Assessing and Improving the Performance of Speech Recognition for Incremental Systems

Timo Baumann, Michaela Atterer, David Schlangen
Institut für Linguistik
Universität Potsdam
Potsdam, Germany
{timo,atterer,das}@ling.uni-potsdam.de

Abstract

In incremental spoken dialogue systems, partial hypotheses about what was said are required even while the utterance is still ongoing. We define measures for evaluating the quality of incremental ASR components with respect to the *relative correctness* of the partial hypotheses compared to hypotheses that can optimize over the complete input, the *timing* of hypothesis formation relative to the portion of the input they are about, and hypothesis *stability*, defined as the number of times they are revised. We show that simple incremental post-processing can improve stability dramatically, at the cost of timeliness (from 90 % of edits of hypotheses being spurious down to 10 % at a lag of 320 ms). The measures are not independent, and we show how system designers can find a desired operating point for their ASR. To our knowledge, we are the first to suggest and examine a variety of measures for assessing incremental ASR and improve performance on this basis.

1 Introduction

Incrementality, that is, the property of beginning to process input before it is complete, is often seen as a desirable property of dialogue systems (e.g., Allen et al. (2001)), as it allows the system to (a) *fold* processing time (of modules such as parsers, or dialogue managers) into the time taken by the utterance, and (b) *react* to partial results, for example by generating back-channel utterances or speculatively initiating potentially relevant database queries.

Input to a spoken dialogue system normally passes an automatic speech recognizer (ASR) as a first processing module, thus the module's incrementality determines the level of incrementality that can be reached by the system as a whole. Using an ASR system incrementally poses interesting challenges, however. Typically, ASRs use dynamic programming and the maximum likelihood hypothesis to find the word sequence with the lowest expected likelihood of the sequence containing errors (sentence error). Due to the dynamic programming approach, what is considered the best hypothesis about a given stretch of the input signal can change during the recognition process, as more right context which can be used as evidence becomes available.

In this paper, we argue that normally used metrics for ASR evaluation such as word error rate must be complemented with metrics specifically designed for measuring incremental performance, and offer some such metrics. We show that there are various subproperties that are not independent of each other, and that trade-offs are involved if either of those is to be optimized. Finally, we propose ways to improve incremental performance (as measured by our metrics) through the use of smoothing techniques.

To our knowledge, incremental evaluation metrics of ASR for incremental systems have not yet been covered in the literature. Most closely related, Wachsmuth et al. (1998) show results for an ASR which fixes its results after a given time Δ and report the corresponding word error rate (WER). This unfortunately confounds the incremental and non-incremental properties of their ASR's performance.

The remainder of this paper is structured as follows: In section 2, we give an overview of incremenality with respect to ASR, and develop our evalua-

tion metrics. Section 3 describes the setup and data that we used in our experiments, and reports and discusses some basic measures for different variants of the setup. In section 4 we propose and discuss two orthogonal methods that improve incremental performance: using right context and using message smoothing, which show different properties with regard to our measures. Finally, in section 5 we sum up and point to future directions.

2 Incrementality and Evaluation Measures for Incremental ASR

In a modular system, an *incremental module* is one that generates (partial) responses while input is still ongoing and makes these available to other modules (Kilger and Finkler, 1995). ASR modules that use token passing (Young et al., 1989) can easily be adapted to output a new, *live* hypothesis after processing of every input frame (often that is every 10 ms). In an incremental system we are able to get partial results from these hypotheses as soon as they become available – or rather as soon as they can be trusted. As mentioned above, hypotheses are only tentative, and may be revised when more right context becomes available. Modules consuming the output of an incremental ASR hence must be able to deal with such revisions. There is a first trade-off here: Depending on how costly revision is for later modules (which after all may need to revise any hypotheses which they themselves based on the now-revised input), it may be better to reduce the incrementality a bit – in the sense that partial information is produced less often, and hence new words for example are recognised later – if that buys stability (fewer revisions). Also, ignoring some incremental results that are likely to be wrong may increase system performance. Defining these notions more precisely is the aim of this section.

