MeetEval: A Toolkit for Computation of Word Error Rates for Meeting Transcription Systems

Thilo von Neumann[1], *Christoph Boeddeker*[1], *Marc Delcroix*[2], *Reinhold Haeb-Umbach*[1]

[1]Paderborn University, Germany
[2]NTT Corporation, Japan

vonneumann@nt.upb.de, boeddeker@nt.upb.de, marc.delcroix@ieee.org, haeb@nt.upb.de

Abstract

MeetEval is an open-source toolkit to evaluate all kinds of meeting transcription systems. It provides a unified interface for the computation of commonly used Word Error Rates (WERs), specifically cpWER, ORC WER and MIMO WER along other WER definitions. We extend the cpWER computation by a temporal constraint to ensure that only words are identified as correct when the temporal alignment is plausible. This leads to a better quality of the matching of the hypothesis string to the reference string that more closely resembles the actual transcription quality, and a system is penalized if it provides poor time annotations. Since word-level timing information is often not available, we present a way to approximate exact word-level timings from segment-level timings (e.g., a sentence) and show that the approximation leads to a similar WER as a matching with exact word-level annotations. At the same time, the time constraint leads to a speedup of the matching algorithm, which outweighs the additional overhead caused by processing the time stamps.

Index Terms: speech recognition, word error rate, meeting transcription

1. Introduction

The Word Error Rate (WER) is a common metric for evaluation of Automatic Speech Recogniton (ASR) systems. While the standard WER, defined as the number of wrongly recognized words divided by the total number of words, is used for single-speaker ASR, it is not directly applicable to multi-speaker meeting transcription. Modifications of the standard WER, such as the Concatenated minimum-Permutation WER (cpWER) and Optimal Reference Combination WER (ORC WER), have been proposed in order to assess meeting transcription performance with a WER metric.

There is, however, no toolkit that allows easy computation of these WER measures. The NIST Scoring Toolkit (SCTK)[1] implements some metrics. Those tools are, however, not built for modern meeting transcription systems, suffer from large memory usage and lack support for some common system output formats. The cpWER is available in the Kaldi speech recognition toolkit [1], but not easily accessible. WER metrics that emerged recently, such as the ORC WER [2] or MIMO WER [3], have no published implementation outside of MeetEval[2].

We present MeetEval with the aim of providing a unified and easy-to-use interface for different WER definitions, including widely used metrics (cpWER) and newly emerging metrics (ORC WER). By this we hope to facilitate assessing the performance of such systems in a reproducible and comparable way. Having different metrics implemented in the same toolkit with the same interface allows easy switching between metrics and thus a deeper analysis of the ASR errors.

We additionally propose the Time-Constrained minimum-Permutation WER (tcpWER), an extension of the cpWER that identifies words as correct or substituted only when the temporal alignment with the reference transcription is plausible. While a similar idea has been mentioned and used in the past [4,5], its impact on the metric has never been assessed for meeting transcription. Since for modern systems, precise word level temporal annotations are often not available (as opposed to what was assumed in earlier works [5]), we propose a way to approximate the word-level annotations from coarse segment-level annotations using a word-length-based heuristic. We analyse and discuss practical issues for reference annotations and show that a collar is necessary to obtain a WER that matches the WER one would get with high precision annotations. Furthermore, the time constraint leads to a speedup of the matching algorithm.

2. Scenario

MeetEval is tailored to the evaluation of systems with WER that transcribe meetings. Fig. 1 gives an overview of the most commonly used recognizers, their output formats and the appropriate metrics. A typical recognizer receives a recording of a meeting and outputs a transcript, where different systems produce different representations and levels of detail. The hypothesis transcript is then evaluated against a reference transcript with a (WER-based) metric.

The recording at the input is typically relatively long (at least a few minutes and up to hours) and contains multiple speakers with complex unpredictable speaking patterns which may lead to speech overlaps and arbitrarily long speech pauses. The hypothesis transcript consists of one or multiple streams $\mathcal{H}_1, \ldots, \mathcal{H}_J$, where each stream contains a part of the transcript. Words can be grouped arbitrarily on these streams, but they are commonly grouped either

- in *Diarization-style*, i.e., *by speaker*, as in [6],
- in *CSS-style*, i.e., such that overlapping speech is placed on different streams inspired by the Continuous Speech Separation (CSS) pipeline [7], or
- in *SOT-style*, where all transcripts are serialized into one output stream as it is done by systems trained with Serialized Output Training (SOT) [8, 9]. The stream is often serialized in temporal order, but the recognizer can in principle choose any order, e.g., output several sentence from one speaker before moving back in time to transcribe another speaker.

