# TTS4PRETRAIN 2.0: ADVANCING THE USE OF TEXT AND SPEECH IN ASR PRETRAINING WITH CONSISTENCY AND CONTRASTIVE LOSSES

*Zhehuai Chen, Yu Zhang, Andrew Rosenberg, Bhuvana Ramabhadran, Pedro Moreno, Gary Wang*

Google, Inc.

## ABSTRACT

An effective way to learn representations from untranscribed speech and unspoken text with linguistic/lexical representations derived from synthesized speech was introduced in *tts4pretrain* [1]. However, the representations learned from synthesized and real speech are likely to be different, potentially limiting the improvements from incorporating unspoken text. In this paper, we introduce learning from supervised speech earlier on in the training process with consistency-based regularization between real and synthesized speech. This allows for better learning of shared speech and text representations. Thus, we introduce a new objective, with encoder and decoder consistency and contrastive regularization between real and synthesized speech derived from the labeled corpora during the pretraining stage. We show that the new objective leads to more similar representations derived from speech and text that help downstream ASR. The proposed pretraining method yields Word Error Rate (WER) reductions of 7-21% relative on six public corpora, Librispeech, AMI, TEDLIUM, Common Voice, Switchboard, CHiME-6, over a state-of-the-art baseline pretrained with wav2vec2.0 and 2-17% over the previously proposed *tts4pretrain*. The proposed method outperforms the supervised SpeechStew by up to 17%. Moreover, we show that the proposed method also yields WER reductions on larger data sets by evaluating on a large resource, in-house Voice Search task and streaming ASR.

***Index Terms***— Speech Recognition, Speech Synthesis, Self-supervised, Consistency Regularization

## 1. INTRODUCTION

State of the art automatic speech recognition (ASR) performance has been achieved using multi-stage training with pretraining techniques such as wav2vec, wav2vec 2.0[2, 3], w2v-bert [4], *tts4pretrain* [5] followed by subsequent fine-tuning. The typical structure of this training involves training the encoder of an end-to-end ASR model with unsupervised speech using self-supervised criteria (e.g., contrastive losses) followed by subsequent fine-tuning on transcribed speech using a supervised objective such as RNNT, CTC or HAT [6].

In our previous work, *tts4pretrain* demonstrated a mechanism to inject unspoken text into the first stage of training (pretraining). However, one of the limitations of this work was that the pretraining criteria does not explicitly encourage the encoder to generalize from synthetic speech to real speech. Central to this work are modifications to the objective function applied during pretraining to promote learning from synthetic to real speech to improve to fine-tuning.

Consistency regularization [7, 8, 9] has been used to provide improvements with data augmentation in ASR training. Building on this result, we investigate the utility of consistency regularization during pretraining with unspoken text by training on synthesized and real speech. Since this requires transcribed speech to calculate consistency between corresponding real and synthetic utterances, we are motivated to bring supervised data into the pretraining process.

Under the typical pretraining/fine-tuning process, pretraining uses unsupervised data only and fine-tuning uses supervised data only. The proposed improvements to *tts4pretrain* break this relationship, bringing supervised data into the pretraining process. A similar concept albeit in the fine-tuning stage was explored in SpeechStew [10] which employed two pretraining stages, a first using unsupervised data, the second using out-of-domain supervised data to generate the best performance on CHiME-6. Moreover, by including supervised data during the pretraining stage and including an auxiliary decoder in the pretrained model, we are able to evaluate the performance of the model prior to fine-tuning.

The novel contributions of this paper are:

- Incorporation of supervised training into pre-training allowing the model to generalize from both supervised and unsupervised speech and text.
- Use of consistency regularization and self-supervised contrastive loss to learn shared speech and text representations.
- Investigation of the relationship between pretraining loss functions and final fine-tuned model performance.