2.1 Relative Correctness

We define a hypothesis at time t (hyp_t) as consisting of a sequence w_{hyp_t} of words predicted by the ASR at time t.[1] As an example figure 1 shows

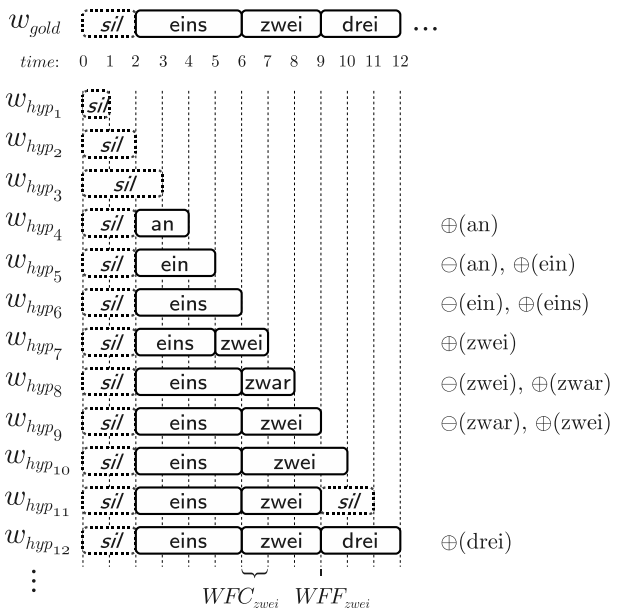

Figure 1: *Live* ASR hypotheses during incremental recognition. Edit messages (see section 2.2) are shown on the right when words are added (⊕) or revoked (⊖). For the word "zwei" WFC and WFF (see section 2.3) are shown at the bottom.

a sequence of incrementally produced hypotheses. (Note that this is an artificial example, showing only a few illustratory and interesting hypotheses. In a real recognition system, the hypothesis frequency is of course much higher, with much repetition of similar hypotheses at consecutive frames.)

The question now is how we can evaluate the quality of a hypothesis at the time t it is produced. It is reasonable to only expect this hypothesis to say something (correct or not) about the input up to time t – unless we want the ASR to *predict*, in which case we want it to make assumptions about times beyond t (see section 4.1). There are two candidates for the yardstick against which the partial hypotheses could be compared: First, one could take the *actually* spoken words, computing measures such as word error rate. The other option, which is the one taken here, is to take as the gold standard the final hypothesis produced by the ASR when it has all evidence avail-

[1] In this paper, we only deal with one-best ASR. We believe that there are no principled differences when generalising to n-best hypotheses, but will explore this in detail in future work.

We also abstract away from changes in the hypothesised start and end times of the words in the sequence. It often happens that the ASR's assumptions about the position of the word boundaries change, even if the word sequence stays constant. If, as we assume here, later modules do not use this timing information, we can consider two hypotheses that only differ in boundary placement as identical.

able (i.e., when the utterance is complete). This is more meaningful for our purpose, as it relates the informativity of the partial hypothesis to what can be expected if the ASR can do all its internal optimisations, and not to the factually correct sequence that the ASR might not be able to recognise even with all information present. This latter problem is already captured in the conventional non-incremental performance measures.

In our metrics in this paper, we hence take as gold standard (w_{gold}) the final, non-incremental hypothesis of the ASR (which, to reiterate this point, might be factually incorrect, that is, might contain word errors). We define a module's incremental response at time t (w_{hyp_t}) as *relatively correct* (*r-correct*), iff it is equal to the non-incremental hypothesis up to time t: $w_{hyp_t t} = w_{gold_t}$. Hence, in figure 1 above, hypotheses 1, 2, 6, 7, 9 and 12 are r-correct.[2] We call the normalised rate of r-correct responses of a module its (average) *r-correctness*.