Diarization- and CSS-stlye recognizers typically deliver segment-level time annotations, where a segment is a group of

[1]https://github.com/usnistgov/SCTK
[2]https://github.com/fgnt/meeteval

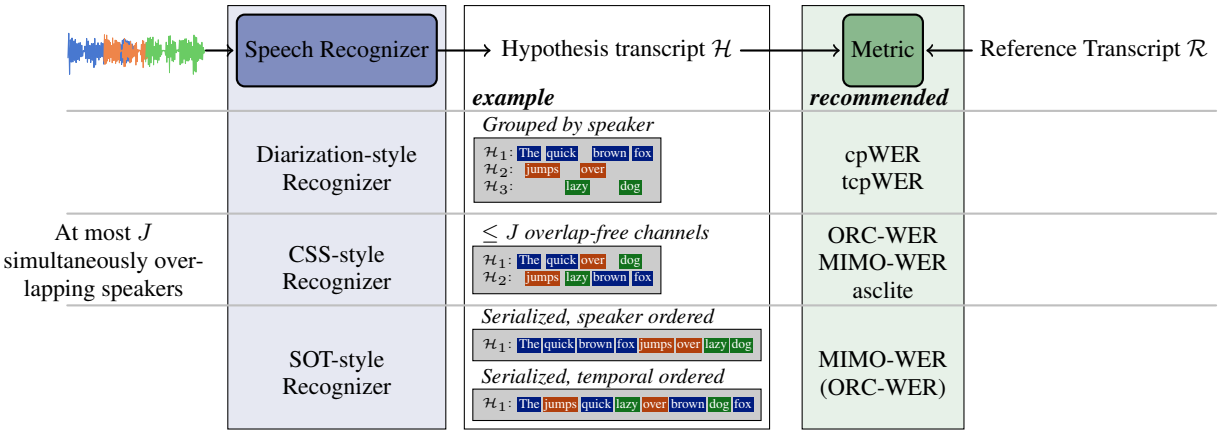

Figure 1: *MeetEval is tailored to the meeting transcription scenario. The input to a speech recognizer contains speech of multiple speakers with potential overlap. The output formats of the different recognizer styles are visualized using a simple example. Letters represent words, words represent segments/utterances and colors represent speakers.*

words that are close to each other (e.g., an utterance). The segment boundaries are found by the recognizer.

We assume that the references contain speaker labels, and for some metrics we also assume a known word order within a segment and segment-level timings of reasonable accuracy from which we approximate word-level timings.

3. Word Error Rates for Meeting Transcription

MeetEval implements a variety of word error rate metrics for the evaluation of meeting transcription systems. This section gives a short overview of the core metrics, where we use the following notation: We use $\mathcal{R}_{i,u}$ to denote the u-th reference utterance of speaker i and $\mathcal{H}_{j,u'}$ to denote the u'-th segment of the j-th system/hypothesis output stream. $U_{\mathcal{R}_i}$ is the number of utterances/segments in $\mathcal{R}_i$. The references/segments are ordered by begin time. Subscripts are neglected where they are not important or applicable. For any word sequence $\mathcal{R}$, $|\mathcal{R}|$ is the number of words and $\mathcal{R}(r)$ is the r-th word in $\mathcal{R}$. We denote sequence concatenation with $\oplus$.

Given the scenario from Section 2, an assignment problem arises because the mapping of utterances to outputs is not unique, e.g., the streams of a Diarization-style recognizer can be permuted along the speaker axis. This assignment problem is solved in different ways by different WER definitions.

3.1. Standard WER

The standard WER is a common metric to evaluate the performance of conventional single-speaker speech recognizers. It can only be applied to pairs of a single reference utterance $\mathcal{R}$ and a single hypothesis segment $\mathcal{H}$. It is defined as the number of word errors in relation to the number of total words in $\mathcal{R}$:

$$\text{WER} = \frac{\text{\#total word errors}}{\text{\#total words}} = \frac{\sum_{\mathcal{R},\mathcal{H}} \text{lev}(\mathcal{R},\mathcal{H})}{\sum_{\mathcal{R}} |\mathcal{R}|}, \quad (1)$$

where $\text{lev}(\mathcal{R},\mathcal{H})$ is the Levenshtein distance [10] between $\mathcal{R}$ and $\mathcal{H}$, and $\sum_{\mathcal{R},\mathcal{H}}$ is the summation of all pairs of reference and hypothesis in a dataset. The Levenshtein distance lev is the minimum number of word edit operations (substitution, deletion, insertion) to change the hypothesis sequence into the reference sequence. Note, though, that the decomposition into substitutions, insertions and deletions is not unique.

