CrisperWhisper: Accurate Timestamps on Verbatim Speech Transcriptions

Laurin Wagner[1], *Bernhard Thallinger*[1], *Mario Zusag*[1]

[1]nyra health

lwagner@nyra.health, bthallinger@nyra.health, mzusag@nyra.health

Abstract

We demonstrate that carefully adjusting the tokenizer of the Whisper speech recognition model significantly improves the precision of word-level timestamps when applying dynamic time warping to the decoder's cross-attention scores. We fine-tune the model to produce more verbatim speech transcriptions and employ several techniques to increase robustness against multiple speakers and background noise. These adjustments achieve state-of-the-art performance on benchmarks for verbatim speech transcription, word segmentation, and the timed detection of filler events, and can further mitigate transcription hallucinations. The code is available open source[1].

Index Terms: speech recognition, word-level timestamp precision, disfluency detection

1. Introduction

Training deep-learning models on large-scale, weakly supervised speech datasets have proven very effective for extracting rich representations, which perform well on versatile speech processing tasks, such as automatic speech recognition (ASR) [1, 2, 3] or speaker verification [4, 5]. Notably, Radford et al. [6] trained Whisper, a sequence-to-sequence (Seq2Seq) transformer model [7] on 680,000 hours of weakly supervised speech recognition data, demonstrating strong generalisation abilities across domains, languages and datasets.

Recent works [8, 9] show that Whisper eliminates many filler words, recurring utterances and other artifacts, which Lea et al. [10] refer to as an *intended* transcription style, suitable for contexts where clarity of intent is prioritized over detailed speech analysis. This style, however, does not support the detection, categorization, or analysis of disfluencies and therefore omits many clinically relevant aspects of speech. *Verbatim* speech transcriptions capture all articulated utterances and can efficiently be used for clinical assessment of speech [11, 12]. A speech disfluency occurs when there's an interruption in the normal rhythm of speech, typically manifesting as filled pauses, word repetitions, or corrections. Filled pauses, beyond their linguistic interest [13], provide insight into the language planning process and are indicators of cognitive load [14, 15, 16]. Therefore, analyzing the timing and frequency of disfluencies, particularly common fillers [17] such as 'uh' and 'um', offers valuable insights into a speaker's cognitive processes. Many clinically relevant biomarkers like speech rate or productive time ratio [12] rely on time accurate detection of all aspects of speech. Wagner et al. [12] show that fluency markers derived from timing information alone are sufficient to differentiate between four different aphasia sub-types with an F_1-score of 81.6. Ge et al. [18] developed a filled pause detection dataset and a pipeline combining ASR, a Voice Activity Detection (VAD) model, and a classifier. Speech regions that remained untranscribed by the ASR model are fed to a VAD model. The resulting voice-active regions are then further classified to identify filled pause events. This approach outperforms a convolutional recurrent neural network (CRNN), which operates on log-mel spectograms with 128 bins computed from 1 second clips and a forced-aligner based method called Gentle [19] combined with an acoustic model. However, distinguishing between filler words and other disfluency types such as false starts, remains challenging for this uncontextualized system. In contrast, Whisper's contextual capabilities could offer improved speech analysis capabilities in noisy scenarios.

Whisper does not provide word-level timestamps natively. To this end, WhisperX [20] uses force-alignment between Whisper's transcriptions and a connectionist temporal classification (CTC) based phoneme model. This forced phoneme alignment transfers the timing information from the CTC-based Wav2Vec2.0 model [1] onto Whisper's transcripts. Their VAD-based cut and merge approach allows for segmenting audio efficiently before transcription, improving both speed and accuracy of the transcriptions. However, this method faces challenges, since discrepancies between model transcripts can further degrade timestamp precision. Additionally, employing a second model increases complexity and the VAD-based segmentation approach, while efficient, lacks robustness in noisy environments. Moreover, Wav2Vec2.0 tends to be less noise robust than Whisper, further degrading timestamps in noisy scenarios. Another approach that gained popularity for inferring word-level timestamps uses Dynamic Time Warping(DTW) on the cross-attention scores of the Whisper decoder [21, 22]. Further, in the context of disfluencies, Koenecke et al. [23] examined Whisper's issues of producing hallucinated content when transcribing speech from people with aphasia. Utilizing samples of AphasiaBank [24], they showed that roughly 1% of the produced transcripts contained hallucinated content.