## 2. RELATED WORK

Self-supervised pretraining techniques such as wav2vec2.0 [3], Contrastive Predicting Coding (CPC) [11, 12], Autoregressive Perdictive Coding (APC) [13] and their variants leverage untranscribed speech to improve ASR. There has been growing interest in learning joint speech and text representations [14, 15, 1]. SPLAT [14], aimed at the spoken language understanding task, introduces an alignment (conistency) loss at the sequence and token levels to better align the representations from speech and text in a shared latent semantic space. In [1] an alternate approach utilizes speech synthesis to inject lexical and phonetic information derived from text into the speech encoder. Combining objectives derived from supervised and unsupervised data is widely used in semi-supervised learning literature, e.g. [16]. Contemporaneous with this work, [17] extends the idea to self-supervised pretraining with the motivation to simplify the training procedure, and similarly [18] proposes a single stage training procedure combining both self-supervised and supervised training.

Recent approaches that introduce consistency regularization at the decoder and encoder stages have shown varied degrees of success. These include techniques such as: Unsupervised Data Augmentation (UDA) [19] that regularize model predictions to be invariant to small perturbations of input or hidden states; Consistent Data Augmentation (CODA) [7] that introduced *decoder* consistency regularization to promote better generalization from synthetic to real speech; and SimCLR [8] and SimSiam [9] that enforce *encoder*

Thanks to Fadi Biadsy, Yinghui Huang, Jesse Emond and Murali Karthick Baskar for many discussions and infrastructure related assistance.

consistency. In this paper, we draw upon these mechanisms to bring together representations learned from speech and text.

## 3. PROPOSED METHOD

*Tts4pretrain 2.0* builds on *tts4pretrain* [1] (Section 3.1) by introducing two new loss components along with the data required to optimize them. These include $\mathcal{L}_{paired}$, a loss term based on supervised data and $\mathcal{L}_{cons}$ consistency regularization (Section 3.2).

$$\begin{aligned}\mathcal{L}_{tts4pretrain2} =& \mathcal{L}_{tts4pretrain}(\mathcal{X}_{unsup} \cup \mathcal{X}_{text}) + \\ & \lambda_1 \mathcal{L}_{paired}(\mathcal{X}_{sup}) + \lambda_2 \mathcal{L}_{cons}(\mathcal{X}_{sup})\end{aligned} \tag{1}$$

where the datasets $\mathcal{X}_{unsup}$ includes untranscribed speech , $\mathcal{X}_{sup}$ contains transcribed speech, and $\mathcal{X}_{text}$ includes unspoken text. They are combined with $\lambda_1$=1.0, $\lambda_2$=0.1. Figure 1 summarizes the proposed method.

The central idea of this work is to leverage supervised speech to enforce shared speech and text representations. Firstly, we include self-supervised and supervised losses from supervised speech $\mathcal{X}_{sup}$ similar to self-supervised and supervised joint training [17, 18].

$$\begin{aligned}\mathcal{L}_{paired} = \mathcal{L}_{\mathtt{w2v}}(x \mid \theta_e) + \mathcal{L}_{\mathtt{aux}}(y|x, \theta_e, \theta_d) \\ (x, y) \in \mathcal{X}_{sup}\end{aligned} \tag{2}$$

Moreover, we leverage data augmentation to enforce consistency between original encoder activation $\mathbf{e}_x$ and the activation from augmented speech $\mathbf{e}_{x^*}$ in $\mathcal{L}_{cons}$, where $x^* = \mathtt{Aug}(x, y)$ comes from data augmentation. When using TTS as augmentation, this encourages the encoder to produce similar representations on from real and synthesized speech.

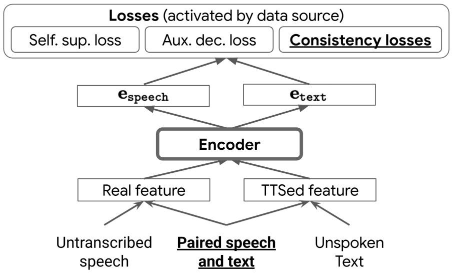

**Fig. 1**. Tts4pretrain 2.0 joint speech and text pretraining architecture. Losses are activated and deactivated following Equation 1.