The Wagner-Fischer algorithm [11] can be used to efficiently compute lev. It creates a two-dimensional distance matrix $L \in \mathbb{R}^{(|\mathcal{R}|+1)\times(|\mathcal{H}|+1)}$, where each entry $L_{r,h}\ \forall\ r \in \{0,\dots,|\mathcal{R}|\}, h \in \{0,\dots,|\mathcal{H}|\}$ is the Levenshtein distance of the sub-sequences up to the r-th and h-th word of $\mathcal{R}$ and $\mathcal{H}$, respectively, where $r = 0$ or $h = 0$ means comparing with the empty string. The computation starts with $L_{0,h} = h$ and $L_{r,0} = r$, and the entries of $L_{r,h}$ are recursively computed:

$$L_{r,h} = \min \begin{cases} L_{r-1,h-1} + C_{\text{corr}} & \text{if } \mathcal{R}(r) = \mathcal{H}(h) \\ L_{r-1,h-1} + C_{\text{sub}} & \text{if } \mathcal{R}(r) \neq \mathcal{H}(h) \\ L_{r,h-1} + C_{\text{ins}} & \\ L_{r-1,h} + C_{\text{del}}, & \end{cases} \quad (2)$$

where C_{corr}, C_{sub}, C_{ins} and C_{del} are the costs of a correct match, a substitution, an insertion, and a deletion operation, respectively. The costs are typically chosen as $C_{\text{sub}} = C_{\text{ins}} = C_{\text{del}} = 1$ and $C_{\text{corr}} = 0$. The final value is $\text{lev}(\mathcal{R},\mathcal{H}) = L_{|\mathcal{R}|,|\mathcal{H}|}$.

The standard WER is the basis for all other WER definitions. In the following, we only state the Levenshtein distance for simplicity. The corresponding WER is the Levenshtein distance divided by the total number of words, see Eq. (1).

3.2. cpWER

One common metric for evaluation of *Diarization-style* output is the cpWER [6]. The reference streams $\mathcal{R}_1, ..., \mathcal{R}_I$ and hypothesis streams $\mathcal{H}_1, ..., \mathcal{H}_I$ are grouped by I speakers, where segment/utterance transcriptions are concatenated. Then, the standard WER is computed between the hypothesis and all permutations $\pi \in P(I)$ of reference streams. The permutation which minimizes the WER is reported:

$$\text{lev}^{(\text{cp})} = \min_{\pi \in P(I)} \sum_{i=1}^{I} \text{lev}\left(\bigoplus_{u=1}^{U_{\mathcal{R}_{\pi(i)}}} \mathcal{R}_{\pi(i),u}, \bigoplus_{u'=1}^{U_{\mathcal{H}_i}} \mathcal{H}_{i,u'} \right). \quad (3)$$

Empty streams are inserted for the reference or hypothesis when the number of speakers is over- or under-estimated, respectively.

The naive computational complexity is $\mathcal{O}(I!)$ since $|P(I)| = I!$. MeetEval, however, uses the Hungarian algorithm [12, 13] to find the permutation in polynomial time. The computation of all I^2 pairwise Levenshtein distances has a complexity of $\mathcal{O}(I^2)$ and dominates the total run time in practice.

3.3. MIMO WER

The recently proposed MIMO WER [3] measures the WER for multi-speaker scenarios without considering segment-level speaker assignment errors. It solves the assignment problem on a segment level, according to

$$\mathrm{lev}^{(\mathrm{MIMO})} = \min_{\rho_1,\dots,\rho_J} \sum_{j=1}^{J} \mathrm{lev}\left(\bigoplus_{(i,u)\in\rho_j} \mathcal{R}_{i,u}, \bigoplus_{u'=1}^{U_{\mathcal{H},j}} \mathcal{H}_{j,u'} \right), \quad (4)$$

where ρ_j contains index pairs (speaker index i and utterance index u) of the reference utterances that are assigned to the j-th hypothesis. Each reference is assigned exactly once and the references must be sorted by begin time within a speaker. The order of utterances is unconstrained across speakers, which is necessary when evaluating *SOT-style* systems that can output utterance of different speakers in an arbitrary order. The MIMO WER penalizes splitting utterances over multiple streams since a reference utterance is always assigned to one output stream continuously, The distance $\mathrm{lev}^{(\mathrm{MIMO})}$ is computed with a multi-dimensional variant of the Levenshtein distance dynamic programming algorithm [3].

