Introducing Semantics into Speech Encoders

Derek Xu[1], Shuyan Dong[2], Changhan Wang[2]*, Suyoun Kim[2]*, Zhaojiang Lin[2]*, Bing Liu[2], Akshat Shrivastava[2], Shang-Wen Li[2], Liang-Hsuan Tseng[3], Guan-Ting Lin[3], Alexei Baevski[2], Hung-yi Lee[3], Yizhou Sun[1], Wei Wang[1]

[1]University of California, Los Angeles
[2]Meta AI
[3]National Taiwan University

Abstract

Recent studies find existing self-supervised speech encoders contain primarily acoustic rather than semantic information. As a result, pipelined supervised automatic speech recognition (ASR) to large language model (LLM) systems achieve state-of-the-art results on semantic spoken language tasks by utilizing rich semantic representations from the LLM. These systems come at the cost of labeled audio transcriptions, which is expensive and time-consuming to obtain. We propose a task-agnostic unsupervised way of incorporating semantic information from LLMs into self-supervised speech encoders without labeled audio transcriptions. By introducing semantics, we improve existing speech encoder spoken language understanding (SLU) performance by over 5% on intent classification (IC), with modest gains in named entity resolution (NER) and slot filling (SF), and spoken question answering (SQA) FF1 score by over 2%. Our approach, which uses no ASR data, achieves similar performance as methods trained on over 100 hours of labeled audio transcripts, demonstrating the feasibility of unsupervised semantic augmentations to existing speech encoders.

1 Introduction

Realizing artificial intelligence (AI) that can understand and respond to spoken language is a north star for many speech and natural language processing (NLP) researchers. A particularly effective framework for this is the encoder-decoder architecture, where an encoder represents input audio signals as high-dimensional embeddings and a decoder converts said embeddings to outputs for different downstream tasks. Benchmarks for such systems include spoken language understanding, where intent, named entities, or slot values are predicted from input utterances (Yang et al., 2021; Bastianelli et al., 2020; Shon et al., 2022), and spoken question answering, where the start and end frames of an input audio passage answering an input audio question are predicted (Lin et al., 2022a).

A particularly notable setup of the encoder-decoder framework is the universal representation setup (Yang et al., 2021), where a shared self-supervised speech encoder is pretrained upstream once and frozen for all downstream tasks, then a different lightweight decoder is fine-tuned on each downstream task. This setup is appealing for building speech systems as maintaining a separate large specialized model for every task is not computationally efficient. The universal representation setup has been widely adopted in other areas of research, such as computer vision (Goyal et al., 2019; Ericsson et al., 2021) and NLP (Rogers et al., 2020; Qiu et al., 2020), and production when there are many downstream tasks or domains (Molino et al., 2019). The current state-of-the-art speech encoders under this setup are W2V2 and HUBERT (Yang et al., 2021; Baevski et al., 2020; Hsu et al., 2021), which are transformer-based models trained with self-supervised learning (SSL) on raw audio and have achieved impressive performance on various tasks.

Recently, analytical works found SSL speech encoders capture primarily acoustic, not semantic, information (Pasad et al., 2021). Thus, researchers proposed end-to-end systems (Chung et al., 2020b; Kim et al., 2021; Qian et al., 2021; Le et al., 2022; Seo et al., 2022; Lin et al., 2022a) that introduce semantic information through large language models (LLMs), such as ROBERTA (Liu et al., 2019) or BART (Lewis et al., 2019), which are pretrained to capture language semantics (Clark et al., 2019). This is typically accomplished by the pipeline approach (Bastianelli et al., 2020), which passes audio input through the SSL speech encoder, then bridge module, then LLM. The bridge module converts speech encoder embedding outputs into LLM token inputs (Lugosch et al., 2019; Rao et al., 2021; Lin et al., 2022a; Seo et al., 2022).

*Equal Contribution

Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 11413–11429
July 9-14, 2023 ©2023 Association for Computational Linguistics

Unsupervised ASR models (ASR-U) (Liu et al., 2020b; Baevski et al., 2021; Liu et al., 2022) have also seen recent success. The state-of-the-art ASR-U model uses generative adversarial networks (GANs) (Goodfellow et al., 2020) to generate text transcription from input audio (Liu et al., 2022).

Current works combining SSL speech encoders and LLMs do not satisfy the universal representation framework, as they either (1) rely on ASR data on the downstream task, which is expensive to collect and maintain, (2) are not lightweight, requiring training the whole system end-to-end, or (3) are not general, as they do not consider a wide variety of downstream tasks (Lugosch et al., 2019; Rao et al., 2021; Lin et al., 2022a; Seo et al., 2022). Similarly, ASR-U was proposed for speech recognition and the focus is not improving SSL speech encoders (Baevski et al., 2021; Liu et al., 2022).

We propose introducing ***S***emantics into ***S***peech ***E***ncoders, SSE, a task-agnostic unsupervised way of incorporating semantic information from LLMs into self-supervised speech encoders without labeled audio transcriptions. Concretely, SSE adopts the pipeline approach to obtain semantic embeddings, with an ASR-U bridge connector to extract information from LLMs. As ASR-U is inherently noisy, SSE introduces attention residual connection (He et al., 2016; Vaswani et al., 2017) between the speech encoder and LLM. SSE also efficiently aligns the LLM with the speech encoder through adapter modules (Houlsby et al., 2019). SSE improves W2V2 (Baevski et al., 2020) and HUBERT (Hsu et al., 2021) on 3 SLU tasks across 3 datasets, all under the universal representation setup. SSE also outperforms state-of-the art no-ASR method, DUAL (Lin et al., 2022a), in SQA.

While recent works use ASR-U to augment existing speech encoders with phoneme-level LLMs (Feng et al., 2022; Meng et al., 2022; Shi et al., 2022; Hsu et al., 2022), subword-level LLMs contain much more pertinent and measurable semantic information (Clark et al., 2019). Other works in SQA rely on clustering to assign audio frames to frequent subword tokens, but this requires heavy finetuning on the downstream task (Lin et al., 2022a).

To the best of our knowledge, we are the first to propose a task-agnostic SSL speech encoder which directly interfaces with subword-based LLMs, unblocking many other applications and future work in this domain. To this end, attention residual connections and adapters are essential to successfully extracting semantic information from noisy intermediary transcriptions. We summarize our contributions below:

- We propose using ASR-U components to augment SSL speech encoders for generating subword tokens with semantic information.
- The augmented SSL speech encoders can be connected with powerful LLMs seamlessly and yields state-of-the-art performance under the universal representation setup.
- We show attention residual connections and adapters are essential to combining and aligning speech and text encoders.

2 Related Works

2.1 Self-Supervised Speech Encoders

SSL speech encoders (Liu et al., 2020a; Chung et al., 2020a; Ling and Liu, 2020; Liu et al., 2021, 2020c; Chung et al., 2019; Baevski et al., 2019; Schneider et al., 2019; Baevski et al., 2020; Hsu et al., 2021; Qian et al., 2022; Zhang et al., 2022) are trained to learn and reconstruct pooled clustered representations of input audio from the original audio. The intuition for this objective comes from linguistics, where speech can be broken down into phoneme groups, where different chunks of input audio represent different phoneme groups. W2V (Schneider et al., 2019) trains a convolutional neural network model to reconstruct the quantized cluster representations. W2V2 (Baevski et al., 2020) uses transformers and a discrete codebook quantization module. HUBERT (Hsu et al., 2021) improves W2V2 by disentangling the clustering and SSL objectives and using a BERT-style encoder (Devlin et al., 2018). The speech processing universal performance benchmark (SUPERB) (Yang et al., 2021; Lin et al., 2022b; Tsai et al., 2022) shows SSL speech encoders are the most effective method for solving multiple downstream tasks with minimal fine-tuning. A recent analytical work finds SSL speech encoders successfully encode acoustic information, but lack semantic information (Pasad et al., 2021). In response, CONTENTVEC (Qian et al., 2022) propose disentangling the speaker and semantic content of audio via an SSL objective. SPEECHLM (Zhang et al., 2022) propose training a multi-modal speech and text encoder.