### 3.1. TTS4Pretrain [1]

*Tts4pretrain* introduces unspoken text into speech-only self-supervised pretraining by introducing two additional components. First, it uses synthesized utterances from the unspoken text along with the untranscribed speech utterances during pretraining, training with a standard wav2vec 2.0 self-supervised criteria. Second, since unspoken text necessarily includes a ground truth transcript, an auxiliary ASR-based loss is included for the synthesized inputs.

$$\begin{aligned}\mathcal{L}_{tts4pretrain} = & (1 - \delta) \cdot \mathcal{L}_{\mathtt{w2v}}(x \mid \theta_e) + \\ & \delta \cdot \left(\mathcal{L}_{\mathtt{w2v}}(x^* \mid \theta_e) + \mathcal{L}_{\mathtt{aux}}(y|x^*, \theta_e, \theta_d)\right) \\ & x \in \mathcal{X}_{unsup}, x^* = \mathtt{TTS}(y, z),\ y \in \mathcal{X}_{text}, z \sim Z.\end{aligned} \tag{3}$$

The acoustic input $x$ is drawn from untranscribed speech corpus $\mathcal{X}_{unsup}$ and synthetic utterances $x^*$ is generated via speech synthesis (TTS) of text $y$ drawn from an unspoken text dataset $\mathcal{X}_{text}$. The TTS model is conditioned on $z \sim Z$, a speaker value and a VAE-based latent variable for prosodic control [1] (cf. Section 5). The pretraining batch generation process samples new speaker and prosody conditioning values each time an unspoken text utterance is observed during training resulting in diverse synthesized utterances on subsequent observations. Each batch contains both synthetic and real utterances. The loss contributions are masked using a $\delta$ indicator so losses are calculated for the appropriate batch elements.

### 3.2. Decoder Consistency Regularization

Decoder consistency regularization operates on hypothesized labels from the ASR model. Consistent Data Augmentation (CoDA) [7] uses Kullback–Leibler divergence (KL) as a consistency loss on augmented outputs. The CoDA formulation is as follows:

$$\mathcal{L}_{\mathtt{cons}} = \mathcal{D}_{\mathrm{KL}}\left(p_\theta(y|x) || p_\theta(y|x^*)\right). \tag{4}$$

The use of consistency regularization was shown in [7] to substantially improve ASR training from synthetic speech. Here we apply it to the auxiliary decoders of *tts4pretrain* to implicitly enforce consistent encoder representations.

Another option to encourage consistency under augmentation is to enforce the *encoder* representations learned from speech and text consistent with each other similar to SimClr [8, 20]. Decoder consistency is more related to the final ASR obective, and by operating at the label-level, it can handle unaligned augmentation, like TTS or time-warping. However, encoder consistency does not require hypothesized labels, allowing it to be applied to all the data $\mathfrak{X}_{sup}, \mathfrak{X}_{unsup}, \mathfrak{X}_{text}$. We compare decoder consistency with a novel encoder consistency measure in Section 6.2.

**Table 1**. Description of supervised speech and unspoken text in various corpora. For unsupervised speech, we always use 60K Libri-Light for public corpora and 1.7M Youtube for internal corpus.

| Corpus | Sup. Audio | Unspoken Text Composition | (utts) |
|---|---|---|---|
| Librispeech | 96h | Libri text [21, 22] | 40M |
| TEDLIUM | 452h | TED text [23] | 14M |
| AMI | 100h | SpeechStew labels + Libri text | 43.5M |
| CommonVoice | 1500h | SpeechStew + TED + Libri text | 57.5M |
| Swb/Fisher | 2000h | SpeechStew + TED + Libri text | 57.5M |
| CHiME-6 | 40h | SpeechStew + TED + Libri text | 57.5M |
| Marathi | 16kh | Typed search query | 20M |

