Neural Language Representations and Scaling Semi-Supervised Learning for Speech Recognition

by

Charles Peyser

A dissertation submitted in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

Department of Computer Science

New York University

May 2024

Professor Kyunghyun Cho

Professor Michael Picheny

Abstract

Speech recognition research has been focused for several years on the incorporation of unpaired speech and text data alongside conventional supervised datasets. Dominant methods have emphasized auxiliary tasks for refining speech and/or text representations during model training. These methods have generally performed strongly when paired with very small supervised datasets, but do not yield the same improvements against strong, supervised baselines.

We argue in this thesis that the path to scaling these methods lies in the speech and text representations themselves. We investigate statistical properties of these representations, and show that downstream ASR performance corresponds to a model's ability to jointly represent speech and text. We analyze existing methods for semisupervised ASR, and develop an algorithm to improve them at scale by aligning speech and text in representation space.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

The introduction of deep learning into speech recognition was made possible by the existence of large, paired speech datasets Hinton et al. (2012) used for conventional supervised learning. Recent years, however, have seen a shift in focus from supervised machine learning to semi-supervised methods that make use of large unpaired corpora. This has been true across domains, with semi-supervised methods setting benchmarks in image tasks Liu et al. (2023), machine translation and NLP Edunov et al. (2018), and acoustic modeling Zhang et al. (2022).

In speech recognition a number of methods have emerged to incorporate both unpaired audio Schneider et al. (2019) and text Toshniwal et al. (2018). These techniques have achieved impressive results when very little supervised data is available, even allowing for performant ASR in the strictly unsupervised setting Baevski et al. (2021). However, the relative performance improvements shown by these methods significantly decreases in settings in which a large supervised corpus is available and

is to be augmented by unpaired data.

At the same time, the field of multimodal deep learning has shown that different types of information may be mapped by a neural network to a common representation. This has been used with images and text for visually-grounded language modeling Alayrac et al. (2022) and image generation from text Ramesh et al. (2022). The same ideas have begun to be explored in ASR to create shared representations for speech and text to improve unpaired text injection Chen et al. (2022a); Sainath et al. (2022).

In this thesis, we argue that a shared representation of speech and text is key to scaling semi-supervised methods to large-scale ASR systems trained with a large supervised dataset. Specifically:

- We begin in Chapter 3 with a comparison of several state-of-the-art semi-supervised learning methods at industrial scale. We show that in this setting the methods are useful, but for reasons other than raw WER improvements. In terms of WER, the methods largely do not scale.

- Suspecting that the failure in scaling is due to a lack of encoder adaptation, we proceed in Chapter 4 to assess the usefulness of a specific method, called dual learning, which specifically aims to refine the encoder. We measure this method in a setting with a large supervised dataset and show how the gains attributable to the method are largely redundant with language model fusion, demonstrating the failure of dual learning to leverage the unsupervised data to improve the encoder representation.

- Having developed a hypothesis on what is failing in the methods we've studied, in Chapter 5 we undertake an abstract study that accumulates evidence that the strength of the shared representation translates to downstream WER improvement.

- Applying the lessons of Chapter 5 to the failure of Chapter 3, in Chapter 6 we develop an algorithm on top of a contemporary text-injection framework that strengthens the shared speech/text representation in the encoder. We demonstrate that this method scales text injection to several large-data settings, addressing the limitations of text injection that we developed in the first two chapters.

1.1 List of Contributions

- **Cal Peyser**, Ronny Huang, Andrew Rosenberg, Tara N. Sainath, Michael Picheny, Kyunghyun Cho.

  Towards Disentangled Speech Representations, *INTERSPEECH, 2022*

- **Cal Peyser**, Ronny Huang, Tara N. Sainath, Michael Picheny, Kyunghyun Cho.

  Dual Learning for Large Vocabulary On-Device ASR, *SLT, 2022*

- **Cal Peyser**, Michael Picheny, Kyunghyun Cho, Rohit Prabhavalkar, Ronny Huang, Tara N. Sainath.

  A Comparison of Semi-Supervised Learning Techniques for Streaming ASR at

Scale, *ICASSP, 2023* (recognized as a top 3% paper in the conference)

- **Cal Peyser**, Kevin Hu, Zhong Meng, Andrew Rosenberg, Rohit Prabhavalkar, Tara N. Sainath, Michael Picheny, Kyunghyun Cho.
  Improving Joint Speech-Text Representations Without Alignment, *INTERSPEECH 2023*

Chapter 2

Background

2.1 Automatic Speech Recognition

The problem of automatic speech recognition (ASR) is conventionally defined as the prediction of a text sequence $\mathbf{y} = y_0, ..., y_m$ corresponding to audio inputs $\mathbf{x} = x_0, ..., x_n$. Preparing a system capable of this prediction usually involves the use of supervised dataset $\mathcal{D}^{sup} = \{(\mathbf{x_0}, \mathbf{y_0}), (\mathbf{x_1}, \mathbf{y_1}), ...\}$ consisting of a large number of corresponding audio and text samples. We generally phrase our predictive system as a probability distribution over possible text transcripts conditioned on input audio, subject to some parameters θ:

$$\mathbf{y}^*_\theta = \text{argmax}_{\mathbf{y}}(P(\mathbf{y}|\mathbf{x}, \theta))$$

where $\mathbf{y}^*_\theta$ is the model's text prediction given the audio input $\mathbf{x}$ and parameters θ. The procedure of "training" such a system aims to find the parameters θ that

maximize the model's performance on the dataset $\mathcal{D}^{sup}$.

$$\theta^* = \text{argmin}_\theta \sum_{\mathcal{D}^{sup}} \mathcal{L}(\mathbf{y}^*_\theta, \mathbf{y})$$

That is, we seek a parameterization θ^* for which the model's predictions minimize some loss function $\mathcal{L}$ that measures the difference between the model's predictions and the ground truth. The evolution of speech recognition over the past three decades can be seen as a series of increasingly powerful models for the posterior $P(\mathbf{y}|\mathbf{x}, \theta)$ and techniques for determining the associated parameterization θ.

2.1.1 GMM-HMMs: An Explicit Relationship between Speech and Text

Speech recognition differs from many other machine learning problems in that it is fundamentally concerned with two modalities - speech and text. We might imagine that a strong speech model must "understand" three things:

1. The speech modality, in order to parse phonetic information from raw audio.
2. The text modality, in order to emit coherent transcripts.
3. The relationship between speech and text, in order to emit the most likely coherent transcript given phonemic information derived from the input.

The earliest ASR systems based on statistical models (see Roe and Wilpon (1993) for a survey) were premised on this understanding and explicitly decomposed the posterior according to Bayes' Rule:

$$P(\mathbf{y}|\mathbf{x}) \propto P(\mathbf{y})P(\mathbf{x}|\mathbf{y})$$

where the prior $P(\mathbf{y})$ can be determined by a language model (LM) and the conditional likelihood of the audio $P(\mathbf{x}|\mathbf{y})$ can be determined by an acoustic model (AM). In practice, n-gram LMs have been combined with AMs implemented by looking up phoneme pronunciations in a lexicon, and connecting a series of phone-dependant hidden Markov models (HMMs), for which states can be modeled as Gaussian mixture models (GMMs). To achieve deployable performance levels, fitting these models required on the order of hundreds of millions of words of text for the LM and tens of thousands of audio utterances Young (1996). For example, the standard training set for the 1993 ARPA benchmark contained 30,000 utterances Pallett et al. (1994).

2.1.2 DNN-HMMs: The Introduction of Neural Networks

ASR research leading up to the deep learning boom was marked by slow adoption of neural networks into components of the conventional HMM-based system, while still preserving the fundamental separation between speech and text modeling implied by Bayes' Rule. While applications of single-layer neural networks for the prediction of HMM states date back to the late 1990s Ding (1999), ASR using neural networks that exceeded the previous state-of-the-art systems emerged in the early 2010s with the application of deep belief networks (DBNs) as an alternative to GMMs Mohamed et al. (2010, 2012). As deep learning began to emerge, even stronger results were achieved replacing the DBNs with multi-layered feed-forward

neural networks (DNNs) trained from scratch using backpropagation Hinton et al. (2012).

2.1.3 End-to-End Neural Models

During the mid 2010s, advances in algorithms and hardware yielded large improvements in the state-of-the-art for several classic machine learning problems, using entirely neural systems trained end-to-end by backpropagation Bahdanau et al. (2014); Krizhevsky et al. (2012). At the core of these breakthroughs was the discovery that performance gains emerge from many neural layers stacked on top of each other to perform "deep" learning.

Such advances quickly made their way into speech recognition, with the core enabling technology being the LSTM-RNN from the language modeling literature Hochreiter and Schmidhuber (1997). One key difference between language modeling and speech recognition with respect to end-to-end neural modeling is the large space of possible alignments between audio frames and text outputs, which was first addressed for this class of model in Graves et al. (2006) using "connectionist temporal classification" (CTC). This work showed for the first time substantial word-error rate (WER) improvements in a end-to-end neural system over a hybrid HMM-based system.

If the alignment problem is one major distinction between speech recognition and language modeling, the inherent distinction between the input domain of audio and the output domain of text is the other. While this distinction was modeled explicitly in the GMM-HMM systems described above, the advent of end-to-end

neural systems blurred the line, relying on the deep architecture to decide internally which components model audio and which model text. While Graves et al. (2006) did away with the explicit distinction, it quickly became clear that the prior of domain separation still had value. Graves et al. (2013), for example, showed that an end-to-end CTC model was still outperformed by the older hybrid systems in tasks that emphasized acoustic modeling, such as on noisy test sets.

The superiority of end-to-end neural systems was eventually cemented by re-asserting the separation between audio and text modeling, borrowing in particular the "encoder-decoder" architecture from machine translation. In this formulation, the end-to-end system in fact consists of two connected components: an "encoder", which learns a representation of the input domain, and "decoder" which consumes the encoder representation and autoregressively predicts the output sequence Cho et al. (2014). In the latter half of the 2010s encoder-decoder systems became the clear state-of-the-art across numerous speech benchmarks. Some systems such as Sak et al. (2017) made use of RNN-T Graves (2012a) decoders to capture the inherent language modeling capacity of the decoder, while others such as Chan et al. (2015a) used an attention mechanism to learn alignments between the output text and the encoder representation. It was the combination of both of these ideas that eventually led to productionizable end-to-end ASR, and systems like this are widely deployed in industry today Chiu et al. (2017); Prabhavalkar et al. (2017).

2.2 Semi-Supervised ASR

The progress made with end-to-end encoder-decoder speech recognition systems cannot be credited to advances in neural architecture alone. It has been long understood that the performance of large models scales as a power law with the amount of data that it is trained on Banko and Brill (2001); Goodman (2001); Kaplan et al. (2020), and speech models are no exception Hestness et al. (2017). As neural speech models became larger and more sophisticated, the data sets on which they were trained grew as well. For example, while we have mentioned that Pallett et al. (1994) used 30,000 utterances (each probably being a few seconds long), Google documented a system in Narayanan et al. (2018) trained on 162,000 *hours* of speech.

In single-modality tasks like language modeling, the usefulness of more data has led to a near-exponential growth in dataset size over time Zhao et al. (2023). This is largely powered by the availability of near-unlimited free text data available on the internet Radford et al. (2019). By contrast, *cross-modality* tasks require supervision, and for ASR in particular labeling speech recordings with transcripts is a time-consuming and expensive process. However, the individual domains involved each lend themselves to large-scale *unpaired* corpora. This has given rise to research interest in *semi*-supervised ASR with unpaired data from either or both modalities. We provide a brief overview of this literature here, and go into more detail on several lines of research in Chapter 3.

2.2.1 Unpaired Audio Injection

A natural analog exists between speech and image recognition, since in both problems we seek to map from a high dimensional data distribution into relatively more constrained text labels. As such, autoencoders, which have been influential in image modeling Gogoi and Begum (2017) presented a natural place to start for incorporating a large unpaired speech corpus into ASR training. However, speech recognition differs from image recognition in two key ways that hamper effective feature extraction with conventional convolutional autoencoders: the sequential nature of both the input and the target, and the need to discard large parts of the audio signal to arrive at relevant representations (e.g. background noise). Chorowski et al. (2019) addressed these issues by adapting the WaveNet autoregressive decoder van den Oord et al. (2016) from the text-to-speech literature and introducing a quantized variational bottleneck after encoding. While this work did not measure downstream ASR performance, it was the first to show that discriminative acoustic features could be learned from unpaired audio using a reconstruction task.

At the same time, increasingly sophisticated constrastive criteria were developed for image problems, yielding strong results on image classification Chen et al. (2020); Gidaris et al. (2018). Early applications to audio include Contrastive Predictive Coding van den Oord et al. (2018), which exploits the temporal structure in audio with a task to distinguish nearby samples in an audio clip from distractors. Like Chorowski et al. (2019), this work also did not directly assess downstream ASR performance. However, it did show that representations arising from this contrastive task separates speakers as well as phonemes neatly in its representation space.

The ideas of CPC were famously applied the next year in Wav2Vec Schneider et al. (2019), which used the CPC contrastive loss in ASR pretraining. This work was followed by several Wav2Vec variants Baevski et al. (2019, 2020) that for the first time demonstrated the ability to train a strong ASR system with very little supervised data. For example, Baevski et al. (2020) achieved 4.8/8.2 WER on Librispeech test-clean/test-other with only ten minutes of supervised audio when 60,000 hours of unsupervised audio are provided for contrastive pre-training. However, these methods offered little gain in the setting where a substantial supervised dataset is available. Baevski et al. (2020) in particular achieved 2.1/4.8 WER on Librispeech when using the all 960 hours of supervised data together with the same 60,000 hour pretraining set. This is compared to 1.9/3.9 achieved by the contemporary Conformer model on the supervised data alone Gulati et al. (2020).

This trend largely continued with the successors to the Wav2Vec series of models. HuBERT Hsu et al. (2021) replaces the quantized CPC-like targets of Wav2Vec 2.0 with acoustic clusters. WavLM Chen et al. (2021a) further introduces a denoising task into the HuBERT recipe. As before, these methods show very strong improvements without a large supervised corpus, but the improvements are much less substantial when a such a dataset is available.

2.3 Unpaired Text Injection

On its surface, using unpaired text in ASR training is a simpler proposition than using unpaired audio. Modern end-to-end ASR systems (e.g. Chan et al.

(2015b); Graves (2012b)) are separated into an encoder and decoder, where the encoder is concerned with speech inputs and the decoder essentially functions as a language model that autoregressively emits a text sequence. This structure creates a natural opportunity to perform conventional language modeling on unpaired text and incorporate the resulting LM into the ASR system.

The first unpaired text injection methods for encoder-decoder ASR models involved directly integrating an external LM into the ASR system. In shallow fusion, which was originally proposed for machine translation Gülçehre et al. (2015), a language model is run during beam search at inference time, and its logits are directly interpolated into the decoder logits. More sophisticated, training-time methods include cold fusion and deep fusion, in which hidden LM states are fused with decoder states during training Sriram et al. (2018). Direct comparisons of language model fusion for ASR have shown that the simpler shallow fusion is generally the most performant Toshniwal et al. (2018). Shallow fusion has been proven effective for ASR even against large supervised baselines, but requires careful tuning of beam search settings Peyser et al. (2020).

The success of shallow fusion over training-time methods begs the question: can unpaired text be used only to inform an ASR system's modeling of language during decoding, or can it be incorporated more deeply into the model's parameters themselves to refine how it encodes audio? Recent systems have taken a multi-modal approach, attempting to project unpaired text into the acoustic encoder's representation space so that downstream parameters may learn agnostically from the two domains. JOIST Sainath et al. (2022) includes a text encoder with random input

upsampling alongside the model's audio encoder, and performs conventional supervised training alongside an unsupervised masked text reconstruction task. MAESTRO Chen et al. (2022a) includes a learned duration model to further match the unpaired text representation to the acoustic representation.

