Compressing Self-Supervised Models

Tsu-Quan Lin[1], Hao Tang[2], Hung-yi Lee[1]

[1]National Taiwan University, [2]The University of Edinburgh

Goal of the project

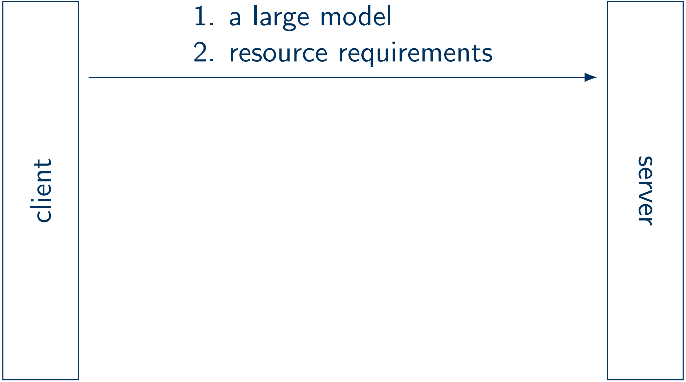
- Self-supervised models are large (and getting larger).
- There usually exists a much smaller network that can perform equally well as the large one.
- The goal is to **compress** self-supervised models for speech.

Model compression

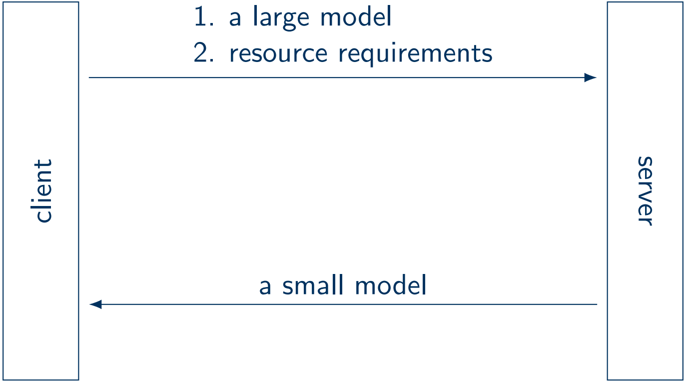
- Common compression techniques have little in common.
  - Pruning
  - Knowledge distillation
  - Low-rank approximation

Model compression

Model compression



Model compression



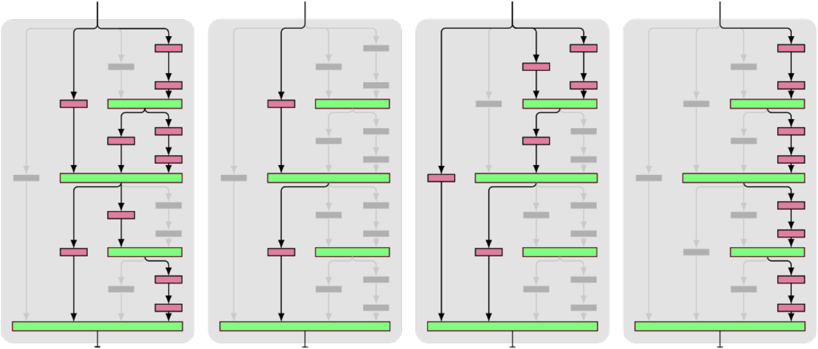
Model compression

- Common compression techniques have little in common.
  - Pruning
  - Knowledge distillation
  - Low-rank approximation
- Any technique qualifies as model compression as long as a smaller model is returned.
  - Neural architecture search
  - Caching models of different sizes

Model compression

- Common compression techniques have little in common.
  - Pruning
  - Knowledge distillation
  - Low-rank approximation
- Any technique qualifies as model compression as long as a smaller model is returned.
  - Neural architecture search
  - Caching models of different sizes
- It is a task of finding a model that satisfies a set of resource constraints.
  - Number of floating point operations
  - Number of compute cores
  - Memory consumption

