Test-Time Training for Speaker Adaptation in ASR Systems

Rajan Vivek
Stanford University
rvivek@stanford.edu

Matthew Harvill
Stanford University
mharvill@stanford.edu

Abstract

We partnered with Ello (Ello, 2024) to evaluate test-time training (TTT) with self-supervision approaches for ASR adaptation to different speakers using a small amount of unlabeled data. Ello has developed a child reading assistant that uses ASR to track child speech and assist their pronunciation. However, Ello's Wav2Vec2-based ASR system struggles to generalize to all speakers as children have highly varied speech patterns and reading levels. We experiment with different training strategies to apply TTT, an out-of-distribution generalization technique involving taking gradient steps on test data with a self-supervised objective at test time, to the pretrained Wav2Vec2 model. We find that training a transformer-based ASR decoder over a frozen Wav2Vec2-base encoder at train time and then adapting the encoder with its pretraining objective over test samples at test time can result in WER reductions of 2% on held-out test data and 5-20% on seen (but unlabeled) test samples. However, due to limited compute our custom decoder bottlenecks overall system performance, with a WER penalty that outweighs the gains of TTT. We conclude that TTT with SSL is best suited for pre-trained models that already have a decoder (or for organizations with ample compute). To overcome this weakness, we recommend further experimentation with decoder weight initialization strategies and TTT approaches that do not require a decoder, which would allow TTT to be applied to the many existing pre-trained audio encoders.

1 Introduction

We partnered with Ello to improve their reading assistant for children. The Ello reading assistant is an app used on tablets that tracks children's reading progression in real-time and assists with their pronounciation of words. Given that children come from many backgrounds– e.g. different accents, speech impediments, and reading abilities– and use Ello in different conditions– e.g. with a noisy background, physical case obstructing the microphone, or parent reading along– it is important for Ello to adapt to different users. We propose to use Test-Time Training with Self-Supervision (TTT) (Sun et al., 2020) to adapt the ASR system used in Ello to different users and conditions. Specifically, as the system acquires test samples, it performs small updates to the end-to-end ASR model in order to account for the discrepancy between its training distribution and the audio distribution of these samples. Unlike conventional speaker adaptation approaches, this strategy does not require labeled speaker data. This makes it ideal for the Ello use case, where transcriptions are not available.

TTT with self-supervision requires an architecture and training strategy that is distinct from how audio foundation models like Wav2Vec2 (Baevski et al., 2020) are commonly used. Specifically, it requires a Y-shaped architecture with a shared encoder and two decoders– one for the downstream task and one for the self-supervision task. At train time, the encoder must be specialized for either both tasks or only the self-supervised task, in contrast to typical fine-tuned Wav2Vec2 models that are specialized for the downstream task only. In this project, we compare TTT-based training recipes to non-TTT alternatives for using Wav2Vec2 for ASR on out-of-distribution speakers. Specifically, our research questions are:

1. What training strategies should be used at train time and at test time in order to see reliable gains for Wav2Vec2 on out-of-distribution test data?
2. How do these gains compare to allocating the train-time compute to fine-tuning on in-distribution train data?
3. How do these gains compare to fine-tuning on test data, i.e. does TTT overcome the need for labels on test data?

2 Related Work

State-of-the-art ASR performance is achieved by large models pre-trained on a large amount of audio data, i.e. $10^5 - 10^6$ hours, with self-supervision (Baevski et al., 2020; Hsu et al., 2021) or weak supervision (Radford et al., 2022) objectives. While powerful, these systems often falter on data that is farther from their training distribution. Speaker adaptation refers to adapting a speech-based system to the audio characteristics of a specific speaker or set of speakers in order to maximize system performance on their data. The most straightforward and reliable method of speaker adaptation is fine-tuning. (Jain et al., 2023) showed that the both Whisper and Wav2Vec2 models could be adapted to child speech with significant performance gains using 10-55 hours of child speech.

However, adapting to specific speakers with far less fine-tuning data is more difficult. Prior work has shown that speaker-embeddings called xvectors (Snyder et al., 2018) and feature space maximum likelihood linear regression (fMLLR) features can improve performance on specific speakers (Baskar et al., 2022). However, these methods require 1) large architecture changes in order to be used in pre-trained systems and 2) labeled speaker data. Thus, they are not ideal for our application.