We show that by adjusting Whisper's tokenizer and carefully fine-tuning Whisper on artificial perturbations for noise robustness and single-speaker focus (i) the word-level timestamps can be improved significantly using a single model (ii) the verbatim transcription style reaches state-of-the-art results on more verbatim datasets such as the AMI Meeting Corpus [25] or TED-LIUM [26], while maintaining the same accuracy on datasets such as Librispeech [27] or CommonVoice [28] (iii) the model achieves near perfect filled pause detection accuracy and (iv) we can substantially mitigate hallucinations. We call the resulting model CrisperWhisper for its *crisp* timestamps and are open-sourcing a synthetic dataset with accurate word-level timestamps as well as the code for CrisperWhisper.

[1]https://github.com/nyrahealth/CrisperWhisper

2. CrisperWhisper

2.1. DTW for Timestamp Prediction

2.1.1. Intuition

Whisper employs an encoder-decoder structure, where the encoder incorporates multiple Transformer encoder blocks to process audio. Initially, audio is re-sampled to 16 kHz and converted into an 80-channel log-magnitude Mel spectrogram and downsample via convolutions. The encoder operates on these downsampled spectograms in 25 ms windows with a stride of 20 ms, meaning that each processed state represents 25 ms of audio, which is shifted by 20 ms steps. The Whisper large model series uses a byte-level Byte Pair Encoding (BPE) text tokenizer [29], which produces the targets during training. Whisper's Transformer decoder uses cross-attention layers. The resulting cross-attention scores effectively reflect the decoder's focus on specific segments of the encoder output during the token prediction process. The intuition is that this focus is indicative of the decoder's strategy to prioritize encoder output regions most relevant to the current token's prediction. The goal is therefore to use the network's cross attention scores to assess which 25 ms audio frames were important to decode the current token and use the aggregate of these frames as a timestamp [22].

2.1.2. DTW and Cost Matrix Construction

We employ the DTW [30] algorithm to find the optimal cost, monotonic and continuous alignment between two sequences, requiring a cost matrix to measure the alignment expense between elements of these sequences. In our case these sequences are the encoder outputs $E = \{e_1, e_2, \ldots, e_n\}$ representing the encoded acoustic signal and the sequence of decoder token predictions $D = \{d_1, d_2, \ldots, d_m\}$. Given a set of suitable attention heads from the decoder $H = \{h_1, h_2, \ldots, h_l\}$ the cost matrix is defined as follows. Each of the d_i is associated with a set of cross-attention vectors $A_i = \{a_{i1}, a_{i2}, \ldots, a_{il}\}$, where a_{ik} denotes the attention score from the k-th attention head when decoding the i-th token, with each $a_{ik} \in \mathbb{R}^n$. We average these vectors ($\bar{A}_i = \frac{1}{l}\sum_{k=1}^{l} a_{ik}$) and normalize them to construct the cost matrix C:

$$C := -\begin{bmatrix} \frac{\bar{A}_1}{\|\bar{A}_1\|_2} \\ \vdots \\ \frac{\bar{A}_m}{\|\bar{A}_m\|_2} \end{bmatrix}$$

Crucially and in contrast to [22], we remove all tokens corresponding to punctuation from D before constructing the cost matrix since punctuation has no clear acoustic representation and should therefore not be given a timestamp in the alignment.

2.2. Retokenization

Taking a closer look at the tokens in the vocabulary of Whisper, we identify that many tokens are prefixed with a space. When applying the BPE algorithm to all CommonVoice14 transcripts [28] using Whisper's vocabulary, we observe that only 13% of spaces in the original transcripts are mapped to the explicit space token. For instance, tokenizing the sentence 'This is a long pause.' with Whisper's original tokenizer results in ['This', ' is', ' a', ' long', ' pause', '.'], where spaces are included at the start of tokens rather than as standalone entities. This tokenization approach impacts the application of DTW for aligning audio segments to tokens, as it inadvertently integrates pauses at the beginning of tokens into their timings. We observe that spaces are exclusively found at the beginning of tokens but never at the end or in the middle. Therefore, we propose to simply strip all tokens in the vocabulary of spaces, except the space token itself, and keep only the unique tokens. We adjust the merges in the tokenizer to be congruent with this reduced vocabulary. This simple adjustment ensures that all spaces will be tokenized individually, theoretically enabling the DTW algorithm to detect pauses between words. Retokenizing our example with the adjustment yields ['This', ' ', 'is', ' ', 'a', ' ', 'long', ' ', 'pause', '.']. The difference between the DTW paths on the cross attention scores of Whisper's large-v2 version with its original tokenizer can be found in Figure 1 with the fine-tuned CrisperWhisper model and its adjusted tokenizer in Figure 2, visualising the close alignment with the ground truth timings. We further re-purpose tokens for 'uh' and 'um' to canonically transcribe filled pause events.