Anytime Inference



(Larsson et al., 2017)

Scope of the project

- The landscape of compressing self-supervised models
- Anytime Transformers

Outline

- Desirable properties of self-supervised models
- Properties of common compression techniques
- Plans and experiments

Desirable properties of self-supervised models

Desirable properties of self-supervised models

- Wide applicability through fine-tuning
  - ASR
  - speaker verifitication

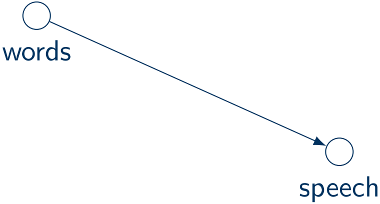
Desirable properties of self-supervised models

- Wide applicability through fine-tuning
  - ASR
  - speaker verifitication

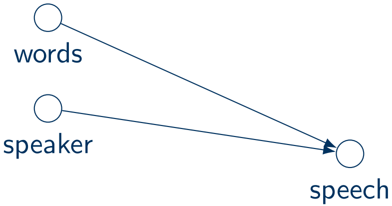
- Improved **accessibility** of high-level concepts
  - acoustic unit discovery
  - phone or word segmentation
  - discrete units for TTS

Representation of speech

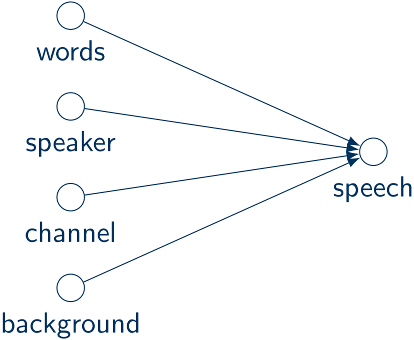
Representation of speech



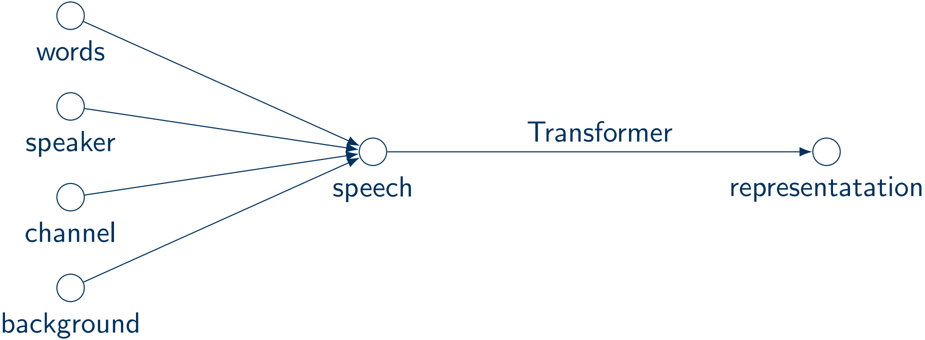
Representation of speech



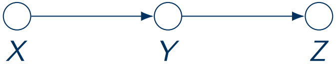
Representation of speech



Representation of speech

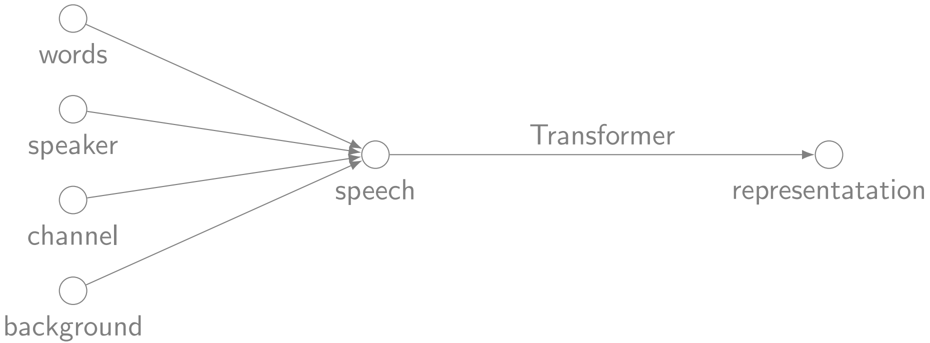


Data processing inequality



$$I(X, Y) \geq I(X, Z)$$

Representation of speech



$$I(\{\text{words}, \text{speaker}, \ldots\}, \log \text{Mel}) \geq I(\{\text{words}, \text{speaker}, \ldots\}, \text{representation})$$

