### Self-Supervised Learning for Speech

Hao Tang

#### Automatic Speech Recognition—ASR Lecture 17 17 March 2024

Hao Tang Self-Supervised Learning for Speech

- How self-supervised learning came about
- What self-supervised learning is
- Examples of SSL on speech
  - Contrastive Predictive Coding
  - wav2vec 2.0
  - HuBERT
- How SSL models are used
  - Fine-tuning
  - Probing

• Training on one task can sometimes help learn other tasks.

- Training on one task can sometimes help learn other tasks.
- Two examples
  - Word embeddings (Mikolov et al., 2013)
  - Supervised pre-training in computer vision (Girshick et al., 2014)

### Background

- Girshick *et al.* (2014) found that a pre-trained image classifier can be fine-tuned for object detection.
- They named the idea supervised pre-training.

	mAP
DPM v5	33.7
DPM ST	29.1
DPM HSC	34.3
R-CNN pool <sub>5</sub>	44.2
R-CNN fc <sub>6</sub>	46.2
R-CNN fc7	44.7
R-CNN fine-tuned pool <sub>5</sub>	47.3
R-CNN fine-tuned fc <sub>6</sub>	53.1
R-CNN fine-tuned fc7	54.2



\* 注入 \* 注入 -

# Supervised Pre-Training





▶ ★ 문 ▶ ★ 문 ▶

# Supervised Pre-Training



- Training on one task can sometimes help learn other tasks.
- The tasks in pre-training is called the **pretext task**, while the other tasks that might benefit from pre-training are called **downstream tasks**.
- Does pre-training (i.e., the pretext task) needs to be supervised?



- The pretext task in this case is to predict the relative position of patches.
- Inspired by word2vec, Doersch *et al.* (2015) coins the approach self-supervised learning.
- Why does this work?

• If the models knows \_\_\_\_\_, then it should be able do well on \_\_\_\_\_.

- If the models knows \_\_\_\_\_, then it should be able do well on \_\_\_\_\_.
- If the models knows something about images, then it should be able do well on context prediction.

- If the models knows \_\_\_\_\_, then it should be able do well on \_\_\_\_\_.
- If the models knows something about images, then it should be able do well on context prediction.
- We train a model to do context prediction and hope that the model can know something about images.



(Pathak et al., 2016)

æ

• • = • • = •



(Larsson et al., 2016)

・ロト ・回ト ・ヨト ・ヨト

### Prediction Features in the Future



(Vondrick et al., 2016)

э

イロト イポト イヨト ・

### Prediction Features in the Future

- Predicting the future in the feature space seems like a reasonable pretext task for self-supervised learning.
- To formalize this, we want to train f to predict  $\phi(x_{t+1})$  from  $\phi(x_t)$ , where  $\phi(x_t)$  is the feature of  $x_t$ .
- If we train both f and  $\phi$  to minimize

$$\|f(\phi(x_t)) - \phi(x_{t+1})\|_2^2, \tag{1}$$

there are trivial solutions where  $\phi(x) = cI$  for any constant c.

• Vondrick *et al.* (2016) use a pre-trained network for  $\phi$  and only trains f while holding  $\phi$  fixed.

### Contrastive Predictive Coding (CPC)



(van den Oord et al., 2018)

э

- 4 同 ト 4 ヨ ト 4 ヨ ト

- The goal of CPC is to predict the future in the future space.
- It suffers the same problem, having trivial solutions.
- Instead of predicting the future with mean-squared error, van den Oord *et al.* (2018) adopt a contrastive approach, to distinguish the correct one from others.

# Contrastive Predictive Coding (CPC)

- Suppose we want to use  $c_t$  to predict  $z_{t+3}$ .
- We know that minimizing  $||z_{t+3} Wc_t||_2^2$  leads to a degenerate solution.
- Instead, we want  $z_{t+3}^{\top}Wc_t$  to be high, and  $z^{\top}Wc_t$  to be low for any other z.
- The correct sample, in this case  $z_{t+3}$ , is typically called the **positive example**, while the others are called **negative examples**.