## 4. DATA

**ASR:** The supervised and unsupervised speech data used in this paper include publicly available, well-benchmarked corpora and an in-house corpus detailed in Table 1. The first six rows correspond to the public corpora included in SpeechStew [10]: LibriSpeech [21], LibriLight [22], AMI [24], TEDLIUM [25], Common Voice [26], Switchboard/Fisher [27] and CHiME-6 [28]. Each row in Table 1 details the amount of untranscribed speech and unspoken text avaialble for each corpora. The last row refers to the in-house data sets. The supervised audio comprising 16K hours is derived from in-house data sets representative of Google's voice search (VS) traffic in Marathi. This corpus is derived from voice search utterances that are anonymized and hand-transcribed. The unsupervised audio is derived from YouTube videos in Marathi and Hindi. The unspoken text used in pretraining, comprises of anonymized and aggregated, typed search query data. These text queries were selected from a

**Table 2**. Performance on 5 public corpora. The "matched" finetune *tts4pretrain* 2.0 results (row 3) are pretrained using supervised data from only the corresponding finetune corpus. The SpeechStew finetune results (row 5) use all supervised data from SpeechStew during pretraining.

| Finetune | | Librispeech | | AMI | | TED | Swb/Fisher | | Common |
|---|---|---|---|---|---|---|---|---|---|
| corpora | Method | *clean* | *other* | *ihm* | *sdm* | | *swb* | *callhm* | Voice |
| Matched | Conformer XL + Libri-Light w2v2 | 1.7 | 3.5 | 11.1 | 25.1 | 6.3 | 5.2 | 13.0 | 9.1 |
| | + tts4pretrain | 1.6 | 3.2 | 10.5 | 24.6 | 6.1 | 5.1 | 12.8 | 8.9 |
| | **+ tts4pretrain 2.0** | **1.6** | **3.1** | **10.1** | **23.8** | **5.8** | **4.9** | **11.1** | **8.5** |
| SpeechStew [10] | Conformer XL + Libri-Light w2v2 | 1.7 | 3.3 | 9.6 | 23.8 | 5.7 | 4.9 | 10.8 | 8.5 |
| | **+ tts4pretrain 2.0** | 1.7 | **3.2** | **8.7** | **21.9** | **5.1** | **4.5** | **8.0** | **8.4** |

much larger pool of 170M queries using the data selection method described in [5].

**TTS:** We use distinct TTS models for English and Marathi experiments. For English, we train using the public LibriTTS [29] corpus. For Marathi, we train on in-house, 30-hour data set comprising 7 Marathi professional speakers (4 female, 3 male).

## 5. MODEL

**ASR:** The ASR model comprises of a Conformer encoder, an LSTM decoder and a feed forward joint network [30]. The encoder is stack of "conformer blocks". Each block consists of a series of multi-headed self attention, depthwise convolution and feed-forward layers. We use the Conformer XL architecture described in [1] with 24 layers of full-context conformer blocks (600M parameters) for the experiments on public corpora to enable comparisons. The experiments on the Voice Search task use a smaller, streaming Conformer model with 12 layers of left-context conformer blocks, totalling 120M parameters to reduce computational costs and memory usage. All models are trained on 80-dimensional log-mel filter bank coefficients and predict word-piece targets [31]. Pre-training and fine-tuning steps use Adam as the optimizer with the transformer learning rate schedule described in [32]. Following [1], we use a reduced peak learning rate of 4e-4 for *tts4pretrain*. Unless otherwise stated, the unspoken text used for learning text-based representations for each task is given in Table 1.

**TTS:** The TTS model uses a multispeaker Tacotron2 TTS architecture described in [1] with hierarchical VAE [33]. The input sequence is a concatenation of embeddings of X-SAMPA phones and a speaker conditioning value. This embedding is passed to an encoder comprising three convolutional layers with 512 filters of size (5, 1), followed by a bidirectional LSTM layer with 512 units (256 forward, 256 backwards). The encoder outputs are consumed by a 2-layer recurrent decoder using a GMM monotonic attention [34] function. The decoder is followed by a PostNet containing 5 convolutional layers with 512 filters of size (5, 1).

## 6. RESULTS

Section 6.1 summarizes the results on 5 public corpora from SpeechStew. CHiME-6, which is treated as unseen low-resource dataset in SpeechStew [10], is discussed in Section 6.2. We apply the proposed method on a larger-scale in-house dataset in Section 6.3.