Consequence of the data processing inequality

Consequence of the data processing inequality

- ~~Richer representation~~

Consequence of the data processing inequality

- ~~Richer representation~~
- Nothing is richer than the input log Mel spectrograms themselves.

Consequence of the data processing inequality

- ~~Richer representation~~
- Nothing is richer than the input log Mel spectrograms themselves.
- High-level concepts, such as phonetic information, prosodic information, speaker characteristics, are more **accessible**.

Consequence of the data processing inequality

- ~~Richer representation~~
- Nothing is richer than the input log Mel spectrograms themselves.
- High-level concepts, such as phonetic information, prosodic information, speaker characteristics, are more **accessible**.
- Typically, accessibility is measured by a linear probing model.

Desirable properties of self-supervised models

- Wide applicability through fine-tuning
  - ASR
  - speaker verifitication
- Improved **accessibility** of high-level concepts
  - acoustic unit discovery
  - phone or word segmentation
  - discrete units for TTS

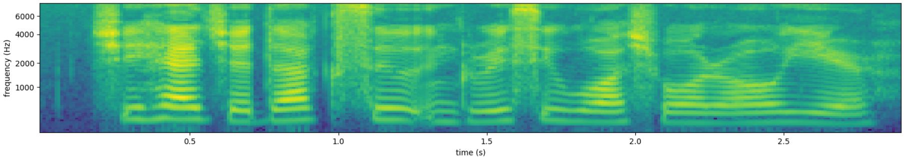
Research questions

- Are the returned small models still susceptible to fine-tuning?
- Do the returned small models preserve accessibility of high-level concepts?

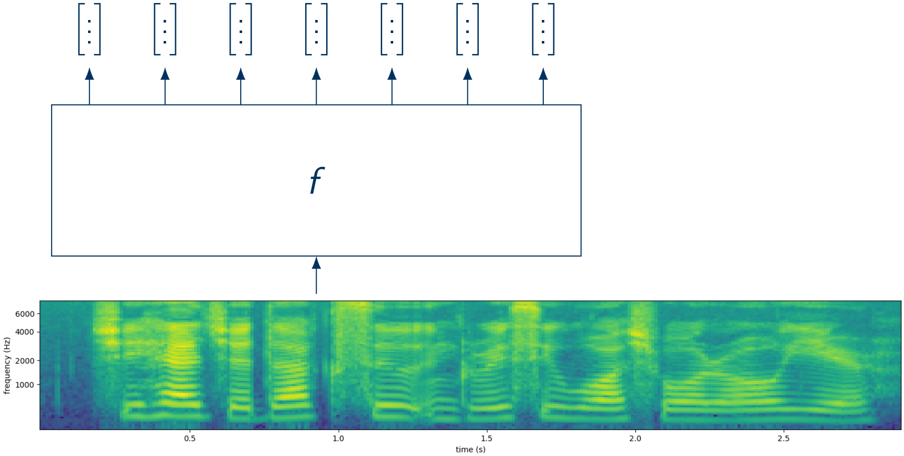
Baseline preparation

$$
\begin{tikzcd}
\begin{array}{c}\text{APC}\\ \text{LSTM}\end{array} \arrow[r] \arrow[d] & \begin{array}{c}\text{APC}\\ \text{Causal Transformer}\end{array} & \\
\begin{array}{c}\text{Target-quantized APC}\\ \text{LSTM}\end{array} \arrow[r] & \begin{array}{c}\text{Target-quantized APC}\\ \text{Causal Transformer}\end{array} \arrow[r] & \text{HuBERT}
\end{tikzcd}
$$