Chapter 3

Investigating the Scaling of Semi-Supervised Learning

Our summary of contemporary semi-supervised learning methods in Chapter 2 cites a large number of methods that have each been shown to benefit ASR training in the low-resource setting. The natural question that arises when considering these results is how these methods scale to the high-resource setting. Some of the cited works, like Baevski et al. (2020) and Chen et al. (2021a) include specific results on large supervised baselines showing that these methods in fact do not scale. Others, like Sainath et al. (2022), show improvements over large supervised baselines on tail sets, but do not clearly demonstrate the same against the head of the distribution. However, as of the beginning of this study there lacked a comprehensive comparison of state-of-the-art semi-supervised methods as they scale in the size of the supervised dataset.

We therefore begin our investigation with a comparison of several leading semi-supervised methods in a controlled setting geared towards production implementation. Unlike previous work, we apply these methods to a state-of-the-art, 160M-parameter streaming Conformer Gulati et al. (2020) model that is already trained on a very large supervised corpus. We further depart from previous work by training supervised and unsupervised tasks jointly, which is being increasingly shown to be preferable to the conventional fine-tuning approach on very large datasets Bai et al. (2021). We find that under these conditions, none of the studied methods improve general WER at all. However, we report improvements in the decoder's computational load and in lattice density, as well as in several targeted WER measurements assessing performance on known categories of particularly difficult utterances. Through this comparison and analysis, we hope to offer a more nuanced and comprehensive view of the usefulness of unpaired audio and text in industrial ASR.

3.1 Selected Methods

For this study, we specify three state-of-the-art semi-supervised ASR methods encompassing both unpaired audio and unpaired text. In this section, we go into more detail about some of the lines of research referenced in Chapter 2 in order to contextualize these three methods.

3.1.1 Text Injection

As described in Chapter 2, text injection in ASR is traditionally done with language model "fusion", either at inference time Gülçehre et al. (2015) or training time Sriram et al. (2018); Variani et al. (2020). These methods involve the explicit separation of the model parameters into an acoustic model trained on paired data and a language model trained on unpaired text. The improvements yielded by these methods come at the cost of the additional language model parameters at inference time.

A simultaneous line of work has sought an alternative to fusion in which unsupervised text is used to train an acoustic model directly. One major line of work focuses on creating pseudolabels for unpaired text through synthesized audio. This has been studied by generating a raw audio signal Xu et al. (2020) or higher level lexical features Chen et al. (2021c). Work adapting cycle consistency losses from machine translation have trained ASR and TTS together with a fully end-to-end objective Baskar et al. (2019); Hori et al. (2019). We choose TTS-based augmentation as the first method to study in this work (see Section 3.2.2.1).

Finally, a third class of methods for unpaired text injection makes use of auxiliary, text only objectives to train an ASR encoder without generating TTS pseudolabels. Most such works have sought to train an ASR encoder to agnostically represent either audio or text, such that unpaired text is processed similarly to audio Bapna et al. (2021); Tang et al. (2022); Yusuf et al. (2022). As described above, JOIST Sainath et al. (2022) is a recent method which does this using a masked language modeling task in the spirit of BERT Devlin et al. (2018). We study JOIST in this work since it

is one of the few methods that has been studied with very large supervised datasets and with on-device sized streaming models (see Section 3.2.2.2).

3.1.2 Audio Injection

We briefly introduced the literature on audio injection in Chapter 2, and a detailed review is available in Mohamed et al. (2022). As we've described, recent work is largely built on the success of the Wav2Vec series of models Baevski et al. (2020); Schneider et al. (2019), which work by modeling masked segments of audio using a contrastive loss. One line of further work investigated audio clustering to generate targets for the contrastive loss Chen et al. (2021a); Hsu et al. (2021) while another investigated methods for computing that signal by quantizing the audio inputs Baevski et al. (2019). BEST-RQ Chiu et al. (2022) in particular finds that fixed random projection to a pre-initialized codebook works effectively as a quantizer. We choose BEST-RQ as the third method to study in this work (see Section 3.2.2.3).

3.2 Model

We require a framing of the semi-supervised ASR problem that encompasses all three of the methods in question. In this section, we provide such a framing, develop our model architecture, and specify the multi-task optimization problem that it is trained for.

3.2.1 Architecture

We are interested in the setting in which unsupervised data in both the speech and text domains is available alongside a large supervised corpus. We denote as $(\mathbf{x}, \mathbf{y}) \in \mathcal{S}$ the supervised pair of a speech utterance $\mathbf{x}$ and text label $\mathbf{y}$ in the supervised dataset $\mathcal{S}$. We similarly denote unsupervised speech examples as $\mathbf{x} \in \mathcal{U}^S$ and unsupervised text examples $\mathbf{y} \in \mathcal{U}^T$.

We extend the cascading conformer proposed in Narayanan et al. (2021) to support semi-supervised multitask training. This model supports "causal" (streaming) prediction, as well as "non-causal" (full-context) prediction. To this end, we define four neural modules:

1. E_C, the "causal" encoder, which consumes streamed audio features with no right-context.

2. E_{NC}, the "non-causal" encoder, which consumes the outputs of E_C with 900ms of right-context.

3. D_C, a decoder for the causal encoder. During inference, this decoder may be used to generate immediate predictions as the user speaks.

4. D_{NC}, a decoder for the non-causal encoder. During inference, this decoder may be used to revise the predictions of the causal decoder with short latency.

Unlike Narayanan et al. (2021), we would like our model to consume representations of either audio or text. For this we follow JOIST, seeking mechanisms to cause the E_C to be agnostic to the input modality. We choose to include two neural

"frontends", one for audio features and one for text. As in JOIST, we upsample text frontend outputs by repetition so that audio and text representations will be of approximately the same length.

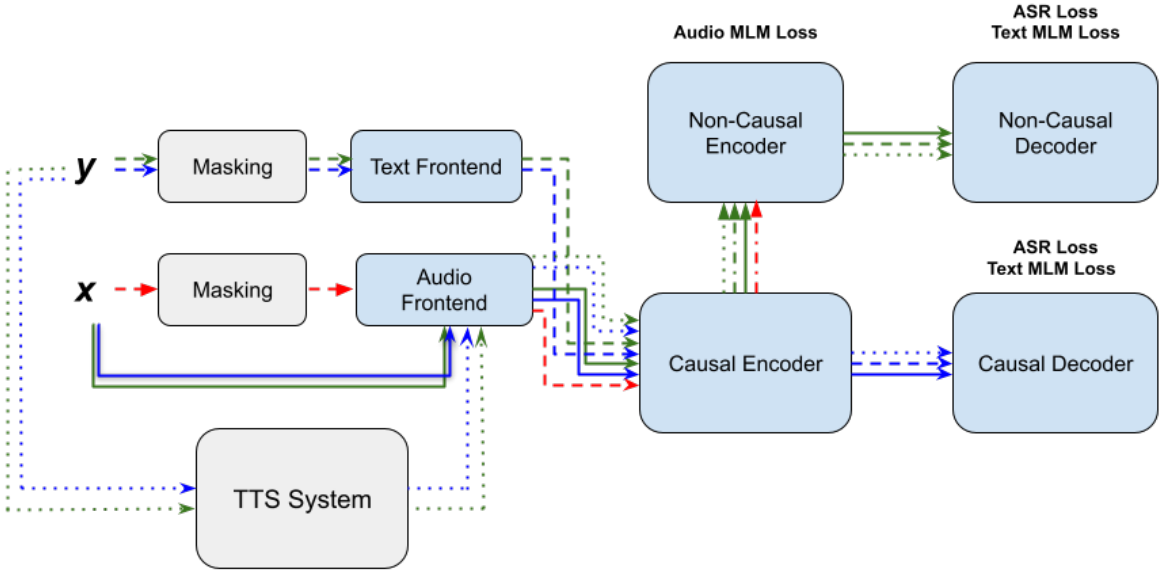


Figure 3.1: A high-level model architecture supporting multiple semi-supervised methods for both streaming (blue paths) and non-streaming prediction (green paths). Solid paths represent BEST-RQ, dashed paths represent JOIST, and dotted paths represent TTS augmentation.

3.2.2 Tasks

In this framework, causal and non-causal ASR are trained as they are in Narayanan et al. (2021). In particular, for causal ASR, x is processed by the audio frontend, encoded by E_C, and decoded by D_C, while non-causal ASR is processed analogously with the non-causal modules E_{NC} and D_{NC}. The model is trained end-to-end with an RNN-T Graves (2012a) loss. This is represented by the solid blue (causal) and solid green (non-causal) paths in Figure 3.1. For semi-supervised tasks we require different formulations.

3.2.2.1 TTS Augmentation: Text Injection

Using a pre-trained TTS system with frozen parameters, we generate an audio clip $\hat{x}$ corresponding to each unsupervised text segment $y \in \mathcal{U}^T$. We then treat $(\hat{x}, y)$ as a supervised audio-text pair and train the causal and non-causal ASR tasks. This is represented by the dotted blue (causal) and dotted green (non-causal) paths in Figure 3.1.

We found that in order to achieve reasonable training speed it is important that the TTS system convert input word-pieces not into raw audio but instead into the (much shorter) sequence of acoustic features that is consumed by the audio frontend. This is due to the fact that since the decoder of our TTS system which produces audio features is autoregressive, audio sequence length has critical implications for training speed and quickly becomes a bottleneck.

3.2.2.2 JOIST: Text Injection

Following the design of JOIST in Sainath et al. (2022), we pass masked unpaired text examples through a text frontend, which consists simply of a learned projection. The results are treated identically to audio features; that is, they are passed in turn to the causal (E_C and D_C) and non-causal (E_{NC} and D_{NC}) encoders and compared to the original text sequence via an RNN-T loss. This is represented by the dashed blue (causal) and dashed green (non-causal) paths in Figure 3.1.

We find that it is critical for WER that JOIST consume phonemic representations of y, as opposed to text tokens, corroborating the findings of Sainath et al. (2022). We include a text-to-phoneme lookup in the model which processes text before masking. The JOIST loss still operates with respect to the standard word-piece representation - that is, the JOIST loss learns to generate word pieces from a masked phoneme sequence.

3.2.2.3 BEST-RQ: Audio Injection

We model our audio injection after BEST-RQ as presented in Chiu et al. (2022). Audio features are masked and processed by the frontend. They are then encoded by the casual and non-causal encoders of the ASR stack. Additionally, audio features are processed by a randomly initialized projection with frozen weights and then discretized by rounding to the nearest entry in a fixed codebook. The encoder is then trained to predict the quantized targets inside the masked region. This is represented by the dashed red path in Figure 3.1.

3.2.3 Training Scheme

There are many approaches to multi-task semi-supervised learning, mostly focused on the pretrain-finetune paradigm Baevski et al. (2019, 2020); Chen et al. (2021a); Schneider et al. (2019). While this methodology has achieved state of the art results on datasets such as Librispeech, we found that on our large dataset it is prone to forgetting representations learned in pretraining during finetuning, which is consistent with the findings in Bai et al. (2021) for very large training sets. We therefore restrict our study to joint training of ASR together with the unsupervised tasks. Note that even though joint training includes ASR, we find that it is still beneficial and convenient to initialize from a strong ASR baseline.

At each iteration during training we sample a separate batch from each dataset, $b_{\mathcal{S}} \in \mathcal{S}$, $b_{\mathcal{U}^S} \in \mathcal{U}^S$, and $b_{\mathcal{U}^T} \in \mathcal{U}^T$. We then propagate each batch through the model, performing the preprocessing specified for TTS augmentation and JOIST on $b_{\mathcal{U}^T}$ and that specified for BEST-RQ on $b_{\mathcal{U}^S}$. We apply the relevant losses to each task and sum them according to preset weights.

3.3 Experiments

This section details the implementation, training, and evaluation of the architecture described above.

3.3.1 Model

Following the components in Figure 3.1 the architecture of our model is as follows.

The causal audio encoder E_C consists of six conformer Gulati et al. (2020) layers with model dimension 2048 and eight attention heads. The noncausal audio encoder E_{NC} adds a further nine such conformer layers. The decoders D_C and D_{NC} are each HAT Variani et al. (2020) decoders with prediction and joint networks with model dimension 640. These four components and the audio frontend, which together make up the inference-time model, contain about 164M parameters.

The TTS system is based on Tacotron 2 Shen et al. (2017). The encoder consists of three convolutions followed by a single RNN layer, while the decoder consists of a single RNN layer with attention to the encoder outputs followed by a post-net consisting of five convolutional layers.

3.3.2 Training

We train our model with a supervised dataset $\mathcal{S}$ consisting of about 4M utterances, totalling about 200k hours of speech. We also use an unsupervised audio set $\mathcal{U}^S$ of about 600M utterances and an unsupervised text set $\mathcal{U}^T$ of about 230B examples.

At timestep t, the audio head of our model consumes 512-dimensional features consisting of four 128-dimensional log-mel features representing the range $[t-2, t+1]$. The log-mel features are computed at 10ms intervals and on 32ms frames. We subsample stacked features by a factor of 3, so that each feature represents 30ms in the input. During BEST-RQ, we mask a single span consisting of 15% of the input features. Text inputs are represented by a wordpiece model of size 4096.

Our baseline model is trained for 800k steps with a batch size of 2048 for each of

$\mathcal{S}$, $\mathcal{U}^S$, and $\mathcal{U}^T$. Our semi-supervised experiments are trained for a further 35k steps, using task splits specified in Section 6.4.

3.3.3 Evaluation

We evaluate our models on several test sets, seeking to measure performance under the acoustic and language conditions which are typically targeted using unsupervised data. Our voice search test set (**VS**) is sampled from anonymized traffic to Google production services. The **NOISY** set consists of anonymized traffic with artificial noise added. Our remaining test sets are synthesized using a TTS system from anonymized text traffic to Google services, and are selected according to a criterion meant to target difficult language conditions. The rare proper nouns set (**RPN**) consists of examples that contain a proper noun (as determined by a neural proper noun tagger) that occurs fewer than five times in $\mathcal{S}$. The Rare-LM set (**R_-LM**) consists of examples containing a unigram that occurs fewer than five times in both $\mathcal{S}$ and $\mathcal{U}^T$, while the (**C_LM**) consists of examples containing a unigram that occurs fewer than five times in $\mathcal{S}$ but at least 150 times in $\mathcal{U}^T$. **RPN** and **C_LM** measure tail performance, while **C_LM** is intended to measure the degree to which information from $\mathcal{U}^T$ has been incorporated into the model.

3.4 Results

We denote JOIST with the letter **A**, TTS augmentation with **B**, and BEST-RQ with **C**. We find the best results when each of these experiments are trained with

40% task weighting each on causal and non-causal ASR, with the remaining 20% split across unsupervised tasks. The weightings of the unsupervised tasks are given in Table 3.1.

Model	C-JOIST	NC-JOIST	TTS	BEST-RQ
E-A	1/2	1/2	0	0
E-B	0	0	1	0
E-C	0	0	0	1
E-AB	1/4	1/4	1/2	0
E-AC	1/4	1/4	0	1/2
E-ABC	1/6	1/6	1/3	1/3

Table 3.1: Task Weights. C-JOIST and NC-JOIST refer to the causal and non-causal variants.

We denote our baseline experiment **E-0**, which splits its weight equally between causal and non-causal supervised ASR.

We give our WER results in Table 3.2. We are unsurprised to find that given a very large supervised corpus and limited model capacity, none of our methods improve performance on the unspecialized voice search test set. We find considerable improvement, however, under tail conditions. JOIST consistently performs best on the acoustically clean but linguistically difficult TTS tail-word test sets, which agrees with the intuition that JOIST acts to improve the model's text representation. However, JOIST in fact degrades performance on the acoustically challenging Noisy test set. BEST-RQ seems beneficial only when combined with JOIST, where it appears to recover lost performance on noisy data while retaining some of the improvements on the tail-word sets.

