Incremental Processing

Katinka Böhm and Konstantin Möllers

University of Hamburg, Department of Informatics
Vogt-Kölln-Straße 30, D-22527 Hamburg
{5boehm,1kmoelle}@informatik.uni-hamburg.de

Introduction

Incremental processing (IP) [1] describes the successive processing of small amounts of input to produce output directly. Thereby incremental units (IU) are small chunks of information (e.g. words in an utterance) that are connected in a IU network that represents the current output at a given time.

Multiple Timelines: Hypotheses are expanded and revised over multiple steps, which results in a two-dimensional representation where in each time step t several assumptions represent the history during incremental recognition, see Figure 1.

Reversibility: IP facilitates revising of previously made hypotheses to dynamically adapt and change the output when new knowledge is gained.

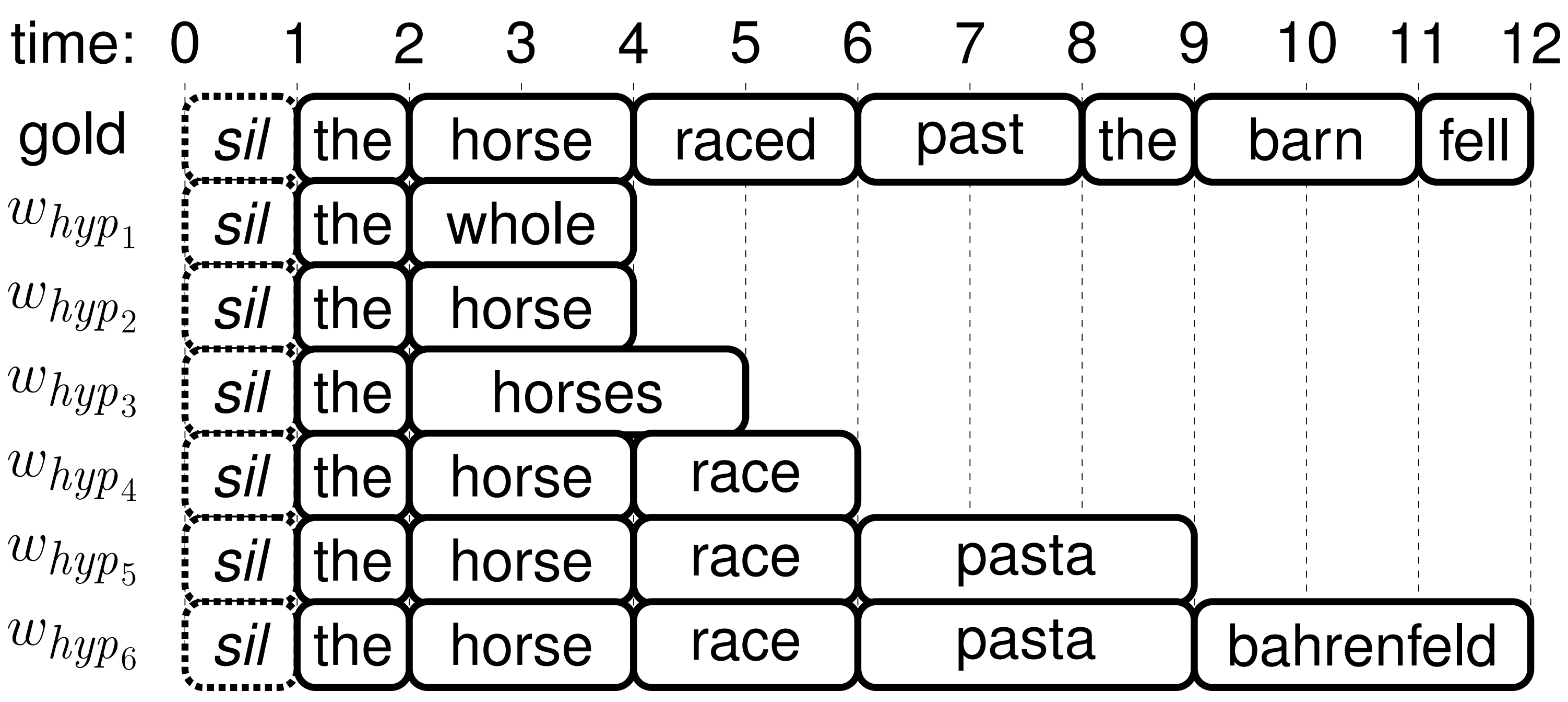


Figure 1: *hypothesis time line*

Evaluation Metrics

Similarity Metrics

What should be known at time t (gold standard) and what is actually known? Word Error Rate **(WER)** is used for incremental speech recognition.

Timing Metrics

When do events happen in incremental output relative to the time specified in the gold standard? Differences w.r.t. the gold standard are measured for the First Occurrence **(FO)** of an output increment and the Final Decision **(FD)** on an output increment.