2.2 Large Language Models

In contrast to speech encoders, pretrained LLMs are shown to capture rich semantic information (Clark et al., 2019). These methods optimize variants of the masked language modeling (MLM) objective to train a large transformer model. BERT (Devlin et al., 2018) uses MLM to learn a transformer encoder. RoBERTa (Liu et al., 2019) introduces dynamic masking and a larger text corpus. BART (Lewis et al., 2019) supports generative modeling and adds a denoising objective, making it less susceptible to noisy text inputs. Longformer (Beltagy et al., 2020) is pretrained for long documents by increasing the document length limit during pretraining. LLMs have been successfully integrated with speech models for specific semantic tasks (Chung et al., 2020b; Kim et al., 2021; Qian et al., 2021; Le et al., 2022; Seo et al., 2022; Lin et al., 2022a), but not under the universal representation framework.

2.3 Task-Specific Speech Models

Task-specific SLU systems outperform generic SSL speech encoders typically by using a LLM. These systems rely on ASR data to reliably interface the LLM. Lugosch (Lugosch et al., 2019) trains a LSTM bridge module to convert audio features into phonemes then text. CTI's (Seo et al., 2022) bridge module uses ASR logits to compute a weighted average of token embeddings. In addition to improving the bridge module, other works attempt to also distill LLM embeddings into speech representations (Chung et al., 2020b; Cha et al., 2021; Kim et al., 2021; Agrawal et al., 2022). For optimizing targeted metrics, researchers have also experimented with reinforcement learning (Rao et al., 2021). While combinations of these methods achieve impressive performance, they do not satisfy the universal representation setup.

2.4 Unsupervised ASR

Recent work show the viability of unsupervised speech recognition. w2v2-U (Baevski et al., 2021) accomplished this by running Principal Component Analysis (PCA), k-means clustering, and mean pooling to convert w2v2 (Baevski et al., 2020) features into phoneme-granularity features, then trains a GAN model to output phoneme text from the post-processed model (Baevski et al., 2021). The state-of-the-art method for phoneme-level unsupervised ASR is w2v2-U2.0 (Liu et al., 2022) which directly trains a CNN to output phonemes from w2v2 features and uses a reconstruction loss to tie the input audio with corresponding generated text. Both methods use WFSTs to decode the phonemes into raw text. While there have been preliminary attempts (Feng et al., 2022; Meng et al., 2022) to use w2v2-U2.0 with phoneme language models[1], we are the first to combine it with semantically-rich subword-based LLMs.

2.5 Adapters

Adapters are intermediary layers added to a large pretrained encoder. Adapter weights are learned during fine-tuning while the rest of the pretrained model is frozen. Adapters serve the dual purpose of efficient fine-tuning and preventing overfitting. First used by computer vision researchers (Rebuffi et al., 2017), adapters now enjoy much success in the natural language processing community by efficiently tuning LLMs (Houlsby et al., 2019). In particular, the multilingual speech translation community found that adapters can effectively align SSL speech encoders and LLMs for spoken translation tasks (Li et al., 2020; Le et al., 2021).

3 Proposed Method

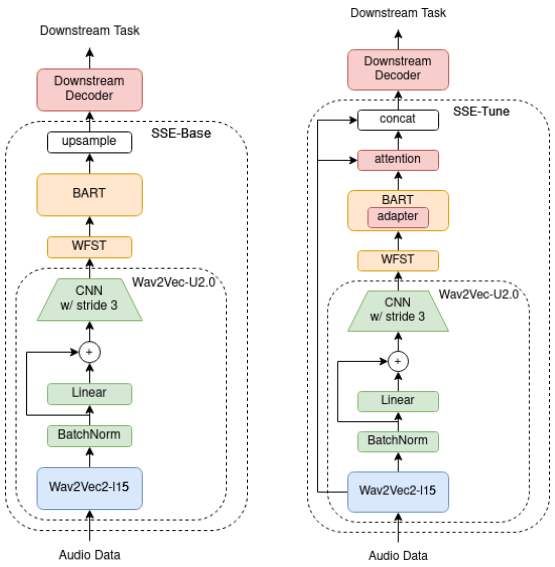
We propose to introduce semantics into SSL speech encoders by using ASR-U to interface with LLMs. Section 3.2 describes how to use ASR-U to link a speech encoder with a LLM. Section 3.3 describes how to combine both acoustic and semantic information and deal with ASR transcriptions errors. Finally, Section 3.4 describes how to align LLMs with the speech encoder for downstream tasks.

3.1 Problem Setting

Following the universal representation framework (Yang et al., 2021), our model consists of a large speech encoder, $\mathcal{E} : \mathcal{X} \rightarrow \mathcal{Z}$, mapping input audio, $X \in \mathcal{X}$, to embeddings, $Z \in \mathcal{Z}$, and a light-weight task decoder, $\mathcal{D}_\omega : \mathcal{Z} \rightarrow \mathcal{Y}_\omega$, mapping embeddings to downstream task outputs, $Y_\omega \in \mathcal{Y}_\omega$. The speech encoder, $\mathcal{E}$, is pretrained once, then shared on all downstream tasks. The task decoder, $\mathcal{D}_\omega$, is fine-tuned on its respective task, $\omega \in \Omega$. During fine-tuning, the majority of model weights are frozen. This ensures the model can be efficiently stored and deployed.

During pretraining, the speech encoder is trained on unlabelled audio, $X \in \mathcal{X}$, and unlabeled text,

[1] https://huggingface.co/voidful/phoneme_byt5



(a) SSE-BASE (b) SSE-TUNE

Figure 1: Depiction of the SSE. The blue component is the speech encoder, W2V2L15, trained with SSL. The green component is the bridge module trained with a GAN objective. The orange component is the LLM, BART, pretrained on a text corpus. The red components are trained on the downstream task and lightweight. Note, all non-red components are frozen during downstream fine-tuning.

$T_u \in \mathcal{T}_u$. During finetuning, the model is trained on the labelled downstream dataset, $(X, Y_\omega) \in \mathcal{X} \times \mathcal{Y}_\omega$. Notice, costly labelled ASR data is not required during pretraining or finetuning.

3.2 Unsupervised Semantic Representation as a Bridge

To incorporate semantic information into SSL speech encoders, $\mathcal{E} : \mathcal{X} \to \mathcal{Z}$, we wish to leverage subword-based LLMs, $\mathcal{M} : \mathcal{S} \to \mathcal{Z}$, that capture language semantics (Devlin et al., 2018; Liu et al., 2019; Lewis et al., 2019; Beltagy et al., 2020). The major challenge is the mismatch of input spaces. Speech encoders take raw audio as input, $X \in \mathcal{X}$. LLMs take subword tokens as input, $S \in \mathcal{S}$. SSE uses W2V2-U2.0 (Liu et al., 2022) as a bridge module (Seo et al., 2022), $\mathcal{B} : \mathcal{Z} \to \mathcal{S}$, to convert speech encoder embedding output into LLM subword tokens in a pipelined approach, $\mathcal{E}_{\text{SSE}} = \mathcal{E} \circ \mathcal{B} \circ \mathcal{M}$.