In production systems, model performance goes beyond raw WER, since it is

Model	**VS**	**Noisy**	**RPN**	**R_LM**	**C_LM**
E-0	6.0	8.2	21.2	38.3	55.8
E-A	**-0.0%**	+1.2%	**-4.7%**	**-5.0%**	**-2.3%**
E-B	**-0.0%**	**-1.2%**	-0.5%	-2.1%	-0.7%
E-C	**-0.0%**	+1.2%	+0.1%	-0.0%	-0.4%
E-AB	**-0.0%**	+1.2%	-3.8%	-4.2%	-2.0%
E-AC	**-0.0%**	-0.0%	-2.8%	-2.9%	-1.2%
E-ABC	**-0.0%**	+2.4%	-3.3%	-3.7%	-1.4%

Table 3.2: Word Error Rate measurements for all task combinations.

often not a 1-best hypothesis but rather the produced lattice that is used to generate predictions or fed directly to a downstream NLU task. In Table 3.4, we measure the richness of the lattice by computing "lattice density", which we define as the number of arcs in the lattice divided by the number of wordpieces in the ground truth. On this measure, we find that all three methods offer considerable improvement in voice search. For difficult utterances, we find that combinations of methods largely outperform single methods. This agrees with the intuition that many training criteria lead to a greater diversity of plausible predictions, and invites investigation into the combination of these methods for applications like biasing or intent classification which can benefit from a rich lattice.

Finally, since an autoregressive decoder is often a computational bottleneck in on-device systems, we seek to determine the impact of our methods on the work the decoder has to do. In Table 3.3, we measure the average number of states expanded by the decoder during beam search. We find that all three methods provide meaningful improvements over the baseline on this metric, with the best results coming from JOIST. This suggests, unsurprisingly, that the decoder explores the

Model	**VS**	**Noisy**	**RPN**	**R_LM**	**C_LM**
E-0	162	187	297	357	325
E-A	-7.2%	-5.3%	**-11.1%**	**-10.1%**	**-8.9%**
E-B	-7.2%	-5.3%	-9.8%	-8.4%	-7.4%
E-C	**-9.9%**	**-6.9%**	-6.3%	-5.8%	-4.9%
E-AB	-7.2%	-4.8%	-6.1%	-5.0%	-4.0%
E-AC	**-9.9%**	-6.4%	-10.8%	-9.2%	-8.3%
E-ABC	-8.5%	-5.9%	-9.8%	-8.7%	-7.7%

Table 3.3: The average number of states traversed by the decoder for each method. Fewer states reflects more pruned paths, in turn reflecting more confident decoder decisions.

Model	**VS**	**Noisy**	**RPN**	**R_LM**	**C_LM**
E-0	3.2	3.3	6.2	8.1	9.7
E-A	**+12.5%**	**+15.2%**	**+3.2%**	+3.7%	+3.1%
E-B	**+12.5%**	+12.1%	**+3.2%**	+3.7%	+3.1%
E-C	**+12.5%**	+12.1%	**+3.2%**	+3.7%	+3.1%
E-AB	**+12.5%**	+12.1%	+1.6%	+3.7%	+3.1%
E-AC	**+12.5%**	+12.1%	**+3.2%**	**+4.9%**	**+4.1%**
E-ABC	**+12.5%**	+12.1%	**+3.2%**	**+4.9%**	+3.1%

Table 3.4: The average decoding lattice density for each method. Denser lattices are generally more useful for downstream NLU tasks.

fewest states when the model has a strong language representation.

3.5 Conclusions

We begin our investigation with the application of several contemporary semi-supervised training methods to a realistic, state-of-the-art production ASR system. We find that unlike in the conventional setting, with a large full-context model and only a small amount of supervised data, these methods do not offer improvement on unspecialized WER. Having shown this, we articulate the core questions of this work: why do these methods fail to scale, and how might they be improved?

As a first step, we note with interest that despite failure to improve WER on the head, these methods are hardly useless. Instead, these techniques offer meaningful utility for tail-condition performance, lattice density, and decoder computational load. These results suggest that the primary function of the studied techniques is not to refine the audio encoder, as might be thought from the design the of the methods, but rather to improve language modeling capacity in the decoder. In this sense, these methods overlap with shallow fusion, which as we've noted in Chapter 2 is a method of text injection that has been shown to reliably scale against strong baselines.

Chapter 4

Dual Learning

The results of Chapter 3 suggest that mainstream semi-supervised ASR methods, when applied at scale, are largely constrained to refining language modeling in the decoder. In this chapter, we test that theory against a less-known semi-supervised method that is specifically designed to leverage both unpaired audio and text in order to refine the encoder representation. We will, for the first time, determine how this method performs in the high-resource, high-parameter, streaming setting. We will find that while the method remains useful even against a baseline trained on all 960 hours of Librispeech, improvements are largely redundant with shallow fusion, which is strictly confined to the decoder computation. As we will see, the design of dual learning emphasizes the encoder representation by training parallel encoders for both modalities, and including the encoder in all training tasks. That even a method so targeted at the encoder as dual learning should only significantly effect the decoder will refine our understanding of why contemporary semi-supervised methods for ASR

scale poorly.

4.1 Background: Dual Learning

The literature on semi-supervised ASR largely involves separate methods for unpaired text and audio, as described above. While works such as Bapna et al. (2021) have incorporated methods from both categories into the same system, few individual methods make use of unpaired audio and text together.

Dual learning Qin (2020) is a notable exception. Originally proposed as back-translation for machine translation Xia et al. (2016), dual learning exploits the "dual" nature of speech-to-text and text-to-speech, co-training models for the two tasks. Staring with Ren et al. (2020) and continuing with Xu et al. (2020), dual learning systems for speech use the TTS component to provide supervision for unpaired text examples and the ASR component to provide supervision for unpaired audio examples, permitting training in both domains. These systems have achieved strong performance on as little as few minutes of paired data. However, they have yet to demonstrate large gains against a strong supervised baseline.

4.2 Methods

In this section, we describe our implementation of ASR pretraining based on dual learning.

4.2.1 Architecture

In order to perform ASR, TTS, and reconstruction in both domains, our implementation must include encoders and decoders for both audio and text. As in Chapter 3 we imitate Narayanan et al. (2021) and implement streaming with an architecture that can emit a provisional hypothesis immediately and then revise it after a short delay. We do this to in order to make a fair comparison to our previous study and in order to examine the method in a production-like setting. In order to improve the likelihood of success on the two more difficult tasks (ASR and TTS), we adapt these components from existing ASR and TTS architectures. Our audio encoders and text decoder are adapted from conformer Gulati et al. (2020); Narayanan et al. (2021). Our text encoder and audio decoder are adapted from Tacotron 2 Shen et al. (2017).

Formally, we frame our problem around the same three datasets as in Chapter 3. First, $\mathcal{S}$ consists of paired text and audio examples (x, y), where $x = (x_0, ..., x_m)$ is an audio sample of length m and $y = (y_0, ..., y_n)$ is the corresponding text transcript of length n. Second, $\mathcal{U}_\mathcal{T}$ gives unpaired text examples y, and finally $\mathcal{U}_\mathcal{A}$ gives unpaired audio examples x. We then define the components of the model as functions. We define the audio encoders E_A^{C}, which has only left-context and E_A^{NC}, which has 900ms of right-context. We also define the text encoder E_T, the decoders D_A and D_T, and the linear transformations $T_{A \to T}$ which maps an audio embedding to a text embedding and $T_{A \leftarrow T}$ which maps a text embedding to an audio embedding.

We may then proceed to define the model's objectives.

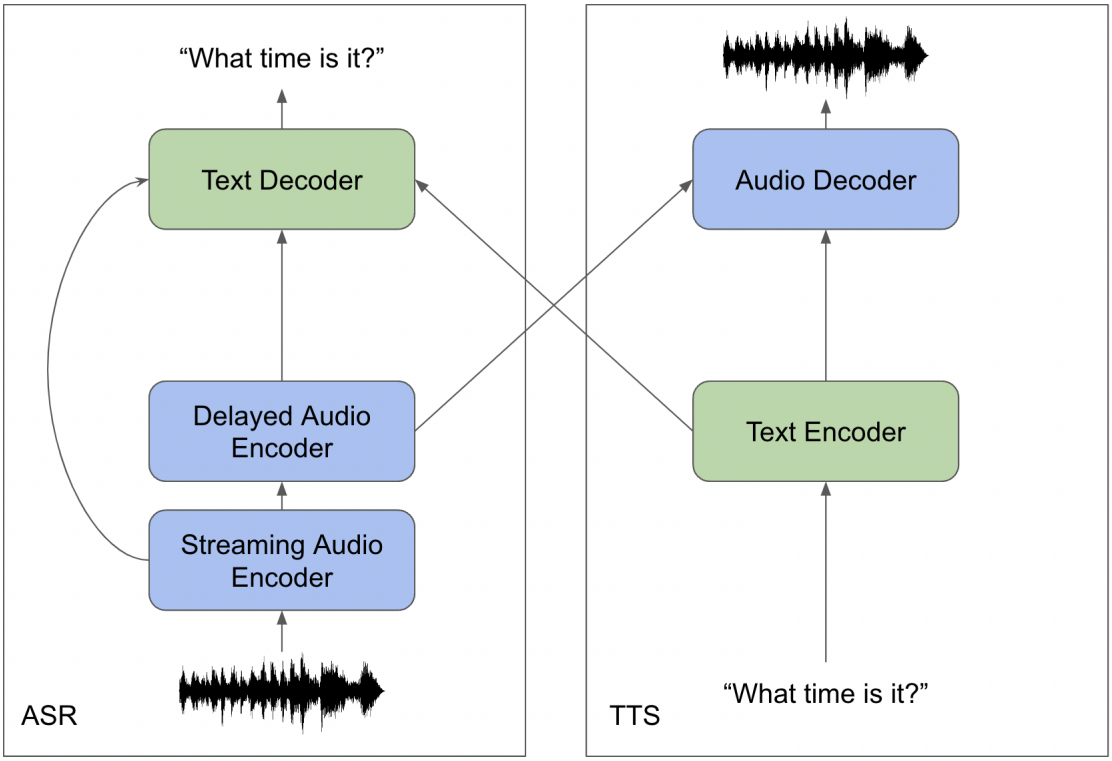


Figure 4.1: The architecture of our dual learning model. Blue components participate in audio reconstruction, while green components participate in text reconstruction.

4.2.1.1 Supervised ASR and TTS

We follow Narayanan et al. (2021) to build a model capable of emitting an ASR hypothesis in real time while streaming finalized predictions with 900ms latency. To this end, we split ASR training into two losses. For $(x, y) \in \mathcal{S}$, the immediate streaming task is given by:

$$\mathcal{L}_{\text{ASR}}^{\text{C}} = L_{\text{xent}}(y, D_T \circ E_A^{\text{C}}(x))$$

while the delayed task is given by:

$$\mathcal{L}_{\text{ASR}}^{\text{NC}} = L_{\text{xent}}(y, D_T \circ E_A^{\text{NC}} \circ E_A^{\text{C}}(x))$$

where $\circ$ denotes function composition. That is, the streaming task uses only the first, left-context encoder while the delayed task adds a further encoder with 900ms of right-context. Here we have defined L_{xent} as the cross-entropy loss over text units.

Since this work focuses on pretraining for ASR systems, we do not seek to stream the TTS task. Instead we define the single full-context task:

$$\mathcal{L}_{\text{TTS}} = L_{\text{MSE}}(x, D_A \circ E_T(y))$$

where we have defined L_{MSE} as the mean squared error loss over continuous audio features.

4.2.1.2 Unsupervised ASR and TTS

Dual learning for speech and text involves the use of the TTS system to provide pseudo-labels for the ASR system and visa versa. Specifically, for an unpaired audio example $x \in \mathcal{U}_{\mathcal{A}}$ we derive the pseudo-label $\hat{y}$ by beam search over the outputs of the ASR model. We may then define the objectives:

$$\mathcal{L}_{\text{U-TTS}} = L_{\text{MSE}}(x, D_A \circ E_T(\hat{x}))$$

Similarly, for an unpaired text example $y \in \mathcal{U}_{\mathcal{T}}$ we may derive the pseudo-label $\hat{x}$ by performing inference in the TTS system. We may then define the objective:

$$\mathcal{L}^{\text{C}}_{\text{U-ASR}} = L_{\text{xent}}(y, D_T \circ E^{\text{C}}_A(\hat{y}))$$

and

$$\mathcal{L}^{\text{NC}}_{\text{U-ASR}} = L_{\text{xent}}(y, D_T \circ E^{\text{NC}}_A \circ E^{\text{C}}_A(\hat{y}))$$

4.2.1.3 Reconstruction

To perform text reconstruction, we must pass representations from the ASR encoder to the TTS decoder and visa versa. We find that when initializing using pre-trained ASR and TTS systems, the components struggle to adapt to each other and the model fails to converge. We find that this problem is alleviated simply by placing a single linear transformation $T_{A \to T}$ between the audio encoder and audio de-

coder, and another transformation $T_{A \leftarrow T}$ between the text encoder and text decoder. With this in mind, we define the text reconstruction task:

$$\mathcal{L}_{\text{Text Recon}} = L_{\text{MSE}}(y, D_T \circ T_{A \leftarrow T} \circ E_T(y))$$

for $(x, y) \in \mathcal{S}$, with $\mathcal{L}_{\text{U-Text Recon}}$ defined analogously for $x \in \mathcal{U}_{\mathcal{T}}$. We similarly define the audio reconstruction task:

$$\mathcal{L}_{\text{Audio Recon}} = L_{\text{xent}}(x, D_A \circ T_{A \rightarrow T} * E_A^{\text{NC}} \circ E_A^{\text{C}}(x))$$

for $(x, y) \in \mathcal{S}$, with $\mathcal{L}_{\text{U-Audio Recon}}$ defined analogously for $x \in \mathcal{U}_{\mathcal{A}}$.

4.2.2 Training

We might naively seek to train the above tasks together by alternating tasks across sequences of batches. As in Chapter 3, we find that such a training scheme fails to achieve convergence, as each task is forgotten during the training of the others, and we again instead combine all tasks in a single batch. This time the batch is split in thirds, the first coming from $\mathcal{S}$, the second from $\mathcal{U}_{\mathcal{A}}$, and the last from $\mathcal{U}_{\mathcal{T}}$. For the first third, we jointly optimize the supervised tasks:

$$\mathcal{L}_S = \frac{\mathcal{L}_{\text{ASR}}^{\text{C}} + \mathcal{L}_{\text{ASR}}^{\text{NC}}}{2} + \mathcal{L}_{TTS} + \mathcal{L}_{\text{Text Recon}} + \mathcal{L}_{\text{Audio Recon}}$$

for the second third, we optimize the unsupervised audio tasks:

$$\mathcal{L}_A = \mathcal{L}_{\text{U-TTS}} + \mathcal{L}_{\text{U-Audio Recon}}$$

for the last third, we optimize the unsupervised text tasks:

$$\mathcal{L}_S = \frac{\mathcal{L}^{\text{C}}_{\text{U-ASR}} + \mathcal{L}^{\text{NC}}_{\text{U-ASR}}}{2} + \mathcal{L}_{\text{U-Text Recon}}$$

We find that this method achieves convergence, so long as we initialize the model's components from an ASR and TTS system trained on $\mathcal{S}$. Otherwise, the model generates incorrect pseudo-labels early in training, preventing progress.

4.2.3 Language Modeling

Since dual learning involves the incorporation of unpaired text at training time, we naturally want to compare our method to the incorporation of unpaired text at inference time. To this end we evaluate our models with shallow fusion Gülçehre et al. (2015) with a pretrained LM. We also use a Hybrid Autoregressive Transducer (HAT) Variani et al. (2020) text decoder, which permits the factorization of our models' internal LM. Ultimately, at inference time for audio sample x we seek:

$$y^* =_y \log P(y|x) + \alpha P_{ELM}(y) - \beta P_{ILM}(y)$$

where $P(y|x)$ gives our acoustic model posterior, $P_{ELM}(y)$ gives the likelihood of a transcript in the external LM, $P_{ILM}(y)$ gives the likelihood of the transcript in the internal LM (as formulated in HAT), and α and β are hyperparameters.

Model	Baseline	Shallow Fusion	Internal LM
BASELINE	8.4	6.3	5.8
E-ALL	7.5	5.6	5.5
E-DL	8.1	6.2	6
E-RECON	10.3	7.4	7.2

(a) Test Clean

Model	Baseline	Shallow Fusion	Internal LM
BASELINE	22.9	19.5	18.3
E-ALL	20.4	16.3	16.2
E-DL	21.9	18.3	17.9
E-RECON	27	22.5	21.9

(b) Test Other

Table 4.1: WER percentage results on the Librispeech test sets. Baseline evals include no language model. Shallow Fusion evaluations include LM interpolation with $\alpha = 0.2$. Internal LM evaluations further subtract out the internal LM with $\beta = 0.1$.