Researchers have found that approaches based on TTT with self-supervision can result in large performance gains of out-of-distribution data with *no labeled test data*. These findings have been shown in image (Gandelsman et al., 2022), time series (Gong et al., 2023), and (more recently) speech (Dumpala et al., 2023) domains. Test Time Training with Self-Supervision (Sun et al., 2020) was the first work to show that training on unlabeled test data at test time with self-supervision can achieve large gains on out-of-distribution data. Specifically, one can use a Y-shaped architecture with a shared encoder, one decoder for a self-supervised task, and one decoder for a downstream task. By training this model on both tasks at train time and then taking gradient steps with the self-supervised objective at test time, the model can overcome train-test distribution shifts on the downstream task.

Subsequent works showed that this approach fails when the test distribution is non-i.i.d. (Gong et al., 2023) or too far from the train distributions (Liu et al., 2021), and proposed various strategies to mitigate these issues including feature alignment, learned normalization, and maintaining a rotating queue of old test samples to include in each TTT update batch.

TTT with Masked Autoencoders (Gandelsman et al., 2022) was the first paper to show that large TTT gains in the image domain could be achieved by adapting to a *single* test sample. By repeating a single test image with many different masks in the same batch and training the encoder solely for the supervised task so that the downstream decoder learned to rely on these features, the authors achieved object recognition gains of 10-20% on OOD image samples. TTT for Speech (Dumpala et al., 2023) further showed that Gandelsman et al. (2022) could be adapted to speech classification problems with a spectrogram masking objective and using BitFit (Zaken et al., 2022) to improve training stability. The authors trained their own encoder from scratch.

Thus, TTT with SSL satisfies our requirements for OOD generalization with a small amount of unlabeled test data. It remains an open question of how to adapt an off-the-shelf audio foundation model such as Wav2Vec2 using TTT with self-supervised learning.

3 Approach

We attacked the problem of out-of-distribution test speech data with two main approaches. These approaches shared a few commonalities. First, both approaches used TTT with self-supervision (single shared encoder with a main task decoder and self-supervised task decoder). Secondly, both approaches used pretrained Wav2Vec2-base for the shared encoder and self-supervised task head. Since Wav2Vec2-base was pretrained on a contrastive task for learning quantized speech representations, we treated the bulk of the model (mostly transformer layers) as the shared encoder, with only the final projections and quantization modules as the self-supervision head.

Our first approach involved training a CTC decoder on top of our (frozen) shared Wav2Vec2-base encoder and is depicted in Figure 1. We chose to freeze the encoder while training our CTC decoder because it forced the decoder to rely solely on the encoder's features specialized for the self-supervised task. This way, when the encoder is trained on the self-supervised task at test time, our decoder can transform better self-supervised representations into better CTC predictions. This is consistent with the findings of Gandelsman et al.

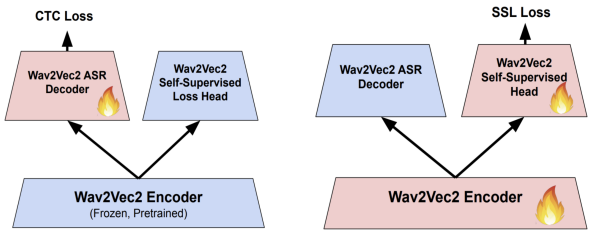

Figure 1: Our TTT approach inspired by Gandelsman et al. (2022) (Train-time updates with CTC Decoder are shown on the left in red and test-time updates with SSL head are shown on the right. Blue indicates frozen and red indicates unfrozen.)

(2022).

We designed our CTC decoder to have 4 transformer layers with hidden size 512, a projection layer from the outputs of the encoder to our decoder (768 to 512), and fixed sinusoidal position embeddings. We decided on this architecture because it was expressive enough to achieve reasonable CTC performance, while fitting within our compute constraints (two consumer GPUs).

Our second approach was very similar to our first approach. For this approach we used the exact same architecture and test time updates. We also initialized our model using the trained weights from our first approach (trained CTC decoder and Wav2Vec2-base for the encoder+self-supervised decoder). However, instead of freezing the encoder and training purely on the CTC task, for this approach we trained the model end-to-end on a multitask objective including the CTC and self-supervised losses. We decided on this approach for two main reasons. First, the literature supported it, i.e. the authors of TTT (Sun et al., 2020) originally trained their model from scratch on a multitask objective. Secondly, this approach would overcome the limitations of our CTC decoder's size by leveraging the encoder for CTC, while maintaining solid performance on the self-supervision task. Thus, this method would improve baseline CTC performance while retaining flexibility for TTT improvements.