Following W2V2-U2.0, the bridge module, $\mathcal{B}$ uses a GAN (Goodfellow et al., 2020) to output phoneme sequences, $P \in \mathcal{P}$, conditioned on input audio, $X \in \mathcal{X}$. The GAN does not directly predict subword-level transcriptions, because subword barriers are not easily deducible from acoustic speech embeddings and requires implicitly learning phoneme-to-subword mappings. Instead, the bridge module, $\mathcal{B}$, uses a Weighted Finite State Transducer (WFST), $\mathcal{W} : \mathcal{P} \to \mathcal{S}$, which is fed known phoneme-to-subword mappings, to map the generator outputs into subword tokens. The generator, $\mathcal{G} : \mathcal{Z} \to \mathcal{P}$, and the discriminator, $\mathcal{C} : \mathcal{P} \to [0, 1]$, are both convolutional neural networks (CNNs). The GAN model is trained on the same regularized GAN objective as in W2V2-U2.0 (Liu et al., 2022).

Model Component	% of Parameters
SSE-BASE	90.40%
residual attention	0.73%
BART adapters	0.18%
downstream decoder	8.69%

Table 1: Comparing the parameter count of different components of SSE-TUNE (W2V2L15). In total, there are 505.3 million parameters. Notice, the decoder is much more lightweight than the encoder. Residual attention and adapters also introduce minimal parameter overhead during finetuning.

The vanilla version of our final model is composed of (1) SSL speech encoder, $\mathcal{E} : \mathcal{X} \to \mathcal{Z}$ pretrained on unlabelled audio data, (2) a CNN+WFST bridge module, $\mathcal{B} = \mathcal{W} \circ \mathcal{G} : \mathcal{Z} \to \mathcal{S}$, trained on unlabelled text and audio data, and (3) a LLM, $\mathcal{M} : \mathcal{S} \to \mathcal{Z}$, pretrained on unlabelled text data. We also add an upsampling layer, $\mathcal{U} : \mathcal{Z} \to \mathcal{Z}$ to make the sequence length of the LLM output match the speech encoder output, such that $\mathcal{E}$ and $\mathcal{E}_{\text{SSE}}$ share the same output space.

We choose the 15th layer of the W2V2 (Baevski et al., 2020) as our speech encoder, as the last layers overfit the self-supervised training objective hence providing worse acoustic representations (Fan et al., 2020; Baevski et al., 2021; Pasad et al., 2021). We choose BART (Lewis et al., 2019) as our LLM, as it is trained to denoise noisy input subword tokens, and we expect the bridge module to introduce some noise. We call this version of our model SSE-BASE. A depiction can be found in Figure 1a.

3.3 Combining Semantics and Acoustics with Residual Attention

We hypothesize certain tasks may require more acoustic information than others. For instance, named entity recognition (NER) requires the model

Model	FSC (Acc)	SLURP-IC (Acc)	SLURP-SF (F1)	SLUE-NER (F1)
W2V2L24	95.28%	39.77%	36.48%	46.10%
W2V2L15	95.60%	49.97%	62.43%	78.77%
HUBERT	98.76%	58.11%	66.97%	**82.88%**
SPEECHLM (HUBERT-BASE)*	97.6%	-%	-%	-%
SPEECHLM (PBERT-BASE)*	98.6%	-%	-%	-%
CONTENTVEC (HUBERT-BASE)	99.10%	34.03%	63.83%	75.19%
SSE-BASE	95.99%	55.28%	61.59%	79.62%
SSE-TUNE (W2V2L15)	98.71%	63.64%	64.48%	80.10%
SSE-TUNE (HUBERT-BASE)	98.30%	58.69%	64.64%	76.61%
SSE-TUNE (HUBERT)	**99.44%**	**64.33%**	**68.82%**	82.02%

Table 2: Experimental Results on FSC, SLURP, and SLUE datasets. We group the models by existing SSL encoders and their semantically-enriched counterparts. Note, including semantics via LLMs consistently improves downstream performance on both W2V2 and HUBERT. SLUE-NER relies on primarily semantic information. Hence, while SSE-TUNE (W2V2L15) improves W2V2L15, HUBERT and SSE-TUNE (HUBERT) perform similarly. Methods with superscript '*' indicates reported results in corresponding papers.

Model	NMSQA	
	FF1	AOS
DUAL-64	39.0%	33.0%
DUAL-128	55.9%	49.1%
DUAL-512	17.3%	12.5%
SSE-BASE (ADAP)	57.2%	46.4%
SSE-BASE (ADAP) †	**62.0%**	**54.7%**
PIPELINE †	**64.2%**	**57.1%**

Table 3: Comparing unsupervised SQA models to supervised PIPELINE model. † denotes the model uses a LLM that was finetuned on the SQUAD-V1.1 text-only QA dataset. We compare the baseline, DUAL, with 3 different number of clusters choices, to SSE-BASE (ADAP) trained with either unlabeled audio or text.

to implicitly transcribe parts of the input speech, a primarily acoustic task. Since the pipelined model may suffer from transcription errors introduced by ASR-U, naively using the pipelined approach introduces an information bottleneck at the bridge module. Hence, we propose adding a residual connection (He et al., 2016) between SSE-BASE and the speech encoder, $\mathcal{E}$.

This can be done in two ways: (1) upsampling semantic embeddings and concatenating with speech embeddings, $Z = [Z_{\mathcal{E}}||\mathcal{U}(Z_{\mathcal{M}})]$, or (2) using multihead attention (Vaswani et al., 2017) to merge the two embeddings, $Z = [Z_{\mathcal{E}}||\text{MHA}(Z_{\mathcal{E}}, Z_{\mathcal{M}}, Z_{\mathcal{M}})]$, where $Z_{\mathcal{E}} \in \mathcal{Z}$ is the output of the W2V2L15 (Baevski et al., 2020) and $Z_{\mathcal{M}} \in \mathcal{Z}$ is the output of BART (Lewis et al., 2019). The former is a simpler but more naive method. The latter is more effective as the attention layers are able to learn the alignment between speech and semantic embeddings. Notice, (2) introduces more learnable parameters to the finetuning-step, but we find the number of new parameters inconsequential compared to the size of the lightweight decoder.

3.4 Aligning Pretrained Text Model with Adapters

Inspired by works from speech translation (Li et al., 2020; Le et al., 2021), we hypothesize that the LLM can easily be adapted for speech tasks through the use of adapters. We adopt the general recipe for adapters, where an adapter (Houlsby et al., 2019), composed of a LayerNorm and 2-layer ReLU neural network, is added to the end of each feed forward layer in the LLM and finetuned on downstream tasks. This introduces additional parameters to finetuning, but we find the number of new parameters inconsequential compared to the size of the lightweight decoder. We call the model using both residual attention and adapters SSE-TUNE, and outline it in Figure 1b.

4 Experiments

4.1 Dataset

To show the effectiveness of introducing semantics into speech encoders, we evaluate 3 SLU tasks, intent classification (IC), slot filling (SF), and named entity recognition (NER), and SQA

Augmentation	FSC-IC (Acc)	SLURP-IC (Acc)	SLURP-SF (F1)	SLUE-NER (F1)
W2V2L15	95.60%	49.97%	62.43%	78.77%
SSE-BASE	95.99%	55.28%	61.59%	79.62%
SSE-BASE (Byt5)	95.80%	35.50%	59.15%	76.44%
SSE-BASE (T5lephone)	95.94%	41.19%	60.87%	77.88%
SSE-BASE (RES)	97.55%	59.59%	63.37%	79.66%
SSE-BASE (RESATT)	**98.97%**	62.39%	64.21%	80.04%
SSE-BASE (ADAP)	96.07%	60.28%	63.85%	79.97%
SSE-TUNE	98.71%	**63.64%**	**64.48%**	**80.10%**

Table 4: Ablation studies on choice of language model, residual attention, and adapters. By better representing semantics, subword-based LLMs outperform phoneme- and unicode-based LLMs. Notice, both residual attention and adapters are important. While SSE-BASE (RESATT) introduces slightly more parameters than SSE-BASE (RES), it provides tangible performance improvement by better aligning the acoustic and semantic embeddings.

task across 4 datasets: Fluent Speech Commands (FSC) (Lugosch et al., 2019), Spoken Language Understanding Resource Package (SLURP) (Bastianelli et al., 2020), Spoken Language Understanding Evaluation (SLUE) (Shon et al., 2022), and Natural Multi-speaker Spoken Question Answering (NMSQA) (Lin et al., 2022a), covering a wide variety of speakers, microphones, and environments

4.2 Encoder Setup and Baselines

4.2.1 Spoken Language Understanding

To show SSE improves SSL speech encoders, we augment two state-of-the art speech encoders under the universal representation setup: W2V2 and HUBERT. Following prior works that found intermediary layers of W2V2 contain better representations (Pasad et al., 2021; Baevski et al., 2021), we consider the 15th layer and the last layer of W2V2, named W2V2L15 and W2V2L24 respectively.