3.4. Optimal Reference Combination WER (ORC WER)

The ORC WER [2, 14] is a special case of the MIMO WER which keeps the temporal order across speakers intact. It can be computed with Eq. (4) by ignoring the reference speaker labels:

$$\mathrm{lev}^{(\mathrm{ORC})} = \min_{\rho_1,\dots,\rho_J} \sum_{j=1}^{J} \mathrm{lev}\left(\bigoplus_{u\in\rho_j} \mathcal{R}_{u}, \bigoplus_{u'=1}^{U_{\mathcal{H},j}} \mathcal{H}_{j,u'} \right). \quad (5)$$

The complexity does not depend on the number of reference speakers and is polynomial in the number of utterances [3].

Note that ORC WER and MIMO WER only differ in certain edge cases when the system modifies the order of utterances (e.g., certain SOT systems [8]) or where the utterance order is ambiguous in the reference. The ORC WER overestimates the WER in these cases [3]. When the utterance order is well defined, the ORC WER is preferred because its complexity is significantly smaller when the number of speakers I is large and the number of system output streams J is small.

3.5. When to use which WER?

Fig. 1 gives an overview of widely used recognizer styles and which WER we recommend for each. For *Diarizaiton-style* recognizers, the cpWER can be computed and gives a good indication of the system performance. The ORC WER and MIMO WER can theoretically be computed to gain insight what the WER would be if all speaker labels were correct. But in practice, this is often infeasible because the complexity explodes with increasing number of system output streams J.

For *CSS-style* and *SOT-style* recognizers, where the output is not grouped by speakers, ORC WER or MIMO WER have to be used. While MIMO WER is desired in such a situation, this is often infeasible (large J). ORC WER is well suited and often equal to the MIMO WER when the output follows the temporal order of the physical signal and has a reasonable execution time when the number of system output streams J is small.

An *SOT-style* recognizer is one of the few exceptions where the recognizer can jump backward in time so that the MIMO WER is the only applicable metric. Since the *SOT-style* output consists of a single stream ($J = 1$), the execution time of MIMO WER is reasonable.

The `asclite` tool is, according to its documentation[3], able to handle all recognizer styles while performing a word-level matching ignoring speaker labels. But in practice, it is computationally demanding and not all promoted features work as expected. For example, the authors of [7] had to pre-process the hypothesis transcriptions before they could apply the tool.

4. Time-constrained WER

All WER metrics presented so far can match words as correct or substituted across arbitrary temporal distances, even though it is unlikely or impossible that the matched words stem from the same acoustic event (e.g., a speaker says common words like "the" and "and" many times during an hour-long session). A human would recognize such a matching as implausible, so the metric should forbid such a matching as well.

To achieve this, we propose to incorporate a temporal constraint into the Levenshtein distance for WER computation. We specifically propose the Time-Constrained minimum-Permutation WER (tcpWER), where the temporal constraint is introduced in the cpWER by replacing the Levenshtein distance in Eq. (3) with its time-constrained variant.

We assume that temporal annotations are present for each word in the form of a beginning and ending time. We discuss later in Section 4.1 how segment-level annotations can be used. Words in the reference and hypothesis are only allowed to match (as correct or substituted) if they overlap or the gap between them is smaller than a collar c. This is equivalent to setting the cost for a substitution and a correct match to $C_{\text{sub}} = C_{\text{corr}} \geq C_{\text{ins}} + C_{\text{del}}$ where words are not allowed to match.

From the time constraint follows a straightforward optimization: Cells in L can be skipped where no words overlap since it is always valid to pick an insertion and deletion over a substitution or a correct match. The skipped cells in Eq. (2) form continuous regions in the upper right and bottom left of the Levenshtein matrix, so that only a diagonal band is left.

In the following, we describe how to deal with segment-level time annotations, then we discuss how to choose the size of the collar, followed by a more general discussion.

4.1. Pseudo-word-level annotations

Word-level time-annotations are often not available, both for reference and hypothesis. Obtaining them for datasets is expensive, error-prone and usually not deterministic. Widely used single-speaker datasets like LibriSpech [15] and WSJ0/1 [16] do not provide them. While a so-called forced alignment can be used to obtain annotations, they are not reproducible and are not reliable for real recordings that contain overlapping speech and noise, like CHiME-6 [6]. On the hypothesis side, obtaining such detailed timing information has become difficult with the rise of attention-based models that provide no direct connection between input and output tokens.