### 6.1. SpeechStew Corpora

The first set of experiments in Table 2 treat each of the 5 public corpora separately, denoted as "matched" in the table. For example, we pretrain the conformer XL using wav2vec 2.0 on Libri-Light or *tts4pretrain* on the unspoken text listed in Table 1. We finetune the model using Librispeech only.

We compare *tts4pretrain 2.0* (row 3) with state-of-the-art wav2vec 2.0 (row 1) and the previously proposed *tts4pretrain* [1] (row 2). As in [1], injecting text in pretraining does help train a better speech encoder for ASR which results in 2-7.2% relative word error rate reduction (RER) on top of wav2vec 2.0 (row 1) with the smallest gains on Switchboard and the largest on Librispeech. We attribute this disparity to the amount of additional in-domain unspoken text in each corpus.

The proposed *tts4pretrain 2.0* obtains 1.6-8.6% RER over *tts4pretrain* and overall 7-10.2% RER over wav2vec 2.0 The proposed method achieves improvements on those corpora hardly improved by vanilla *tts4pretrain* such as Switchboard and Common Voice (without additional in-domain unspoken text). This demonstrates that tts4pretrain 2.0 can better leverage out-of-domain text during pretraining. On Librispeech, *tts4pretrain* with LM shallow fusion achieved test/testother WERs of 1.6/3.1 [1]. *Tts4pretrain 2.0* matches this *without* using an LM.

Next, we follow [10] to build the supervised SpeechStew baseline in Row 4, which pretrains the encoder using wav2vec 2.0 on Libri-Light and finetunes the model using all SpeechStew corpora. The proposed *tts4pretrain 2.0* pretrains the model using Libri-Light, unspoken text from Librispeech and TEDLIUM domains, and supervised SpeechStew corpora, followed by fine-tuning on SpeechStew corpora. *Tts4pretrain 2.0* (Row 5) matches or outperforms SpeechStew (Row 4) across all 5 corpora (8 test sets) and reduces WER by up to 17% relatively. Moreover, comparing *tts4pretrain 2.0* in the matched finetune setting (Row 3) with the supervised SpeechStew baseline (Row 4), the overall WER difference is less than 1% relative. This suggests that incorporating in-domain, supervised corpora in pre-training instead of fine-tuning [10] is more effective. This is most pronounced on AMI and TED where the amount of transcribed speech data is reduced by 90%.

### 6.2. CHiME-6 with Ablation Study

CHiME-6 is treated as unseen low-resource dataset in SpeechStew [10] whose domain is very different from both unsupervised Libri-Light, and supervised SpeechStew. We report results with an ablation study in Table 3. All experiments finetune the pretrained encoder on CHiME-6 supervised data. *Tts4pretrain 2.0* also includes CHiME-6 during pretraining. Following [10], the SpeechStew pretrain setting (the last row) conducts Libri-Light wav2vec 2.0 pretraining, followed by two-stage supervised finetune on SpeechStew and CHiME-6.

The best *tts4pretrain 2.0* setting (Row 6) achieves significant gains over speech-only pretrain (row 1) and *tts4pretrain* (row 2). [35] found that mismatch between pretrain and finetune domains can degrade ASR performance. Our result shows that *tts4pretrain 2.0* is able to reduce this degradation. This result also outperforms SpeechStew that uses more supervised data (last row).

We also explore the use of *encoder* consistency measures for CHiME-6. We propose Hierarchical Contrastive Consistency Regularization (HCCR). The structure of HCCR is similar to SimCLR or SimSiam, wherein encoder activations from original and augmented speech ($\mathbf{e}$ and $\mathbf{e}^*$) are projected through an auxiliary network to generate $\mathbf{z}$ and $\mathbf{z}^*$. Positive and negative pairs are then constructed and a contrastive loss is calculated. Specific to HCCR, we use a CNN projection network,calculate projections over increasing-length segments of $\mathbf{e}$ (30, 60, 120ms) yielding 3 views, $V$, and draw negative examples from the same utterance for short segments, and from other utterances in the batches with 120ms segments.

