Hamburg University
Faculty of Mathematics,
Informatics and Natural Sciences
Department of Informatics

Exposé

Incremental Speech Recognition from Several Combined Sources

Natalia Orlova

2orlova@informatik.uni-hamburg.de
Degree Program Master Informatics
Matr.-No. 6459416

First supervisor Hamburg University: Dr. Timo Baumann
Second supervisor Hamburg University: <>

1 Introduction

1.1 Motivation

In recent years, Automatic Speech Recognition (ASR) systems have improved increasingly, being used in everyday applications: Siri, Google ASR (McGraw and Gruenstein 2012). However, most of such systems work asynchronously in respect to output, computing the result after the utterance is already finished. In the area of Human-Machine Interaction, where intermediate system reactions of ASRs are desirable, incremental output of the intermediate results becomes increasingly important. Benefits of incremental speech recognition include post-processing time savings, faster system feedback and more natural dialogue between humans and intelligent systems. Commercial Google recognition engine, working in incremental mode, produces accurate results in non-specific domains, but demonstrates high latency (McGraw and Gruenstein 2012). Non-commercial open source systems, like Sphinx-4, on the other hand, demonstrate very short delays and can be timed to specific applications, but are less reliable in accuracy. The challenge is to combine the advantages of both systems and to overcome the latency problem of Google-ASR.

1.2 Problem Statement

As already stated above, the main problem of the incremental recognition is the trade-off between system timeliness and recognition quality. In other words, we are interested in development of such an engine that comes to a stable and accurate synchronous result with a minimum possible latency.

1.3 Objectives

The aim of the master thesis is to improve the timeliness of incremental speech recognition by developing a state-of-the art combination of Google Incremental ASR (McGraw and Gruenstein 2012) and an incremental speech recogniser, based on Sphinx-4 (Baumann, Atterer, and Schlangen 2009). The architecture of the incremental recogniser is shown in the Figure 1. The first aim is to get the detailed timing information for incremental Google-ASR results, using forced alignment technique. The second objective is to be able to reset the SearchGraph of the Sphinx-4 recogniser, by going back to the saved state and choosing an alternative path. The third central aim is to manipulate SearchGraph. Manipulation includes doing reset, depending on the timing information about forced-aligned

phonemes, coming from Google ASR and changing the hypotheses path. Furthermore it should be able to make a combination of Google-ASR and Sphinx-4, using Google incremental output as a guidance for SearchGraph manipulation and hypotheses correction, when Google incremental output differs from Sphinx-4.