As mentioned in Section 3, we show 2 versions of our model, SSE-BASE and SSE-TUNE. The former uses the pipelined approach to connect W2V2L15 with BART (Lewis et al., 2019) with no additional modifications. The latter introduces an attention residual connection and learnable adapters to combine acoustics and semantics together and align the LLM with the speech encoder respectively. We either connect the residual connection to the output of W2V2L15, yielding SSE-TUNE (W2V2L15), or to the output of HUBERT, yielding SSE-TUNE (HUBERT).

To show the importance of using LLMs, we compare against 2 very recent approaches for improving SSL speech encoders without LLMs, SPEECHLM (Zhang et al., 2022) and CONTENTVEC (Qian et al., 2022). As HUBERT-BASE was used as the base speech encoder by both baselines, we also provide results where SSE-TUNE is used to augment HUBERT-BASE.

4.2.2 Spoken Question Answering

To show the effectiveness of SSE, we compare it against DUAL (Lin et al., 2022a), the state-of-the-art SQA model which does not use ASR data. While both SSE and DUAL obtain frame-level tokens from speech input, SSE uses ASR-U to obtain its tokens, whereas DUAL uses clustering. As a result, SSE's output tokens exists in the LLM's existing vocabulary, whereas DUAL's output tokens does not. Hence, DUAL must retrain the LLM on its output tokens.

We compare DUAL to the closest analogous SSE model, which is SSE-BASE but with adapter layers, SSE-BASE (ADAP). Similar to DUAL, both methods modify the LLM weights. Unlike DUAL, SSE-BASE (ADAP) is lightweight, tuning only around 10% of the total parameters. To produces frame-level predictions, we remove the upsampling layer from SSE-BASE (ADAP). We choose W2V2L15 as our speech model and BART as our LLM, as it is robust to ASR errors.

We also show a PIPELINE model, which trains a W2V2 model on ASR data and a LONGFORMER LLM on text-only question answering data. It is worth noting that since evaluation is based on the frame-level, SSL speech encoders are not a baseline since they operate at the audio level.

4.3 Decoder Setup

To satisfy the universal representation setup, we adopt lightweight SLU decoders from SUPERB (Yang et al., 2021). For IC, the decoder

Bridge Module	ASR data	FSC		SLURP			SLUE	
		WER	IC Acc	WER	IC Acc	SF F1	WER	NER F1
W2V2-ASR	960h	9.19%	99.34%	45.83%	66.18%	65.62%	15.51%	80.58%
W2V2-ASR	100h	11.89%	99.10%	53.22%	63.20%	63.87%	17.70%	79.67%
W2V2-ASR	10h	59.06%	98.50%	74.77%	59.91%	63.42%	53.00%	79.76%
SSE-TUNE	nothing	21.28%	98.71%	51.51%	63.64%	64.48%	31.22%	80.10%

Table 5: Analysis on WER of SSE's bridge module. All models adopt the same speech encoder, LLM, residual attention, and adapter components as SSE-TUNE (W2V2L15), but convert speech embeddings into subword tokens in different ways. W2V2-ASR finetunes W2V2 with an ASR head using letter-based CTC with varying amounts of ASR data. As seen in this table, ASR errors correlate with downstream performance. Hence, more accurate ASR-U models or methods to alleviate ASR errors, such as residual attention, would greatly benefit SSE.

is sum pooling followed by a multilayer perceptron classifier trained with cross entropy loss. For the SF and NER tasks, the decoder is recursive neural network (RNN) that transcribes input audio into text. The decoder identifies named entities or slot values by surrounding them with named special tokens and is trained with connectionist temporal classification loss. For SQA, we adopt the same decoder as DUAL (Lin et al., 2022a), which is a linear layer classifying each subword embedding as the start or end or neither of an answer span.

5 Results

5.1 Spoken Language Understanding

5.1.1 Improving SSL Speech Encoders

As seen in Table 2, SSE significantly improves the SLU performance of both W2V2 and HUBERT, confirming that including semantic information drastically improves existing SSL speech encoder performance. Specifically, SSE-TUNE (W2V2L15) improves W2V2L15 on all tasks. SSE-TUNE (HUBERT) improves HUBERT on 3 out of 4 tasks, and is the best performing model overall. Comparing SSE-TUNE with SSE-BASE shows residual attention and adapters effectively counteracts bridge module transcription errors.

The relative performance gain for IC is more than SF or NER. Unlike IC, both SF and NER require the speech encoder to transcribe identified audio snippets, and transcription is a primarily acoustic task. Hence SF and NER require less semantic information than IC. Nevertheless, combining both acoustic and semantic information, as done by SSE-TUNE, provides the most consistent performance improvement, since the skip connection can learn which type of information is more needed.

5.1.2 Importance of LLMs

As seen in Table 2, SSE-TUNE (HUBERT-BASE) outperforms alternative approaches augmenting speech encoders, SPEECHLM (HUBERT-BASE) and CONTENTVEC (HUBERT-BASE). Unlike these alternative approaches, SSE-TUNE incorporate information from LLMs, which we found to be very beneficial for capturing semantic information as they are carefully pretrained objectives on large amounts of unlabelled text data.

It is noteworthy that SSE-TUNE is a general framework which can augment any speech encoder of our choice, including SPEECHLM and CONTENTVEC. Similarly, SSE-TUNE can directly integrate new LLMs without costly pretraining. We leave incorporating such encoders into SSE-TUNE as future work.

5.2 Spoken Question Answering

As seen in Table 3, SSE outperforms recent unsupervised clustering-based approaches, DUAL. In contrast to DUAL's HUBERT cluster tokens, SSE's ASR-U tokens are better aligned with LLMs and share the same space. Thus, SSE can better utilizes pretrained LLMs. Furthermore, SSE does not require carefully tuning the number of HUBERT cluster counts, as the vocabulary size of the LLM is fixed and consistent with ASR-U.

5.3 Ablation Study

5.3.1 Choice of Language Model

We find subword-based LLMs contain more information than phoneme-based LLMs (Clark et al., 2019). We empirically verify this by replacing our subword-based LLM, BART (Lewis et al., 2019), with popular character-based LLM, ByT5 (Xue et al., 2022), and phoneme-based LLM, T5lephone (Hsu et al., 2022) in SSE-BASE. As

seen in Table 4, the subword-based LLM perform the best as each subword token is more semantically meaningful than a phoneme or character. We believe T5lephone outperforms the Byt5 as it has better robustness to ASR-U errors. Overall, subword-based LLMs are the best choice for embedding semantic information in transcribed text.