4.3 Experimental Setup

In this section, we give the details of our experimental setup.

4.3.1 Model

The ASR branch of our model is a cascading conformer adapted from Narayanan et al. (2021), specifically sized to be realistic for an on-device streaming application. The streaming encoder is small to ensure fast inference. It consists first of 3 convolutional layers followed by 7 conformer layers with a 512-dimensional representation for a total of 56M parameters. The delayed encoder is larger, and is parameterized by 10 conformer layers with a 640-dimensional representation for a total of 99M parameters. Following Botros et al. (2021), the HAT decoder consists of an embedding network and joint network, contributing another 9M parameters.

The TTS branch of our model is adapted from Tacotron 2 Shen et al. (2017). The text encoder first maps wordpieces into a 512-dimensional embedding space, followed by three convolutional layers and a single bidirectional LSTM layer, totalling 8M parameters. The audio decoder consumes the audio sample autoregressively through a "pre-net", which consists of two fully-connected layers with 50% dropout at each layer. We find that this aggressive dropout is critical to convergence, since during multi-task training with teacher forcing the TTS decoder has a strong tendency to rely entirely on the autoregressive signal instead of the encoder representation. This yields poor performance at inference, which in turn creates poor pseudo-labels for unpaired text. Scheduled sampling Bengio et al. (2015) was investigated as an

alternative, but was found to be less effective than simple dropout. The audio sample is then passed to two LSTM layers, which also consume the encoder representation via cross-attention. After the LSTM has generated an audio prediction, that result is further processed by a full-context "post-net" consisting of five convolutional layers. In total, the TTS decoder has about 26M parameters.

4.3.2 Data

We use 960 hours of supervised audio from Librispeech as $\mathcal{S}$, 60k hours of unsupervised audio from Librilight Kahn et al. (2020) as $\mathcal{U}_{\mathcal{A}}$, and 80M transcripts from the Librispeech LM set as $\mathcal{U}_{\mathcal{T}}$. We process the audio into a 128-dimensional log-mel feature per 10ms of audio. We stack every third such feature with the three features before it, yielding 512-dimensional features at 30ms intervals. We then apply SpecAugment Park et al. (2019) with mask parameter $F = 27$ and ten time masks, as in Gulati et al. (2020). This forms the inputs to the audio encoder.

4.3.3 Evaluation

We evaluate our models using a beam search with a beam size of 8. For fusion experiments, we use an external language model trained on $\mathcal{U}_{\mathcal{T}}$. The LM is a causal transformer Vaswani et al. (2017) with 8 layers, 16 attention heads, and a model dimension of 1024, totaling about 100M parameters.

4.4 Results

We evaluate our model relative to a baseline ASR system trained on 960 hours of Librispeech (**BASELINE**). The difference between our baseline WER and those reported in full-context works like Chen et al. (2021a) reflect the added difficulty of streaming results as well as the reduced model size. We contrast this with a model trained on all tasks defined above (**E-ALL**). We perform ablations by also training a model on only the supervised and dual learning tasks (**E-DL**, excluding $\mathcal{L}_{\text{Text Recon}}$, $\mathcal{L}_{\text{U-Text Recon}}$, $\mathcal{L}_{\text{Audio Recon}}$, and $\mathcal{L}_{\text{U-Audio Recon}}$) and another only on the supervised and reconstruction tasks (**E-RECON**, excluding $\mathcal{L}^{\text{C}}_{\text{U-ASR}}$, $\mathcal{L}^{\text{NC}}_{\text{U-ASR}}$, and $\mathcal{L}_{\text{U-TTS}}$). Results are given in Table 4.1.

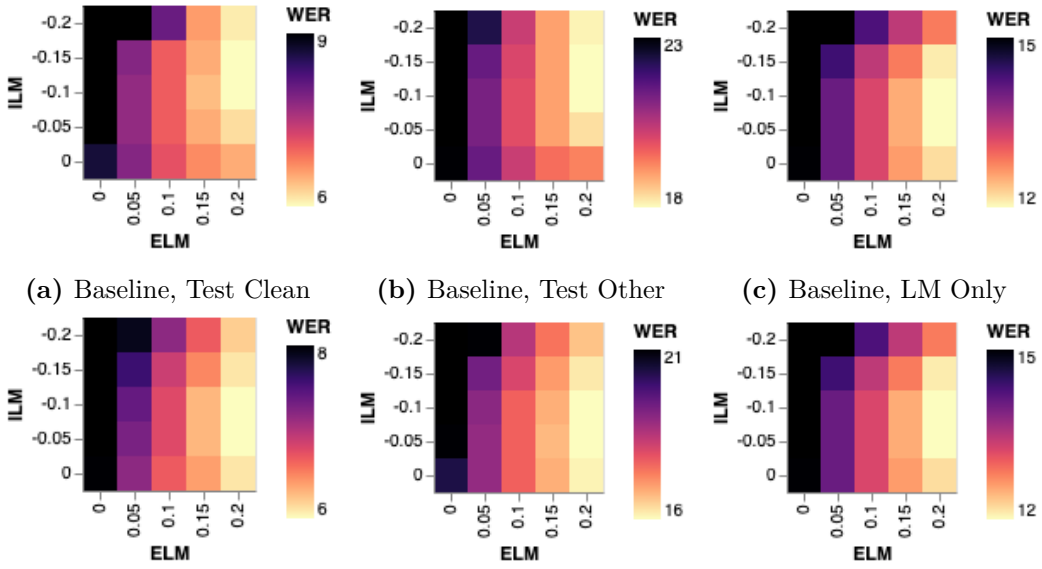
We find our method to improve performance on the test-clean/test-other test sets by 10.7%/5.2% without an LM and 11.1%/16.4% with an LM included via shallow fusion. Interestingly, we find that while dual learning alone (**E-DL**) yields improvements, reconstruction alone (**E-RECON**) does not. Nevertheless, the combination of dual learning and reconstruction (**E-ALL**) yields better results than either alone. This suggests that reconstruction itself distracts from the ASR tasks but synergizes with dual learning. This may reflect the fact that on the multi-speaker, long utterances of Librispeech, a joint model benefits from extra exposure to the unsupervised training data in order to produce strong pseudo-labels for dual learning.

4.4.1 Effect of the Language Model

We note with interest that while the application of shallow fusion preserves the gains yielded by our method, further subtracting out the internal language model via HAT only partially preserves those gains. That is, subtracting out the internal language model substantially closes the gap between the baseline and our method. Figure 4.2 illustrates this effect by plotting WER for a parameter sweep of external LM interpolation (shallow fusion) weights and internal LM interpolation (HAT) weights. Quantitatively, subtracting out the internal LM with a factor of $\beta = 0.1$ from a model with shallow fusion improves **BASELINE** by 7.9%/6.2% while only improving **E-ALL** by 1.2%/0.6%.

This result suggests that our method largely benefits the internal language representation of the ASR system's decoder. In doing so, it is mostly redundant to contemporary language model fusion methods in the large-data regime. That's not to say that dual learning offers no benefits over fusion. Unlike conventional language model fusion, our method bakes knowledge of that data into the parameters of the decoder, providing much the same effect as combining the ASR system with a pretrained language model with no modifications to the architecture.

This result also suggests that the improvements due to our method come mostly, but not entirely, from the unsupervised text data, as opposed to the unsupervised audio. This is consistent with our design; unsupervised text yields pseudo-labeled examples for the ASR task, while unsupervised audio yields pseudo-labeled examples for the TTS task.



(a) Baseline, Test Clean **(b)** Baseline, Test Other **(c)** Baseline, LM Only

(d) Dual Learning, Test Clean **(e)** Dual Learning, Test Other **(f)** Dual Learning, LM Only

Figure 4.2: The effects of external LM (ELM) and internal LM (ILM) interpolation at inference time for our supervised baseline and our dual learning model. WER on three test sets is plotted for various settings of ELM and ILM interpolation weights.

4.4.2 Tail Analysis

Since unsupervised data is often used to address parts of the data distribution that are absent from the supervised training set, we seek to understand the effect of our method on "tail" words, which we define as words that are underrepresented in the training data relative to their frequency in the language as a whole. To this end, we make use of the **LM_C** tail set from Chapter 3, consisting of synthesized audio transcripts from a Tacotron TTS system as in Shen et al. (2017). Results on this tail test set are given in Table 4.1, and measurements of internal and external LM integration are included in Figure 4.2.

Model	Baseline	Shallow Fusion	Internal LM
BASELINE	16.1	13.8	12.9
E-ALL	15.3	12.6	12.4
E-DL	15.5	13.1	12.8
E-RECON	17.4	14.5	14.2

Table 4.2: WER percentage results on the synthesized Tail test set.

As above, we find that our method yields improvements across the board. However, we note that the improvements are smaller than they are on Librispeech test sets. In particular, without an LM we improve by 5.0%, with shallow fusion we improve by 8.7%, and with internal language model subtraction we improve by 3.9%. This perhaps reflects a domain mismatch, since the tail sets are derived from Google datasets instead of Librispeech's audiobooks, as described in 3.

4.5 Conclusion

On the surface, our experiments show that the usefulness of dual learning scales to a scenario with 960 hours of supervised audio. However, upon looking closer we see that the method is only marginally stronger then shallow fusion. This casts doubt on the understanding that methods like dual learning use unpaired data to improve encoder representations of audio, and suggests instead that they mostly improve the decoder's internal language model. It also leaves unsolved the problem of how to improve encoder representations with unpaired data in a manner that scales to large supervised datasets.

Chapter 5

Disentanglement

After having carefully assessed several semi-supervised ASR methods encompassing both speech and audio we have accumulated evidence that alternative loss criteria based on unsupervised modeling yield refinement only in the function of a model's decoder. We hypothesize that scaling these methods will require a deeper refinement that extends to the model's encoder representations. We ask what properties of encoder representations are both achievable with unpaired data and beneficial to downstream ASR performance.

With this question in mind, we undertake an abstract study of the representations formed by a dual learning model in a contrived, small learning setting. We provide evidence that the strength of unsupervised learning comes not only from exposure to new data, but also from the development of a joint representation with observable statistical properties. This understanding will motivate our approach towards improving semi-supervised ASR methods for the large-data setting.

5.1 Introduction

Before the proliferation of large language models trained by simple next-word prediction, advances in language processing were due mostly to auxiliary unsupervised tasks that developed a representation of text. Methods like ELMo Peters et al. (2018), GPT Radford and Narasimhan (2018), and BERT Devlin et al. (2018) work by using an unsupervised task that develops a representation of text that is useful for downstream tasks in a way that is agnostic to what that task is.

The jumping-off point for this study is the observation that in applying these lessons to unsupervised pretraining in speech, great progress has been made with the discovery that in a data-intensive domain like audio, it is best to learn a representation that discards unimportant parts of the signal. Contrastive estimation Gutmann and Hyvärinen (2010), in which a full reconstruction is not learned, has yielded representations that achieve strong results in speaker identification and speech recognition van den Oord et al. (2018). State-of-the-art methods combine contrastive learning with masked language modeling as in Wav2Vec 2.0 Baevski et al. (2020) and Adaptive SpecAugment Zhang et al. (2022).

Such successes can be seen as signaling a movement away from task-agnostic representations and towards "lossy" representations, in which a model learns not only to summarize relevant portions of a signal but also to discard portions that are irrelevant to the downstream task. This distinction is particularly clear in the world of multi-modal representation learning, where we seek a representation specifically of the intersection between two domains (e.g. audio and images Morgado et al. (2020); Peri et al. (2021) or audio and text Chung (2019)). However, while there are several

natural methods for learning a representation that models components of a signal that are required for a task, it is difficult to craft a method that compels a model to specifically exclude irrelevant components. Approaches in this space have commonly relied on techniques like adversarial learning to exclude particular parts of a signal thought to be irrelevant, as in Wang et al. (2021).

In this study, we develop a novel architecture specifically designed to learn a measurably disentangled representation of audio using supervised data. Our model is based on the paradigm of dual learning, which as described in Chapter 4 seeks to exploit the "duality" between ASR and TTS. Traditionally, this is done by training a model that performs both ASR and TTS with a shared encoder that is tasked with representing inputs from both the speech and text domains Ren et al. (2020); Xu et al. (2020). Our model adds a secondary encoder, which is intended to capture specifically those parts of the audio signal that are irrelevant to the transcript. While the primary encoder is utilized for both ASR and TTS, this secondary encoder is used only for audio reconstruction, which is a task that requires both that part of the audio signal that predicts the transcript and the "residual" signal that does not. We argue that disentanglement is facilitated by the explicit modeling of the residual signal by the secondary encoder, and demonstrate this disentanglement by training a speaker-ID classifier on the outputs of both the primary and secondary encoders.

Other studies have shown that in scenarios where more than one solution to an optimization problem is possible (such as generalized vs. overfit solutions Weber et al. (2018) and selection of significant units in a DNN Frankle and Carbin (2018)), the stochasticity of parameter initialization and minibatch selection can be decisive.

We present empirical evidence that speech signal disentanglement is such a problem. We find that both entangled and disentangled solutions to our dual learning problem are possible, and that the superior, disentangled solution is arrived at randomly. We then observe that the disentangled solution has the unique statistical property of using a large amount of its variational capacity in both encoders. Finally, we show that enforcing this property during training with an additional loss term substantially improves ASR quality.

Possible immediate applications of our joint modeling task include refinement of back-transcription based semi-supervised learning systems such as speech chains Tjandra et al. (2017) and Sequential MixMatch Chen et al. (2021b). However, we are most interested in probing to see what attributes of a joint speech and text representation contribute to WER improvements. We will make the observation that WER improvements in our setup come alongside greater "jointness" in the speech and text representation.

5.2 Architecture

In this section, we describe a joint ASR, TTS, and reconstruction model built on the dual-learning paradigm. Our architecture is depicted in Figure 5.1.

5.2.1 Architecture Summary

Our model is trained to consume either text or audio input, and to emit both text and audio. In that way, for a given input the model either performs ASR and

speech reconstruction (audio input) or TTS and text reconstruction (text input).

These tasks are performed by way of a pair of encoders, each of which yield a data representation. The "joint" (or primary) encoder can consume either audio or text, while the "audio-only" (or secondary) encoder consumes only audio input. There are also two decoders, one corresponding to each of the domains. The text decoder consumes the output of the joint encoder only, while the audio decoder consumes the outputs of both the joint and audio-only encoders, combined by way of a "embedding combination module", which consists simply of three transformer layers.

For tasks which lack text input (ASR and audio reconstruction), the joint encoder consumes zeros instead of text. For tasks which lack audio input (TTS and text reconstruction), the joint encoder consumes zeros instead of audio, and the embedding combination module consumes zeros instead of the outputs of the audio-only encoder.

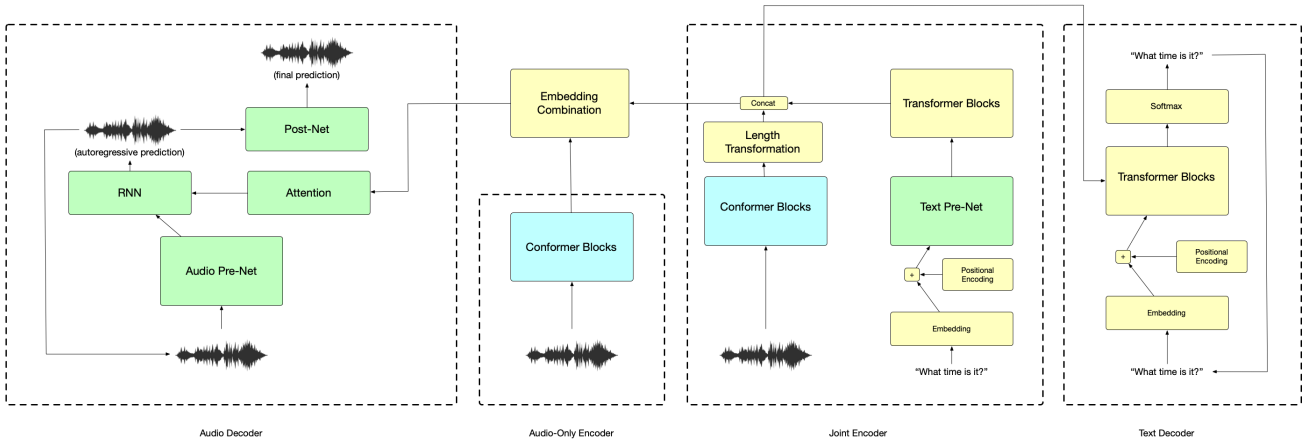


Figure 5.1: Our joint ASR/TTS model architecture, including an audio-only and joint encoder to support a disentangled representation. Blue components are adapted from Conformer Gulati et al. (2020), green components are adapted from Tacotron 2 Shen et al. (2017).