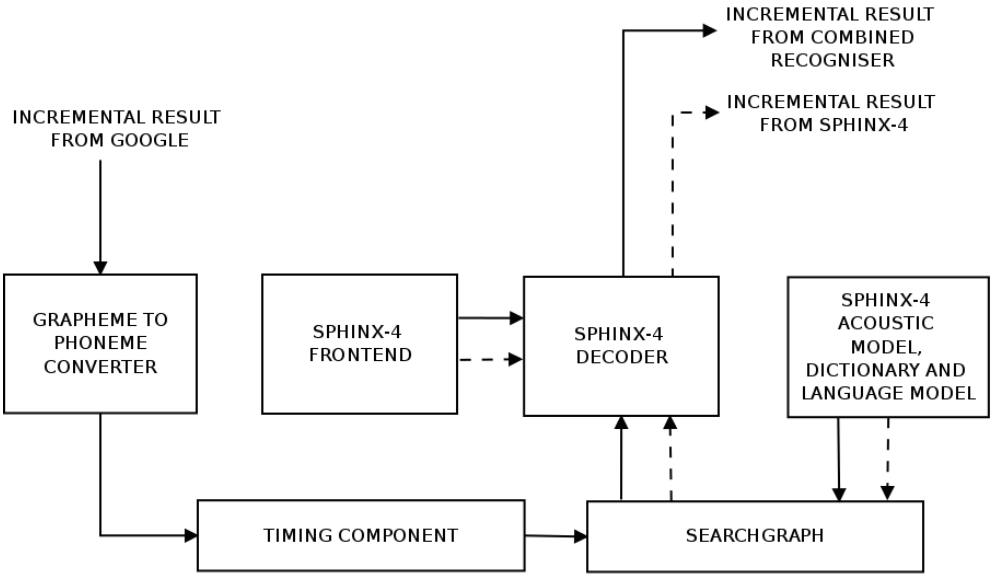

Figure 1: Incremental recogniser from combined sources

The final objective is to evaluate and to visualize timeliness of the implemented incremental recognizer, using a combined technique in comparison with Google-ASR system or Sphinx-4 based system alone.

1.4 Structure

This exposé is structured as follows. Section 2 provides review of scientific papers, related to the thesis topic. Section 3 is a brief statement of the research questions. Section 4 presents an example to illustrate the problem of timeliness. Section 5 contains methods to be used for this research. Section 6 describes the evaluation approach and expected results. Section 7 contains preliminary table of contents for the upcoming thesis. In the section 8 a timeline for the thesis is presented. Section 9 summarizes the exposé.

2 Related Work

The architecture of contemporary speech recognition systems includes basic ASR components: acoustic input, acoustic model, language model and decoder. ASR decoder gets as input feature vectors of audio signal and the results of acoustic and language modelling and produces a decoded word and phone alignment as an output (Jurafsky and Martin 2009).

Most traditional ASRs, including commercial, are not incremental. They produce the output result after the utterance is finished, resulting in a delay up to 750-1500 ms. In additional they may use multiple-pass decoding, which is not possible incrementally (Skantze and Schlangen 2009). Incremental dialogue processing allows reducing of feedback time, bringing the dialogues in interactive environments closer to natural ones. Apart from abstract models of incremental dialogue processing there exist an implementation of a limited micro-domain system, recognising sequence numbers (Skantze and Schlangen 2009).

A good example of traditional ASR architecture is the implementation of the open-source CMU Sphinx-4 recogniser (Lamere et al. 2003). Sphinx-4 was not designed as an incremental system, but its modular design allows extension and adding new components, required for an incremental speech recogniser. (Baumann, Atterer, and Schlangen 2009).

Apart from Sphinx-4 the leading speech recognition systems is commercial Google Voice Search (Schalkwyk et al. 2010).

State-of-art Google Search by Voice, trained using huge amounts of audio data, presents the best results in the terms of accuracy for a standard non-incremental task. However, when intermediate results are produced synchronously in an incremental mode we see a trade-off between stability of the input and latency (McGraw and Gruenstein 2012).

Further restriction of Google is its task-orientated recognition, primarily aimed to interactive Google command search. This generally means that simple transfer of Google technology to domain-specific natural dialogue systems and context-specific HRI environments leads to lower accuracy.

Recently, there has been proposed a solution for a domain-dependent applications, using a combination of Google phonetic post processing and Sphinx-4. Original frontend of Sphinx-4 is replaced by phoneme frontend, which converts the Google result to its phonetic representation. Test results of the combined approach have shown a significant improvement of recognition results for non-incremental tasks. Combination technique is considered to be transferable to the incremental problem solution (Twiefel et al. 2014).

3 Research Question

The main research question of the master thesis is timeliness improvement of an incremental speech recogniser, implementing forced alignment of Google incremental results, Sphinx4 searchGraph manipulation and developing a combination of Google ASR and Sphinx-4 recogniser (cmp.Twiefel et al. 2014).

As in incremental recogniser intermediate results are produced while the speaker has not yet finished the utterance, the focus of the research becomes detailed timing of input and output. Monitoring of its own input in relation to output is a crucial point for a system to be able to revoke or commit the state of the recogniser (Skantze and Schlangen 2009).

Under the above considerations, we are dealing with the following research subquestions. The first problem is forced alignment of Google incremental results with audio input. The idea is to get an accurate timing information for Google incremental results and to improve in such a way the existing Google timing. For calculating of the actual timing standard deviation and nets latency are to be taken into account. Second problem is studying the possibility of Sphinx4 SearchGraph manipulations and hypothesis path reset within schinx4. Third problem is improvement of Sphinx4 recogniser performance by combining Sphinx4 and Google.