As defined above, the criterion for r-correctness is still pretty strict, as it demands of the ASR that words on the right edge are recognised even from the first frame on. For example, $w_{hyp_{10}}$ in figure 1 is not r-correct, because $w_{gold_{10}}$ (that part of w_{gold} that ends where $w_{hyp_{10}}$ ends) already spans parts of the word "drei" which has not yet been picked up by the incremental recognition. A relaxed notion of correctness hence is *prefix-correctness*, which requires only that w_{hyp_t} be a prefix of w_{gold_t}. (Hypotheses 3 and 10 in figure 1 are p-correct, as are all r-correct hypotheses.) It should be noted though that p-correctness is too forgiving to be used directly as an optimization target: in the example in figure 1, a module that only ever produces empty hypotheses would trivally achieve perfect p-correctness (as this is always a prefix of w_{gold}).

2.2 Edit Overhead

The measures defined so far capture only static aspects of the incremental performance of a module and do not say anything about the dynamics of the recognition process. To capture this, we look at the changes between subsequent partial hypotheses. There are three ways in which an hypothesis hyp_{t+1} can be different from hyp_t: there can be an *extension* of the word sequence, a *revokation*, or a *revision* of the last words in the sequence.[3] These differences can be expressed as *edit messages*, where extending a sequence by one word would require an *add* message ($\oplus$), deleting the last word in the sequence a *revoke* message ($\ominus$), and exchange of the last word would require two messages, one to revoke the old and one to add the new word.[4]

Now, an incrementally perfect ASR would only generate extensions, adding new words at the right edge; thus, there would be exactly as many edit messages as there are words in w_{gold}. In reality, there are typically many more changes, and hence many spurious edits (see below for characteristic rates in our data). We call the rate of spurious edits the *edit overhead* (EO). For figure 1 above, this is $\frac{8}{11}$: There are 11 edits (as shown in the figure), while we'd expect only 3 (one $\oplus$ for each word in the final result). Hence, 8 edits are spurious.

This measure corresponds directly to the amount of unnecessary activity a consumer of the ASR's output performs when it reacts swiftly to words that may be revoked later on. If the consumer is able to robustly cope with parallel hypotheses (for example by building a lattice-like structure), a high EO may not be problematic, but if revisions are costly for later modules (or even impossible because action has already been taken), we would like EO to be as low as possible. This can be achieved by not sending edit messages unconditionally as soon as words change in the ASR's current hypothesis, using strategies as outlined in section 4. Obviously, deferring or suppressing messages results in delays, a topic to which we turn in the following section, where we define measures for the response time of ASR.

2.3 Timing Measures

So far, our measures capture characteristics about the complete recognition process. We now turn to the timing of the recognition of individual words. For this, we again take the output of the ASR when all signal is present (i.e., w_{gold}) as the basis. There

[2] The timing in hypothesis 7 is not correct – but this does not matter to our notion of correctness (see footnote 1).

[3] As fourth and most frequent alternative, consecutive hypotheses do not change at all.

[4] Revision could also be seen as a third atomic operation, as in standard ASR evaluation (then called "substitution"). To keep things simple, we only regard two atomic operations.

are two things we may be interested in. First, we may want to know when is the first time that a certain word appears in the correct position in the sequence (or equivalently, when its first correct *add* edit message is sent), expressed in relation to its boundaries in w_{gold}. We measure this event, the first time that the ASR was right about a word, relative to its gold beginning. We call the measure *word first correct response* (WFC). As a concrete example take hyp_7 in figure 1. At this point, the word "zwei" is first hypothesised. Compared to the beginning of the word in w_{gold}, this point (t_7) has a delay of 1 frame (the frames are illustrated by the dashed lines).

As explained above, it may very well be the case that for a brief while another hypothesis, not r-correct w.r.t. w_{gold}, may be favoured (cf. the word "zwar" in the example in the figure). Another measure we hence might also be interested in is when our word hypothesis starts remaining stable or, in other words, becomes final. We measure this event relative to the end of the word in the gold standard. We call it *word first final response* (WFF). In our example, again for "zwei", this is t_9, which has a distance of 0 to the right boundary of the word in w_{gold}.

In principle, we could use both anchor points (the left vs. the right edge of a word) for either measure or use a word-relative scale, but for simplicity's sake we restrict ourselves to one anchor point each.