5.3.2 Residual Attention and Adapters

To more carefully analyze the affect of residual attention and adapters in SSE-TUNE, we run experiments on all SLU datasets with and without each component. We denote these two design choices as (ResAtt) and (Adap) respectively. As seen in Table 4, both components provide ample performance improvement over SSE-BASE.

We also try the naive residual connection approach described in Section 3.3 by directly concatenating the LLM upsampled semantic embeddings to the speech embeddings. We call this approach SSE-BASE (RES). This method is less effective than SSE-BASE (RESATT) as it does not learn how to align speech and semantic embeddings, but still improves SSE-BASE, further validating our hypothesis that merging acoustic and semantic information is beneficial.

As seen in parameter breakdown for the SSE-TUNE (W2V2L15) model in Table 1, the number of new learnable parameters introduced by (ResAtt) and (Adap) are unsubstantial compared to the size of the lightweight downstream decoder. Specifically, the downstream task decoder accounts for 9.60% of the total model parameters. SSE-TUNE introduces only 10.47% more parameters than SSE-BASE during fine-tuning and 0.91% to the total model parameter count, but often provides significant performance improvement.

5.4 Comparison with Supervised ASR Methods

To quantify the effect of transcription errors introduced by the bridge module, we compute the word error rate (WER) of the bridge connector in SSE-TUNE, and compare it against standard W2V2 supervised ASR models (Baevski et al., 2020) trained on 10 minutes, 100 hours, and 960 hours of labeled ASR data. Table 5 confirms that less noisy transcripts, transcripts with lower WER, correlates with better downstream performance. The unsupervised model, which uses 960 hours of unlabelled data, can reach similar WER as a supervised model trained on 100 hours of labelled data, indicating the effectiveness of the bridge module. On SLURP and SLUE, the relative drop in WER ($> 20\%$) is substantially more than the relative drop in downstream performance ($< 5\%$), verifying SSE-TUNE's tolerance to noisy transcriptions. The robustness to ASR errors come from our choice of LLM, BART, which is trained to handle noisy inputs, residual connection to acoustic embeddings, and LLM alignment with adapters.

Model	IC (Acc)	SF (F1)
W2V2L15	49.97%	62.43%
HUBERT	58.11%	66.97%
SSE-TUNE (W2V2L15)	63.64%	64.48%
SSE-TUNE (HUBERT)	64.33%	68.82%
Kaldi+HerMiT	78.33%	70.84%
CTI	82.93%	71.12%

Table 6: Comparison with specialized SLU models not under the universal representation setup, Kaldi+HerMiT and CTI. Results are on the SLURP dataset.

5.5 Comparison to Specialized SLU Models

To better quantify the performance improvement introduced by SSE, we compare against 2 specialized SLU models that do not abide by the universal representation framework: Kaldi+HerMiT, which is a pipelined Kaldi ASR (Povey et al., 2011) and HerMiT NLU (Vanzo et al., 2019) model reported in the SLURP paper (Bastianelli et al., 2020), and CTI (Seo et al., 2022), which is an end-to-end pipelined W2V2 (Baevski et al., 2020) ASR and ROBERTA (Liu et al., 2019) NLU model. To the best of our knowledge, CTI is the state-of-the-art SLU model.

In addition to unlabelled text, unlabelled audio, and downstream data, both Kaldi+HerMiT and CTI require 40 hours of downstream SLURP ASR data (Bastianelli et al., 2020). Kaldi+HerMiT requires an additional 24,000 hours of ASR data (Povey et al., 2016). CTI requires an additional 960 hours of ASR data (Panayotov et al., 2015). Neither use lightweight fine-tuning. Thus, such specialized SLU models are less general, more expensive, and require much more data. As seen in Table 6, SSE helps bridge the gap between tailor-made models and more practical SSL speech encoders. We believe ASR-U errors plays a major role in the remaining gap, as the ASR-supervised Kaldi+HerMiT and CTI models have WER of 16.20% and 16.67% respectively, compared to

Most Common Mix-ups	% Mistakes
qa_factoid, general_quirky	+5.83%
calendar_set, calendar_query	-20.00%
general_quirky, calendar_query	+8.57%
weather_query, calendar_query	-34.72%
play_music, play_audiobook	-7.27%
play_music, play_radio	-14.03%
calendar_set, calendar_remove	-32.26%
play_music, play_game	-18.87%
...	...

Table 7: Top-8 most common mix-ups, descending, by either HUBERT or SSE-TUNE (HUBERT) on SLURP-IC's test set. A mix-up is when the model either misclassifies label "A" as "B" or misclassifies label "B" as "A". For each mix-up, we compute the percentage of less mistakes made by SSE-TUNE (HUBERT) than HUBERT. For example, SSE-TUNE (HUBERT) misclassifies calendar_set as calendar_query or vice-versa 20% less frequently than HUBERT. The "general-quirky" label is assigned to Out-of-Distribution inputs.

SSE's ASR-U bridge with a WER of 51.51%.

5.6 Error Analysis

To better understand the semantic information captured by SSE, we study predictions made by both HUBERT and SSE-TUNE (HUBERT) on SLURP-IC's test set. We find HUBERT errors are made primarily between intents within the same or similar domains (e.g. calendar_set vs calendar_query). The performance bottleneck lies with distinguishing finer-grained in-domain intents. Table 7 shows that SSE-TUNE is better at differentiating finer-grained intents.

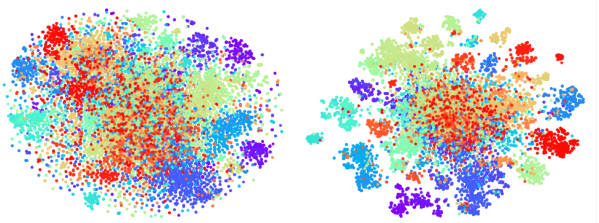
SSE-TUNE's misclassifications come primarily from errors made by its ASR-U bridge component. As seen in Table 8, the ASR-U WER of incorrect predictions made by HUBERT is much lower than that of incorrect predictions made by SSE-TUNE. When ASR-U returns resonable transcriptions (typically <50% WER), SSE-TUNE can correctly classify inputs that HUBERT cannot. Hence, the effectiveness of SSE is tightly coupled with the effectiveness of ASR-U.

5.7 Representation Visualization

To better see the impact of including semantic representations, we visualize the pooled audio snippet embedding for intent classification on SLURP-IC using t-distributed stochastic neighbor embedding (t-SNE) (Van der Maaten and Hinton, 2008). We denote the ground truth label of each audio snippet by the color of its pooled embedding. As seen in Figure 2, the clusters produced by semantic embeddings are more spread out and better separated than those produced by just acoustic speech embeddings, indicating that SSE introduces new semantic information that existing speech encoders lack.

		SSE-TUNE (HUBERT)	
		✓	✗
HUBERT	✓	40.8% (6395)	75.2% (1204)
	✗	48.2% (2019)	78.1% (3460)

Table 8: WER between the output of SSE-TUNE (HUBERT)'s bridge module and ground truth transcriptions. Each WER is evaluated on a subset of SLURP-IC testing samples where the model classifies correctly (✓) or incorrectly (✗). We denote the number of pairs belonging to each subset, in the thousands, in parentheses.



(a) W2V2L15 embeddings (b) SSE-TUNE embeddings

Figure 2: t-SNE visualizations of pooled audio snippet embeddings for SLURP-IC. Each point corresponds with one embedding. The color denotes the ground truth class of the corresponding audio snippet. Subfigure 2a and 2b shows SSE-TUNE is better at differentiating intents by incorporating semantic information.