4 Example

4.1 Forced Alignment

In the following section a simple example is provided to sketch the problems to be researched in the master thesis. For this example a simple audio file 'DE_1234.wav' is used, where the numbers 1-2-3-4, pronounced one after another in German language, are recorded.

Picture 2 visualizes time alignments, created manually (transcription '.manual') and with the help of existing forced alignment module (transcription '.sphinx'), implemented in Inprotk, using sphinx4 recogniser alone. Incremental results, coming direct from Google with particular network latency, contain only information about hypotheses timing. Transcription .google shows forced alignment, applied to Google results and calculated by the existing time-alignment module (Baumann, Atterer, and Schlangen 2009). As it can be seen from the picture the alignment of the sphinx approximates the true alignment, whereas the timing, calculated for Google, is not only delayed by the net latency, but also

incorrect in duration.

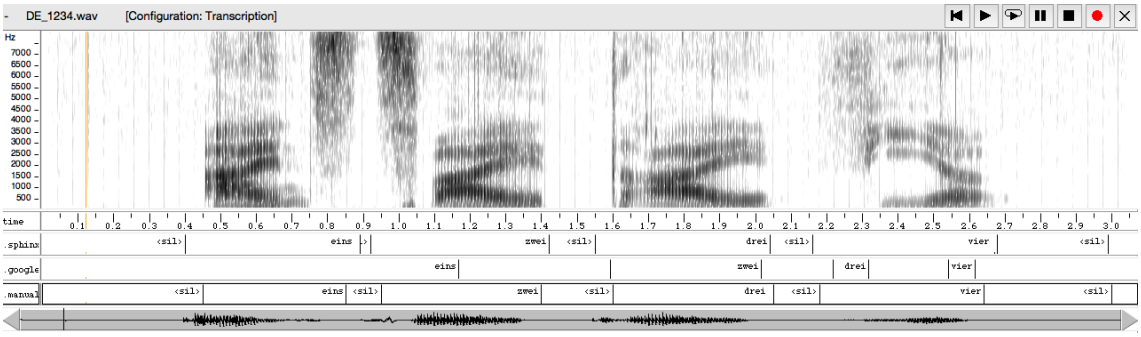

Figure 2: Audio Input Alignment

Pictures 3 and 4 show the incremental output of both ASRs: sphinx4 and Google in detail. This pictures also illustrate that outputs differs not only in the number of hypotheses and nets latency time.