$$l_{t,\mathbf{z},\mathbf{z}^*} = -\log \frac{\exp(\text{sim}(z_t^*, z_t)/\tau)}{\sum_{k=1}^{T}\exp(\text{sim}(z_t^*, z_k)/\tau)} \tag{5}$$

$$\mathcal{L}_{\texttt{enc\_cons}} = \sum_{v=1}^{V}\sum_{t=1}^{T^{(v)}} l_{t,\mathbf{z}^{*(v)},\mathbf{z}^{(v)}} \tag{6}$$

HCCR loss Equation 6 is calculated over paired, unpaired and synthesized speech. It is added to Equation 1 with a coefficient of $1e$-3. We call this configuration *tts4pretrain 2.1* in Table 3 finding encoder consistency to be additive with *tts4pretrain 2.0* on this challenging task.

**Table 3**. Performance (WER) on CHiME-6 with an ablation study of consistency regularization contributions.

| Pretrain | $\mathcal{L}_{*\texttt{\_cons}}$ | $\mathcal{L}_{paired}$ | CHiME-6 *eval* |
|---|---|---|---|
| w2v2 | n/a | n/a | 49.0 |
| tts4pretrain | n/a | n/a | 47.3 |
| tts4pretrain 2.0 | × | ✓ | 47.0 |
|  | CoDA | ✓ | 43.2 |
|  | RNNT loss | ✓ | 41.2 |
|  | + CoDA | ✓ | 40.1 |
| tts4pretrain 2.1 | + HCCR | ✓ | **39.1** |
| SpeechStew | n/a | n/a | 41.5 |

Figure 2 visualizes the encoder representations of TTS and real speech using t-SNE [36]. After introducing consistency regularization in *tts4pretrain 2.0*, the speech and text representations (red and blue) stay closer to each other compared to those from vanilla *tts4pretrain* (teal and magenta). The shorter distance in the projected space suggests that *tts4pretrain 2.0* is effectively generating shared speech and text representations.

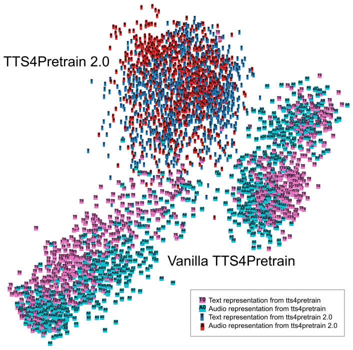

**Fig. 2**. t-SNE of speech and text encoder representations of vanilla *tts4pretrain* and *tts4pretrain 2.0* .

### 6.3. Voice Search

Table 4 shows the performance of *tts4pretrain 2.0* on the in-house Marathi Voice Search task. These experiments use a streaming (left-context) conformer encoder rather than a full-context encoder. Thus, we replace the full-context Conformer blocks with left-context blocks and replace their batch normalization with group normalization [37]. We also use the wav2vec 1.0 self-supervised training rather than wav2vec 2.0. We observe that *tts4pretrain* and *tts4pretrain 2.0* still provide improvements, even when trained on 8-160x more supervised data and 28x more unsupervised data.

**Table 4**. Voice Search Results on Streaming Marathi Models

| Method | VS |
|---|---|
| Baseline conformer XL | 22.4 |
| + Speech pretrain | 21.9 |
| + tts4pretrain | 21.7 |
| **+ tts4pretrain 2.0** | 21.5 |

### 6.4. Monitoring pretraining through auxiliary decoders

One challenge for pretraining is that the optimized self-supervised loss is a weak predictor of fine-tuned performance. Nevertheless, *tts4pretrain 2.0* provides informative pretraining signals. Figure 3 shows the correlation between auxiliary decoder performance in pretrain (yellow, blue and red in the figure) and the final finetune results (green). The RNNT loss on real audio and pretrain auxiliary decoder WER both have high correlation with the finetune result whose minima are all at 250k steps. This suggests either of these can be a good metric to monitor pretraining for early stopping. Moreover, we have used the divergence between RNNT loss on real and synthesized speech to identify overfitting to synthetic data. This can be an indicator of a lack of generalization from synthetic to real speech.