# Contrastive Predictive Coding (CPC)

- We want  $z_{t+3}^{\top}Wc_t$  to be high, and  $z^{\top}Wc_t$  to be low for any other z.
- In other words, we want

$$\log \frac{\exp(z_{t+3}^{\top}Wc_t)}{\sum_{z \in \mathcal{N} \cup \{z_{t+3}\}} \exp(z^{\top}Wc_t)}$$
(2)

to be high, where N is the set of negative samples.

• The negative samples can be other frames of the same utterance, frames from other utterances of the same speaker, or frames from other utterances of different speakers.

• The final objective is

$$\sum_{t=1}^{T-3} \log \frac{\exp(z_{t+3}^{\top} W c_t)}{\sum_{z \in \mathcal{N} \cup \{z_{t+3}\}} \exp(z^{\top} W c_t)}.$$
(3)

- The number of frames into the future (3 in  $z_{t+3}$ ) is a *necessary* hyperparameter.
- The negative examples are typically just all the frames in the batch.



- The model architecture is a 12-layer Transformer.
- Instead of future prediction, wav2vec 2.0 uses masked prediction.
- The design is heavily inspired by BERT (Devlin et al., 2019).
- There is a quantization layer after the CNN.

- Vector quantization is a procedure that converts a vectors into an integer.
- The integer is called a **code**, and every integer corresponds to a **code vector**.
- The set of integers and their corresponding vectors comprise a **codebook**.
- Given a vector, vector quantization finds the closest code vector in the codebook and returns the code.

### Vector Quantization



Hao Tang Self-Supervised Learning for Speech

æ

▶ ▲ 문 ▶ ▲ 문 ▶

### Vector Quantization



æ

▶ ▲ 문 ▶ ▲ 문 ▶

### Vector Quantization



Hao Tang Self-Supervised Learning for Speech

æ

< E > < E >

• wav2vec 2.0 still uses the contrastive objective

$$\sum_{t \in \mathcal{M}} \log \frac{\exp(\cos(q_t, c_t))}{\sum_{q \in \mathcal{N} \cup \{q_t\}} \exp(\cos(q, c_t))},$$
(4)

where M is the indices of the masked frames and N is the set of negative samples.



• HuBERT uses a cross entropy

$$\sum_{t \in M} \log \frac{\exp(\cos(q_t, c_t))}{\sum_{q \in V} \exp(\cos(q, c_t))},$$
(5)

where M is the indices of the masked frames and V is the codebook.

★ ∃ ► < ∃ ►</p>

- A pre-trained model can serve as an initialization of another model on a new task. This approach is often called fine-tuning.
- A pre-trained model can also be used as a feature extractor.



▲御▶ ▲屋▶ ▲屋▶

## Fine-Tuning



#### wav2vec 2.0 Results

Madal	Unlabeled LM data LM	dev		test		
Model		clean	other	clean	other	
10 min labeled						
Discrete BERT []	LS-960	4-gram	15.7	24.1	16.3	25.2
BASE	LS-960	4-gram	8.9	15.7	9.1	15.6
		Transf.	6.6	13.2	6.9	12.9
Large	LS-960	Transf.	6.6	10.6	6.8	10.8
	LV-60k	Transf.	4.6	7.9	4.8	8.2
1h labeled						
Discrete BERT [4]	LS-960	4-gram	8.5	16.4	9.0	17.6
BASE	LS-960	4-gram	5.0	10.8	5.5	11.3
		Transf.	3.8	9.0	4.0	9.3
Large	LS-960	Transf.	3.8	7.1	3.9	7.6
	LV-60k	Transf.	2.9	5.4	2.9	5.8
10h labeled						
Discrete BERT	LS-960	4-gram	5.3	13.2	5.9	14.1
Iter. pseudo-labeling 58	LS-960	4-gram+Transf.	23.51	25.48	24.37	26.02
	LV-60k	4-gram+Transf.	17.00	19.34	18.03	19.92
BASE	LS-960	4-gram	3.8	9.1	4.3	9.5
		Transf.	2.9	7.4	3.2	7.8
Large	LS-960	Transf.	2.9	5.7	3.2	6.1
	LV-60k	Transf.	2.4	4.8	2.6	4.9
100h labeled						
Hybrid DNN/HMM [34]	-	4-gram	5.0	19.5	5.8	18.6
TTS data augm. [30]	-	LSTM			4.3	13.5
Discrete BERT	LS-960	4-gram	4.0	10.9	4.5	12.1
Iter. pseudo-labeling 58	LS-860	4-gram+Transf.	4.98	7.97	5.59	8.95
	LV-60k	4-gram+Transf.	3.19	6.14	3.72	7.11
Noisy student [42]	LS-860	LSTM	3.9	8.8	4.2	8.6
BASE	LS-960	4-gram	2.7	7.9	3.4	8.0
		Transf.	2.2	6.3	2.6	6.3
Large	LS-960	Transf.	2.1	4.8	2.3	5.0
	LV-60k	Transf.	1.9	4.0	2.0	4.0