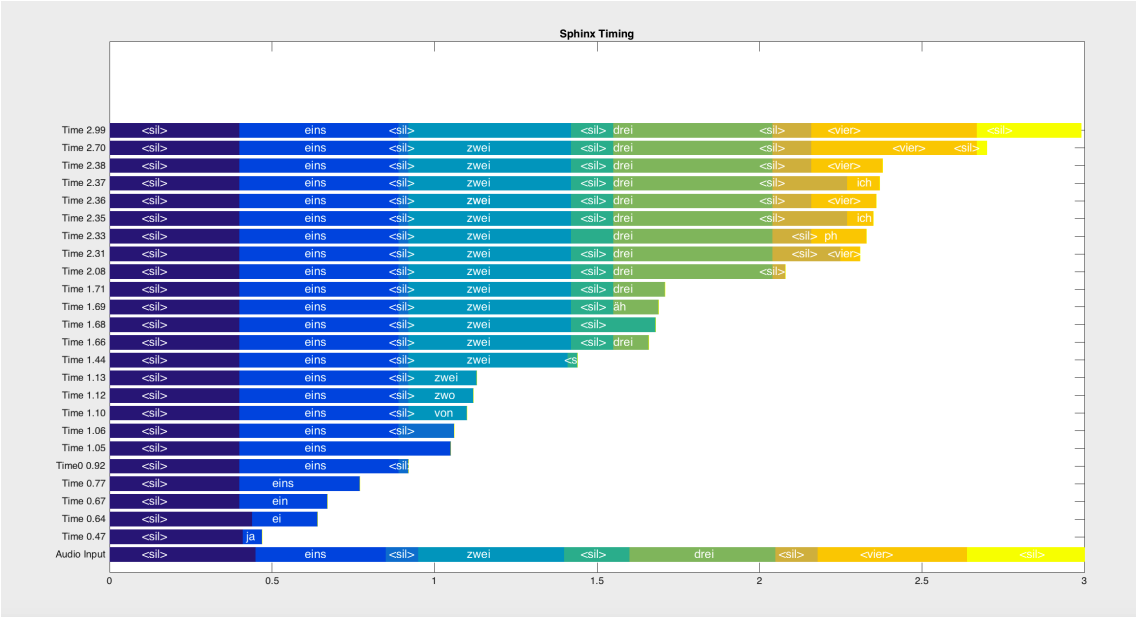

Figure 3: Audio Input and Sphinx Recognizer Timing Visualisation

The difference between Google timing and true alignment is not always constant. For example, if we calculate the mean fault between Google timing and true timing with the formula: $\frac{Delay - True\ Timing}{Number\ of\ Word\ Matches}$ the timing is still not accurate with respect to word duration. The results of the mean fault correction are shown in the picture 5 (transcription .mean).

To implement the forced alignment for Google ASR several facts are to be considered. First, we are dealing with the network latency. Second it is not enough to correct the result by a constant factor as the standard deviation of hypotheses