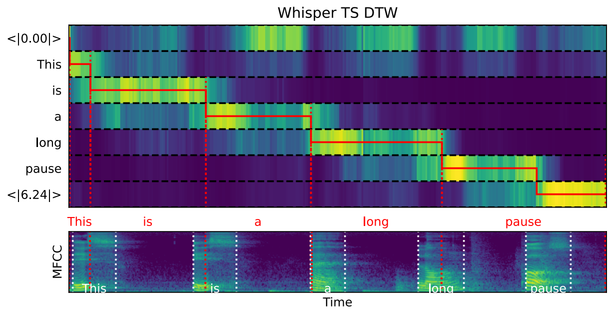


Figure 1: *Example of a DTW path through the cross-attention weights matrix of Whisper large-v2 as in [22]. White lines represent the ground truth.*

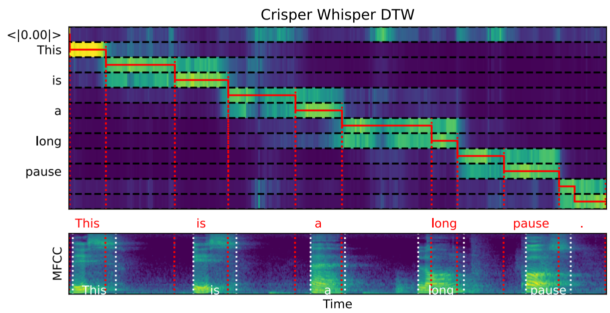


Figure 2: *Example of a DTW path through the cross-attention weights matrix after CrisperWhisper retokenization. White lines represent the ground truth.*

2.3. Pause Heuristics

To address the overestimation of pause durations by the DTW algorithm due to non-distinct attentions, we introduced a heuristic that splits the duration of pauses evenly between the preceding and subsequent words, setting a cap at 160 ms. This cap is based on the observed distribution of pause durations, effectively distinguishing between insubstantial "artifact" pauses and meaningful speech pauses. Durations surpassing this cap are identified and timed as genuine pauses, ensuring a more accurate representation of speech rhythm.

3. Training

3.1. Datasets

Our training dataset consists of two spontaneous speech datasets with a verbatim transcription style, namely the **AMI Meeting Corpus** [25] and a specially adapted version of the **PodcastFillers Corpus** [18], along with a cleaned segment of the **CommonVoice14 Corpus** [28] English subset. Additionally, we use two noise datasets, **FSDnoisy18k** [31] and **AudioSet** [32], to make the model more noise invariant. Specifically, for the AMI dataset, we utilize the training split of the AMI-IHM subset, which contains approximately 29,000 meeting recording clips with canonical transcriptions of filler events. Moreover, we utilize a subset of the PodcastFillers dataset, which comprises approximately 35,000 instances of filler words such as 'uh' or 'um' used in podcast episodes. The dataset includes timings and automatically generated timed transcriptions for the podcast episodes. Our process for reformatting this dataset involves the following steps:

1. **Sampling Context:** For each timed filler word in the training set, we generate three distinct audio segments by choosing varying context lengths ranging from 1 to 5 seconds from both before and after the filler. This selection is made with care to avoid including partial words on both ends, slightly adjusting the sampled context length when needed to include partial words fully. This expands our dataset to approximately 105,000 samples.
2. **Cut Audio Clips with Aligned Transcripts:** The chosen audio segments, along with their aligned transcripts, are extracted. Filler words are explicitly marked as either 'uh' or 'um' in their respective positions within the transcripts.
3. **Transcript Correction:** To address inaccuracies in punctuation and capitalization within the original transcripts, we utilized GPT-4 [33] after observing that the original transcript quality adversely affected Whisper's ability to correctly apply punctuation.