6 Conclusion

We presented a compelling case for introducing semantics into SSL speech encoders and an effective method of doing so. Our approach boosts the performance of existing speech encoders on multiple SLU and SQA tasks and datasets. We provide reasoning for what tasks may benefit more or less from incorporating semantics. Furthermore, our approach is task agnostic and can augment any existing SSL speech encoder. With SSE-TUNE, we show merging acoustic and semantic information and effectively aligning LLMs to the speech encoder on downstream tasks can further boost performance with minimal parameter overhead. As it can generalize to many downstream tasks, SSE provides an important step towards AI that can understand and respond to spoken language.

References

Bhuvan Agrawal, Markus Müller, Samridhi Choudhary, Martin Radfar, Athanasios Mouchtaris, Ross McGowan, Nathan Susanj, and Siegfried Kunzmann. 2022. Tie your embeddings down: Cross-modal latent spaces for end-to-end spoken language understanding. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7157–7161. IEEE.

Alexei Baevski, Wei-Ning Hsu, Alexis Conneau, and Michael Auli. 2021. Unsupervised speech recognition. *Advances in Neural Information Processing Systems*, 34:27826–27839.

Alexei Baevski, Steffen Schneider, and Michael Auli. 2019. vq-wav2vec: Self-supervised learning of discrete speech representations. *arXiv preprint arXiv:1910.05453*.

Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. 2020. wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in Neural Information Processing Systems*, 33:12449–12460.

Emanuele Bastianelli, Andrea Vanzo, Pawel Swietojanski, and Verena Rieser. 2020. Slurp: A spoken language understanding resource package. *arXiv preprint arXiv:2011.13205*.

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Sujeong Cha, Wangrui Hou, Hyun Jung, My Phung, Michael Picheny, Hong-Kwang Kuo, Samuel Thomas, and Edmilson Morais. 2021. Speak or chat with me: End-to-end spoken language understanding system with flexible inputs. *arXiv preprint arXiv:2104.05752*.

Yu-An Chung, Wei-Ning Hsu, Hao Tang, and James Glass. 2019. An unsupervised autoregressive model for speech representation learning. *arXiv preprint arXiv:1904.03240*.

Yu-An Chung, Hao Tang, and James Glass. 2020a. Vector-quantized autoregressive predictive coding. *arXiv preprint arXiv:2005.08392*.

Yu-An Chung, Chenguang Zhu, and Michael Zeng. 2020b. Splat: Speech-language joint pre-training for spoken language understanding. *arXiv preprint arXiv:2010.02295*.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D Manning. 2019. What does bert look at? an analysis of bert's attention. *arXiv preprint arXiv:1906.04341*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Linus Ericsson, Henry Gouk, and Timothy M Hospedales. 2021. How well do self-supervised models transfer? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5414–5423.

Zhiyun Fan, Meng Li, Shiyu Zhou, and Bo Xu. 2020. Exploring wav2vec 2.0 on speaker verification and language identification. *arXiv preprint arXiv:2012.06185*.

Tzu-hsun Feng, Annie Dong, Ching-Feng Yeh, Shu-wen Yang, Tzu-Quan Lin, Jiatong Shi, Kai-Wei Chang, Zili Huang, Haibin Wu, Xuankai Chang, et al. 2022. Superb@ slt 2022: Challenge on generalization and efficiency of self-supervised speech representation learning. *arXiv preprint arXiv:2210.08634*.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2020. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144.

Priya Goyal, Dhruv Mahajan, Abhinav Gupta, and Ishan Misra. 2019. Scaling and benchmarking self-supervised visual representation learning. In *Proceedings of the ieee/cvf International Conference on computer vision*, pages 6391–6400.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR.

Eduard Hovy, Mitch Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. 2006. Ontonotes: the 90% solution. In *Proceedings of the human language technology conference of the NAACL, Companion Volume: Short Papers*, pages 57–60.

Chan-Jan Hsu, Ho-Lam Chung, Hung-yi Lee, and Yu Tsao. 2022. T5lephone: Bridging speech and text self-supervised models for spoken language understanding via phoneme level t5. *arXiv preprint arXiv:2211.00586*.

Wei-Ning Hsu, Yao-Hung Hubert Tsai, Benjamin Bolte, Ruslan Salakhutdinov, and Abdelrahman Mohamed. 2021. Hubert: How much can a bad teacher benefit asr pre-training? In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6533–6537. IEEE.

Minjeong Kim, Gyuwan Kim, Sang-Woo Lee, and Jung-Woo Ha. 2021. St-bert: Cross-modal language model pre-training for end-to-end spoken language understanding. In *ICASSP 2021-2021 IEEE International*

Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 7478–7482. IEEE.

Duc Le, Akshat Shrivastava, Paden Tomasello, Suyoun Kim, Aleksandr Livshits, Ozlem Kalinli, and Michael L Seltzer. 2022. Deliberation model for on-device spoken language understanding. *arXiv preprint arXiv:2204.01893*.

Hang Le, Juan Pino, Changhan Wang, Jiatao Gu, Didier Schwab, and Laurent Besacier. 2021. Lightweight adapter tuning for multilingual speech translation. *arXiv preprint arXiv:2106.01463*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Xian Li, Changhan Wang, Yun Tang, Chau Tran, Yuqing Tang, Juan Pino, Alexei Baevski, Alexis Conneau, and Michael Auli. 2020. Multilingual speech translation with efficient finetuning of pretrained models. *arXiv preprint arXiv:2010.12829*.

Guan-Ting Lin, Yung-Sung Chuang, Ho-Lam Chung, Shu-wen Yang, Hsuan-Jui Chen, Shuyan Dong, Shang-Wen Li, Abdelrahman Mohamed, Hung-yi Lee, and Lin-shan Lee. 2022a. Dual: Discrete spoken unit adaptive learning for textless spoken question answering. *CoRR*.

Guan-Ting Lin, Chi-Luen Feng, Wei-Ping Huang, Yuan Tseng, Tzu-Han Lin, Chen-An Li, Hung-yi Lee, and Nigel G Ward. 2022b. On the utility of self-supervised models for prosody-related tasks. *arXiv preprint arXiv:2210.07185*.

Shaoshi Ling and Yuzong Liu. 2020. Decoar 2.0: Deep contextualized acoustic representations with vector quantization. *arXiv preprint arXiv:2012.06659*.

Alexander H Liu, Yu-An Chung, and James Glass. 2020a. Non-autoregressive predictive coding for learning speech representations from local dependencies. *arXiv preprint arXiv:2011.00406*.

Alexander H Liu, Wei-Ning Hsu, Michael Auli, and Alexei Baevski. 2022. Towards end-to-end unsupervised speech recognition. *arXiv preprint arXiv:2204.02492*.

Alexander H Liu, Tao Tu, Hung-yi Lee, and Lin-shan Lee. 2020b. Towards unsupervised speech recognition and synthesis with quantized speech representation learning. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7259–7263. IEEE.

Andy T Liu, Shang-Wen Li, and Hung-yi Lee. 2021. Tera: Self-supervised learning of transformer encoder representation for speech. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:2351–2366.

Andy T Liu, Shu-wen Yang, Po-Han Chi, Po-chun Hsu, and Hung-yi Lee. 2020c. Mockingjay: Unsupervised speech representation learning with deep bidirectional transformer encoders. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6419–6423. IEEE.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Loren Lugosch, Mirco Ravanelli, Patrick Ignoto, Vikrant Singh Tomar, and Yoshua Bengio. 2019. Speech model pre-training for end-to-end spoken language understanding. *arXiv preprint arXiv:1904.03670*.

Yen Meng, Hsuan-Jui Chen, Jiatong Shi, Shinji Watanabe, Paola Garcia, Hung-yi Lee, and Hao Tang. 2022. On compressing sequences for self-supervised speech models. *arXiv preprint arXiv:2210.07189*.

Piero Molino, Yaroslav Dudin, and Sai Sumanth Miryala. 2019. Ludwig: a type-based declarative deep learning toolbox. *arXiv preprint arXiv:1909.07930*.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. *arXiv preprint arXiv:1904.01038*.

Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. 2015. Librispeech: an asr corpus based on public domain audio books. In *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5206–5210. IEEE.

Ankita Pasad, Ju-Chieh Chou, and Karen Livescu. 2021. Layer-wise analysis of a self-supervised speech representation model. In *2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 914–921. IEEE.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32.

Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulić, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. 2020. Adapterhub: A framework for adapting transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 46–54.

Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, et al. 2011. The kaldi speech recognition toolkit. In *IEEE 2011 workshop on automatic speech recognition and understanding*, CONF. IEEE Signal Processing Society.

Daniel Povey, Vijayaditya Peddinti, Daniel Galvez, Pegah Ghahremani, Vimal Manohar, Xingyu Na, Yiming Wang, and Sanjeev Khudanpur. 2016. Purely sequence-trained neural networks for asr based on lattice-free mmi. In *Interspeech*, pages 2751–2755.

Kaizhi Qian, Yang Zhang, Heting Gao, Junrui Ni, Cheng-I Lai, David Cox, Mark Hasegawa-Johnson, and Shiyu Chang. 2022. Contentvec: An improved self-supervised speech representation by disentangling speakers. In *International Conference on Machine Learning*, pages 18003–18017. PMLR.

Yao Qian, Ximo Bianv, Yu Shi, Naoyuki Kanda, Leo Shen, Zhen Xiao, and Michael Zeng. 2021. Speech-language pre-training for end-to-end spoken language understanding. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7458–7462. IEEE.

Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. 2020. Pre-trained models for natural language processing: A survey. *Science China Technological Sciences*, 63(10):1872–1897.

Milind Rao, Pranav Dheram, Gautam Tiwari, Anirudh Raju, Jasha Droppo, Ariya Rastrow, and Andreas Stolcke. 2021. Do as i mean, not as i say: Sequence loss training for spoken language understanding. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7473–7477. IEEE.

Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. 2017. Learning multiple visual domains with residual adapters. *Advances in neural information processing systems*, 30.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in bertology: What we know about how bert works. *Transactions of the Association for Computational Linguistics*, 8:842–866.

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

Seunghyun Seo, Donghyun Kwak, and Bowon Lee. 2022. Integration of pre-trained networks with continuous token interface for end-to-end spoken language understanding. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7152–7156. IEEE.

Jiatong Shi, Chan-Jan Hsu, Holam Chung, Dongji Gao, Paola Garcia, Shinji Watanabe, Ann Lee, and Hung-yi Lee. 2022. Bridging speech and textual pre-trained models with unsupervised asr. *arXiv preprint arXiv:2211.03025*.

Suwon Shon, Ankita Pasad, Felix Wu, Pablo Brusco, Yoav Artzi, Karen Livescu, and Kyu J Han. 2022. Slue: New benchmark tasks for spoken language understanding evaluation on natural speech. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7927–7931. IEEE.

Hsiang-Sheng Tsai, Heng-Jui Chang, Wen-Chin Huang, Zili Huang, Kushal Lakhotia, Shu-wen Yang, Shuyan Dong, Andy T Liu, Cheng-I Jeff Lai, Jiatong Shi, et al. 2022. Superb-sg: Enhanced speech processing universal performance benchmark for semantic and generative capabilities. *arXiv preprint arXiv:2203.06849*.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

Andrea Vanzo, Emanuele Bastianelli, and Oliver Lemon. 2019. Hierarchical multi-task natural language understanding for cross-domain conversational ai: Hermit nlu. *arXiv preprint arXiv:1910.00912*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Linting Xue, Aditya Barua, Noah Constant, Rami Al-Rfou, Sharan Narang, Mihir Kale, Adam Roberts, and Colin Raffel. 2022. Byt5: Towards a token-free future with pre-trained byte-to-byte models. *Transactions of the Association for Computational Linguistics*, 10:291–306.

Shu-wen Yang, Po-Han Chi, Yung-Sung Chuang, Cheng-I Jeff Lai, Kushal Lakhotia, Yist Y Lin, Andy T Liu, Jiatong Shi, Xuankai Chang, Guan-Ting Lin, et al. 2021. Superb: Speech processing universal performance benchmark. *arXiv preprint arXiv:2105.01051*.

Ziqiang Zhang, Sanyuan Chen, Long Zhou, Yu Wu, Shuo Ren, Shujie Liu, Zhuoyuan Yao, Xun Gong, Lirong Dai, Jinyu Li, et al. 2022. Speechlm: Enhanced speech pre-training with unpaired textual data. *arXiv preprint arXiv:2209.15329*.

A Appendix

A.1 Acknowledgments

We thank Zoey Zhiyu Chen for checking experimental results and Jiatong Shi for providing helpful discussion. This work was done during an internship at Meta AI and was partially supported by NSF 1829071, 1937599, 2106859, 2200274, 2211557, 2119643, 2303037, DARPA #HR00112290103/HR0011260656, NASA, SRC, Okawa Foundation Grant, Amazon Research Awards, Cisco research grant, Picsart Gifts, Snapchat Gifts, and an NEC research award.

A.2 ASR-U Bridge Training Objective Details

We adopt the same unsupervised training scheme as W2V2-U2.0 (Liu et al., 2022). Specifically, we train the generator, $\mathcal{G}$, on GAN loss, $\mathcal{L}_{gan}$, a gradient penalty term, $\mathcal{L}_{gp}$, for better convergence, a smoothness penalty term, $\mathcal{L}_{sp}$, to encourage consecutive speech segments to generate the same phonemes, a phoneme diversity term, $\mathcal{L}_{pd}$, to diverse phoneme usage in output transcripts by maximizing entropy, and a self-supervised reconstruction loss, $\mathcal{L}_{ss}$, to encourage the generated phonemes to match the input audio.

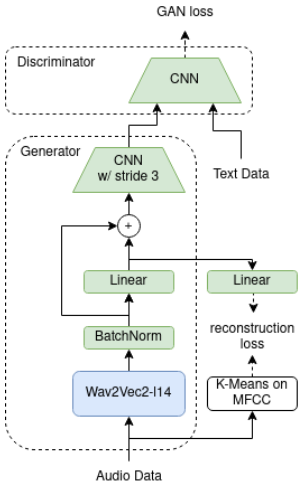


Figure 3: Outline of W2V2-U2.0 training procedure. The CNN module generates phoneme logits for the input audio. The bridge connector is trained on the GAN objective with reconstruction loss and regularization. During inference, the generator and linear layer used for reconstruction are discarded.

The reconstruction term uses a separate linear head to classify each speech embedding into 1 of 64 clusters, ζ_t, obtained from running k-means on the Mel-frequency cepstral coefficient (MFCC) features of the input audio (Hsu et al., 2021; Liu et al., 2022). The final GAN training objective, $\min_{\mathcal{G}} \max_{\mathcal{C}} \mathcal{L}$, is summarized in Equation 1. The training procedure for the bridge module is outlined in Figure 3. Similar to W2V2-U2.0 (Baevski et al., 2021), SSE bridge models are trained on unlabelled audio and text from the Librispeech (Panayotov et al., 2015) dataset.

$$
\begin{aligned}
\mathcal{L} &= \mathcal{L}_{gan} + \lambda\mathcal{L}_{gp} + \gamma\mathcal{L}_{sp} + \eta\mathcal{L}_{pd} + \delta\mathcal{L}_{ss} \\
\mathcal{L}_{gan} &= \mathop{\mathbb{E}}_{T_u}[log\mathcal{C}(T_u)] + \mathop{\mathbb{E}}_{X}[log(1-\mathcal{C}(\mathcal{G}(X))] \\
\mathcal{L}_{gp} &= \mathop{\mathbb{E}}_{\substack{X,T_u \\ \mu\sim U(0,1) \\ \mu'=1-\mu}} [(||\nabla\mathcal{C}(\mu\mathcal{G}(X) + \mu' T_u|| - 1)^2] \\
\mathcal{L}_{sp} &= \sum_{(p_t,p_{t+1})\in\mathcal{G}(X)} ||p_t - p_{t+1}||^2 \\
\mathcal{L}_{pd} &= \frac{1}{|B|}\sum_{S\in B} -H_{\mathcal{G}}(\mathcal{G}(S)) \\
\mathcal{L}_{ss} &= -\sum_t log P_{\mathcal{G}}(\zeta_t|X). \qquad (1)
\end{aligned}
$$

A.3 Hyperparameter Settings

A.3.1 Speech Encoder

We augment both W2V2 (Baevski et al., 2020) and HUBERT (Hsu et al., 2021) by introducing semantics. Specifically, we use the W2V2-Large LV-60 model and HUBERT-Large models, which are pretrained on just unlabelled audio and implemented with the fairseq library (Ott et al., 2019).