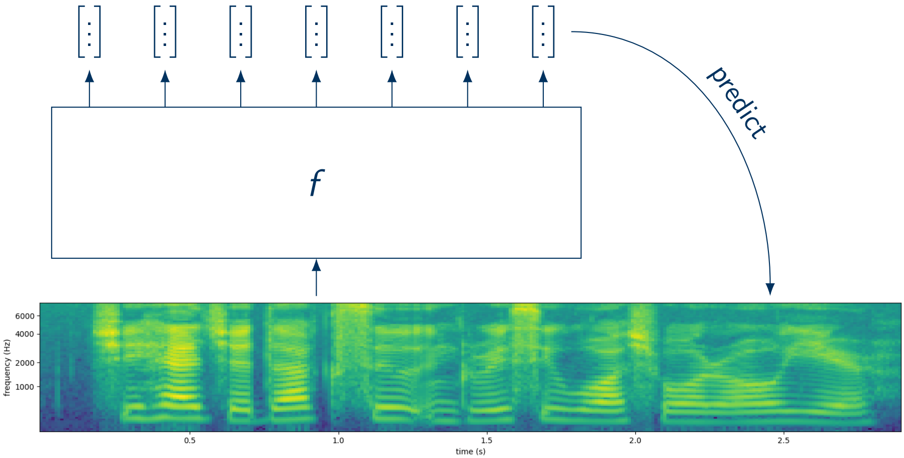
Autoregressive predictive coding (APC) (Chung et al., 2019)



Autoregressive predictive coding (APC) (Chung et al., 2019)



Autoregressive predictive coding (APC) (Chung et al., 2019)



Autoregressive predictive coding (APC) (Chung et al., 2019)

$$h_1, \ldots, h_t = f(x_1, \ldots, x_t)$$

Autoregressive predictive coding (APC) (Chung et al., 2019)

$$h_1, \ldots, h_t = f(x_1, \ldots, x_t)$$

$$\sum_{t=1}^{T-k} \|x_{t+k} - W h_t\|_2^2$$

Autoregressive predictive coding (APC) (Chung et al., 2019)

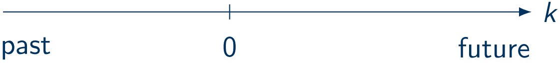
$$h_1, \ldots, h_t = f(x_1, \ldots, x_t)$$

$$\sum_{t=1}^{T-k} \|x_{t+k} - W h_t\|_2^2$$

Autoregressive predictive coding (APC) (Chung et al., 2019)

$$h_1, \ldots, h_t = f(x_1, \ldots, x_t)$$

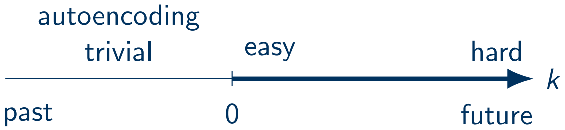
$$\sum_{t=1}^{T-k} \|x_{t+k} - W h_t\|_2^2$$



Autoregressive predictive coding (APC) (Chung et al., 2019)

$$h_1, \ldots, h_t = f(x_1, \ldots, x_t)$$

$$\sum_{t=1}^{T-k} \|x_{t+k} - W h_t\|_2^2$$



Target-quantized APC (Yeh and Tang, 2022)

$$h_1, \ldots, h_t = f(x_1, \ldots, x_t)$$

Target-quantized APC (Yeh and Tang, 2022)

$$h_1, \ldots, h_t = f(x_1, \ldots, x_t)$$

$$c_{t+k} = \operatorname*{argmin}_{i=1,\ldots,C} \|x_{t+k} - v_i\|_2^2$$

Target-quantized APC (Yeh and Tang, 2022)

$$h_1, \dots, h_t = f(x_1, \dots, x_t)$$

$$c_{t+k} = \operatorname*{argmin}_{i=1,\dots,C} \|x_{t+k} - v_i\|_2^2$$

$$\sum_{t=1}^{T-k} \log \frac{\exp(w_{c_{t+k}}^\top h_t)}{\sum_{i=1}^{C} \exp(w_i^\top h_t)}$$