4 Experiments

We ran five main experiments to evaluate both TTT and non-TTT approaches at improving Wav2Vec2's performance on out-of-distribution (OOD) test data. We selected Gigaspeech's Youtube audio as our test data, since it was fairly OOD for Wav2Vec2, which was trained on Librispeech audio books. For each of three distributions (Youtube Education, Gaming, and Science/Technology videos), we selected 5 videos to serve as our "speakers." (We use speaker/video interchangeably for the remainder of this paper). For each video, we selected 150 samples to serve as "seen" test data (i.e. unlabeled test data that TTT can adapt to) and 50 as "held-out" test data.

Note - We decided not to evaluate on Ello's child speech data because it required labeling and we received it late in the quarter. However, this did not affect our ability to demonstrate the effectiveness of our method(s).

Our experiments are as follows:

1. Fine-tuning Wav2Vec2-960h for ASR on each speaker. This strong baseline shows the performance we could achieve if we had labeled test data, though for all remaining experiments we assume that test labels are not available.

2. Applying TTT directly to Wav2Vec2-960h. This baseline explores whether TTT can be applied to Wav2Vec2-960h, testing the hypothesis asserted by Gandelsman et al. (2022) that TTT does not work when the encoder is fine-tuned on the downstream task.

3. Adding a randomly-initialized 4-layer decoder to Wav2Vec2-base (hereby called Wav2Vec2-ED for Encoder-Decoder) and fine-tuning all weights on the ASR task with 360 hours of Librispeech audio. This experiment evaluates the set-up of Gandelsman et al. (2022) but omits freezing the encoder at train time.

4. Multi-task training Wav2Vec2-ED on both the self-supervised pre-training objective and CTC objective for ASR on 360 hours of Librispeech audio and then applying TTT. We use a non-weighted sum of the loss terms. This set-up matches that used in Sun et al. (2020).

5. Freezing Wav2Vec2-ED's encoder at train time and fine-tuning on the ASR task with 360 hours of Librispeech audio, and then applying TTT (with an unfrozen encoder at test time). This set-up matches that used in Gandelsman et al. (2022).

Table 1 shows our results for these experiments. Figures 3 and 4 plot TTT gains for specific settings,

while Figure 5 shows qualitative results for specific transcriptions. Finally, Figure 6 visualizes TTT gains for four of our models, averaged across all three distributions.

4.1 TTT doesn't work on off-the-shelf Wav2Vec2

We first observe that applying TTT directly to Wav2Vec2-960h is not effective, increasing WER on 2 of our 3 distributions. This result was expected: because Wav2Vec2-960h was fine-tuned with CTC loss on the downstream task, taking gradient steps with the SSL objective at test-time will only confuse the CTC decoder (linear head), which has learned to rely on recognition-only features rather than SSL-features.

4.2 Approach inspired by Gandelsman et al. (2022) achieves large gains

Next, we observe that Wav2Vec2-ED + TTT with the encoder frozen at train time results in the best WER reductions of all our TTT experiments. The WER on held-out test data decreases by 1.9% on average, comparable to the 2.1% reduction achieved by fine-tuning. Thus, TTT (largely) **overcomes the need for test labels**. The gains on seen test samples are even more impressive: Figure 2 visualizes the WER of samples before and after running TTT on them. We see an average WER reduction of 5% and individual reductions as large as 20% on these samples. Note that the Ello application runs on a tablet so TTT must be performed offline; thus, the held-out results shown in Table 1 represent a more practical setting than those in Figure 2.

4.3 Our Custom Decoder Hurts Performance

Despite the nice improvements from TTT, it is clear that the WERs of Wav2Vec2-ED + TTT are dramatically worse than those of our Wav2Vec2-960h baselines. **The performance drop from our decoder outweighs the gains of TTT.** This performance drop occurs because 1) our decoder's performance was hindered by our limited compute and 2) the requirement of Gandelsman et al. (2022)'s approach that the encoder is frozen at train time limits the expressivity of the model. Removing this requirement (third column of Table 1), we observe that the WERs improve by a large amount but TTT begins to harm performance, as expected.