5.2.2 Encoder Architecture

The joint encoder is split into two sub-encoders, one to encode audio and one to encode text, each implementing a state-of-the-art encoding scheme. The audio sub-encoder is based on Conformer Gulati et al. (2020), and consists of 17 conformer blocks with sub-sampling so that the length of the audio input sequence is reduced by a factor of four. The text sub-encoder is based on Tacotron 2 Shen et al. (2017), and consists of a embedding projection and positional encoding followed by a pre-net and transformer module. As in Shen et al. (2017), the pre-net consists of three blocks of a 1D convolution with a 5x1 filter and a dropout layer that zeros out 10% of its input. The transformer block borrows from the original transformer architecture in Vaswani et al. (2017) and consists of three blocks of multi-headed self-attention followed by a feed-forward layer.

In order to produce a representation that is agnostic to the input domain, we would like to ensure the joint encoder emits a representation of approximately equal length for both domains. Otherwise, for example, the audio decoder might learn to model audio reconstruction and TTS separately based on encoding length. To this end, we adapt the length-transformation component from Shu et al. (2019) by which a representation is compressed into a shorter sequence where each element is a weighted average of elements from the original sequence. In particular, a sequence $z_1, ..., z_M$ is converted to to $\bar{z}_1, ..., \bar{z}_N$ (with $N < M$) as:

$$\bar{z}_j = \sum_{k=1}^{M} \sigma(\alpha_k^j) z_k$$
$$\alpha_k^j = -\frac{1}{2s}(k - \frac{|x|}{N} j)^2$$

where s is a learnable spread parameter and σ represents softmax normalization across all weights α_k^j for fixed j.

The audio-only encoder consists simply of four conformer blocks. These blocks do not include sub-sampling, so the output of the audio-only encoder is the same length as the audio input.

5.2.3 Decoder Architecture

The audio and text decoders are adapted from Tacotron 2 Shen et al. (2017) and Transformer Vaswani et al. (2017) respectively.

The audio decoder consists of a pre-net, autoregressive RNN, and post-net. The autoregressive component consumes its own previous output, and passes it though a simple audio pre-net which consists simply of a projection and dropout layer. We then attend to the outputs of the embedding combination module and concatenate the obtained context vector to the processed audio. This input is passed to a small recurrent network (two LSTM layers) which emits the autoregressive prediction. As in Wang et al. (2017), we find that tuning the dropout in the audio pre-net is critical to convergence, since without dropout in the autoregressive input the model simply learns to copy the previous frame. We find the best results with 10% dropout.

As in Shen et al. (2017), we find significant improvement in TTS when the autoregressive decoder output is further processed by a non-autoregressive convolutional post-net. We use a stack of five convolutions to refine the autoregressive prediction. During training, we jointly optimize the cross-entropy of both predictions.

The text decoder is a conventional Transformer Vaswani et al. (2017) decoder, consisting of two blocks each containing a projection, self-attention, and cross-attention.

5.3 Experiments

We've described how an audio input passed to our model is represented separately by the joint encoder and audio-only encoder. When optimized to perform the four tasks of ASR, TTS, and audio and text reconstruction, we may naturally imagine two classes of solutions that the model might arrive at:

- A "disentangled" representation, in which the joint encoder output (which will be consumed by the text decoder) represents that part of the audio signal relevant to the transcript, while the audio-only encoder output (which is only consumed by the audio decoder) represents that part of the audio signal that is not relevant to the transcript. For example, the joint encoder might represent phonetics, while the audio-only encoder might represent prosody, background noise, and channel effects.

- An "entangled" representation, in which that part of the audio signal relevant to transcription is not particularly favored by either representation.

We seek to observe which of these two representations is learned by our model. To this end, we train our model fifteen times on the given joint task, arriving at fifteen different solutions to the optimization problem. We then freeze the parameters of the model, and for each of the fifteen instances we train:

- A classifier to determine the speaker ID for a speech example given the model's joint encoder output.
- A classifier to determine the speaker ID for a speech example given the model's audio-only encoder output.

For a model that has learned a disentangled representation, we expect to be able to predict speaker ID best from the audio-only encoder output, since speaker information is ostensibly required for audio reconstruction but irrelevant to transcription.

5.3.1 Entanglement Classifiers

Speaker IDs are learned using a custom classifier that applies a positional embedding to the selected encoder output followed by three transformer blocks with multi-headed self-attention, five convolutions with a 3x3 filter and stride of 2 and finally a projection and softmax layer.

5.3.2 Model Settings

As in Shen et al. (2017) and Wang et al. (2017), we process audio inputs into mel spectrograms with a short-term Fourier transform (STFT) using a frame size of 50 ms

and frame hop of 12.5 ms. We then apply a Han windowing function before applying a mel filterbank, yielding 80-dimensional vectors for our model's audio input.

For text, we choose to use grapheme-level inputs such that the outputs of the embedding layers are 72-dimensional vectors. While a wordpiece representation might have yielded stronger ASR results, we found that graphemes most reliably ensured convergence of all tasks.

All components use a model dimension of 256, with the model containing about 68 million parameters in total. Each model is trained with a batch size of 256 split across 16 third-generation tensor processing units (TPUs). After 150k steps, we freeze the joint model and train each disentanglement classifier for 100k steps.

5.3.3 Training

To jointly optimize our four tasks, we split each batch into two halves. The first half consists of text inputs and represents the TTS and text reconstruction tasks, while the second half consists of audio inputs and represents the ASR and audio reconstruction tasks. For all elements in the batch, we optimize the loss

$$L = L_{\text{text}} + \frac{L_{\text{audio\_ar}}}{2} + \frac{L_{\text{audio\_final}}}{2}$$

where L_{text} is the cross-entropy loss for the text output, $L_{\text{audio\_ar}}$ is the cross-entropy loss for the audio output before the convolutional post-net, and $L_{\text{audio\_final}}$ is the cross-entropy loss for the audio output after the convolutional post-net. We

find this setup to train more quickly and to converge better than regimens in which tasks alternate across batches.

5.3.4 Data

As in Chapter 4 we train on Librispeech. We see Librispeech as ideal for this experiment since it contains a diverse set of speakers such that there is a significant part of the audio signal to represent outside of the transcript. We train our models in particular on the "clean" subset of the training data, which contains about 460 hours of speech. As we will describe below, we do this in order to allow for many training runs. For WER measurements we evaluate on the "clean" test set.

5.4 Results

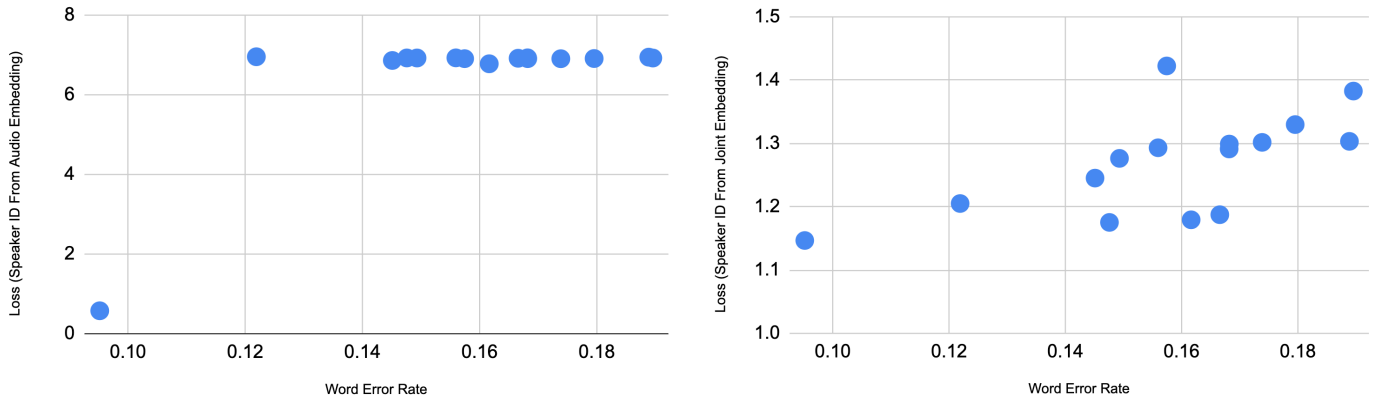
In this section, we report the results of our experiments and analyze the learned representations.

We point out that by the nature of this experiment, our model had to be trained from scratch many times, leading to considerable resource constraints. These constraints forced us to simplify the training procedure by using only the clean Librispeech data, a small batch size, and a small number of training steps. This combined with the additional TTS and reconstruction tasks leads to WER values considerably worse than the state of the art for ASR only. With that in mind, we draw conclusions based on the changes in WER and representation properties across different solutions.

5.4.1 Classification

Figure 5.2 plots the WER of the joint model against the two classification losses described above after training with frozen encoder parameters. We quickly make the observation that of our fifteen runs, one has an unusually strong result with a WER of about 9%.

The speaker ID classification task shows a clear pattern. The strongest model achieves a training loss that is more than ten times better than the next strongest model on the task from its audio embedding, and more than two times better than on its own joint embedding, suggesting that speaker information has been mostly disentangled from the transcript and localized to the audio embedding.



(a) Speaker ID from Audio

(b) Speaker ID from Joint

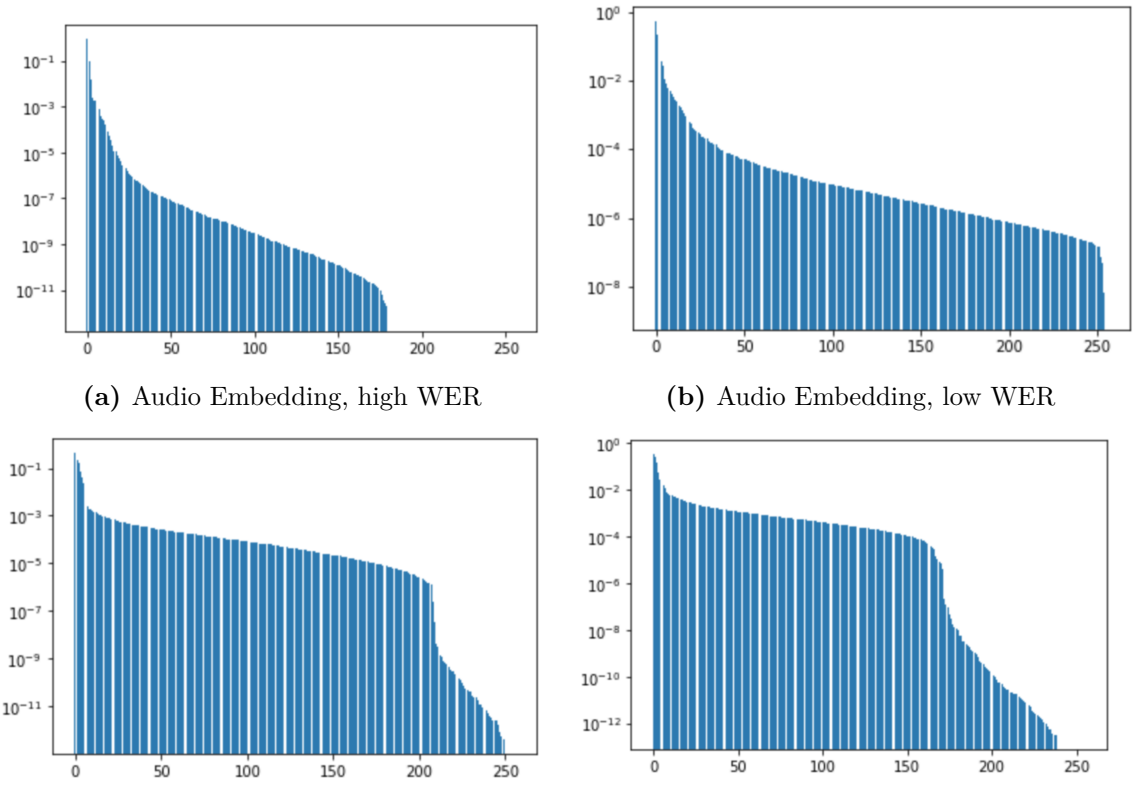
Figure 5.2: The relationship between WER and classifier training loss on the four disentanglement tasks measured. Each blue dot is a single converged solution for our architecture. The WER of that solution is plotted against the training loss of a speaker-id classifier trained on either the audio-only or joint encoder output. We see that the solution that achieves the best WER is also the only solution for which speaker ID is successfully derived from the audio-only representation.

5.4.2 Representation Properties

Having seen that our model can sometimes, subject to the randomness of training, achieve a much better WER than average, we seek to understand the nature of that stronger, disentangled representation. In particular, we suspect that in a model without the desired disentanglement, the audio-only embedding is underused.

To this end, we sample the audio-only and joint representations of our best model and of one of our other models. Since each input contains a large number of frames, we are able to collect several thousand 256-dimensional vectors from just a few examples. For each representation, we perform an SVD on those vectors and normalize the squared singular values. In this manner, we obtain a measurement of the proportion of variance in the representation attributable to each of its 256 dimensions. We consider a representation with significant variance in a large number of dimensions to be more used by a model than one in which only a few dimensions vary.

The results of these measurements are plotted in Figure 5.3. We see a stark difference between the distribution of variance in a weak, non-disentangled representation and our strong, disentangled representation. In particular, the disentangled solution has very few significant dimensions in its audio-only embedding, with the first three dimensions capturing more than 95% of the variance. By contrast, the disentangled solution has a much larger number of significant dimensions, with almost 50 dimensions containing more than 0.1% of the total variance each.



(a) Audio Embedding, high WER

(b) Audio Embedding, low WER

(c) Joint Embedding, high WER

(d) Joint Embedding, low WER

Figure 5.3: The distribution of the singular values of encoder output frames. Each encoder is considered with a high WER and low WER solution. The squared values of the singular values are plotted in decreasing order to illustrate the number of dimensions explaining a significant portion of the embedding's variance.

5.4.3 Correlation Loss

Having observed that strong performance occurs together with a relatively uncorrelated audio-only embedding, we naturally wonder if optimizing for that property at training time will yield better WER. To test this, we interpolate an additional "correlation loss" into the the training of our joint model:

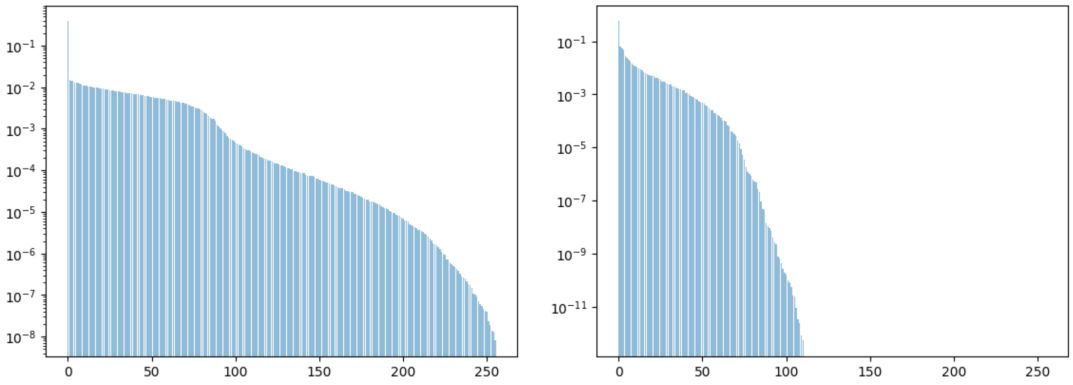
$$L_{\text{corr}} = \alpha \sum_{b \in B} \sum |corr[A \cdot X, A \cdot X] - I|$$

where $b \in B$ are the elements in the batch, X is the matrix formed by stacking the audio representation of the batch element along the time axis, A is a learnable linear projection, I is the identity matrix, and the inner summation adds up each (unsigned) element of the given matrix. This loss is intended to act as regularization that pushes the off-diagonal elements of the correlation matrix to zero, yielding a representation with uncorrelated elements. We achieve the strongest results setting the hyperparameter $\alpha = 10^{-5}$.