Target-quantized APC (Yeh and Tang, 2022)

$$h_1, \dots, h_t = f(x_1, \dots, x_t)$$

$$c_{t+k} = \operatorname*{argmin}_{i=1,\dots,C} \|x_{t+k} - v_i\|_2^2$$

$$\sum_{t=1}^{T-k} \log \frac{\exp(w_{c_{t+k}}^\top h_t)}{\sum_{i=1}^{C} \exp(w_i^\top h_t)}$$

Baseline preparation

APC
LSTM

$\downarrow$

Target-quantized APC
LSTM

Baseline preparation

APC
LSTM

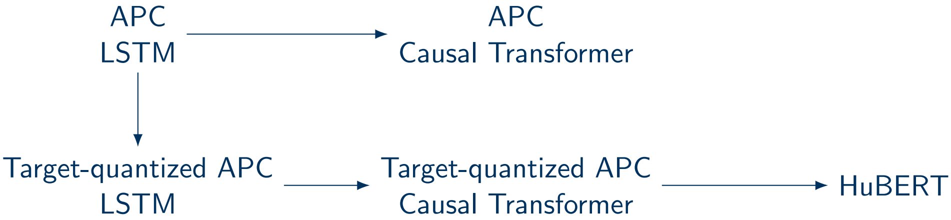
APC
Causal Transformer

Target-quantized APC
LSTM

Baseline preparation

$$\begin{tikzcd}
\begin{matrix}\text{APC}\\ \text{LSTM}\end{matrix} \arrow[r] \arrow[d] & \begin{matrix}\text{APC}\\ \text{Causal Transformer}\end{matrix} \\
\begin{matrix}\text{Target-quantized APC}\\ \text{LSTM}\end{matrix} \arrow[r] & \begin{matrix}\text{Target-quantized APC}\\ \text{Causal Transformer}\end{matrix}
\end{tikzcd}$$

Baseline preparation



More baselines

- Small models (fewer heads, fewer layers)
- Lottery ticket hypothesis
- Knowledge distillation
- Low-rank approximation

Lottery ticket hypothesis (Frankle and Carbin, 2018)

Lottery ticket hypothesis (Frankle and Carbin, 2018)

- Weight pruning (with ℓ_1) + re-training (Han et al., 2015)

Lottery ticket hypothesis (Frankle and Carbin, 2018)

- Weight pruning (with ℓ_1) + re-training (Han et al., 2015)
- Significant size reduction (80% to 90%)

Lottery ticket hypothesis (Frankle and Carbin, 2018)

- Weight pruning (with ℓ_1) + re-training (Han et al., 2015)
- Significant size reduction (80% to 90%)
- Not GPU-friendly

Lottery ticket hypothesis (Frankle and Carbin, 2018)

- Weight pruning (with ℓ_1) + re-training (Han et al., 2015)
- Significant size reduction (80% to 90%)
- Not GPU-friendly
- Great for network architecture search

Lottery ticket hypothesis (Frankle and Carbin, 2018)

- Weight pruning (with ℓ_1) + re-training (Han et al., 2015)
- Significant size reduction (80% to 90%)
- Not GPU-friendly
- Great for network architecture search
- Sensitive to re-initialization

Knowledge distillation (Ba and Caruana, 2014)

Knowledge distillation (Ba and Caruana, 2014)

- Training on the output of another model

Knowledge distillation (Ba and Caruana, 2014)

- Training on the output of another model
- Better than regular training

Knowledge distillation (Ba and Caruana, 2014)