For CommonVoice14, we employed Whisper's medium model on the English train subset to identify and remove samples likely not transcribed verbatim. Any sample with a character error rate exceeding 3% compared to its label was excluded.

3.2. Implementation Details

To deploy CrisperWhisper as a comprehensive speech analysis model in practical applications, it is essential that the model is trained to prioritize the primary speaker's voice and to be generally noise robust. To this end we use the noisy/overlapped speech simulation proposed in WavLM [2] during fine-tuning from the whisper-large-v2 checkpoint [34]. As noise data, we use FSDnoisy18k, AudioSet, random Gaussian noise and random speech samples drawn from the dataset. To counteract hallucinations, we introduce noise-only samples (containing no speech) with empty transcriptions in 1% of the training samples. The training process spans 6,000 steps with a batch size of 256, utilizing a 0.00005 learning rate, a linear learning rate decay with an 800-step warmup phase, amounting to approximately 2 epochs.

4. Evaluation

4.1. Datasets

AMI Meeting Corpus: We use the official test split on the AMI-IHM subset with approximately 11,500 samples. **AMI disfluency subset**: The AMI Meeting Corpus contains filler words transcribed in a canonical way. We extend the filler words by using GPT-4 to label repetitions, false starts and revisions on the transcripts of the AMI-IHM test set. Our disfluency subset consists of all files and transcripts that contain at least one of the labeled disfluencies and contain more than 5 words, which are approximately 4,000 samples. **PodcastFillers Corpus**: We use the same approach as described in Section 3.1 for creating annotated filler samples on the official test subset, choosing 1 second of context before and after each filler, which results in approximately 5,000 samples. **Synthetic dataset**: To compare word-level timestamp accuracy, we created 200 samples of spontaneous speech transcripts with GPT-4, which contain natural pauses in sentences, indicated by '...' in the transcripts. These transcripts were subsequently synthesized with ElevenLabs `https://www.elevenlabs.io`, creating naturally sounding spontaneous speech samples for which timestamps were manually annotated. **AphasiaBank Corpus**: AphasiaBank [24] comprises a collection of interviews between clinicians and subjects afflicted with aphasia, as well as healthy control subjects. We are using the same files that Koenecke et al. [23] have identified to cause hallucinated content when transcribed with Whisper large-v2. **TED-LIUM**: We use the test split of TED-LIUM Release 3 [26], using the segmented manual test transcripts included in the release. **LibriSpeech**: We use both of the popular LibriSpeech [27] 'test clean' and 'test other' splits for evaluation.

4.2. Metrics

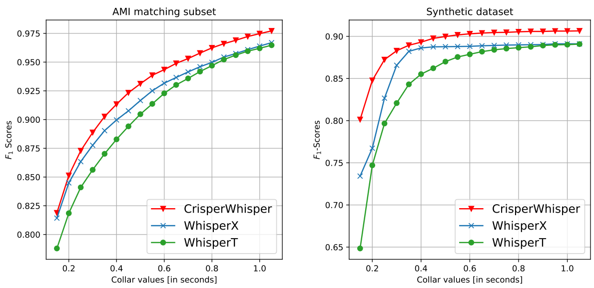
To evaluate transcription accuracy, we use word error rate (**WER**) and insertion error rate (**IER**) to quantify word omissions. For timing accuracy, we use the **F_1-score**, which is well defined via basic confusion matrix terminology. In this context, we define a **true positive** as a predicted word that both overlaps temporally with a reference word and matches its content. Each reference word can only contribute to a single true positive. A **false positive** is defined as a predicted word that does not have temporal overlap or content match with any reference word. Conversely, a **false negative** is a reference word that does not have temporal overlap or content match with any predicted word. Temporal overlap occurs when the start (onset) and end (offset) timestamps of a prediction fall within a predefined collar of the corresponding timestamps in the reference. Additionally, we evaluate localization accuracy using the **mean Intersection over Union (mIoU)** metric, which compares each predicted word against the reference for both string match and temporal overlap, calculating the IoU based on timestamps, or assigning a score of 0 if no match exists. The highest IoU score for each word represents its IoU, ensuring each word is matched only once.