Hao Tang Self-Supervised Learning for Speech

æ

《曰》《聞》《臣》《臣》。

### wav2vec 2.0 Results

Madal	Unlabeled LM	dev		test		
Model		clean	other	clean	other	
10 min labeled						
Discrete BERT 🖪	LS-960	4-gram	15.7	24.1	16.3	25.2
BASE	LS-960	4-gram	8.9	15.7	9.1	15.6
		Transf.	6.6	13.2	6.9	12.9
Large	LS-960	Transf.	6.6	10.6	6.8	10.8
	LV-60k	Transf.	4.6	7.9	4.8	8.2
1h labeled						
Discrete BERT [4]	LS-960	4-gram	8.5	16.4	9.0	17.6
BASE	LS-960	4-gram	5.0	10.8	5.5	11.3
		Transf.	3.8	9.0	4.0	9.3
Large	LS-960	Transf.	3.8	7.1	3.9	7.6
	LV-60k	Transf.	2.9	5.4	2.9	5.8
10h labeled						
Discrete BERT	LS-960	4-gram	5.3	13.2	5.9	14.1
Iter. pseudo-labeling [58]	LS-960	4-gram+Transf.	23.51	25.48	24.37	26.02
	LV-60k	4-gram+Transf.	17.00	19.34	18.03	19.92
BASE	LS-960	4-gram	3.8	9.1	4.3	9.5
		Transf.	2.9	7.4	3.2	7.8
Large	LS-960	Transf.	2.9	5.7	3.2	6.1
	LV-60k	Transf.	2.4	4.8	2.6	4.9
100h labeled						
Hybrid DNN/HMM [34]	-	4-gram	5.0	19.5	5.8	18.6
TTS data augm. [30]	-	LŠTM			4.3	13.5
Discrete BERT	LS-960	4-gram	4.0	10.9	4.5	12.1
Iter. pseudo-labeling [58]	LS-860	4-gram+Transf.	4.98	7.97	5.59	8.95
	LV-60k	4-gram+Transf.	3.19	6.14	3.72	7.11
Noisy student [42]	LS-860	LSTM	3.9	8.8	4.2	8.6
BASE	LS-960	4-gram	2.7	7.9	3.4	8.0
		Transf.	2.2	6.3	2.6	6.3
Large	LS-960	Transf.	2.1	4.8	2.3	5.0
	LV-60k	Transf.	1.9	4.0	2.0	4.0

æ

《曰》《聞》《臣》《臣》。



<回>< E> < E> < E> <



• • = • • = •





- The hidden vectors are often better representations than the input at solving a task.
- Nowadays, hidden vectors and representations are often used interchangeably.
- In other words, when we take one of the layers of HuBERT when inputting a speech utterance, we get a representation of this particular utterance.
- Don't forget that wave samples and Mel spectrograms are also speech representations.

# Probing

- What can we decode from a given representation?
- For example, can we decode phones from a given representation?
- To answer this question, we train a linear classifier to predict phones.
- A non-trivial accuracy tells us to what extent we can decode phones from a given representation.
- The act is usually called phone probing, and the classifier is called a probing classifier.
- There are no particular restrictions on what a probing classifier should be, but they are typically just a linear classifier.

### Layer-Wise Analysis



- Pre-training followed by fine-tuning is a simple approach to leveraging data from different tasks.
- In particular, self-supervised learning makes it possible to leverage unlabeled data.
- Speech representations learned from self-supervision have found their way to many different applications.