Under normal conditions, we expect WFC to be positive. The better the incremental ASR, the closer to 0 it will be. WFC is not a measure we can easily optimize. We would either have to enumerate the whole language model or use external non-ASR knowledge to predict continuations of the word sequence before the word in question has started. This would increase EO. In principle, we are rather interested in accepting an increase in WFC, when we delay messages in order to decrease EO.

WFF however, can reach values below 0. It converges towards the negative average of word length as an incremental ASR improves. For non-incremental ASR it would be positive: the average distance beween the sentence end and word end. WFF is a measure we can strive to reduce by sending fewer (especially fewer wrong) messages.

Another property we might be interested in optimizing is the time it takes from the first correct hypothesis to stabilize to a final hypothesis. We compute this *correction time* as the difference in time between WFF and WFC.[5] A correction time of 0 indicates that there was no correction, i.e. the ASR was immediately correct about a word, something which we would like to happen as often as possible.

Note that these are measures for each word in each processed utterance, and we will use distributional parameters of these timing measures (means and standard deviations) as metrics for the performance of the incremental setups described later.

2.4 Summary of Measures

In this section, we first described measures that evaluate the overall *correctness* of incrementally produced ASR hypotheses, not taking into account their sequential nature. We then turned to the dynamics of how the current hypothesis evolves in a way which we consider important for a consumer of incremental ASR, namely the *overhead* that results from edits to the hypothesis. Finally, we looked at the timing of individual messages with regard to first correct (potentially unstable) occurrence (WFC) and stability (WFF). In the next section, we use the measures defined here to characterize the incremental performance of our ASR, before we discuss ways to improve incremental performance in section 4.

3 Setup, Corpora and Base Measurements

We use the large-vocabulary continuous-speech recognition framework Sphinx-4 (Walker et al., 2004) for our experiments, using the built-in LexTree decoder, extended by us to provide incremental results. We built acoustic models for German, based on a small corpus of spontaneous instructions in a puzzle building domain,[6] and the Kiel corpus of read speech (IPDS, 1994). We use a trigram language model that is based on the puzzle domain transcriptions. As test data we use 85 recordings of two speakers (unknown to the acoustic model) that speak sentences similar to those in the puzzle domain.

We do not yet use recognition rescoring to optimize for word error rate, but just the ASR's best hypotheses which optimize for low sentence error. Incremental rescoring mechanisms such as that of

[5] In figure 1, the correction time for "zwei" is $9 - 7 = 2$.

[6] Available from `http://www.voxforge.org/home/downloads/speech/`

SER (non-incremental)	68.2 %
WER (non-incremental)	18.8 %
r-correct (cropped)	30.9 %
p-correct (cropped)	53.1 %
edit overhead	90.5 %
mean word duration	0.378 s
WFC: mean, stddev, median	0.276 s, 0.186 s, 0.230 s
WFF: mean, stddev, median	0.004 s, 0.268 s, –0.06 s
immediately correct	58.6 %

Table 1: Base measurements on our data

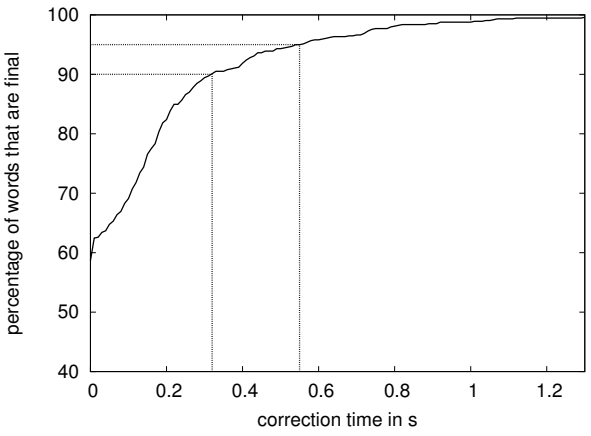

Figure 2: Distribution of correction times (WFF − WFC).

Razik et al. (2008) to optimize ASR performance are orthogonal to the approaches presented in section 4 and could well be incorporated to further improve incremental performance.