4.3. Results

In the following, we are referring to Whisper's large-v2 model with the DTW implementation of [22] as **WhisperT**, the default configuration of WhisperX [20] with an underlying Whisper large-v2 and Wav2vec2.0 alignment model as **WhisperX**.

4.3.1. Word Segmentation Performance

Figure 3a shows how CrisperWhisper outperforms previous state-of-the-art models using different collar values on the test set of the AMI Corpus. We ensure that the normalized transcripts of our prediction, the normalized reference and the normalized predictions of the other models coincide completely. This is to account for the fact that our more verbatim approach gives us an unfair advantage and we want to ensure that we evaluate the localization performance separately from the transcription accuracy. Figure 3b depicts the segmentation performance using different collar values on the clean, synthetic dataset mentioned in Section 4.1 with manually annotated timestamps.



(a) *Matching AMI subset* (b) *Synthetic dataset*

Figure 3: *Word segmentation performance showing the F_1-score for different collar values.*

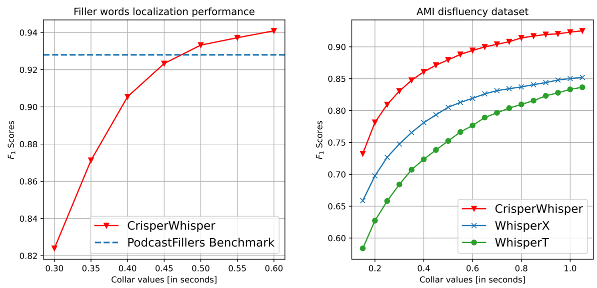
We further evaluate the noise robustness of our model by adding random secondary voice samples from the LibriSpeech 'test clean' subset, white noise, and random noise samples from FSDnoisy18k with a signal-to-noise ratio of 1:5 to the synthetic samples. As detailed in Table 1, CrisperWhisper demonstrates superior robustness in terms of mIoU and F_1-score under noisy conditions. In contrast, WhisperX exhibits a more significant performance decline than WhisperT, attributable to Wav2Vec2.0's lesser noise resilience. Notably, CrisperWhisper achieves markedly higher mIoU metrics and F_1-scores, particularly with narrower collars, underscoring its enhanced accuracy in timestamping pauses compared to other evaluated methods.

Table 1: *Noise robustness of word segmentation performance on synthetic data using a collar of 0.2 seconds.*

Model	**Synthetic**		**Synthetic noisy**	
	$F_1 \uparrow$	mIoU $\uparrow$	$F_1 \uparrow$	mIoU $\uparrow$
WhisperT [34]	74.7	51.4	68.3	49.8
WhisperX [20]	76.7	61.5	59.0	44.3
CrisperWhisper	**84.7**	**63.4**	**79.5**	**60.5**

4.3.2. *Disfluency Segmentation Performance*

For evaluating the filler word detection and segmentation performance, we transcribe the audio examples of our adjusted test split of the Podcast-Fillers Corpus as described in Section 4 with CrisperWhisper and calculated the F_1-scores as described in Section 4.2 for various collar values. The results can be seen in Figure 4a with the reported F_1-score eventually exceeding the acoustic model reported in [18], although the segmentation is worse for collar values smaller than 0.5 seconds. Since our model transcribes verbatim, we also detect and segment other disfluency types, such as repetitions, false starts or partial words. Figure 4b shows the localization performance on disfluent speech samples of the AMI disfluency subset described in Section 4.1.



(a) *PodcastFillers Dataset* (b) *AMI disfluent subset*

Figure 4: *Disfluency Localization Performance*

4.3.3. *Verbatim Transcription Performance*

Table 2 shows the significantly improved ASR performance on spontaneous speech datasets with more verbatim transcriptions. We have further validated that CrisperWhisper's transcription accuracy does not degrade on intended speech datasets. All transcriptions of the 'test other' and 'test clean' subsets of the LibriSpeech Corpus or the test split of CommonVoice14 lie within 0.01 WER of the original Whisper large-v2 model.