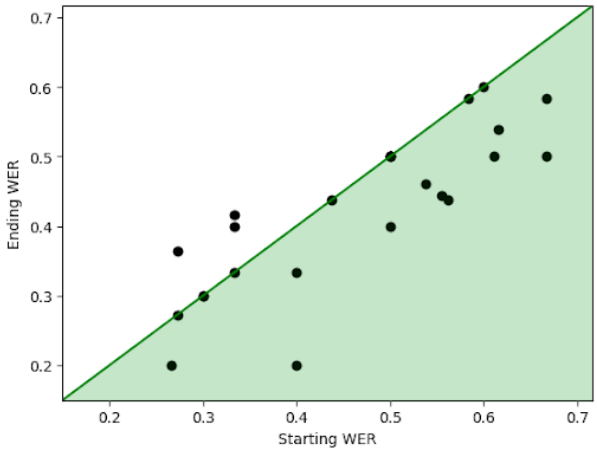

Figure 2: Single sample ending WER (TTT) vs. starting WER (no TTT) on OOD YouTube speaker

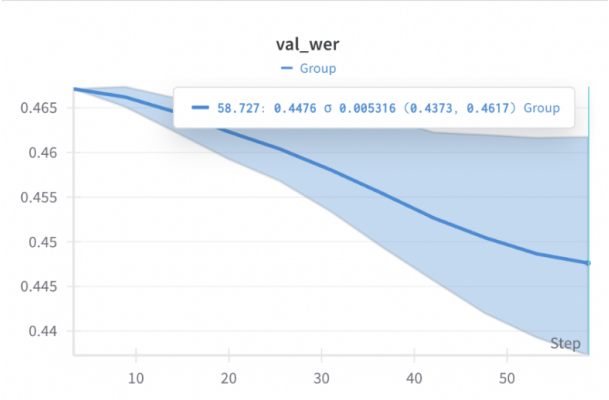

Figure 3: Single sample TTT validation WER (with min and max shaded) from education distribution.

4.4 Multitask Training Partially Recovers Losses from Decoder

Finally, we explore whether we can overcome the performance drop from our decoder by using Sun et al. (2020)'s multi-task TTT approach. Unfreezing the encoder and training it with both the SSL and ASR tasks results in an improvement over Gandelsman et al. (2022)'s in WER. TTT remains beneficial in this setting, albeit to a lesser extent than that of the Gandelsman et al. (2022) approach.

5 Conclusion

For RQ1 we found that training a transformer-based ASR decoder over a frozen Wav2Vec2-base encoder at train time and then adapting the encoder with its self-supervised objective over OOD test samples at test time resulted in the largest WER reductions. However, due to the frozen encoder and custom decoder, this method suffered a significant baseline performance drop. Thus, in response

	Wav2Vec2-960h + FT	Wav2Vec2-960h + TTT	Wav2Vec2-ED + TTT	Wav2Vec2-ED + Multitask FT + TTT	Wav2Vec2-ED + TTT
Train-time Encoder Frozen?	✗	✗	✗	✗	✓
Education	24.1 (↓ 2.1)	26.5 (↑ 0.3)	36.2 (↑ 1.3)	41.8 (↓ 1.1)	56.6 (↓ **2.6**)
Gaming	31.1 (↓ 0.1)	29.3 (↓ 0.9)	40.5 (↑ 0.1)	46.8 (↓ 0.3)	56.5 (↓ **1.4**)
Science/Tech.	22.7 (↓ **4.2**)	27.2 (↑ 0.3)	38.2 (↓ 0.1)	47.8 (↓ 0.5)	52.6 (↓ 1.6)
Average	26.0 (↓ **2.1**)	27.2 (↓ 0.1)	38.3 (↑ 0.4)	45.5 (↓ 0.7)	55.2 (↓ 1.9)

Table 1: WERs on Gigaspeech Youtube test distributions. The ↓ / ↑ values indicate the change in WER after 40 steps of TTT (except in the case of the finetuning baseline in column 1, where it refers to 100 steps of finetuning.) The absolute best WERs are underlined and the best adaption improvements are **bolded.**