To overcome the need for precise word-level timing information, we propose to infer pseudo-word-level annotations from coarse segment-level annotations, see Fig. 2. The simplest approach is (1) to use the full *segment-level annotation* for each word. This, however, does not represent the true word positions. Using (1) for the hypothesis can be easily fooled: A system could predict a single segment that spans the whole recording length, e.g., by merging all recognized segments, which would make the tcpWER equal to the cpWER and render tcpWER

[3] `https://github.com/usnistgov/SCTK/blob/master/doc/asclite.pod`

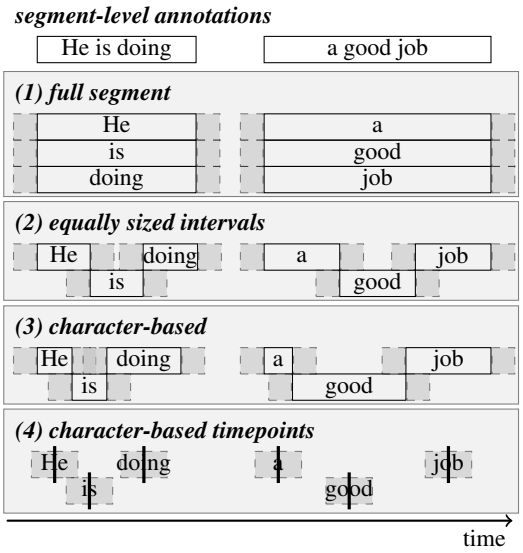

Figure 2: *Visualization of the different pseudo-word-level annotation strategies. The collar is visualized as gray boxes and kept short for better visualization. The character-based annotation strategy correlates best with the actual pronunciation time.*

meaningless. We conclude that annotations within a speaker must be non-overlapping for the hypothesis.

Instead, the true word boundaries can be roughly approximated by (2) *equally sized intervals*, i.e., dividing each segment into number-of-words many intervals of equal size. To incorporate differences in the time required to pronounce words, the (3) *character-based* approximation chooses the segment length proportional to the number of characters in the word. We assume that the number of characters correlates with the pronunciation length, which we confirmed for read English speech. The average difference between the ground truth word length and the character-based approximation is, excluding examples with extreme annotation errors, below 100 ms for both LibriSpeech and TIMIT. Note that the approximation becomes less accurate for longer segments. We recommend (3) for the reference.

The metric could be tricked when using (3) for the hypothesis by filling gaps between segments with single words. This reintroduces implausible matchings that we aim to eliminate with the tcpWER. We recommend to use (4) *character-based timepoints*, i.e., the center points of (3), for the hypothesis.

While a phone-based approximation would be more accurate, it is in practice non-trivial to obtain and not unique. We thus stick to the character-based approximation which can easily be computed for any transcript.

4.2. Collar

A collar $c > 0$ is required to compensate inaccuracies in the (reference) segment-level annotations and approximation errors introduced by the pseudo-word-level annotations. The reference annotations usually extend slightly over the actual pronunciation time. Human annotators tend to over-estimate the length of the speech segments to make sure that all speech is covered. For example, in TIMIT [17] and LibriSpeech [15], the recordings extend over the true speech activity by about 400 ms and 660 ms on average.[4] A similar pattern can be observed for simulated or re-recorded datasets since they are typically based on single-speaker datasets. Additional issues can be introduced by

[4]Actual speech activity determined with Kaldi [1]

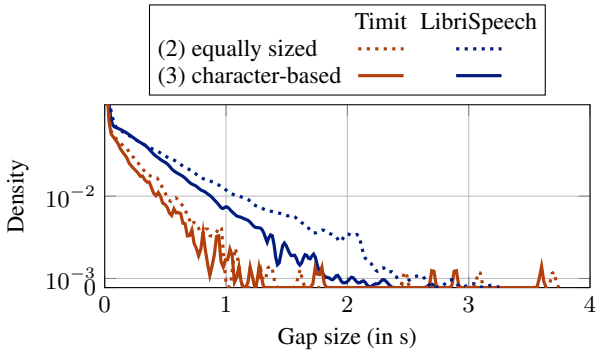

Figure 3: *Density plot of the gap sizes between pseudo-word-level annotations and ground-truth word-level annotations of each word for TIMIT and LibriSpeech.*

the rerecording process, such as constant offsets induced by the playback hardware and sound propagation or a varying offset caused by a sample rate offset. The collar should be chosen large enough to compensate errors introduced by these issues.