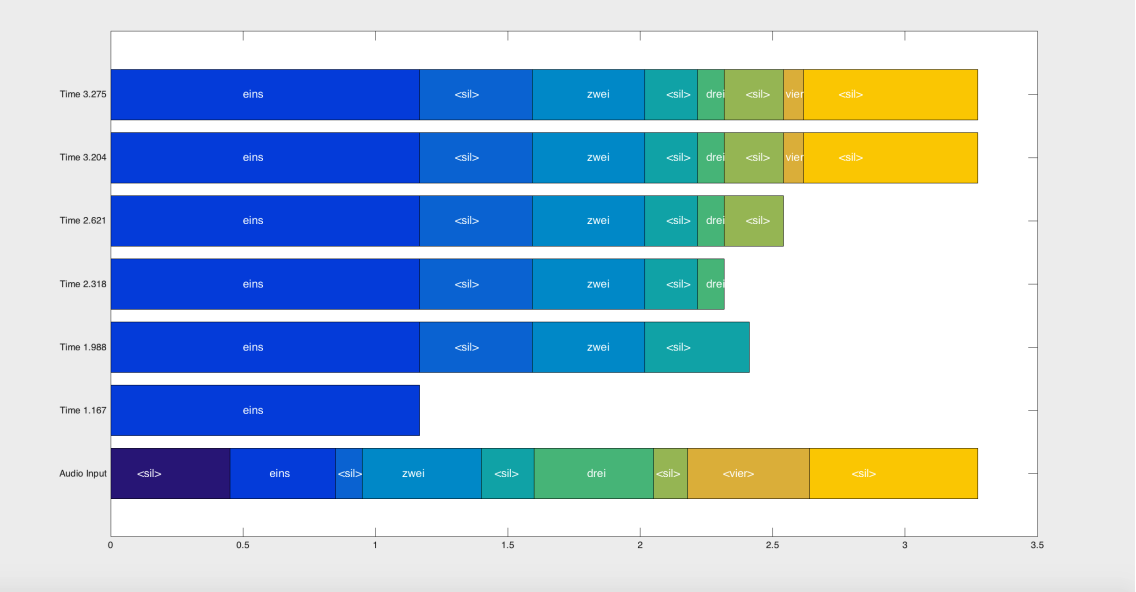

Figure 4: Audio Input and Existing Google Recognizer Timing Visualisation

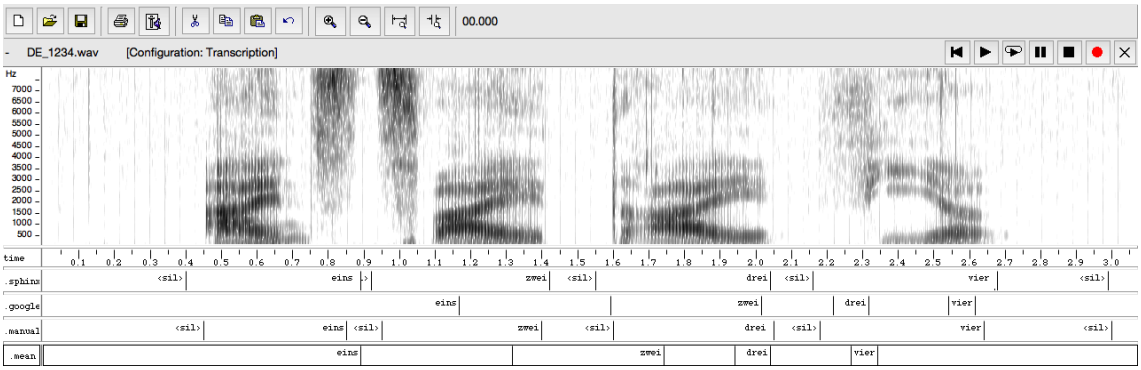

Figure 5: Audio Input Alignment and Mean Fault Correction for Google Results

output, produced by Google, should also be taken into account. Timing, giving information about alignment of Google incremental output to audio input will be further used to manipulate the SearchGraph of the sphinx4.

5 Methods

5.1 Development Environment

Incremental recogniser will be implemented in Java programming language, using an open-source project Incremental Spoken Dialogue processing Toolkit (InproTK), developed by the Universities of Potsdam, Bielefeld and Hamburg. InproTK includes modules for speech recognition, speech understanding, speech processing and dialogue management. Speech recognition module is based on

implemented Java Sphinx-4 recogniser, a package of CMUSphinx speech recognition toolkit.

5.2 Data Structures

Incremental Recogniser architecture includes Sphinx-4 Frontend, Dictionary, Language Model and Acoustic Model, Decoder and SearchGraph and time alignment component (see Figure 1).

5.2.1 Frontend

Frontend computes feature vector representation of speech signal, using acoustic analysis. For this master thesis a standard Sphinx-4 frontend is to be used as a Frontend. It provides methods for manipulating the processors, performing specific transformation functions on input data (Lamere et al. 2003).

5.2.2 Acoustic Models and Hidden Markov Model

Acoustic model finds out the probability for an observing an acoustic vector Y given a sequence of words P(Y | W). For a small vocabulary it can be found by sampling several variants of the words and computing statistical similarity of the inputs with the existing samples. However, when the vocabulary is very large we have to move down to the level of word subunits or phonemes. Acoustic model contains statistical representation of each phoneme in the form a Hidden Markov Models (HMM), which abstracts the physical process of pronouncing of a particular phoneme. In order to get a statistical representation of phonemes, i. e, build an automate with transition probabilities for each phoneme, the speech corpus data is subjected to the training process. (Jurafsky and Martin 2009).

5.3 Language Model

Language model describes the probability of a word in a speech sequence P (W) and attempts to predict the next word in a sentence, depending on the prior context. Language models are based on n-gram sequences of words, where the number of words of the prior context is equal n-1. Language models are built on the basis of an application corpus (Jurafsky and Martin 2009). Depending on the application, language models can be either generalized or domain specific.

5.3.1 Decoder and SearchGraph

On the basis of acoustic model, language model and dictionary the SearchGraph is build, which is used by the decoder together with the frontend information to determine the most probable output sequence. Decoder traverses the network of HMM states and finds the path with the best score, using Viterbi algorithm.

5.3.2 Grapheme to Phomene Converter

Grapheme to phoneme converter converts a sequence of character strings into a sequence of phones, according to the international phonetic alphabet standards. Sphinx4 provides grapheme to phoneme conversion feature as a part of its implementation (Lamere et al. 2003).