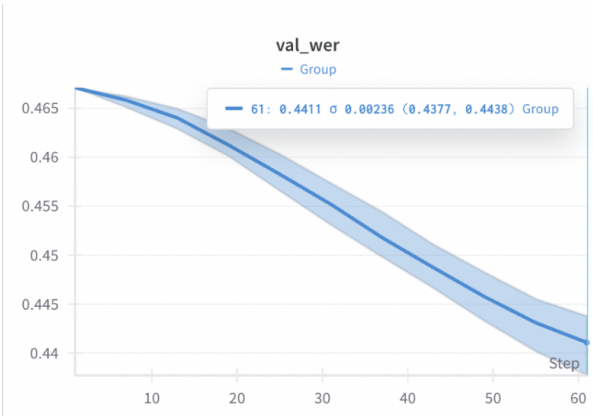

Figure 4: Multiple sample TTT validation WER (with min and max shaded) from education distribution.

to RQ2, we found that with limited compute, we could obtain better overall performance on OOD test samples by unfreezing the encoder and training end-to-end on the train distribution. However, with more compute and/or a pretrained encoder decoder model we believe this method is still promising. Lastly, in response to RQ3, we observed that TTT gains were on par with finetuning gains. Thus, TTT appears promising for ASR systems working with unlabeled OOD test data. Overall, we recommend that Ello considers exploring TTT approaches that do not require a decoder such as Lin et al. (2022), which are likely to achieve comparable gains without the complications of architecture modifications.

6 Acknowledgements and Contributions

We would like to thank Joe Lou and Henry Zhou (Ello, 2024) for advising us on this project, meeting with us on a weekly basis, and providing a GPU for training. Their guidance was integral to us obtaining promising results.

Rajan Vivek completed the majority of the literature review and proposed using TTT with self-supervision for this project. Matt Harvill performed the majority of the dataset exploration. Both team members contributed equally to the development of successful and unsuccessful approaches, model training, model evaluation, and completion of written course deliverables.

References

Alexei Baevski, Henry Zhou, Abdelrahman Mohamed, and Michael Auli. 2020. wav2vec 2.0: A framework for self-supervised learning of speech representations.

Murali Karthick Baskar, Tim Herzig, Diana Nguyen, Mireia Diez, Tim Polzehl, Lukáš Burget, and Jan "Honza" Černocký. 2022. Speaker adaptation for wav2vec2 based dysarthric asr.

Sri Harsha Dumpala, Chandramouli Shama Sastry, and Sageev Oore. 2023. Test-time training for speech. In *Workshop on Efficient Systems for Foundation Models @ ICML2023*.

Ello. 2024. Read with ello.

Yossi Gandelsman, Yu Sun, Xinlei Chen, and Alexei A Efros. 2022. Test-time training with masked autoencoders. In *Advances in Neural Information Processing Systems*.

Taesik Gong, Jongheon Jeong, Taewon Kim, Yewon Kim, Jinwoo Shin, and Sung-Ju Lee. 2023. Note: Robust continual test-time adaptation against temporal correlation.

Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. 2021. Hubert: Self-supervised speech representation learning by masked prediction of hidden units. *CoRR*, abs/2106.07447.

Rishabh Jain, Andrei Barcovschi, Mariam Yiwere, Peter Corcoran, and Horia Cucu. 2023. Adaptation of whisper models to child speech recognition.

Guan-Ting Lin, Shang-Wen Li, and Hung yi Lee. 2022. Listen, adapt, better wer: Source-free single-utterance test-time adaptation for automatic speech recognition.

Yuejiang Liu, Parth Kothari, Bastien Germain van Delft, Baptiste Bellot-Gurlet, Taylor Mordan, and Alexandre Alahi. 2021. TTT++: When does self-supervised test-time training fail or thrive? In *Advances in Neural Information Processing Systems*.

Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. 2022. Robust speech recognition via large-scale weak supervision.

David Snyder, Daniel Garcia-Romero, Gregory Sell, Daniel Povey, and Sanjeev Khudanpur. 2018. X-vectors: Robust dnn embeddings for speaker recognition. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5329–5333.

Yu Sun, Xiaolong Wang, Zhuang Liu, John Miller, Alexei A. Efros, and Moritz Hardt. 2020. Test-time training with self-supervision for generalization under distribution shifts.

Elad Ben Zaken, Shauli Ravfogel, and Yoav Goldberg. 2022. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models.

Appendix

For all TTT settings, we use 40 steps of TTT with an SGD optimizer and learning rate of 3E-4. These were identified through various hyperparameter sweeps, though we note that further hyperparameter tuning may result in increased performance.