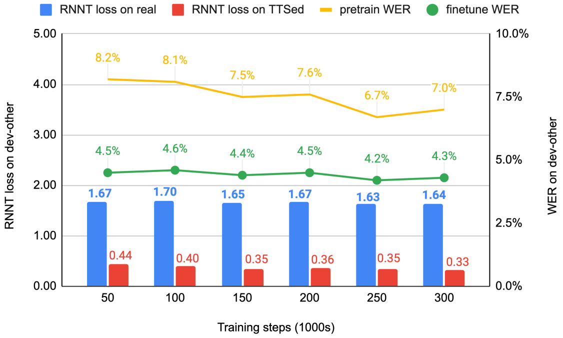

**Fig. 3**. Monitoring pretraining using auxiliary decoder performance.

## 7. CONCLUSION

In this paper we present *tts4pretrain 2.0*. *Tts4pretrain* incorporated two central components: self-supervised pretraining using synthetic speech, and the introduction of one or more auxiliary ASR decoders during pretraining. *Tts4pretrain 2.0* introduces the use of supervised data during pretraining and auxiliary decoder consistency regularization resulting in a **17% relative reduction in WER** on SpeechStew [10]. We introduce encoder consistency regularization (*tts4pretrain 2.1*) and demonstrate additional wins on the CHiME-6 task. Finally, we show that the proposed method is effective in a production setting with nearly 10x more supervised training data.

## 8. REFERENCES

[1] Zhehuai Chen, Yu Zhang, Andrew Rosenberg, Bhuvana Ramabhadran, Gary Wang, and Pedro Moreno, "Injecting Text in Self-Supervised Speech Pretraining," *arXiv preprint arXiv:2108.12226*, 2021.

[2] Steffen Schneider, Alexei Baevski, Ronan Collobert, and Michael Auli, "wav2vec: Unsupervised pre-training for speech recognition," *arXiv preprint arXiv:1904.05862*, 2019.

[3] Alexei Baevski et al., "wav2vec 2.0: A framework for self-supervised learning of speech representations," *arXiv preprint arXiv:2006.11477*, 2020.

[4] Yu-An Chung et al., "W2v-BERT: Combining Contrastive Learning and Masked Language Modeling for Self-Supervised Speech Pre-Training," *arXiv preprint arXiv:2108.06209*, 2021.

[5] Zhehuai Chen et al., "Semi-Supervision in ASR: Sequential MixMatch and Factorized TTS-Based Augmentation," in *Interspeech*, 2021.

[6] Ehsan Variani, David Rybach, Cyril Allauzen, and Michael Riley, "Hybrid autoregressive transducer (HAT)," in *IEEE ICASSP*. IEEE, 2020, pp. 6139–6143.

[7] Gary Wang, Andrew Rosenberg, Zhehuai Chen, Yu Zhang, Bhuvana Ramabhadran, Yonghui Wu, and Pedro Moreno, "Improving Speech Recognition Using Consistent Predictions on Synthesized Speech," in *ICASSP*. IEEE, 2020.

[8] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton, "A Simple Framework for Contrastive Learning of Visual Representations," 2020.

[9] Xinlei Chen and Kaiming He, "Exploring simple siamese representation learning," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

[10] William Chan et al., "SpeechStew: Simply mix all available speech recognition data to train one large neural network," *arXiv preprint arXiv:2104.02133*, 2021.

[11] Luyu Wang, Kazuya Kawakami, and Aäron van den Oord, "Contrastive Predictive Coding of Audio with an Adversary.," in *INTERSPEECH*, 2020, pp. 826–830.

[12] Chaitanya Talnikar, Tatiana Likhomanenko, Ronan Collobert, and Gabriel Synnaeve, "Joint masked cpc and ctc training for asr," in *IEEE ICASSP*, 2021, pp. 3045–3049.

[13] Yu-An Chung and James Glass, "Generative pre-training for speech with autoregressive predictive coding," in *IEEE ICASSP*, 2020, pp. 3497–3501.