The individual recordings in our corpus are fairly short (5.5 seconds on average) and include a bit of silence at the beginning and end. Obviously, recognizing silence is much easier than recognizing words. To make our results more meaningful for continuous speech, we crop away all ASR hypotheses from before and after the active recognition process.[7] While this reduces our performance in terms of correctness (we crop away areas with nearly 100 % correctness), it has no impact on the edit overhead, as the number of changes in w_{curr} remains unchanged, and also no impact on the timing measures as all word boundaries remain the same.

3.1 Base Measurements

Table 1 characterises our ASR module (on our data) in terms of the metrics defined in section 2. Additionally we state *sentence error rate*, as the rate of sentences that contain at least one error, and word error rate computed in the usual way, as well as the mean duration of words in our corpus (as non-incrementally measured for our ASR).

We see that correctness is quite low. This is mostly due to the jitter that the evolving current hypothesis shows in its last few frames, jumping back and forth between highly-ranked alternatives. Also, our ASR only predicts words once there is acoustic evidence for several phonemes and every phoneme (being modelled by 3 HMM states) must have a duration of at least 3 frames. Thus, some errors relative to the final hypothesis occur because the ASR only hypothesizes about words once they already have a certain duration (and hence preceding hypotheses are not r-correct). The difference between r-correctness and p-correctness (20 % in our case) may be largely attributed to this fact.

The edit overhead of 90.5 % means that for every neccessary add message, there are nine superfluous (add or revoke) messages. Thus, a consumer of the ASR output would have to recompute its results ten times on average. In an incremental system, this consumer might itself output messages and further revise decisions as information from other modules becomes available, leading to a tremendous amount of changes in the system state. As ASR is the first module in an incremental spoken dialogue system, reducing the edit overhead is essential for overall system performance.

On average, the correct hypothesis about a word becomes available 276 ms after the word has started (WFC). With a mean word duration of 378 ms this means that information becomes available after roughly $\frac{3}{4}$ of the word have been spoken. Notice though that the median is somewhat lower than the mean, implying that this time is lower for most words and much higher for some words. In fact, the maximum for WFC in our data is 1.38 s.

On average, a word becomes final (i.e. is not changed anymore) when it has ended (mean(WFF) = 0.004). Again, the median is lower, indicating the unnormal distribution of WFF (more often lower, sometimes much higher).

Of all words, 58.6 % were immediately correctly

[7] In figure 1, hypotheses 1, 2 and 3 would be cropped away.

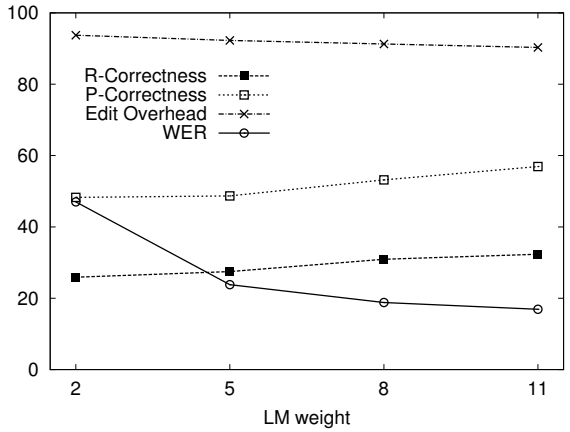

Figure 3: Correctness, Edit Overhead and Word Error Rate (WER) with varied language model weight and unaltered audio.

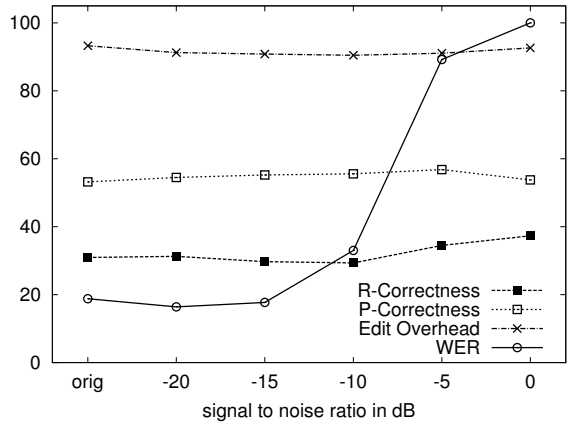

Figure 4: Correctness, Edit Overhead and Word Error Rate (WER) with additive noise (LM weight set to 8).