Transcription:	YOU KNEW EXACTLY WHAT OUR SCHOOL NEEDED IN A LEADER AND YOU MADE SURE WE GOT IT
Base Model Prediction (WER: 23.5%):	YOU KNOW EXACTLY WHAT OUR SHOOL NEED IN A LEADER AND YOU MADE SURE WE GOT ERE
TTT Model Prediction (WER: 5.9%):	YOU KNEW EXACTLY WHAT OUR SCHOOL NEEDED IN A LEADER AND YOU MADE SURE WE GOT ER
Transcription:	THAT IS A TESTAMENT TO YOU THAT YOU HAVE HIRED WELL
Base Model Prediction (WER: 27.3%):	THAT IS A CESTIMENT TO YOU THAT YOU HAVE HIRE DWELL
TTT Model Prediction (WER: 9.1%):	THAT IS A TESTAMENT TO YOU THAT YOU HAVE HIRED WELLE
Transcription:	AND I BELIEVE WE HAVE A COUPLE OF PRINCIPALS WHO WOULD LIKE TO SPEAK AS WELL
Base Model Prediction (WER: 18.8%):	AND I BELIEVE WE HAVE A COUPLE OF PRINCIPALS WHO OLD LIKE TO SPEAK HIS WELL E
TTT Model Prediction (WER: 6.2%):	AND I BELIEVE WE HAVE A COUPLE OF PRINCIPALS WHO LIKE TO SPEAK AS WELL
Transcription:	RELATED TO SCHOOLS SPECIFICALLY THE WEARING OF MASKS BY STUDENTS
Base Model Prediction (WER: 20.0%):	RELATED TO SCHOOLS SPIIFICLY THE WEARING OF MASK BY STUDENTS
TTT Model Prediction (WER: 0.0%):	RELATED TO SCHOOLS SPECIFICALLY THE WEARING OF MASKS BY STUDENTS
Transcription:	THAT IS A CONCERN MOVING FORWARD DUE TO THE EXCESSIVE TIME ON SCREENS
Base Model Prediction (WER: 30.8%):	THAT IS THE CONCERN MOVING FORWARD DU TO TH EXCESSIVE TIME ON SRIA
TTT Model Prediction (WER: 7.7%):	THAT IS A CONCERN MOVING FORWARD DUE TO THE EXCESSIVE TIME ON SRIA

Figure 5: Five examples of our Wav2Vec2-ED model improving after 40 TTT steps

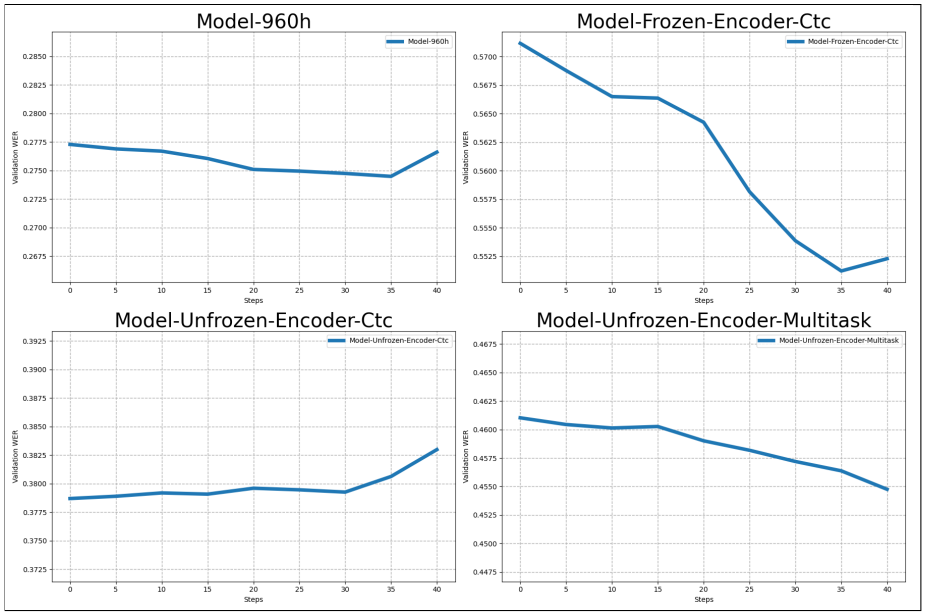

Figure 6: Comparing TTT effectiveness across models with Y-axis scale (WER) kept constant.