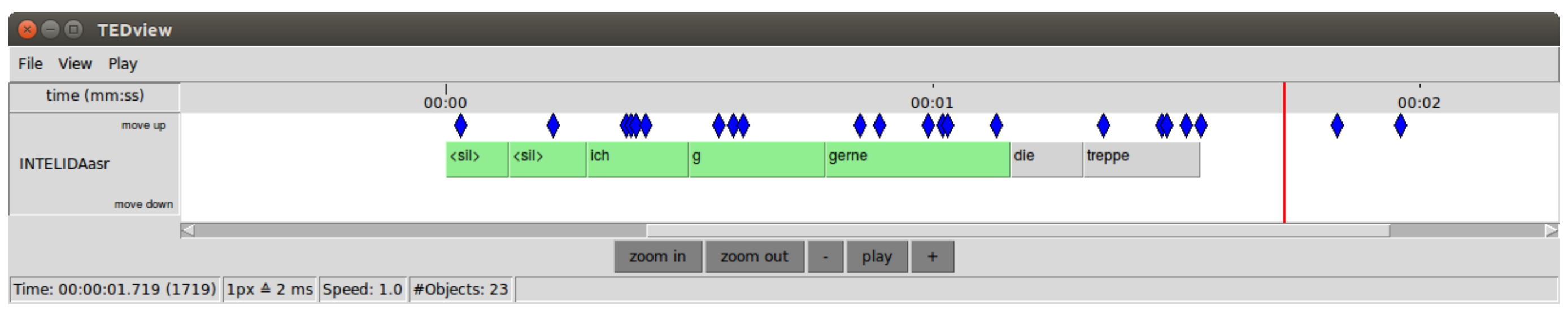
Diachronic Metrics

Which edits happens over the course of processing? Recognize edits to the IU network and measure the Edit Overhead **(EO)** as proportion of unnecessary or harmful edits.
Three different types of edits are considered: **Addition** (extension of a previous output), **Revocation** (retraction of an IU) and **Substitution** (substitute one IU with a new one). A final result of n increments needs at least n additions.

Experimental Setup

We analyzed incremental ASR for examples of "complex" speech input where meaning changes over time (e.g. garden-path sentences). Our goal was to compare incremental ASR of **CMU Sphinx 4** with the **Google Speech Recognition API** using the *inproTK* toolkit.

For Sphinx we applied the German language model and dictonary. We used *wavesurfer* to record our own *.wav* examples and manually created respective gold standards. For evaluation we used the *interactivetool* of the *InTELiDa* evaluation library as well as *TEDview* for visualization, see Figure 2.



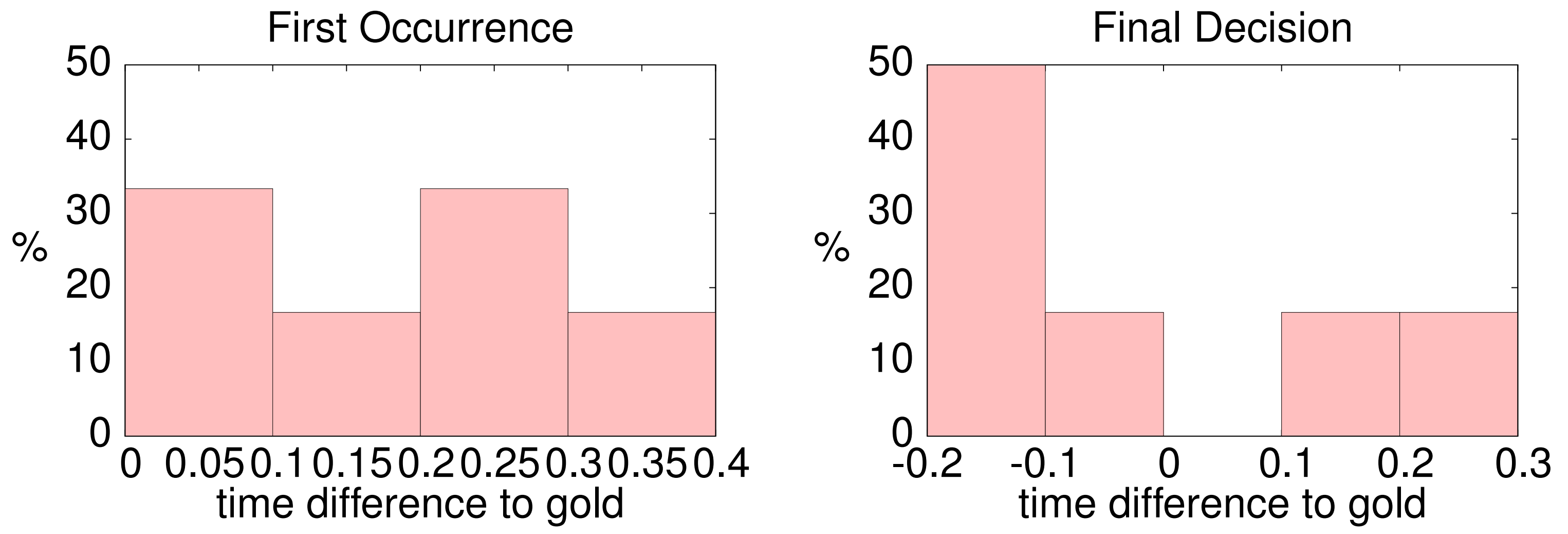
Time: 00:00:01.719 (1719) | 1px ≙ 2 ms | Speed: 1.0 | #Objects: 23