hypothesized by the ASR. Figure 2 plots the percentage of words with correction times equal to or lower than the time on the x-axis. While this starts at the initial 58.6 % of words that were immediately correct, it rises above 90 % for a correction time of 320 ms and above 95 % for 550 ms. Inversely this means that we can be certain to 90 % (or 95 %) that a current correct hypothesis about a word will not change anymore once it has not been revoked for 320 ms (or 550 ms respectively).

Knowing (or assuming with some certainty) that a hypothesis is final allows us, to *commit* ourselves to this hypothesis. This allows for reduced computational overhead (as alternative hypotheses can be abandoned) and is crucial if action is to be taken that cannot be revoked later on (as for example, initiating a response from the dialogue system). Figure 2 allows us to choose an *operating point* for commitment with respect to hypothesis age and certainty.

3.2 Variations of the Setup

In setting up our system we did not yet strive for best (non-incremental) performance; this would have required much more training material and parameter tweaking. We were more interested here in exploring general questions related to incremental ASR, and in developing approaches to improve incremental performance (see section 4), which we see as a problem that is independent from that of improving performance measures like (overall) accuracy.

To test how independent our measures are on details of the specific setting, such as quality of the audio material and of the language model, we varied these factors systematically, by adding white noise to the audio and changing the language model weight relative to the acoustic model. We varied the noise to produce signal to noise ratios ranging from hardly audible (−20 dB), through annoying noise (−10 dB) to barely understandable audio (0 dB).

Figure 3 gives an overview of the ASR-performance with different LM weights and figure 4 with degraded audio signals. Overall, we see that r-correctness and EO change little with different LM and AM performance and correspondigly degraded WER. A tendency can be seen that larger LM weights result in higher correctness and lower EO. A larger LM weight leads to less influence of acoustic events which dynamically change hypotheses, while the static knowledge from the LM becomes more important. Surprisingly, WER improved with the addition of slight noise, which we assume is due to differences in recording conditions between our test data and the training data of the acoustic model.

In the following experiments as well as in the data in table 1 above, we use a language model weight of 8 and unaltered audio.

4 Improving Incremental Performance

In the previous section we have shown how a standard ASR that incrementally outputs partial hypotheses after each frame processed performs with regard to our measures and showed that they remain

stable in different acoustic conditions and with differing LM weights. We now discuss ways of incrementally post-processing ASR hypotheses in order to improve selected measures.

We particularly look for ways to improve EO; that is, we want to reduce the amount of wrong hypotheses and resulting spurious edits that deteriorate later modules' performance, while still being as quick as possible with passing on relevant hypotheses. We are less concerned with correctness measures, as they do not capture well the dynamic evolution, which is important for further processing of the incremental hypothesis. We also discuss tradeoffs that are involved in the optimization decisions.

4.1 Right Context

Allowing the use of some *right context* is a common strategy to cope with incremental data. For example, our ASR already uses this strategy (with very short right contexts) internally at word boundaries to restrict the language model hypotheses to an acoustically plausible subset (Ortmanns and Ney, 2000). In the experiment described here, we allow the ASR a larger right context of size Δ by taking into account at time t the output of the ASR up to time $t - \Delta$ only. That is, what the ASR hypothesizes about the interval $]t - \Delta, t]$ is considered to be too immature and is discarded, and the hypotheses about the input up to $t - \Delta$ have the benefit of a lookahead up to t. This reduces jitter, which is found mostly to the very right of the incremental hypotheses. Thus, we expect to reduce the edit overhead in proportion with Δ. On the other hand, allowing the use of a right context leads to the current hypothesis *lagging behind* the gold standard. Correspondingly, WFC increases by Δ. Obviously, using only information up to $t - \Delta$ has averse effects on correctness as well, as this measure evaluates the word sequences up to w_{gold_t} which may already contain more words (those recognised in $]t - \Delta, t]$). Thus, to be more fair and to account for the lag when measuring the module's correctness, we additionally define *fair* r-correctness which restricts the evaluation up to time $t - \Delta$: $w_{hyp_t\, t-\Delta} = w_{gold_{t-\Delta}}$.