4.2.1. Choosing a collar: pseudo-word-level approximation

The collar should compensate the errors introduced by the pseudo-word-level approximation. To determine which collar size compensates all these errors, we look at the gap sizes between the pseudo-word-level approximation and the true word boundaries[5]. Their distribution is shown for the TIMIT [17] and LibriSpeech [15] datasets in Fig. 3 as a case study. The distributions show large peaks at 0 s, which means that for most words the pseudo-word-level annotations overlap with the true word boundaries with $c = 0$ s. The character-based approximation, as expected, matches the true word boundaries better than the equally sized intervals. Since a system that estimates words and timestamps correctly should have a WER of zero, we obtain a lower bound of 3.6 s and 2.7 s for datasets that are based on TIMIT[6] and LibriSpeech, respectively.

4.2.2. Choosing a collar: Approaching the desired WER

For the following investigation, we use the recognition result for the Libri-CSS [7] dataset obtained with a target-speaker separation (TS-SEP) [18] model followed by the base model from Whisper [19] as a single-speaker speech recognizer which delivers word boundaries needed for our analysis. This system generates diarization-style output.

The top of Fig. 4 shows the tcpWER over the collar. The tcpWER naturally approaches the cpWER for an increasing collar c since the tcpWER = cpWER for $c \to \infty$. The tcpWER decreases quickly for small c and is roughly constant in the range of 3 s to 10 s, which agrees with Section 4.2.1.

The collar should be chosen such that it forbids implausible matchings but does not forbid matching of words that actually stem from the same acoustic event. We first define the "desired WER" and the desired matching, where words are only identified as correct when they temporally overlap (without a collar) using word-level timestamps. Reference word-level timestamps are obtained using forced alignments (including an offset compensation for the playback delay) and for the hypothesis using Whipser ASR. The cpWER and the "desired WER" are shown in Fig. 4 as dashed lines. It should be noted that the difference between the cpWER and the desired WER is relatively small

[5]Determined with forced alignments using the Kaldi toolkit [1]

[6]The value for TIMIT is dominated by a single annotation error

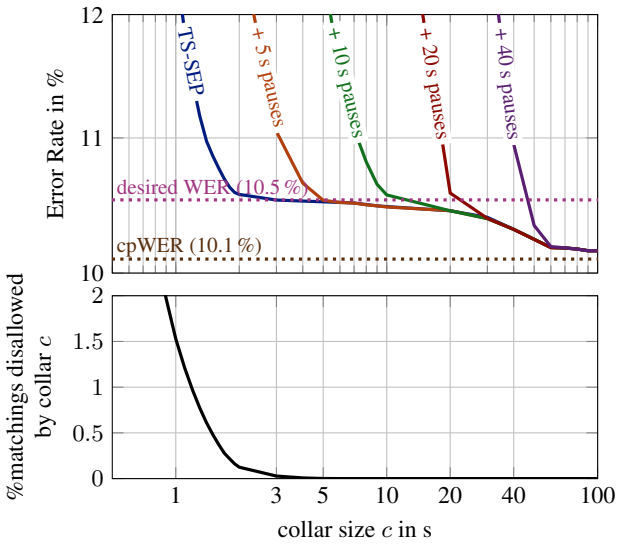

Figure 4: Top: *tcpWER over collar for the TS-SEP model on Libri-CSS. The "desired WER" is determined with oracle word-level timestamps. "+ n s pauses" means that segments were artificially merged to include silence of n seconds length.* Bottom: *Proportion of matchings in the desired WER that would be disallowed by the collar. This value should be 0.*

since the overall performance of the model is relatively good. The tcpWER is close to the desired WER for a collar between 3 s and 5 s which means that this collar size successfully compensate errors introduced by the reference annotation but still forbids temporally implausible matchings.

The bottom of Fig. 4 shows the amount of matchings in the "desired WER" (after offset correction) that would be disallowed by the collar (without offset correction and with pseudo-word-level annotations). This curve includes the errors from Fig. 3, from the estimation of the pseudo-word-level annotations and offset errors introduced by the simulation and recording process of Libri-CSS. Its value should be 0 for the chosen collar value, which gives us a lower bound of 4 s.

To investigate the impact of silence or long pauses in a system output on the metric, we artificially merge segments to include such pauses, denoted by "+ n s pauses" in Fig. 4 for pauses up to n seconds. Large silences should negatively impact the metric which is not reflected in the cpWER but the tcpWER increases more silence. Segments that include large silent regions are penalized because the quality of the pseudo-word-level annotation decreases with increasing amounts of silence.