A.3.2 Large Language Model

We use BART (Lewis et al., 2019) as our LLM since it is pretrained to handle noisy input. In our SLU experiments, we use BART-Base model, which has lower computational overhead. For our SQA experiments, we use BART-Large, since SQA is a more challenging task. Note, unlike baselines that train the whole LLM, SSE freezes all weights in its LLM except adapters optionally, hence SSE has lower overhead. All LLMs were implemented using the huggingface library (Wolf et al., 2019).

A.3.3 Residual Attention and Adapters

We choose the residual attention layer to be the same dimension as our speech encoder, which is 1024 for both W2V2 (Baevski et al., 2020) and HUBERT (Hsu et al., 2021). We implement our general recipe for adapters using the adapter-transformers

package (Pfeiffer et al., 2020) and pyTorch (Paszke et al., 2019).

A.3.4 Bridge Connector

We follow the same hyperparameter settings reported in the w2v2-U2.0 paper (Liu et al., 2022). Specifically, we use a 2-layer CNN with stride 3. The model is trained on unlabelled Librispeech-960 (Panayotov et al., 2015) data for 100,000 epochs with a learning rate of 5e-5 and 3e-4 for the generator and discriminator respectively. Decoding is done using a WFST in the same way as w2v2-U2.0 (Liu et al., 2022). Similar to w2v2-U2.0, we pre-process the Librispeech-960 by removing silences with an unsupervised model, but not during fine-tuning or testing. We believe such techniques could further improve performance, but leave it as future work. The regularized GAN loss function hyperparameters, as stated in Section A.2 are set to $1.0/1.5$, $1.5/2.5$, $0/3$, and $0.3/0.5$ for λ, γ, η, and δ respectively.

A.3.5 SLU Training Details

As mentioned in Section 4.3, we use the standard decoders provided by SUPERB (Yang et al., 2021). We ran a grid search on 5 settings for learning rate on an exponential scale of 2 around the default settings from SUPERB (Yang et al., 2021) and found said default hyperparameters optimal. Specifically, we set the learning rate to $1e-4$, $1e-4$, $2e-4$, and $2e-4$ for FSC-IC, SLURP-IC, SLURP-SF, and SLUE-NER respectively. All methods use the AdamW (Loshchilov and Hutter, 2017) optimizer with gradient clipping set to 1 for 200,000 total steps to convergence. Validation performance is used to pick the best model for all datasets except SLUE, since SLUE test data is not publicly available.

A.3.6 SQA Training Details

As mentioned in Section 4.3, we use a frame-level linear layer classification head as our decoder. We follow DUAL's (Lin et al., 2022a) default hyperparameter settings with a learning rate of 1e-4. We train the models using the same warm-up and decay strategies as DUAL with the AdamW (Loshchilov and Hutter, 2017) optimizer for 5,000 steps to convergence.

A.4 Training Setup and Time

All models were trained on a server with 8 Nvidia Tesla V100 GPUs. The total training time for the bridge module takes around a day. The total training time for downstream tasks take between half a day and one day.

A.5 Dataset Details

As mentioned in Section 4.1, we evaluate SSE on 3 SLU tasks, intent classification (IC), slot filling (SF), and named entity recognition (NER), and the SQA task. The goal of IC is to classify the intent of an input audio snippet. The goal of SF is to extract certain attributes of a given intent from an audio snippet. The goal of NER is to identify named entities in an audio snippet. The goal of SQA is to find the start and end frames of the answer in a spoken passage given a spoken question.

A.5.1 FSC

The FSC dataset (Lugosch et al., 2019) is an IC dataset for a smart home virtual assistant. The input is a single audio file containing spoken English commands and the output class is the intent of the spoken command. The data was obtained through crowd-sourcing from 97 native and non-native English speakers. In total, there are 31 intents. The number of utterances and hours of each split can be found in the Table 9.

A.5.2 SLURP

The SLURP dataset (Bastianelli et al., 2020) is an IC and SF dataset for an in-home personal robot assistant. The input is a single audio file containing spoken English commands and the output is the scenerio, action, and entities. In total, there are 18 different scenarios, 46 different actions (IC), and 56 different entities (SF). The data was collected from 177 native and non-native English speaking Amazon Mechanical Turk workers. The number of utterances and hours of each split can be found in Table 9. SLURP use both headsets and microphones with various placement configurations.

A.5.3 SLUE

The SLUE dataset (Shon et al., 2022) is a NER dataset using European Parliament event recordings. The input is a single audio file containing spoken English passages and the output are the named entities. There are in total 7 categories that were based on the OntoNotes Release 5.0 (Hovy et al., 2006) entity labels. The dataset was collected from the official European Parliament website. The

Dataset	# of Utterances	# of Hours
FSC-train	23,132	14.7
FSC-dev	3,118	1.9
FSC-test	3,793	2.4
SLURP-train	50,628	40.2
SLURP-dev	8,690	6.9
SLURP-test	13,078	10.3
SLUE-train	5,000	14.5
SLUE-dev	1,753	5.0

Table 9: Dataset statistics for FSC, SLURP, and SLUE. Note, for SLUE, only the train and dev splits are publicly available, thus we evaluate on the dev set.

number of utterances and hours of each split can be found in the Table 9.

A.5.4 NMSQA

The NMSQA dataset (Lin et al., 2022a) is a SQA dataset generated from a standard text question answering dataset, SQuAD-v1.1 [2], using Amazon Polly Text-to-Speech [3] for the train and dev split, and 60 human speakers for the test set. NMSQA contains 297.18 hours, 37.61 hours, and 2.67 hours of train, dev, and test split audio respectively. We follow DUAL (Lin et al., 2022a) by evaluating on Frame-level F1 score (FF1) and Audio Overlapping Score (AOS).

[2]A question answering dataset using Wikipedia articles
[3]https://aws.amazon.com/tw/polly

ACL 2023 Responsible NLP Checklist

A For every submission:

☐ A1. Did you describe the limitations of your work?
Left blank.

☐ A2. Did you discuss any potential risks of your work?
Left blank.

☐ A3. Do the abstract and introduction summarize the paper's main claims?
Left blank.

☐ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☐ Did you use or create scientific artifacts?

Left blank.

☐ B1. Did you cite the creators of artifacts you used?
Left blank.

☐ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
Left blank.

☐ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
Left blank.

☐ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
Left blank.

☐ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Left blank.

☐ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
Left blank.

C ☐ Did you run computational experiments?

Left blank.

☐ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Left blank.

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☐ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Left blank.

☐ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Left blank.

☐ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Left blank.

D ☐ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
Left blank.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
Left blank.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
Left blank.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
Left blank.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
Left blank.