Figure 5 details the results for our data with right context between 1.5 s and -0.2 s. (The x-axis plots Δ as negative values, with 0 being "now". Results for a right context (Δ) of 1.2 can thus be found 1.2 to the left of 0, at -1.2.) We see that at least in the fair measure, fixed lag performs quite well at improving both the module's correctness and EO. This is due to the fact that ASR hypotheses become more and more stable when given more right context. Still, even for fairly long lags, many late edits still occur.

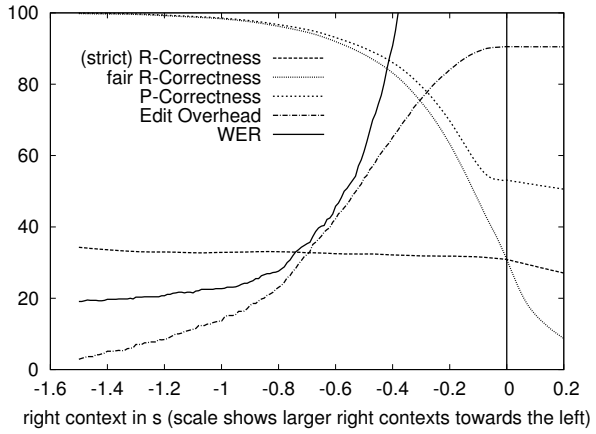

Figure 5: Correctness (see text), Edit Overhead and fixed-WER for varying right contexts Δ.

To illustrate the effects of a system that does not support edits of hypotheses, but instead commits right away, we plot WER that would be reached by a system that always commits after a right context of Δ. As can be seen in the figure, the WER remains higher than the non-incremental WER (18.8 %) even for fairly large right contexts. Also, the WER plot by Wachsmuth et al. (1998) looks very similar to ours and likewise shows a sweet spot suitable as an operating point with a right context of about 800 ms.

As expected, the analysis of timing measures shows an increase with larger right contexts with their mean values quickly approaching Δ (or $\Delta -$ mean word duration for WFF), which are the lower bounds when using right context. Correspondingly, the percentage of immediately correct hypotheses increases with right context reaching 90 % for $\Delta = 580$ ms and 98 % for $\Delta = 1060$ ms.

Finally, we can extend the concept of right context into negative values, predicting the future, as it were. By choosing a *negative* right context, in which we extrapolate the last hypothesis state by Δ into the future, we can measure the correctness of our hypotheses correctly predicting the close future, which is always the case when the current word is still be-

ing spoken. The graph shows that 15 % of our hypotheses will still be correct 100 ms in the future and 10 % will still be correct for 170 ms. Unfortunately, there is little way to tell apart hypotheses that will survive and those which will soon be revised.

4.2 Message Smoothing

In the previous section we reduced wrong edit messages by avoiding most of the recognition jitter by allowing the ASR a right context of size Δ, which directly hurt timing measures by roughly the same amount. In this section, we look at the sequence of partial hypotheses from the incremental ASR, using the dynamic properties as cues. We accomplish this by looking at the edit messages relative to the currently output word sequence. But instead of sending them to a consumer directly (updating the external word sequence), we require that an edit message be the result of N consecutive hypotheses. To illustrate the process with $N = 2$ we return to figure 1. None of the words "an", "ein" or "zwar" would ever be output, because they are only present for one time-interval each. Edit messages would be sent at the following times: $\oplus(\text{eins})$ at t_7, $\oplus(\text{zwei})$ at t_{10} (only then is "zwei" the result of two consecutive hypotheses) and $\oplus(\text{drei})$ at t_{13}. While no words are revoked in the example, this still occurs when a revocation is consecutively hypothesized for N frames.