Table 2: *ASR performance on verbatim datasets.*

Model	**AMI**		**TED-LIUM**	
	WER $\downarrow$	IER $\downarrow$	WER $\downarrow$	IER $\downarrow$
Whisper [34]	16.82	11.77	4.01	3.08
CrisperWhisper	**9.72**	**2.26**	**3.26**	**0.75**

4.3.4. *Hallucinations*

To validate hallucination mitigation, we use the same audio files from AphasiaBank as analyzed by Koenecke et al. [23]. CrisperWhisper does not produce harmful hallucinations on any of the identified speech recordings, but produces repetitive transcription loops on 10 recordings. Since CrisperWhisper is producing accurate word-level timestamps, we are simply removing tokens with a duration below 50 ms, effectively eliminating this type of artefact of hallucinating speech during inactivate speech regions.

5. Conclusion

We proposed CrisperWhisper, a robust end-to-end speech transcription model producing accurate word-level timestamps in a verbatim, single speaker focused transcription style. We trace the problem of unsharp timestamps around disfluencies and pauses back to Whisper's tokenizer and present a strategy to alleviate this problem. One weakness of our approach is the arbitrary selection of attention heads used for alignment. In the near future, we want to investigate the verbatim transcription and segmentation capabilities for quantifying speech deficits, scale the approach with more high quality verbatim data and explore how these capabilities can be transferred to other languages.

6. References

[1] A. Baevski, H. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," 2020. [Online]. Available: https://arxiv.org/abs/2006.11477

[2] S. Chen, C. Wang, Z. Chen, Y. Wu, S. Liu, Z. Chen, J. Li, N. Kanda, T. Yoshioka, X. Xiao *et al.*, "Wavlm: Large-scale self-supervised pre-training for full stack speech processing," *IEEE Journal of Selected Topics in Signal Processing*, vol. 16, no. 6, pp. 1505–1518, 2022.

[3] W. Hsu, B. Bolte, Y. H. Tsai, K. Lakhotia, R. Salakhutdinov, and A. Mohamed, "Hubert: Self-supervised speech representation learning by masked prediction of hidden units," *CoRR*, vol. abs/2106.07447, 2021. [Online]. Available: https://arxiv.org/abs/2106.07447

[4] Y. Zhang, D. S. Park, W. Han, J. Qin, A. Gulati, J. Shor, A. Jansen, Y. Xu, Y. Huang, S. Wang *et al.*, "Bigssl: Exploring the frontier of large-scale semi-supervised learning for automatic speech recognition," *IEEE Journal of Selected Topics in Signal Processing*, vol. 16, no. 6, pp. 1519–1532, 2022.

[5] J. Kang, J. Huh, H. S. Heo, and J. S. Chung, "Augmentation adversarial training for self-supervised speaker representation learning," *IEEE Journal of Selected Topics in Signal Processing*, vol. 16, no. 6, pp. 1253–1262, 2022.

[6] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever, "Robust speech recognition via large-scale weak supervision," *arXiv preprint arXiv:2212.04356*, 2022.

[7] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[8] S. Wollin-Giering, M. Hoffmann, J. Höfting, C. Ventzke *et al.*, "Automatic transcription of english and german qualitative interviews," in *Forum Qualitative Sozialforschung/Forum: Qualitative Social Research*, vol. 25, no. 1, 2024.

[9] A. Romana, K. Koishida, and E. M. Provost, "Automatic disfluency detection from untranscribed speech," *arXiv preprint arXiv:2311.00867*, 2023.

[10] C. Lea, Z. Huang, J. Narain, L. Tooley, D. Yee, D. T. Tran, P. Georgiou, J. P. Bigham, and L. Findlater, "From user perceptions to technical improvement: Enabling people who stutter to better use speech recognition," in *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, 2023, pp. 1–16.

[11] A. Romana, J. Bandon, M. Perez, S. Gutierrez, R. Richter, A. Roberts, and E. M. Provost, "Automatically detecting errors and disfluencies in read speech to predict cognitive impairment in people with parkinson's disease." in *INTERSPEECH*, 2021, pp. 1907–1911.

[12] L. Wagner, M. Zusag, and T. Bloder, "Careful whisper – leveraging advances in automatic speech recognition for robust and interpretable aphasia subtype classification," in *INTERSPEECH*, 2023.

[13] H. H. Clark and J. E. F. Tree, "Using uh and um in spontaneous speaking," *Cognition*, vol. 84, no. 1, pp. 73–111, 2002.

[14] A. Lindström, J. Villing, S. Larsson, A. Seward, N. Åberg, and C. Holtelius, "The effect of cognitive load on disfluencies during in-vehicle spoken dialogue," *INTERSPEECH, 2008*, pp. 1196–1199, 09 2008.