Model	Average WER
Non-Disentangled	15.5%
Disentangled	9.8%
Correlation Loss	11.7%

Table 5.1: WER with and without the Correlation Loss

The distribution of singular values in Figure 5.4 shows clearly that the added loss has the intended effect of decorrelating the audio embedding. It also suggests that this is done by moving information over from the joint embedding, which has become lower-dimensional. WER results are given in Table 5.1. We see that the correlation loss yields on average a 24.5% reduction in WER.



(a) Audio Embedding, with L_{corr} **(b)** Joint Embedding, with L_{corr}

Figure 5.4: Singular value distributions after training with the correlation loss.

5.5 Conclusions

In this chapter, we presented a novel dual-learning architecture capable of learning a disentangled representation of audio. We associated disentanglement directly with strong performance on the ASR task and with a high-dimensional audio embedding.

One can imagine future work in semi-supervised ASR that will train our dual-learning model on supervised data to learn a disentangled audio representation which can then be fine-tuned with both unpaired audio and text data via back-transcription. For the time being, however, we take a broader lesson from these results, which is that exposing the model to more data is likely not the entire story of the usefulness of unpaired data in ASR. Rather, the auxiliary tasks in fact lead to a unification of the domain representations on the *encoder level*, and that the degree of jointness in the representation directly corresponds to better ASR performance. This is a departure from the conventional understanding of semi-supervision that the contribution made by unpaired data is to expand coverage of the data distribution. It also brings to

our attention a consideration that is not directly addressed in the design of the semi-supervised methods we studied in Chapters 3 and 4.

Chapter 6

The Best-Alignment Method

Our experiments in Chapter 5 offered an optimistic outlook on semi-supervised ASR by offering evidence that unpaired data improves downstream WER to the degree that an encoder learns to jointly represent speech and text. This motivates the pursuit of such representations in the methods studied in Chapters 3 and 4.

In this chapter, we ask if the results of Chapter 5 offer a path towards scaling WER improvements. Constraining ourselves to text injection in the context of JOIST Sainath et al. (2022), we design a training algorithm that requires the acoustic encoder not only to represent text (as is required in JOIST), but to represent it *jointly* with the speech domain. We find WER improvements over JOIST in the large-data setting of Chapter 3 as well as in a large-data multilingual setting.

6.1 Introduction

The power of very large models trained on vast unsupervised corpora in a single modality has become increasingly clear. This has been demonstrated in the text domain where language models have achieved unprecedented zero-shot capabilities Brown et al. (2020); Chowdhery et al. (2022), as well as in the audio domain, in which a single model has been shown to be adaptable to a surprisingly wide array of acoustic tasks Borsos et al. (2022); Yang et al. (2021). These successes have given rise to the question of how to apply these methods for problems involving two modalities, which historically have depended on manually paired data.

One very promising solution to this problem is to train a large encoder on both modalities, such that either modality may be provided as an unpaired example, but which learns to map paired examples to similar points in representation space. In the image/text domain, such a representation has proved achievable and capable of attaining state-of-the-art performance on many image and text comprehension tasks in a single model Alayrac et al. (2022); Cho et al. (2021).

In the audio/text domain, joint speech and text models have been utilized for a wide range of tasks Huang et al. (2020); Mariooryad et al. (2022); Renduchintala et al. (2018). In speech recognition, the past few years has seen a trend toward models with a joint speech and text encoder to allow pretraining on unpaired speech and text data Bapna et al. (2020); Chen et al. (2022a); Sainath et al. (2022); Tang et al. (2021). However, speech recognition presents the particular challenge of two sequence modalities, one of which (speech) is typically represented by a much longer sequence than the other (text). This complicates the task of representing both modalities in

the same embedding space, since we cannot make a direct, frame-wise comparison of an encoder's speech representation to its text representation.

This complication has largely been handled either by upsampling or an explicit alignment model. Fixed upsampling of the text inputs has been applied successfully for ASR in Sainath et al. (2022) and SLU in Thomas et al. (2022), proving that an approximate alignment is sufficient for learning a joint representation. On the other hand, Chen et al. (2022b) addresses the problem with a separately-trained alignment model that aims for perfect alignment. In Chen et al. (2022a), it's shown that such an alignment model permits the use of "consistency" regularization in which the encoder's outputs on corresponding speech and text are compared frame-wise and pushed together in representation space. Chen et al. (2022a) goes on to show that "consistency" regularization yields a more closely joined representation space leading to better WER.

Consistency regularization itself follows naturally from the literature on generative models. Systems like autoencoders applied to augmented data (e.g. Chadebec et al. (2022)) explicitly push representations of matched examples together, while contrastive systems like Chen et al. (2020) do the same implicitly. The success of the same idea in speech using an explicit alignment begs the question of if the same can be done with an implicit alignmentment; that is, without knowing the particular alignment between speech and text.

In this paper, we ask if consistency regularization may be applied using the implicit alignments learned in upsampling systems like Sainath et al. (2022) to achieve the performance improvements seen with the explicit alignments in Chen

et al. (2022a). To this end, we develop an algorithm inspired by dynamic time warping Sakoe and Chiba (1978) that finds the *best possible alignment* between an encoder's representation of a paired speech and text example. We measure the quality of this *best alignment* in a system without an explicit alignment model and show that that it is not only learned during training but in fact improves at deeper layers of the network. Inspired by the improvements shown in Chen et al. (2022b) and Chen et al. (2022a), we then show that by changing the criteria of the consistency regularization to encourage consistency under *some* alignment, instead of a direct frame-wise comparison, we can achieve robust WER improvements against strong, semi-supervised baselines in both a monolingual and multilingual setting, all without any learned alignment model. Our results suggest that enforcing consistency in cross-modal representations can be done by simply forgiving misalignment.

6.2 Methods

In this section we present our setup for semi-supervised ASR by joint speech/text modeling, for which we mostly follow Sainath et al. (2022). We then present our proposed best-alignment algorithm and define a corresponding consistency loss inspired by Chen et al. (2022b).

6.2.1 Model Architecture

Figure 6.1 gives our model architecture. Essentially, we perform supervised ASR with streaming and non-streaming decoders, where the encoder is split into "audio-

only"/"text-only" and "shared" components to permit text injection. The simultaneous ASR and text-injection tasks give rise to a joint representation in the shared encoder. Specifically, given a corpus of supervised examples $(x, y) \in \mathcal{S}$ and an unpaired text corpus $y \in \mathcal{U}$, our model contains the following components:

- E_a: The audio encoder, which embeds audio features x.
- E_t: The text encoder, which embeds text features y.
- E_s^{C}: The shared streaming encoder, which may consume either $E_a(x)$ or $E_t(y)$ and maps them to a joint representation. Since this encoder is "streaming", it only receives past acoustic frames.
- E_s^{NC}: The full-context encoder, which consumes the outputs of E_s^{C} and which is given forward acoustic frames.
- D^{C}: The streaming decoder, which consumes the outputs of E_s^{C} and emits streaming ASR hypothesis.
- D_s^{NC}: The non-streaming decoder, which consumes the outputs of E_s^{NC} and emits non-streaming ASR hypothesis.

Our model is trained simultaneously on two tasks: ASR, and masked text reconstruction. For ASR, audio is passed into audio encoder, and hypotheses are compared against ground truth text with the conventional cross-entropy loss. Masked text reconstruction makes use of unpaired text data. A mask (15% of the transcript) is

applied to a phonemic representation of text, which is then passed into the text encoder. The hypothesis is compared to the masked portion of the input again with a cross-entropy loss.

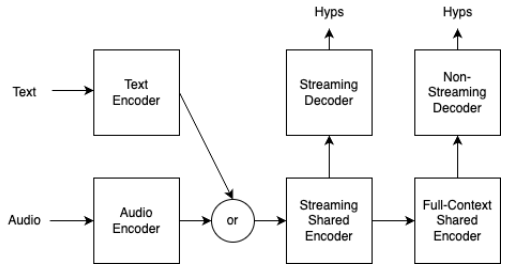


Figure 6.1: Our architecture for semi-supervised ASR, adapted from Sainath et al. (2022) and Chen et al. (2022a).

6.2.2 Consistency Loss

Consider a paired example (x, y), where $x = (x_0, ..., x_n)$ are speech inputs and $y = (y_0, ..., y_m)$ are text inputs and where $n > m$. Let us define the shared representations of audio and text as

$$R_a = E_s(E_a(x)) \quad R_t = E_s(E_t(y))$$

where E_s can represent either the application of only E_s^{C} (for a streaming representation) of E_s^{C} followed by E_s^{NC} (for a non-streaming representation). A "consistency loss", as developed in Chen et al. (2022a) and Chen et al. (2022b), is some loss $\mathcal{L}^{\text{consistency}}(R_a, R_t)$ that measures the similarity of the two representations.

Since the audio x and the text y are sequences of different lengths, we require some notion of an alignment to define a meaningful consistency loss. By alignment,

we mean a specific up-sampling of y such that each audio frame $x[i]$ will correspond to some text frame $y[j]$. With this in mind, we define an alignment $\mathcal{A} = (a_0, a_1, ..., a_n)$ as a list of indexes into y, such that for any audio frame i, $x[i]$ corresponds to $y[a_i]$ in the alignment. We will also add the constraint that $a_i \leq a_{i+1}$ for all i. That is, we constrain $\mathcal{A}$ to be monotonically increasing, so that sequential audio frames may not correspond to text backwards.

This formulation is one of many conceivable ways to define an "alignment" and we've chosen it for the practicality it offers in efficiently computing the best alignment (see Section 6.2.3 below). We note that in this formulation, each audio frame is considered exactly once, while each text frame can be repeated or skipped over entirely.

With this definition in mind, we define the consistency loss for a given alignment as

$$\mathcal{L}_{\mathcal{A}}^{\text{consistency}}(R_a, R_t) = \sum_{x=0}^{n} \frac{\mathcal{L}^{\text{frame}}(R_a[x], R_t[\mathcal{A}[x]])}{n}$$

where $\mathcal{L}^{\text{frame}}$ is some frame-wise similarity measure (in this work, we use L2). That is, $\mathcal{L}_{\mathcal{A}}^{\text{consistency}}(R_a, R_t)$ gives the average frame-wise similarity between R_a and the specific up-sampling of R_t given by $\mathcal{A}$.

The setups in Chen et al. (2022a) and Chen et al. (2022b) use such a consistency loss successfully, taking $\mathcal{A}$ from a neural alignment model. We propose, as an alternative, to optimize the consistency over the *best possible* alignment:

$$\mathcal{L}^{\text{consistency}}(R_a, R_t) = \min_{\mathcal{A}} \mathcal{L}_{\mathcal{A}}^{\text{consistency}}(R_a, R_t)$$

In order to train with such a loss, we require an efficient algorithm to compute the best alignment.

6.2.3 Computing the Best Alignment

Dynamic time warping Sakoe and Chiba (1978) relies on an inductive rule in order to define a recursive algorithm to match two sequences based on a cost function. We do the same, specifying the cost:

$$\mathcal{C}(i,j) = \min_{\mathcal{A}} \mathcal{L}^{\text{consistency}}(R_a[:i], R_t[:j])$$

That is, the cost $\mathcal{C}(i,j)$ gives the consistency loss under the best alignment between the prefix of the audio representation up to the index i and the prefix of the text representation up to the index j. We may then specify an inductive rule:

$$\mathcal{C}(i,j) = \min_{k \leq j} \left[\mathcal{C}(i, k-1) + \mathcal{L}^{\text{frame}}(R_a[i], R_t[k]) \right]$$

That is, the best alignment for the prefixes $R_a[:i]$ and $R_t[:j]$ aligns the previous $i-1$ audio frames to some shorter prefix $R_t[:k]$, and then appends to it the specific alignment of $R_a[i]$ to $R_t[k]$.

We may back out the indexes of the best alignment from this computation. This rule gives rise to a dynamic programming algorithm for finding the best alignment in $\mathcal{O}(nm^2)$ time and memory.

We note that the minimization across all alignments precludes differentiation of the alignment-finding. Instead, we compute the best alignment during forward-

propagation, and then differentiate $\mathcal{L}^{\text{frame}}$ as applied to the aligned frames. That is, we use the pass-through approximation of the gradient:

$$\frac{\partial \mathcal{L}^{\text{consistency}}(R_a, R_t)}{\partial \theta} \approx \frac{\partial \mathcal{L}^{\text{consistency}}_{\mathcal{A}^*}(R_a, R_t)}{\partial \theta}$$

where

$$\mathcal{A}^* = \arg\min_{\mathcal{A}} \mathcal{L}^{\text{consistency}}_{\mathcal{A}}(R_a, R_t)$$

6.3 Experiments

In this section, we provide details of our model settings and data.

6.3.1 Model Settings

We specify component's parameterizations according to the list in Section 2.1:

- E_a: The audio encoder consists of a single conformer Gulati et al. (2020) layer with 8 attention heads and dimension 2048. The audio encoder consumes 128 dimensional log-mels spanning 32ms each and spaced apart by 10ms. We then stack each frame with the frame before it and the two frames after it to yield 512 dimensional representations. Finally, we subsample by taking each third frame, yielding a final frame rate of 30ms.

- E_t: The text encoder consists of a embedding projection followed by a conformer layer. As in Sainath et al. (2022), we find it necessary to supply the

text encoder with phonemic representations of text transcripts. We then continue to follow Sainath et al. (2022) by repeating each phoneme twice as an alignment heuristic.

- E_s^{C}: The shared streaming encoder consists of five conformer layers, with layer-norm applied at the end.
- E_s^{NC}: The full-context shared encoder consists of nine additional conformer layers, with layer-norm applied at the end.
- D^{C}: The streaming decoder is a HAT decoder Variani et al. (2020) in which both the prediction and joint layers have dimension 640.
- D_s^{NC}: The non-streaming decoder, is identical to the streaming decoder.

Together, our model contains about 165M parameters. Training is done in two phases. First, the audio encoder, joint encoders, and decoders are all trained on paired data for 800k steps with a batch size of 2048. The text encoder is then added and the model is further trained with equally weighted supervised and unsupervised loss as described in Section 2.1, with the best alignment loss from Section 2.3 optionally included. The model trains in this manner for 100k further steps with a batch size of 2048 for both the supervised and unsupervised data.

All models are implemented in Tensorflow, with the best alignment algorithm itself implemented as a CPU kernel. We find that the addition of the best alignment computation does not significantly increase training time over the baseline model.

6.3.2 Datasets

Text-injection methods in ASR have historically been applied in two broad settings. Strong baselines are often fine-tuned with very large text corpora to improve performance on difficult words. Alternatively, text-injection may be used for models trained on limited supervised data to improve the internal language model and get closer to a viable system. With these two settings in mind, we study the best alignment loss in two setups:

- A large English corpus consisting of about 200k hours of supervised speech, together with an unsupervised text dataset of about 200B examples. Both are internal Google datasets composed, with the supervised dataset largely of voice-search transcriptions and the unsupervised text dataset gathered from several domains.

  We report results for a **Main** test set derived from the same distribution as the training examples, and a **Noisy** test set of especially noisy examples.

- A multilingual corpus consisting of the following eleven languages: English (**En**), French (**Fr**), Spanish (**Sp**), Arabic (**Ar**), Portuguese (**Po**), German (**De**), Russian (**Ru**), Hindi (**Hi**), Italian(**It**), Mandarin, and Japanese.

  This setting involves no unsupervised text, with the MLM objective applied instead to the supervised transcripts. The dataset consists of about 140M paired examples.

  Bolded abbreviations are given above for languages for which we are able to report WER in 6.3. For simplicity with the large number of test sets, we report

only non-streaming WER from this model.

All datasets are anonymized and human transcribed.

6.4 Results

In this section, we seek to demonstrate that even without consistency regularization, our model learns an alignment between paired speech and text examples. We then seek to show that optimizing this alignment with our proposed best-alignment consistency regularization improves WER.