We argue that short silences in the range of a few seconds do not hurt downstream systems and should be allowed in the system output and the reference. An allowed silence length of 5 s (visualized as "+5 s pauses" in Fig. 4) seems reasonable here since it is unlikely that a real system includes much longer silences and the errors in the reference are typically smaller. A collar of 5 s is thus a reasonable choice for the majority of systems. When applying the tcpWER to other datasets, the user should be aware of the issues mentiond above and adjust the collar value accordingly.

4.3. Relation to other metrics

The idea to use a temporal constraint has appeared before for single utterance recognition, e.g., in `sclite` and [5], but required exact word-level annotations. Due to the issues discussed in Section 4.1, it is often impossible to obtain such annotations for long-form recognition, and thus has been ignored in the past for this scenario. The time-based cost model in `sclite` incorporates the distance between words into the cost for a substitution, which makes the model prefer a deletion and insertion over a substitution when words do not overlap. It has, however, no notion of a collar and the constraint is less strict.

For evaluation of long-form speech recognition, the `asclite` [4] tool uses a temporal constraint, but mainly for speedup reasons. It has no clear notion of a collar and is not designed to improve the quality of the matching.

Table 1: *cpWER and tcpWER on different datasets*

Dataset	(#spks)	Model	cpWER (%)	tcpWER (%)
Libri-CSS	(8)	TS-SEP	10.1	10.5
CHiME-6	(4)	baseline	62.4	66.6
DiPCo	(4)	baseline	56.8	61.7
Mixer 6	(2)	baseline	20.3	20.4

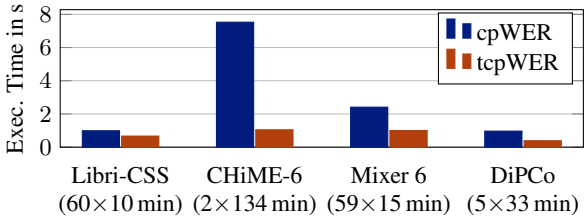

Figure 5: *Execution time of cpWER and tcpWER on different datasets. The number of recordings in the dataset and the average recording length is given in parenthesis.*

4.4. Scoring results

Scoring results for the CHiME-7 baseline system on the CHiME-6 [6], DiPCo [20] and Mixer 6 [21] datasets, and with the TS-SEP model on the Libri-CSS dataset are shown in Table 1. We focus on Diarizaiton-style recognizers here, as they are required for the CHiME-7 challenge. We thus only provide results for cpWER and tcpWER and refer to [3] for ORC WER and MIMO WER. The tcpWER is computed with a collar of $c = 5$ s. As expected, the tcpWER is always larger than the cpWER. Especially for poor recognition performance, as on CHiME-6 and DiPCo, we can observe larger differences.

We observed that higher WERs (more wrongly transcribed words) lead to a higher number of unreasonable matchings. The number of unreasonable matchings is further increased by longer inactivity of a single speaker between utterances. The number of speakers in a recording is a proxy for the amount of silence which is reflected by larger differences between cpWER and tcpWER in Table 1.

4.5. Profiling

The profiling results of cpWER compared with tcpWER are shown in Fig. 5 for LibriCSS, CHiME-6, DiPCo and Mixer 6. Execution times were measured on a single core of a an Intel Core i7-13700K processor, averaged over 10 runs. The execution times of both WERs are in an acceptable range, but the pruning employed in tcpWER leads to a decrease in execution time with increasing total lengths of the evaluated recordings.

5. Conclusions

This paper presents the MeetEval toolkit which implements a variety of Word Error Rate metrics for the evaluation of meet-

ing transcription systems. We propose to incorporate a temporal constraint into the WER computation to improve the matching between words in the reference and hypothesis and prevent unrealistic matchings. We show that the temporal constraint with a reasonably chosen collar leads to realistic WERs. A comprehensive intuition is given for how to choose a reasonable collar.

6. Acknowledgements

Christoph Boeddeker was funded by Deutsche Forschungsgemeinschaft (DFG), project no. 448568305. Computational resources were provided by BMBF/NHR/PC2. We thank Samuele Cornell for kindly providing the transcription results of the CHiME-7 baseline system.