5.3.3 Time alignment Component

The central part of the implementation is timing component, which gives detailed timing information Google incremental result. Time alignment component computes alignment of Google incremental results with respect to audio input. Furthermore timing is intended to manipulate the SearchGraph. Manipulation includes reset, allowing back tracking to the saved state and changing the search path in the graph.

5.3.4 Google-Sphinx Combination

The implementation of the incremental speech recogniser is to be realised as a combination of Google-ASR and Sphinx-4 recogniser. This combination uses a standard Sphinx-4 frontend as input for the decoder, whereas the states of the Search-Graph are manipulated, depending on the incremental output, coming from Google. Acoustic model, Dictionary and Lingustic Model are chosen from the standard ones, offered by Sphinx-4. Google incremental results are preprocessed in the grapheme to phoneme converter and undergo forced time-alignment. The initialisation and implementation of the SearchGraph is realised by Sphinx-4. The choice and alternation of the path in the SearchGraph depends on the Google incremental hypotheses. When the new Google hypotheses differs from the present hypotheses in the SearchGraph, the SearchGraph is manipulated and the hypotheses path is corrected. Furthermore an alternation between single Google-ASR, single Sphinx-4 and a combination of Google-ASR and Sphinx-4 is foreseen.

5.4 Algorithms

5.4.1 Viterbi search algorithm

Search in the Sphinx-4 decoder module is executed, using Viterbi algorithm (Lamere et al. 2003). Viterbi is a dynamic programming search algorithm. It traverses the network of HMM states and finds the path with the best score. Abstract representation of the Viterbi algorithm is shown in the in the figure 6. One axis represent states in the HMM network, another time. Arrows means transitions through the network. Each point in this 2-D space represents the best path probability for the corresponding state at that time. Every state has a best predecessor and starting from the final state and using backtracking it is possible to find the best path sequence for the whole search.

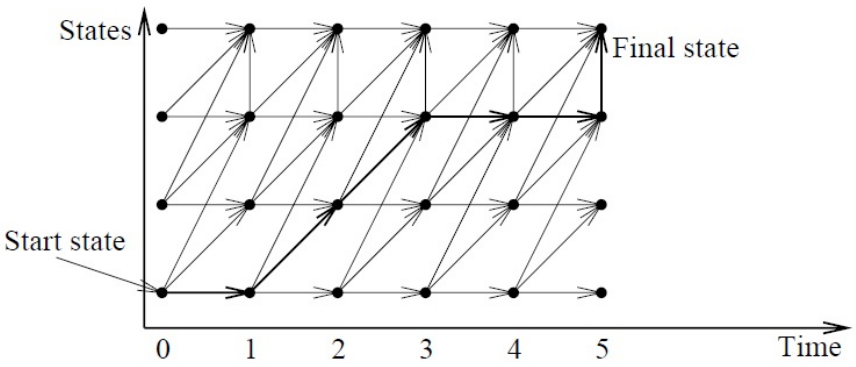

Figure 6: Viterbi Search algorithm (Jurafsky and Martin 2009)

5.4.2 Similarity Metric and Levenstein Distance

Levestein distance is a similarity metric of two stings. Levenstein distance algorithm calculates a Levenstein distance matrix, containing substitutions and matches of characters, occurring in two strings. Each substitution increases Levenstein distance by 1 (Levenshtein 1966) . Based on the Levenstein distance matrix it should be possible to calculate the Google latency. Timing difference will be calculated only for matches of audio transcription and Google incremental results.

5.5 Test Data

For testing purposes a corpus of a total of 27 hours of audio files is available. Each record contains audio file, true alignment and Google incremental results in JSON

Format, including hypotheses timing. For development purposes a smaller part of this data, consisting of 30 audio files, is to be used. For validation purposes corpus will be gradually extended with additional data.