6.4.1 Best-Alignment in an Unregularized Model

For this analysis, we use our baseline model from the monolingual setup as described in Section 6.3.1. Our objective is to measure $\mathcal{L}^{\text{consistency}}$ on a small set of random development examples for R_s and R_t taken at each of the first five conformer layers of the streaming joint encoder. We interpret a lower value for $\mathcal{L}^{\text{consistency}}$ as reflecting a stronger implicit alignment between speech and text.

For each layer l of the five-layer conformer encoder we sample 2000 random pairs of audio and text embeddings and compute the mean μ_l and standard deviation σ_l^2 of the distribution of distances between pairs. We then compare two alignments: the naive frame-wise alignment and our computed best alignment. For each of these alignments $\mathcal{A}$, we report:

$$\mu_l - \frac{\mathcal{L}_{\mathcal{A}}^{\text{consistency}}(R_s, R_t)}{\sigma_l^2}$$

That is, we report the consistency in terms of the number standard deviations away from the mean, such that a result of 0 suggests that the alignment is no better than random and a result below 0 suggests that the alignment is stronger than random.

Table 6.1: Consistency of the linear and best alignments at layers of the shared encoder.

Layer	Frame-wise Alignment	Best Alignment
1	-0.06	-1.47
2	-0.23	-2.15
3	-0.29	-2.61
4	-0.37	-2.67
5	-0.49	-3.06

Table 6.1 presents these measurements. We see that while the consistency of the frame-wise alignment is close to that of the random alignment, the best alignment is considerably better than random. Furthermore, the quality of the best alignment improves steadily as we progress deeper into the model. That is, while text and speech are not modeled jointly at the frame level, there is *some* alignment for which paired speech and text are mapped to similar points in the embedding space, and this alignment improves with the depth of the network.

To illustrate the presence of this alignment, we visualize the relationship between shared encoder's final representations of the speech and text from a single test example. Figure 6.2a plots the distance between each pair of frames in the embeddings, and demonstrates that is indeed a single alignment with low distance. Figure 6.2b shows how the best alignment algorithm recovers this trajectory.

	Main	Noisy
E_0	5.40	8.75
E_10	5.37	8.70
E_1	5.35	**8.42**
E_0.1	**5.27**	8.77
E_0.01	5.32	8.54

(a) Non-Streaming

	Main	Noisy
E_0	7.99	13.33
E_10	8.21	13.08
E_1	8.07	13.00
E_0.1	**7.90**	**12.63**
E_0.01	7.94	12.74

(b) Streaming

Table 6.2: Evaluation Results for the English-Only Setting.

6.4.2 Consistency Regularization Results

We present results for the best-alignment loss at different interpolation weights and for both of the settings specified in Section 6.3.2. For ease of reading, we specify each experiment by a letter and a number. The letter is either **E** for the English-only setting of **M** for the multilingual setting. The number is the interpolation weight of the best-alignment loss as a percentage. For example, **E_0** is the baseline English-only model with unregularized semisupervised finetuning, while **M_0.1** is

	En	Fr	Sp	Ar	Po	De	Ru	Hi	It
M_0	9.1	10.6	6.4	12.6	7.9	14.8	13.0	19.7	10.3
M_10	**8.5**	10.4	**5.8**	**11.8**	**7.7**	**13.4**	**12.5**	19.4	**9.8**
M_1	**8.5**	10.5	6.1	11.9	8.1	13.9	12.7	**19.3**	10.0
M_0.1	8.6	**10.3**	6.2	12.1	8.0	13.9	12.9	19.5	9.9
M_0.01	8.8	10.5	6.3	12.2	7.9	14.0	13.0	19.6	10.3

Table 6.3: Evaluation Results for the multilingual setting. Each language is evaluated for various interpolation weights of the best-alignment loss.

a multilingual model with the best alignment loss interpolated during finetuning at 0.1 percent.

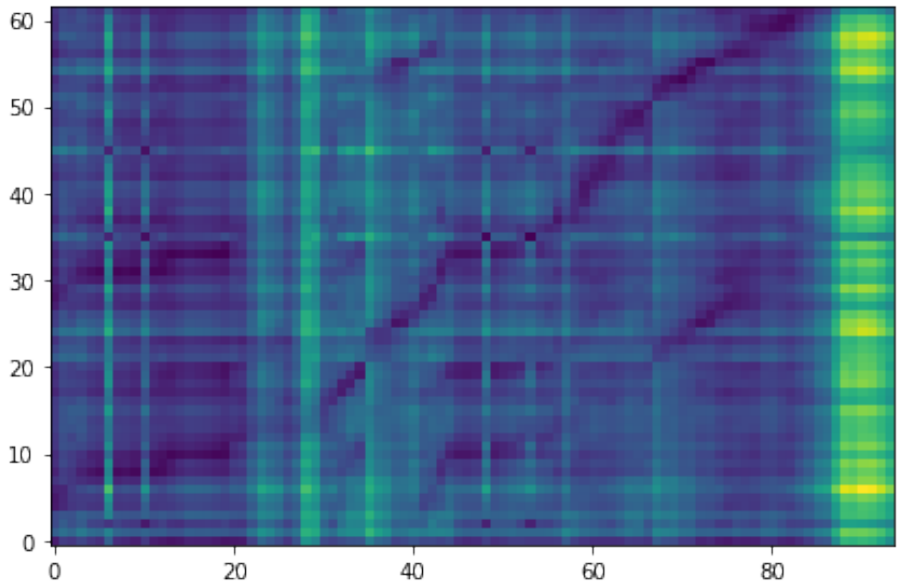
Table 6.2 gives our results in the high-resource, English-only setting. There, we see small but consistent WER improvements with the best-alignment loss, although we note the necessity of selecting the correct interpolation weight. Table 6.3 gives our results in the multilingual setting, where we see larger improvements. We believe that the strength of the method in the multilingual setting is due to the increased difficulty of the problem and the smaller dataset leaving more room for the model to improve.

6.5 Conclusions

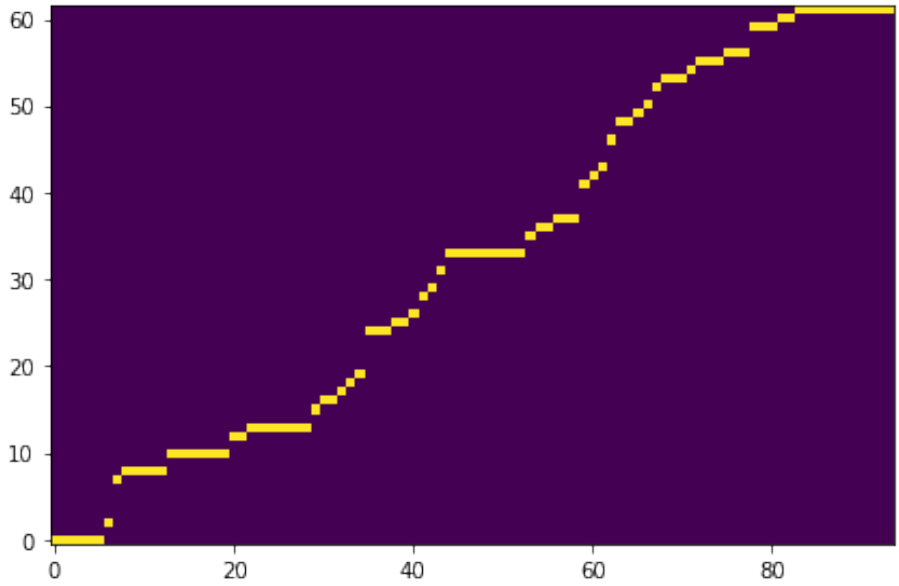
We've shown that a semi-supervised speech/text encoder learns a joint representation of the two modalities that can be observed by choosing the best alignment. We've exploited that fact to enforce domain consistency with an extra loss term which optimizes the modality match for the best alignment. We show consistent improvements over an unregularized joint model across multiple large-data setups

without adding any parameters.

We believe these results bear out the intuitions developed in Chapter 5 that a joint speech and text representation is the effect of semi-supervised training that yields downstream WER improvements. In demonstrating how this intuition applies in practice, we hope to provide a mechanism for using unpaired text to improve an acoustic encoder, even when that encoder is trained on a very large supervised corpus.



(a) Distances



(b) Best Alignment

Figure 6.2: Visualizations of embedding distances (a) and the best alignment (b) between an audio embedding on the horizontal axis and the corresponding text embedding on the vertical axis. Darker points in (a) represent pairs of audio and text frames with nearby embeddings, and yellow points in (b) represent pairs in the recovered best alignment.

Chapter 7

Conclusion

This work is premised on the observation that the literature on semi-supervised ASR has largely focused on the small-data setting, and that many of the methods that have been shown to be effective in that setting do not offer the same WER improvements against a strong supervised baseline. We began our study by explicitly making this observation in a side-by-side implementation of leading methods applied to a deployable, two-pass streaming model pre-trained on Google's large English supervised dataset. We found that none of the methods in question improved WER on the head of the data distribution. However, we discovered several advantageous properties of the models trained with semi-supervised data that not only offer justification to the use of the methods but also shed light on what effect they have. In particular, our results suggest that semi-supervised training in the large-data setting refines the decoder, but does not meaningfully improve encoder representations.

With this in mind, we moved off the beaten path to make a similar study of a semi-

supervised method called dual learning that has seen success in other domains but has only begun to be applied to speech recognition. This method is designed to pass supervised data, unpaired text, and unpaired audio through the model's encoder, and we hypothesized in light of our previous observations that it would scale to the large-data setting. We were, however, disappointed to find that it performs similarly to other methods, and that the benefit it yields is largely redundant with a careful application of shallow fusion. That is, contrary to the prevailing understanding of dual learning, in the large-data setting it is mostly a decoder-only method.

This result forced us to step back and to try to understand what it means exactly to refine an encoder representation. We made this study by attempting to disentangle the encoder representation of a dual learning model, and observed that the property of disentanglement, when it arises, substantially improves model performance. This led us to the counter-intuitive hypothesis that in the large-data setting, semi-supervised learning offers utility in the encoder not from exposure to yet more data, but through the unification of the trained model's representations of audio and text.

With this hypothesis in mind, we reassessed one of the methods measured at the beginning of our study and confirmed that it does not unify encoder representations of audio and text. We designed a regularization to promote this unification, and after retraining the model found that performance significantly improves even against a baseline trained on Google's large supervised dataset, and that this result holds true in a large-scale multilingual experiment as well.

While we end our study here, this research direction has several promising contin-

uations. One clear direction is the application of domain-consistency regularizations to existing semi-supervised methods in order to scale them to the large-data setting. While we emphasized JOIST, our findings apply to any method in which text is encoded alongside audio, such as MAESTRO Chen et al. (2022a) and cold fusion Sriram et al. (2018). It might also be applied to TTS augmentation by aligning and matching the encoder's representation of text with the corresponding representation of the synthesized audio. Also, while our solution focused on the matching of audio and text representations, our results suggest that aligned representation matching would be useful in other methods in which two "views" of the same data are provided with a goal of being processed similarly. For example, audio recordings modified via SpecAugment Park et al. (2019) could be pushed together at the encoder representation level.

Since the bulk of this research was concluded in early 2023, the speech community has moved even more towards semi-supervised learning, with foundation models trained on large unsupervised audio corpora dominating novel architectures. The unification of such representations with correspondingly sophisticated representations of text has the potential to power further advances in ASR. For example, we believe that domain unification can provide a path forward in multi-lingual ASR by improving the incorporation of low-resource languages based on only unpaired audio and text. Unified audio and text representations can clear a path for the incorporation of large generative language models in ASR by projecting input audio into a shared text representation. We hope that this work will contribute towards a richer understanding of the potential in this unification of domains.

Bibliography

Alayrac, J.-B., Donahue, J., Luc, P., Miech, A., Barr, I., Hasson, Y., Lenc, K., Mensch, A., Millican, K., Reynolds, M., Ring, R., Rutherford, E., Cabi, S., Han, T., Gong, Z., Samangooei, S., Monteiro, M., Menick, J., Borgeaud, S., Brock, A., Nematzadeh, A., Sharifzadeh, S., Binkowski, M., Barreira, R., Vinyals, O., Zisserman, A., and Simonyan, K. (2022). Flamingo: a visual language model for few-shot learning. In *Conference on Neural Information Processing Systems (NeurIPS)*.

Baevski, A., Hsu, W., Conneau, A., and Auli, M. (2021). Unsupervised speech recognition. In *Conference on Neural Information Processing Systems (NeurIPS)*.

Baevski, A., Schneider, S., and Auli, M. (2019). vq-wav2vec: Self-supervised learning of discrete speech representations. In *International Conference on Learning Representations (ICLR)*.

Baevski, A., Zhou, H., Mohamed, A., and Auli, M. (2020). wav2vec 2.0: A framework for self-supervised learning of speech representations. In *Conference on Neural Information Processing Systems (NeurIPS)*.

Bahdanau, D., Cho, K., and Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *ArXiv*, 1409.

Bai, J., Li, B., Zhang, Y., Bapna, A., Siddhartha, N., Sim, K. C., and Sainath,

T. N. (2021). Joint unsupervised and supervised training for multilingual ASR. In *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*.

Banko, M. and Brill, E. (2001). Scaling to very very large corpora for natural language disambiguation. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics*, pages 26–33, Toulouse, France. Association for Computational Linguistics.

Bapna, A., Cherry, C., Zhang, Y., Jia, Y., Johnson, M., Cheng, Y., Khanuja, S., Riesa, J., and Conneau, A. (2020). mslam: Massively multilingual joint pre-training for speech and text.

Bapna, A., Chung, Y., Wu, N., Gulati, A., Jia, Y., Clark, J. H., Johnson, M., Riesa, J., Conneau, A., and Zhang, Y. (2021). SLAM: A unified encoder for speech and language modeling via speech-text joint pre-training. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Baskar, M. K., Watanabe, S., Astudillo, R., Hori, T., Burget, L., and Černocký, J. (2019). Semi-supervised sequence-to-sequence asr using unpaired speech and text. In *INTERSPEECH*.

Bengio, S., Vinyals, O., Jaitly, N., and Shazeer, N. (2015). Scheduled sampling for sequence prediction with recurrent neural networks. In *Advances in Neural Information Processing Systems*.

Borsos, Z., Marinier, R., Vincent, D., Kharitonov, E., Pietquin, O., Sharifi, M., Teboul, O., Grangier, D., Tagliasacchi, M., and Zeghidour, N. (2022). Audiolm: a language modeling approach to audio generation.

Botros, R., Sainath, T. N., David, R., Guzman, E., Li, W., and He, Y. (2021). Tied & reduced RNN-T decoder. In *Interspeech*.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakan-

tan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. (2020). Language models are few-shot learners. In *Conference on Neural Information Processing Systems (NeurIPS)*.

Chadebec, C., Thibeau-Sutre, E., Burgos, N., and Allassonniere, S. (2022). Data augmentation in high dimensional low sample size setting using a geometry-based variational autoencoder. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Chan, W., Jaitly, N., Le, Q. V., and Vinyals, O. (2015a). Listen, attend and spell.

Chan, W., Jaitly, N., Le, Q. V., and Vinyals, O. (2015b). Listen, attend and spell. In *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*.

Chen, S., Wang, C., Chen, Z., Wu, Y., Liu, S., Chen, Z., Li, J., Kanda, N., Yoshioka, T., Xiao, X., Wu, J., Zhou, L., Ren, S., Qian, Y., Qian, Y., Wu, J., Zeng, M., and Wei, F. (2021a). Wavlm: Large-scale self-supervised pre-training for full stack speech processing. In *Journal of Selected Topics in Signal Processing (JSTSP)*.

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. E. (2020). A simple framework for contrastive learning of visual representations. In *International Conference on Machine Learning (ICML)*.

Chen, Z., Rosenberg, A., Zhang, Y., Zen, H., Ghodsi, M., Huang, Y., Emond, J., Wang, G., Ramabhadran, B., and Moreno, P. J. (2021b). Semi-supervision in asr: Sequential mixmatch and factorized tts-based augmentation. In *Interspeech*.