[15] M. Corley and O. Stewart, "Hesitation disfluencies in spontaneous speech: The meaning of," *Language and Linguistics Compass*, vol. 2, pp. 589–602, 07 2008.

[16] E. G. Bard, R. J. Lickley, and M. P. Aylett, "Is disfluency just difficulty?" in *Proc. ITRW on Disfluency in Spontaneous Speech (DiSS 2001)*, 2001, pp. 97–100.

[17] K. Womack, W. McCoy, C. Ovesdotter Alm, C. Calvelli, J. B. Pelz, P. Shi, and A. Haake, "Disfluencies as extra-propositional indicators of cognitive processing," in *Proceedings of the Workshop on Extra-Propositional Aspects of Meaning in Computational Linguistics*, R. Morante and C. Sporleder, Eds. Jeju, Republic of Korea: Association for Computational Linguistics, Jul. 2012, pp. 1–9. [Online]. Available: https://aclanthology.org/W12-3801

[18] G. Zhu, J.-P. Caceres, and J. Salamon, "Filler word detection and classification: A dataset and benchmark," in *INTERSPEECH, 2023*, Incheon, Korea, Sep. 2022. [Online]. Available: https://arxiv.org/abs/2203.15135

[19] R. Ochshorn and M. M. Hawkins, "Gentle: A robust yet lenient forced aligner built on kaldi," https://lowerquality.com/gentle/, 2015.

[20] M. Bain, J. Huh, T. Han, and A. Zisserman, "Whisperx: Time-accurate speech transcription of long-form audio," *INTERSPEECH*, 2023.

[21] J. W. Kim, "openai-dtw," https://github.com/openai/whisper/blob/main/notebooks/Multilingual_ASR.ipynb, 2023.

[22] J. Louradour, "whisper-timestamped," https://github.com/linto-ai/whisper-timestamped, 2023.

[23] A. Koenecke, A. S. G. Choi, K. Mei, H. Schellmann, and M. Sloane, "Careless whisper: Speech-to-text hallucination harms," *arXiv preprint arXiv:2402.08021*, 2024.

[24] B. MacWhinney, D. Fromm, M. Forbes, and A. Holland, "Aphasiabank: Methods for studying discourse," *Aphasiology*, vol. 25, no. 11, pp. 1286–1307, 2011.

[25] J. Carletta, S. Ashby, S. Bourban, M. Flynn, M. Guillemot, T. Hain, J. Kadlec, V. Karaiskos, W. Kraaij, M. Kronenthal *et al.*, "The ami meeting corpus: A pre-announcement," in *International workshop on machine learning for multimodal interaction*. Springer, 2005, pp. 28–39.

[26] F. Hernandez, V. Nguyen, S. Ghannay, N. Tomashenko, and Y. Estève, "Ted-lium 3: Twice as much data and corpus repartition for experiments on speaker adaptation," in *Speech and Computer*. Springer International Publishing, 2018, pp. 198–208.

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

[28] R. Ardila, M. Branson, K. Davis, M. Henretty, M. Kohler, J. Meyer, R. Morais, L. Saunders, F. M. Tyers, and G. Weber, "Common voice: A massively-multilingual speech corpus," in *Proceedings of the 12th Conference on Language Resources and Evaluation (LREC 2020)*, 2020, pp. 4211–4215.

[29] P. Gage, "A new algorithm for data compression," *C Users Journal*, vol. 12, no. 2, pp. 23–38, 1994.

[30] T. Giorgino, "Computing and visualizing dynamic time warping alignments in r: The dtw package," *Journal of Statistical Software*, vol. 31, no. 7, 2009.

[31] E. Fonseca, M. Plakal, D. P. Ellis, F. Font, X. Favory, and X. Serra, "Learning sound event classifiers from web audio with noisy labels," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 21–25.

[32] J. F. Gemmeke, D. P. W. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio set: An ontology and human-labeled dataset for audio events," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017, pp. 776–780.

[33] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat *et al.*, "Gpt-4 technical report," *arXiv preprint arXiv:2303.08774*, 2023.

[34] OpenAI, "Whisper large v2," https://huggingface.co/openai/whisper-large-v2, 2024, version of 20.02.2024.