6 Evaluation Approach and Expectations

In the chapter the quality and performance of the implemented Incremental Speech recogniser is to be analysed. As a metric for performance analysis latency time is to be used. Latency is understood here as the time difference between audio incremental input and incremental output of the recogniser. Latency of Google ASR, Sphinx-4 based recogniser alone and a Coogle-Sphinx combination are supposed to be computed and visualises graphically, showing the delays for single incremental results. For this purpose evaluation library InTELiDa (Incremental Timing Evaluation of Linguistic Data), providing a graphical user interface for incremental data analysis, will be used (Baumann 2013). The latencies of the experiments performance results, using a proposed combination of Gooogle ASR an Sphinx-4, are expected to show better results for latency than and Google ASR and Sphinx-4 alone.

7 Preliminary Table of Contents

1. Introduction
    1.1. Motivation
    1.2. Problem Statement
    1.3. Objectives
    1.4. Structure of the work
2. Related Work
    2.1. Literature Review on Speech Recognition
    2.2. Comparison and Summary of existing Approaches to Incremental Speech Recognition
3. Methods
    3.1. Development environment
    3.2. Incremental Speech Recogniser Architecture

3.3. Used Algorithms

4. Implementation of the Incremental Speech Recogniser

   4.1. Time alignment component of the Incremental Speech Recogniser

   4.2. Combination of Google Incremental ASR with Time-aligned Phonetic Post-processing, based on Sphinx-4

5. Experiments and Results

   5.1. Testing Approach

   5.2. Test Results

   5.3. Tests Evaluation

6. Conclusion

   6.1. Conclusion

   6.2. Discussion

   6.3. Future Work

8 Timeline

Preliminary timeline for the master thesis is shown in the 7. Working process is divided into two closely connected phases: experimental and writing.

9 Conclusion

In this exposé we have introduced the objectives for the upcoming master thesis, gave a brief literature review, formulated the research question, gave an overview of the methods and the test approach and present a preliminary table of contents and timeline for the work. In the focus of the research stays the possibility of latency bypassing in an incremental recogniser, using a combination of frontend sources and time alignment of phonemic representation.