We get controversial results for this strategy, as can be seen in figure 6: The edit overhead falls rapidly, reaching 50 % (for each message necessary, there is one superfluous message) with only 110 ms (and correspondingly increasing WFC by the same time) and 10 % with 320 ms. The same thresholds are reached through the use of right context at 530 ms and 1150 ms respectively as shown in figure 5. Likewise, the prefix correctness improvements are better than with using right context, but the r-correctness is poor, even under the "fair" measure. We believe this is due to correct hypotheses being held back too long due to the hypothesis sequence being interspersed with wrong hypotheses (which only last for few consecutive hypotheses) which reset the counter until the add message (for the prevalent and potentially correct word) is sent.[8]

[8]This could be resolved by using some kind of majority smoothing instead of requiring a message to be the result of *all* consecutive hypotheses. We will investigate this in future work.

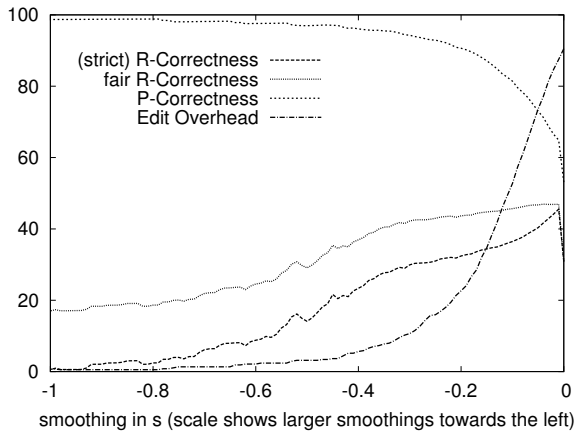

Figure 6: Correctness and Edit Overhead for varying smoothing lengths.

5 Conclusions and Further Directions

We have presented the problem of speech recognition for incremental systems, outlined requirements for incremental speech recognition and showed measures that capture how well an incremental ASR performs with regard to these measures. We discussed the measures and their implications in detail with our baseline system and showed that the incremental measures remain stable regardless of the specific ASR setting used.

Finally, we presented ways for the online post-processing of incremental results, looking for ways to improve some of the measures defined, while hurting the other measures as little as possible. Specifically, we were interested in generating less wrong hypotheses at the cost of possible short delays. While using right context shows improvements with larger delays, using message smoothing seems especially useful for fast processing. We think these two approaches could be combined to good effect. Together with more elaborate confidence handling a system could quickly generate hypotheses and then refine the associated confidences over time. We will explore this in future work.

Acknowledgments

This work was funded by a DFG grant in the Emmy Noether programme. We wish to thank the anonymous reviewers for helpful comments.

References

James Allen, George Ferguson, and Amanda Stent. 2001. An architecture for more realistic conversational systems. In *Proceedings of the Conference on Intelligent User Interfaces*, Santa Fe, USA.

IPDS. 1994. The Kiel Corpus of Read Speech. CD-ROM.

Anne Kilger and Wolfgang Finkler. 1995. Incremental generation for real-time applications. Technical Report RR-95-11, DFKI, Saarbrücken, Germany.

Stefan Ortmanns and Hermann Ney. 2000. Look-ahead techniques for fast beam search. *Computer Speech & Language*, 14:15–32.

Joseph Razik, Odile Mella, Dominique Fohr, and Jean-Paul Haton. 2008. Frame-Synchronous and Local Confidence Measures for on-the-fly Automatic Speech Recognition. In *Proceedings of Interspeech 2008*.

Sven Wachsmuth, Gernot A. Fink, and Gerhard Sagerer. 1998. Integration of parsing and incremental speech recognition. In *Proceedings of the European Signal Processing Conference*, volume 1, pages 371–375, Rhodes, Greece.

Willi Walker, Paul Lamere, Philip Kwok, Bhiksha Raj, Rita Singh, Evandro Gouvea, Peter Wolf, and Joe Woelfel. 2004. Sphinx-4: A flexible open source framework for speech recognition. Technical Report SMLI TR2004-0811, Sun Microsystems Inc.

Steve Young, NH Russell, and JHS Thornton. 1989. Token passing: a simple conceptual model for connected speech recognition systems. *Cambridge University Engineering Department Technical Report CUED/F-INFENG/TR*, 38.