)	46.9% (62.12%)	30.73% (46.42%)	5.88% (12.75%)	26.79% (38.44%)	25.13% (36.4%)	34.25% (48.31%)
Word2Vec (L) + Intra-Confusion (L)	intra	4.97% (7.9%)	69.27% (81.30%)	54.86% (64.85%)	23.86% (40.55%)	7.84% (11.44%)	34.92% (45.02%)	32.77% (42.38%)	37.16% (49.26%)
Word2Vec (L) + Intra-Confusion (L)	hybrid	1.1% (1.64%)	26.54% (40.32%)	20.83% (31.65%)	49.25% (61.91%)	2.29% (3.92%)	15.05% (25.24%)	14.04% (23.55%)	28.12% (39.04%)
Word2Vec (L) + Inter-Confusion (L)	inter	33.01% (43.72%)	50.81% (71.13%)	46.82% (64.98%)	37.15% (52.99%)	9.48% (16.01%)	23.16% (36.41%)	22.07% (34.79%)	35.35% (50.92%)
Word2Vec (L) + Inter-Confusion (L)	intra	21.9% (30.43%)	58.99% (76.12%)	50.68% (65.88%)	33.05% (49.4%)	7.52% (10.46%)	31.23% (44.12%)	29.35% (41.51%)	37.69% (52.26%)
Word2Vec (L) + Inter-Confusion (L)	hybrid	10.48% (15.72%)	30.0% (44.25%)	25.63% (37.86%)	52.73% (67.21%)	3.27% (4.9%)	16.09% (27.77%)	15.08% (25.96%)	31.15% (43.68%)
Word2Vec (L) + Hybrid Intra-Inter (L)	inter	19.24% (26.59%)	61.57% (76.8%)	52.08% (65.54%)	17.85% (27.97%)	7.52% (12.75%)	28.81% (38.94%)	27.12% (36.87%)	32.35% (43.46%)
Word2Vec (L) + Hybrid Intra-Inter (L)	intra	10.09% (13.77%)	68.76% (79.06%)	55.61% (64.42%)	10.34% (20.05%)	5.88% (9.48%)	36.13% (45.41%)	33.73% (42.56%)	33.23% (42.34%)
Word2Vec (L) + Hybrid Intra-Inter (L)	hybrid	12.98% (17.91%)	68.26% (79.62%)	55.87% (65.79%)	11.73% (22.63%)	5.88% (10.46%)	35.28% (43.92%)	32.95% (41.3%)	33.52% (43.24%)

Table A.5: **Analogy Task Results: Model concatenation and joint optimization**
Numbers inside parenthesis indicate top-2 evaluation accuracy;
Numbers outside parenthesis represent top-1 evaluation accuracy.
Acronyms: (F):Fixed embedding, (L):Learn embedding during joint training
Model Concatenation provides gains in Acoustic-Analogy Task and thereby resulting in gains in average accuracy compared to results in Table A.3 for Intra-Confusion and Inter-Confusion models.
Fixed Word2Vec and Unrestricted Joint Optimizations further improves results over model concatenation. Best results in terms of average accuracy is obtained with unrestricted joint optimizations, an absolute improvement of 10%.
Confusion2Vec models surpass Word2Vec even for Semantic&Syntactic analogy task (top-2 evaluation accuracy).

Model	Fine-tuning Scheme	Similarity Tasks	
		Word Similarity	Acoustic Similarity
Baseline Word2Vec (556 dim.)	-	0.5973 (2.8e-33)	-0.4341 (1.3e-44)
Model Concatenation			
Word2Vec (F) + Intra-Confusion (F)	-	0.5102 (2.9e-23)	0.7231 (2.2e-153)
Word2Vec (F) + Inter-Confusion (F)	-	0.5609 (9.8e-29)	0.6345 (2.3e-107)
Word2Vec (F) + Hybrid Intra-Inter (F)	-	0.4142 (4.1e-15)	0.5285 (5.6e-69)
Fixed Word2Vec Joint Optimization			
Word2Vec (F) + Intra-Confusion (L)	inter	0.5676 (1.6e-29)	0.4437 (9.1e-47)
Word2Vec (F) + Intra-Confusion (L)	intra	0.5211 (2.3e-24)	0.6967 (6.5e-138)
Word2Vec (F) + Intra-Confusion (L)	hybrid	0.5384 (3.4e-26)	0.6287 (6.7e-105)
Word2Vec (F) + Inter-Confusion (L)	inter	0.5266 (6.1e-25)	0.5818 (1.6e-86)
Word2Vec (F) + Inter-Confusion (L)	intra	0.5156 (8.3e-24)	0.7021 (6.3e-141)
Word2Vec (F) + Inter-Confusion (L)	hybrid	0.5220 (1.8e-24)	0.6674 (1.4e-122)
Word2Vec (F) + Hybrid Intra-Inter (L)	inter	0.5587 (1.7e-28)	0.302 (2.5e-21)
Word2Vec (F) + Hybrid Intra-Inter (L)	intra	0.4996 (3.1e-22)	0.5691 (4.7e-82)
Word2Vec (F) + Hybrid Intra-Inter (L)	hybrid	0.5254 (8.2e-25)	0.4945 (2.6e-59)
Unrestricted Joint Optimization			
Word2Vec (L) + Intra-Confusion (L)	inter	0.5513 (1.3e-27)	0.7926 (2.4e-204)
Word2Vec (L) + Intra-Confusion (L)	intra	0.5033 (1.4e-22)	0.7949 (2e-206)
Word2Vec (L) + Intra-Confusion (L)	hybrid	0.1067 (0.0528)	0.8309 (8.5e-242)
Word2Vec (L) + Inter-Confusion (L)	inter	0.5763 (1.3e-30)	0.7725 (8.2e-188)
Word2Vec (L) + Inter-Confusion (L)	intra	0.5379 (3.8e-26)	0.7717 (3.5e-187)
Word2Vec (L) + Inter-Confusion (L)	hybrid	0.2295 (2.6e-5)	0.8294 (3.6e-240)
Word2Vec (L) + Hybrid Intra-Inter (L)	inter	0.5338 (1e-25)	0.6953 (3.7e-137)
Word2Vec (L) + Hybrid Intra-Inter (L)	intra	0.4920 (1.6e-21)	0.6942 (1.5e-136)
Word2Vec (L) + Hybrid Intra-Inter (L)	hybrid	0.4967 (5.8e-22)	0.6986 (5.9e-139)

Table A.6: **Similarity Task Results: Model concatenation and joint optimization**
Similarity in terms of Spearman's correlation.
Numbers inside parenthesis indicate correlation $p - value$ for similarity tasks.
Good correlations are observed for both the word similarity and acoustic similarity with model concatenation with and without joint optimization. All the correlations are found to be statistically significant.