- Training on the output of another model
- Better than regular training
- Related to learning data geometry (Phuong and Lampert, 2019)

Knowledge distillation (Ba and Caruana, 2014)

- Training on the output of another model
- Better than regular training
- Related to learning data geometry (Phuong and Lampert, 2019)
- Leading to drastically different student models (Stanton et al., 2021)

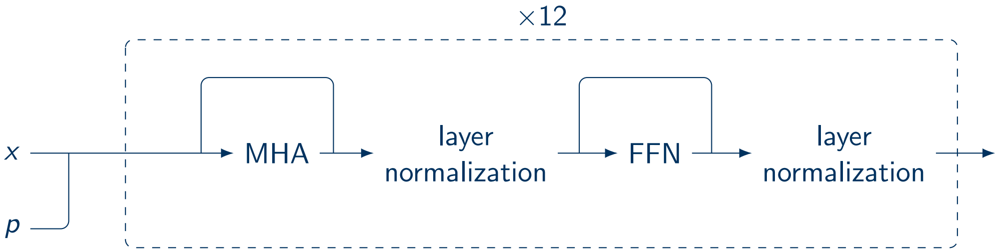
Low-rank approximation (Sainath et al., 2013)

Low-rank approximation (Sainath et al., 2013)

$$
\begin{aligned}
\min_{U,V} \quad & \|W - UV\|_2^2 \\
\text{s.t.} \quad & rank(UV) \leq k
\end{aligned}
$$

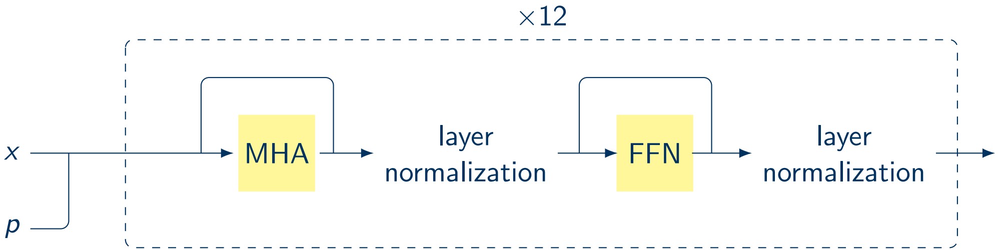
Low-rank approximation (Sainath et al., 2013)

$$\min_{U,V} \quad \|W - UV\|_2^2$$
$$\text{s.t.} \quad rank(UV) \leq k$$



Low-rank approximation (Sainath et al., 2013)

$$\min_{U,V} \quad \|W - UV\|_2^2$$
$$\text{s.t.} \quad rank(UV) \leq k$$



Low-rank approximation (Sainath et al., 2013)

$$
\begin{aligned}
\min_{U,V} \quad & \|W - UV\|_2^2 \\
\text{s.t.} \quad & rank(UV) \leq k
\end{aligned}
$$

- FFN typically occupies a significant amount of memory.
- The approach can be training-free.

Research questions

- Are the returned small models still susceptible to fine-tuning?
- Do the returned small models preserve accessibility of high-level concepts?
- Do we allow re-training at compression time?
- Do we have access to a large pre-trained model at compression time?

Where we are

$$
\begin{tikzcd}
\text{APC LSTM} \arrow[r] \arrow[d] & \text{APC Causal Transformer} & \\
\text{Target-quantized APC LSTM} \arrow[r] & \text{Target-quantized APC Causal Transformer} \arrow[r] & \text{HuBERT}
\end{tikzcd}
$$

Timeline

Apr–Jun Baselines

- Reproducing HuBERT
- Implement lottery ticket hypothesis
- Implement knowledge distillation
- Implement low-rank approximation
- Pilot experiments on LibriSpeech 360 hours

Jun–Aug Workshop period

- Scaling up to LibriSpeech 960 hours
- Analyses

Aug–Oct Wrap-up

- ICASSP submission