Figure 2: *results shown in TEDview*

Sphinx Results

Said: "ich gehe gerne die treppe hoch" (6 words)
Recognized: "ich g gerne die treppe hoch" (6 words)

Analysis:
WER: 16%, Edits: added 7, substituted 14, revoked 1, EO: 72%

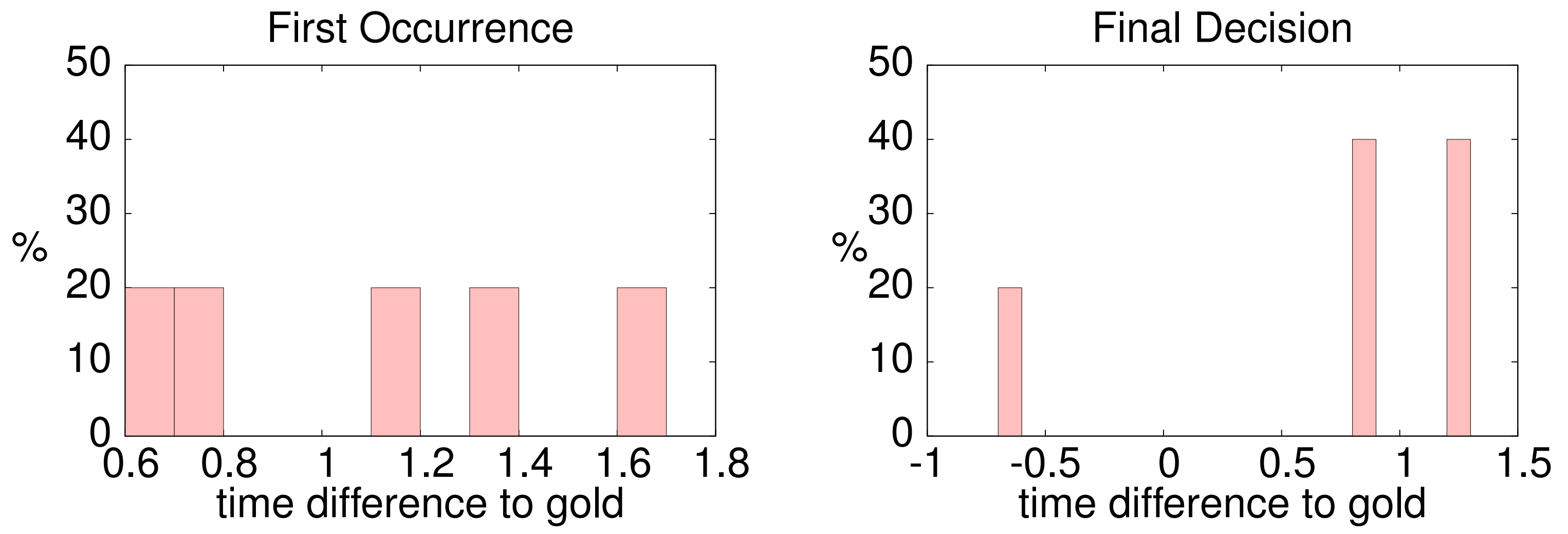


(a) $\mu = 0.18,\ \sigma = 0.12$ (b) $\mu = -0.02,\ \sigma = 0.19$

Google Results

Said: "the horse raced past the barn fell" (7 words)
Recognized: "the horse race pasta bahrenfeld" (5 words)

Analysis:
WER: 71%, Edits: added 5, substituted 3, revoked 2, EO: 50%



(a) $\mu = 1.11,\ \sigma = 0.42$ (b) $\mu = 0.71,\ \sigma = 0.78$

Comparison: Sphinx vs. Google

Google is much slower than Sphinx because of far HTTP requests in between and transformation of data into JSON format. However, most of the times, it produces more accurate output for English inputs than Sphinx does. This is because Google uses Machine Learning to improve their results and also takes geographical information into account.

[1] Timo Baumann, Okko Buß, David Schlangen: *Evaluation and Optimisation of Incremental Processors. Dialogue and Discourse 2(1): 113-141 (2011)*