7. References

[1] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely. The Kaldi Speech Recognition Toolkit. Infoscience. [Online]. Available: https://infoscience.epfl.ch/record/192584

[2] I. Sklyar, A. Piunova, X. Zheng, and Y. Liu, "Multi-Turn RNN-T for Streaming Recognition of Multi-Party Speech," pp. 8402–8406. [Online]. Available: http://arxiv.org/abs/2112.10200

[3] T. von Neumann, C. Boeddeker, K. Kinoshita, M. Delcroix, and R. Haeb-Umbach, "On Word Error Rate Definitions and Their Efficient Computation for Multi-Speaker Speech Recognition Systems," in *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5.

[4] J. G. Fiscus, J. Ajot, N. Radde, and C. Laprun, "Multiple Dimension Levenshtein Edit Distance Calculations for Evaluating Automatic Speech Recognition Systems During Simultaneous Speech," in *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*. European Language Resources Association (ELRA), p. 6.

[5] A. C. Morris, V. Maier, and P. Green, "From WER and RIL to MER and WIL: Improved evaluation measures for connected speech recognition," in *Eighth International Conference on Spoken Language Processing*, 2004.

[6] S. Watanabe, M. Mandel, J. Barker, E. Vincent, A. Arora, X. Chang, S. Khudanpur, V. Manohar, D. Povey, D. Raj, D. Snyder, A. Shanmugam Subramanian, J. Trmal, B. Ben Yair, C. Boeddeker, Z. Ni, S. Horiguchi, N. Kanda, T. Yoshioka, and N. Ryant, "CHiME-6 Challenge: Tackling multispeaker speech recognition for unsegmented recordings," in *6th International Workshop on Speech Processing in Everyday Environments (CHiME 2020)*.

[7] Z. Chen, T. Yoshioka, L. Lu, T. Zhou, Z. Meng, Y. Luo, J. Wu, X. Xiao, and J. Li, "Continuous speech separation: Dataset and analysis," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 7284–7288.

[8] N. Kanda, Y. Gaur, X. Wang, Z. Meng, and T. Yoshioka, "Serialized Output Training for End-to-End Overlapped Speech Recognition," in *Interspeech 2020*. ISCA, 2020, pp. 2797–2801.

[9] N. Kanda, J. Wu, Y. Wu, X. Xiao, Z. Meng, X. Wang, Y. Gaur, Z. Chen, J. Li, and T. Yoshioka, "Streaming Multi-Talker ASR with Token-Level Serialized Output Training," in *Interspeech 2022*, pp. 3774–3778.

[10] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals," in *Doklady Akademii Nauk*, vol. 163. Russian Academy of Sciences, 1965, pp. 845–848.

[11] R. A. Wagner and M. J. Fischer, "The String-to-String Correction Problem," vol. 21, no. 1, pp. 168–173, 1974.

[12] H. W. Kuhn, "The Hungarian Method for the Assignment Problem," no. 2, pp. 83–97, 1955.

[13] J. Munkres, "Algorithms for the assignment and transportation problems," vol. 5, no. 1, pp. 32–38, 1957.

[14] D. Raj, L. Lu, Z. Chen, Y. Gaur, and J. Li, "Continuous Streaming Multi-Talker ASR with Dual-Path Transducers," in *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2022, pp. 7317–7321, iSSN: 2379-190X.

[15] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: An ASR corpus based on public domain audio books," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5206–5210.

[16] J. Garofalo, D. Graff, D. Paul, and D. Pallett, "Csr-i (wsj0) complete."

[17] J. S. Garofolo, L. F. Lamel, W. M. Fisher, D. S. Pallett, N. L. Dahlgren, V. Zue, and J. G. Fiscus, "TIMIT Acoustic-Phonetic Continuous Speech Corpus," p. 715776 KB. [Online]. Available: https://catalog.ldc.upenn.edu/LDC93S1

[18] C. Boeddeker, A. S. Subramanian, G. Wichern, R. Haeb-Umbach, and J. L. Roux, "TS-SEP: Joint Diarization and Separation Conditioned on Estimated Speaker Embeddings." [Online]. Available: http://arxiv.org/abs/2303.03849

[19] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever, "Robust speech recognition via large-scale weak supervision," in *International Conference on Machine Learning*. PMLR, 2023, pp. 28 492–28 518.

[20] M. V. Segbroeck, Z. Ahmed, K. Kutsenko, C. Huerta, T. Nguyen, B. Hoffmeister, J. Trmal, M. Omologo, and R. Maas, "Dipco - dinner party corpus," in *Interspeech 2020*, 2020.

[21] L. Brandschain, D. Graff, C. Cieri, K. Walker, C. Caruso, and A. Neely, "The Mixer 6 corpus: Resources for cross-channel and text independent speaker recognition," in *Proc. of LREC*, 2010.