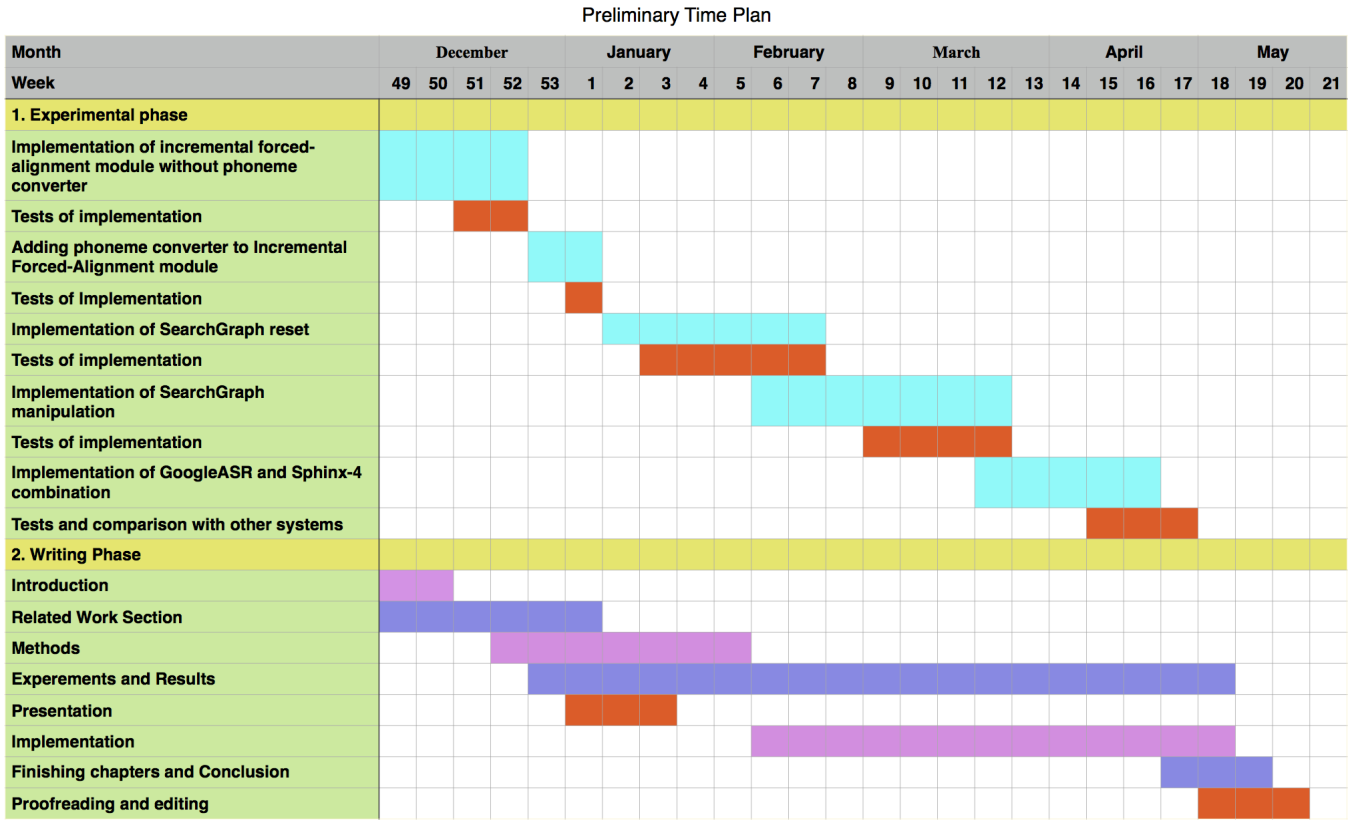

Figure 7: Preliminary timeline for the master thesis

10 Conclusion

In this exposé we have introduced the objectives for the upcoming master thesis, gave a brief literature review, formulated the research question, gave an overview of the methods and the test approach and present a preliminary table of contents and timeline for the work. In the focus of the research stays the possibility of latency bypassing in an incremental recogniser, using a combination of frontend sources and time alignment of phonemic representation.

References

Baumann, Timo (2013). "Incremental Spoken Dialogue Processing: Architecture and Lower-level Components". PhD thesis. Universität Bielefeld, Germany. URL: http://www.timobaumann.de/diss/tb-diss-201305231753-publishedversion-compressed.pdf.

Baumann, Timo, Michaela Atterer, and David Schlangen (2009). "Assessing and Improving the Performance of Speech Recognition for Incremental Systems". In: *Proceedings of NAACL-HLT 2009*. Boulder, USA.

Jurafsky, Daniel and James H. Martin (2009). *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. 2nd. Pearson International.

Lamere, Paul et al. (2003). "Design of the cmu sphinx-4 decoder". In: *IN 8TH EUROPEAN CONF. ON SPEECH COMMUNICATION AND TECHNOLOGY (EUROSPEECH)*.

Levenshtein, V. (1966). "Binary Codes Capable of Correcting Deletions, Insertions, and Reversals". In: *Soviet Physics-Doklady* 10.8, pp. 707–710.

McGraw, Ian and Alexander Gruenstein (2012). "Estimating Word-Stability During Incremental Speech Recognition." In: *INTERSPEECH*. ISCA, pp. 1019–1022.

Schalkwyk, Johan et al. (2010). ""Your Word is my Command": Google Search by Voice: A Case Study". In: *Advances in Speech Recognition: Mobile Environments, Call Centers and Clinics*. Chap. 4, pp. 61–90.

Skantze, Gabriel and David Schlangen (2009). "Incremental Dialogue Processing in a Micro-domain". In: *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*. EACL '09. Athens, Greece: Association for Computational Linguistics, pp. 745–753.

Twiefel, Johannes et al. (2014). "Improving Domain-independent Cloud-Based Speech Recognition with Domain-Dependent Phonetic Post-Processing". In: *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, July 27 -31, 2014, Québec City, Québec, Canada.* Pp. 1529–1536.