[14] Yu-An Chung, Chenguang Zhu, and Michael Zeng, "SPLAT: Speech-Language Joint Pre-Training for Spoken Language Understanding," *arXiv preprint arXiv:2010.02295*, 2020.

[15] Renjie Zheng et al., "Fused acoustic and text encoding for multimodal bilingual pretraining and speech translation," *arXiv preprint arXiv:2102.05766*, 2021.

[16] Daniel S Park et al., "Improved Noisy Student Training for Automatic Speech Recognition," *arXiv preprint arXiv:2005.09629*, 2020.

[17] Dongseong Hwang et al., "Large-scale ASR Domain Adaptation using Self- and Semi-supervised Learning," 2021.

[18] Junwen Bai, Bo Li, Yu Zhang, Ankur Bapna, and Tara N. Sainath, "Joint Unsupervised and Supervised Training for Multilingual ASR," 2022.

[19] Qizhe Xie, Zihang Dai, Eduard Hovy, Minh-Thang Luong, and Quoc V Le, "Unsupervised data augmentation for consistency training," *arXiv preprint arXiv:1904.12848*, 2019.

[20] Dongwei Jiang et al., "Speech SIMCLR: Combining Contrastive and Reconstruction Objective for Self-supervised Speech Representation Learning," 2020.

[21] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur, "Librispeech: an asr corpus based on public domain audio books," in *2015 ICASSP*. IEEE, 2015.

[22] Jacob Kahn, Morgane Rivière, Weiyi Zheng, et al., "Libri-light: A benchmark for asr with limited or no supervision," in *IEEE ICASSP*, 2020, pp. 7669–7673.

[23] Daniel Povey, , et al., "The Kaldi speech recognition toolkit," in *IEEE 2011 workshop on automatic speech recognition and understanding*, 2011.

[24] Jean Carletta et al., "The AMI meeting corpus: A pre-announcement," in *International workshop on machine learning for multimodal interaction*. Springer, 2005.

[25] Anthony Rousseau, Paul Deléglise, and Yannick Esteve, "TED-LIUM: an Automatic Speech Recognition dedicated corpus.," in *LREC*, 2012, pp. 125–129.

[26] Rosana Ardila et al., "Common voice: A massively-multilingual speech corpus," *arXiv preprint arXiv:1912.06670*, 2019.

[27] Christopher Cieri, David Miller, and Kevin Walker, "From Switchboard to Fisher: Telephone collection protocols, their uses and yields," in *Eurospeech*, 2003.

[28] Shinji Watanabe et al., "CHiME-6 challenge: Tackling multispeaker speech recognition for unsegmented recordings," *arXiv preprint arXiv:2004.09249*, 2020.

[29] Heiga Zen et al., "LibriTTS: A corpus derived from LibriSpeech for text-to-speech," *arXiv preprint arXiv:1904.02882*, 2019.

[30] Anmol Gulati, James Qin, Chung-Cheng Chiu, et al., "Conformer: Convolution-augmented transformer for speech recognition," *arXiv preprint arXiv:2005.08100*, 2020.

[31] Taku Kudo and John Richardson, "Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing," *arXiv preprint arXiv:1808.06226*, 2018.

[32] Yu Zhang et al., "Pushing the Limits of Semi-Supervised Learning for Automatic Speech Recognition," *arXiv preprint arXiv:2010.10504*, 2020.

[33] Wei-Ning Hsu et al., "Hierarchical generative modeling for controllable speech synthesis," *arXiv preprint arXiv:1810.07217*, 2018.

[34] Alex Graves, "Generating sequences with recurrent neural networks," *arXiv preprint arXiv:1308.0850*, 2013.

[35] Wei-Ning Hsu et al., "Robust wav2vec 2.0: Analyzing Domain Shift in Self-Supervised Pre-Training," *arXiv preprint arXiv:2104.01027*, 2021.

[36] Laurens Van der Maaten and Geoffrey Hinton, "Visualizing data using t-SNE.," *Journal of machine learning research*, vol. 9, no. 11, 2008.

[37] Yuxin Wu and Kaiming He, "Group normalization," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 3–19.