Chen, Z., Zhang, Y., Rosenberg, A., Ramabhadran, B., Moreno, P., Bapna, A., and Zen, H. (2022a). Maestro: Matched speech text representations through modality matching. In *INTERSPEECH*.

Chen, Z., Zhang, Y., Rosenberg, A., Ramabhadran, B., Moreno, P., and Wang, G. (2022b). Tts4pretrain 2.0: Advancing the use of text and speech in asr pretraining with consistency and contrastive losses. In *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*.

Chen, Z., Zhang, Y., Rosenberg, A., Ramabhadran, B., Wang, G., and Moreno, P. J. (2021c). Injecting text in self-supervised speech pretraining. In *IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*.

Chiu, C., Qin, J., Zhang, Y., Yu, J., and Wu, Y. (2022). Self-supervised learning with random-projection quantizer for speech recognition. In *International Conference on Machine Learning (ICML)*.

Chiu, C., Sainath, T. N., Wu, Y., Prabhavalkar, R., Nguyen, P., Chen, Z., Kannan, A., Weiss, R. J., Rao, K., Gonina, K., Jaitly, N., Li, B., Chorowski, J., and Bacchiani, M. (2017). State-of-the-art speech recognition with sequence-to-sequence models.

Cho, J., Lei, J., Tan, H., and Bansal, M. (2021). Unifying vision-and-language tasks via text generation. In *International Conference on Machine Learning (ICML)*.

Cho, K., van Merrienboer, B., Bahdanau, D., and Bengio, Y. (2014). On the properties of neural machine translation: Encoder-decoder approaches.

Chorowski, J., Weiss, R. J., Bengio, S., and van den Oord, A. (2019). Unsupervised speech representation learning using wavenet autoencoders. In *IEEE/ACM Transactions on Audio Speech and Language Processing*.

Chowdhery, A., Narang, S., Devlin, J., Bosma, M., Mishra, G., Roberts, A., Barham, P., Chung, H. W., Sutton, C., Gehrmann, S., Schuh, P., Shi, K., Tsvyashchenko, S., Maynez, J., Rao, A., Barnes, P., Tay, Y., Shazeer, N., Prabhakaran, V., Reif, E., Du, N., Hutchinson, B., Pope, R., Bradbury, J., Austin, J., Isard, M., Gur-Ari, G., Yin, P., Duke, T., Levskaya, A., Ghemawat, S., Dev, S., Michalewski, H.,

Garcia, X., Misra, V., Robinson, K., Fedus, L., Zhou, D., Ippolito, D., Luan, D., Lim, H., Zoph, B., Spiridonov, A., Sepassi, R., Dohan, D., Agrawal, S., Omernick, M., Dai, A. M., Pillai, T. S., Pellat, M., Lewkowycz, A., Moreira, E., Child, R., Polozov, O., Lee, K., Zhou, Z., Wang, X., Saeta, B., Diaz, M., Firat, O., Catasta, M., Wei, J., Meier-Hellstern, K., Eck, D., Dean, J., Petrov, S., and Fiedel, N. (2022). Palm: Scaling language modeling with pathways.

Chung, Y.-A. (2019). Unsupervised learning of cross-modal mappings between speech and text.

Devlin, J., Chang, M., Lee, K., and Toutanova, K. (2018). BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Ding, L. (1999). Computational models for speech production. In *Computational Models of Speech Pattern Processing*.

Edunov, S., Ott, M., Auli, M., and Grangier, D. (2018). Understanding back-translation at scale.

Frankle, J. and Carbin, M. (2018). The lottery ticket hypothesis: Training pruned neural networks. *CoRR*, abs/1803.03635.

Gidaris, S., Singh, P., and Komodakis, N. (2018). Unsupervised representation learning by predicting image rotations. In *International Conference on Learning Representations (ICLR)*.

Gogoi, M. and Begum, S. A. (2017). Image classification using deep autoencoders. In *2017 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC)*.

Goodman, J. (2001). A bit of progress in language modeling.

Graves, A. (2012a). Sequence transduction with recurrent neural networks.

Graves, A. (2012b). Sequence transduction with recurrent neural networks. In *International Conference on Machine Learning (ICML)*.

Graves, A., Fernández, S., Gomez, F., and Schmidhuber, J. (2006). Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd International Conference on Machine Learning*, New York, NY, USA. Association for Computing Machinery.

Graves, A., Jaitly, N., and Mohamed, A.-r. (2013). Hybrid speech recognition with deep bidirectional lstm. pages 273–278.

Gulati, A., Qin, J., Chiu, C.-C., Parmar, N., Zhang, Y., Yu, J., Han, W., Wang, S., Zhang, Z., Wu, Y., and Pang, R. (2020). Conformer: Convolution-augmented transformer for speech recognition. In *INTERSPEECH*.

Gülçehre, Ç., Firat, O., Xu, K., Cho, K., Barrault, L., Lin, H., Bougares, F., Schwenk, H., and Bengio, Y. (2015). On using monolingual corpora in neural machine translation. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Gutmann, M. U. and Hyvärinen, A. (2010). Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *AISTATS*.

Hestness, J., Narang, S., Ardalani, N., Diamos, G. F., Jun, H., Kianinejad, H., Patwary, M. M. A., Yang, Y., and Zhou, Y. (2017). Deep learning scaling is predictable, empirically.

Hinton, G., Deng, l., Yu, D., Dahl, G., Mohamed, A.-r., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T., and Kingsbury, B. (2012). Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *Signal Processing Magazine, IEEE*, 29:82–97.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9:1735–80.

Hori, T., Astudillo, R. F., Hayashi, T., Zhang, Y., Watanabe, S., and Roux, J. L. (2019). Cycle-consistency training for end-to-end speech recognition. In *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*.

Hsu, W., Bolte, B., Tsai, Y. H., Lakhotia, K., Salakhutdinov, R., and Mohamed, A. (2021). Hubert: Self-supervised speech representation learning by masked prediction of hidden units. In *IEEE/ACM Transactions on Audio, Speech and Language Processing*.

Huang, Y., Kuo, H., Thomas, S., Kons, Z., Audhkhasi, K., Kingsbury, B., Hoory, R., and Picheny, M. (2020). Leveraging unpaired text data for training end-to-end speech-to-intent systems. In *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*.

Kahn, J., Rivière, M., Zheng, W., Kharitonov, E., Xu, Q., Mazaré, P., Karadayi, J., Liptchinsky, V., Collobert, R., Fuegen, C., Likhomanenko, T., Synnaeve, G., Joulin, A., Mohamed, A., and Dupoux, E. (2020). Libri-light: A benchmark for ASR with limited or no supervision. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., and Amodei, D. (2020). Scaling laws for neural language models.

Krizhevsky, A., Sutskever, I., and Hinton, G. (2012). Imagenet classification with deep convolutional neural networks. *Neural Information Processing Systems*, 25.

Liu, H., Son, K., Yang, J., Liu, C., Gao, J., Lee, Y. J., and Li, C. (2023). Learning customized visual models with retrieval-augmented knowledge. In *Conference on Computer Vision and Pattern Recognition Conference (CVPR)*.

Mariooryad, S., Shannon, M., Ma, S., Bagby, T., Kao, D., Stanton, D., Battenberg, E., and Skerry-Ryan, R. (2022). Learning the joint distribution of two sequences using little or no paired data.

Mohamed, A., Lee, H.-y., Borgholt, L., Havtorn, J. D., Edin, J., Igel, C., Kirchhoff, K., Li, S.-W., Livescu, K., Maaløe, L., Sainath, T. N., and Watanabe, S. (2022). Self-supervised speech representation learning: A review. *IEEE Journal of Selected Topics in Signal Processing*, 16(6):1179–1210.

Mohamed, A.-r., Dahl, G., and Hinton, G. (2010). Deep belief networks for phone recognition. *Science*, 4.

Mohamed, A.-r., Dahl, G. E., and Hinton, G. (2012). Acoustic modeling using deep belief networks. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(1):14–22.

Morgado, P., Vasconcelos, N., and Misra, I. (2020). Audio-visual instance discrimination with cross-modal agreement. *CoRR*, abs/2004.12943.

Narayanan, A., Misra, A., Sim, K. C., Pundak, G., Tripathi, A., Elfeky, M., Haghani, P., Strohman, T., and Bacchiani, M. (2018). Toward domain-invariant speech recognition via large scale training.

Narayanan, A., Sainath, T. N., Pang, R., Yu, J., Chiu, C.-C., Prabhavalkar, R., Variani, E., and Strohman, T. (2021). Cascaded encoders for unifying streaming and non-streaming asr.

Pallett, D. S., Fiscus, J. G., Fisher, W. M., Garofolo, J. S., Lund, B. A., and Przybocki, M. A. (1994). 1993 benchmark tests for the ARPA spoken language program. In *Human Language Technology: Proceedings of a Workshop held at Plainsboro, New Jersey, March 8-11, 1994*.

Park, D. S., Chan, W., Zhang, Y., Chiu, C.-C., Zoph, B., Cubuk, E. D., and Le,

Q. V. (2019). SpecAugment: A simple data augmentation method for automatic speech recognition. In *Interspeech*.

Peri, R., Parthasarathy, S., Bradshaw, C., and Sundaram, S. (2021). Disentanglement for audio-visual emotion recognition using multitask setup.

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018). Deep contextualized word representations. *CoRR*, abs/1802.05365.

Peyser, C., Mavandadi, S., Sainath, T. N., Apfel, J., Pang, R., and Kumar, S. (2020). Improving tail performance of a deliberation e2e asr model using a large text corpus. In *INTERSPEECH*.

Prabhavalkar, R., Rao, K., Sainath, T. N., Li, B., Johnson, L. M., and Jaitly, N. (2017). A comparison of sequence-to-sequence models for speech recognition. In *Interspeech*.

Qin, T. (2020). *Dual Learning*. Springer.

Radford, A. and Narasimhan, K. (2018). Improving language understanding by generative pre-training.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. (2019). Language models are unsupervised multitask learners.

Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., and Chen, M. (2022). Hierarchical text-conditional image generation with clip latents. In *Conference on Computer Vision and Pattern Recognition Conference (CVPR)*.

Ren, Y., Tan, X., Qin, T., Zhao, S., Zhao, Z., and Liu, T.-Y. (2020). Almost unsupervised text to speech and automatic speech recognition. In *International*

Conference on Machine Learning (ICML).

Renduchintala, A., Ding, S., Wiesner, M., and Watanabe, S. (2018). Multi-modal data augmentation for end-to-end ASR. In *INTERSPEECH*.

Roe, D. and Wilpon, J. (1993). Whither speech recognition: the next 25 years. *IEEE Communications Magazine*, 31(11):54–62.

Sainath, T. N., Prabhavalkar, R., Bapna, A., Zhang, Y., Huo, Z., Chen, Z., Li, B., Wang, W., and Strohman, T. (2022). Joist: A joint speech and text streaming model for asr. In *IEEE Spoken Language Technology Workshop (SLT)*.

Sak, H., Shannon, M., Rao, K., and Beaufays, F. (2017). Recurrent neural aligner: An encoder-decoder neural network model for sequence to sequence mapping. In *INTERSPEECH*.

Sakoe, H. and Chiba, S. (1978). Dynamic programming algorithm optimization for spoken word recognition. In *IEEE Transactions on Acoustics, Speech, and Signal Processing*.

Schneider, S., Baevski, A., Collobert, R., and Auli, M. (2019). wav2vec: Unsupervised pre-training for speech recognition. In *INTERSPEECH*.

Shen, J., Pang, R., Weiss, R. J., Schuster, M., Jaitly, N., Yang, Z., Chen, Z., Zhang, Y., Wang, Y., Skerry-Ryan, R. J., Saurous, R. A., Agiomyrgiannakis, Y., and Wu, Y. (2017). Natural TTS synthesis by conditioning wavenet on mel spectrogram predictions. *CoRR*, abs/1712.05884.

Shu, R., Lee, J., Nakayama, H., and Cho, K. (2019). Latent-variable non-autoregressive neural machine translation with deterministic inference using a delta posterior. *CoRR*, abs/1908.07181.

Sriram, A., Jun, H., Satheesh, S., and Coates, A. (2018). Cold fusion: Training seq2seq models together with language models. In *INTERSPEECH*.

Tang, Y., Gong, H., Dong, N., Wang, C., Hsu, W.-N., Gu, J., Baevski, A., Li, X., Mohamed, A., Auli, M., and Pino, J. (2022). Unified speech-text pre-training for speech translation and recognition. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Tang, Y., Pino, J. M., Wang, C., Ma, X., and Genzel, D. (2021). A general multi-task learning framework to leverage text data for speech to text tasks. In *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*.

Thomas, S., Kuo, H.-K. J., Kingsbury, B., and Saon, G. (2022). Towards reducing the need for speech training data to build spoken language understanding systems. In *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*.

Tjandra, A., Sakti, S., and Nakamura, S. (2017). Listening while speaking: Speech chain by deep learning. *CoRR*, abs/1707.04879.

Toshniwal, S., Kannan, A., Chiu, C.-C., Wu, Y., Sainath, T., and Livescu, K. (2018). A comparison of techniques for language model integration in encoder-decoder speech recognition. In *IEEE Spoken Language Technology Workshop (SLT)*.

van den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A. W., and Kavukcuoglu, K. (2016). Wavenet: A generative model for raw audio. In *Proc. 9th ISCA Workshop on Speech Synthesis Workshop*.

van den Oord, A., Li, Y., and Vinyals, O. (2018). Representation learning with contrastive predictive coding. In *Conference on Neural Information Processing Systems (NeurIPS)*.

Variani, E., Rybach, D., Allauzen, C., and Riley, M. (2020). Hybrid autoregressive

transducer (hat). In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. *CoRR*, abs/1706.03762.

Wang, J., Li, J., Zhao, X., Wu, Z., Kang, S., and Meng, H. (2021). Adversarially learning disentangled speech representations for robust multi-factor voice conversion.

Wang, Y., Skerry-Ryan, R. J., Stanton, D., Wu, Y., Weiss, R. J., Jaitly, N., Yang, Z., Xiao, Y., Chen, Z., Bengio, S., Le, Q. V., Agiomyrgiannakis, Y., Clark, R., and Saurous, R. A. (2017). Tacotron: A fully end-to-end text-to-speech synthesis model. *CoRR*, abs/1703.10135.

Weber, N., Shekhar, L., and Balasubramanian, N. (2018). The fine line between linguistic generalization and failure in seq2seq-attention models. *CoRR*, abs/1805.01445.

Xia, Y., He, D., Qin, T., Wang, L., Yu, N., Liu, T., and Ma, W. (2016). Dual learning for machine translation. In *Conference on Neural Information Processing Systems (NeurIPS)*.

Xu, J., Tan, X., Ren, Y., Qin, T., Li, J., Zhao, S., and Liu, T.-Y. (2020). Lrspeech: Extremely low-resource speech synthesis and recognition. In *International Conference on Knowledge Discovery and Data Mining*.

Yang, S., Chi, P., Chuang, Y., Lai, C. J., Lakhotia, K., Lin, Y. Y., Liu, A. T., Shi, J., Chang, X., Lin, G., Huang, T., Tseng, W., Lee, K., Liu, D., Huang, Z., Dong, S., Li, S., Watanabe, S., Mohamed, A., and Lee, H. (2021). SUPERB: speech processing universal performance benchmark. In *INTERSPEECH*.

Young, S. (1996). A review of large-vocabulary continuous-speech recognition.

Yusuf, B., Gandhe, A., and Sokolov, A. (2022). Usted: Improving asr with a unified speech and text encoder-decoder. In *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*.

Zhang, Y., Qin, J., Park, D. S., Han, W., Chiu, C.-C., Pang, R., Le, Q. V., and Wu, Y. (2022). Pushing the limits of semi-supervised learning for automatic speech recognition. In *Conference on Neural Information Processing Systems (NeurIPS)*.

Zhao, W. X., Zhou, K., Li, J., Tang, T., Wang, X., Hou, Y., Min, Y., Zhang, B., Zhang, J., Dong, Z., Du, Y., Yang, C., Chen, Y., Chen, Z., Jiang, J., Ren, R., Li, Y., Tang, X., Liu, Z., Liu, P., Nie, J.-Y., and Wen, J.-R. (2023). A survey of large language models.