Wav2vec 2.0 – An Overview

A Foundation Model for ASR

Tirthankar Banerjee

Author={Baevski, Alexei and Zhou, Yuhao and Mohamed, Abdelrahman and Auli, Michael},

Journal={Advances in neural information processing systems},

Year={2020}

Code and models are available at https://github.com/pytorch/fairseq

wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations

Alexei Baevski **Henry Zhou** **Abdelrahman Mohamed** **Michael Auli**

{abaevski,henryzhou7,abdo,michaelauli}@fb.com

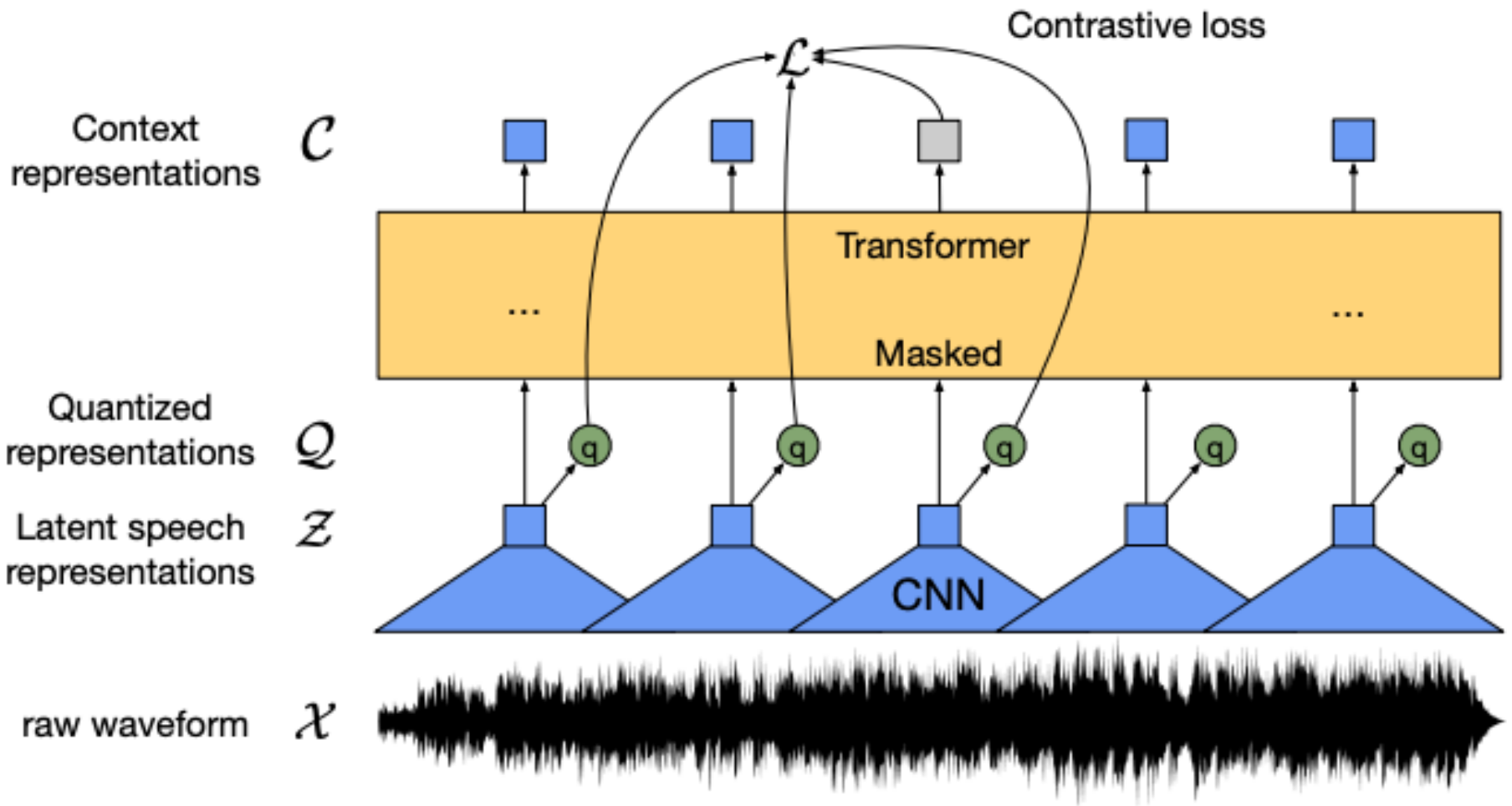
Facebook AI

Abstract

We show for the first time that learning powerful representations from speech audio alone followed by fine-tuning on transcribed speech can outperform the best semi-supervised methods while being conceptually simpler. wav2vec 2.0 masks the speech input in the latent space and solves a contrastive task defined over a quantization of the latent representations which are jointly learned. Experiments using all labeled data of Librispeech achieve 1.8/3.3 WER on the clean/other test sets. When lowering the amount of labeled data to one hour, wav2vec 2.0 outperforms the previous state of the art on the 100 hour subset while using 100 times less labeled data. Using just ten minutes of labeled data and pre-training on 53k hours of unlabeled data still achieves 4.8/8.2 WER. This demonstrates the feasibility of speech recognition with limited amounts of labeled data.

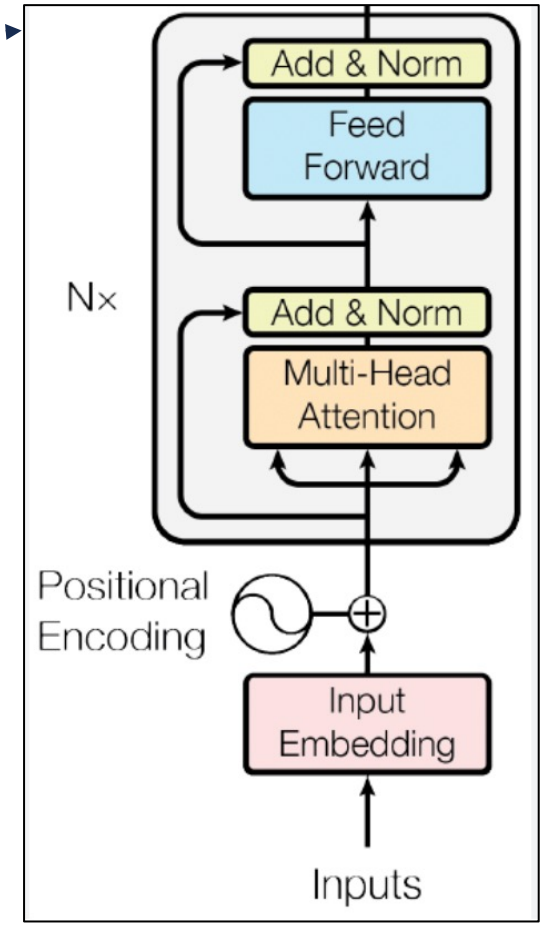
https://arxiv.org/pdf/2006.11477.pdf

Architecture Schematic



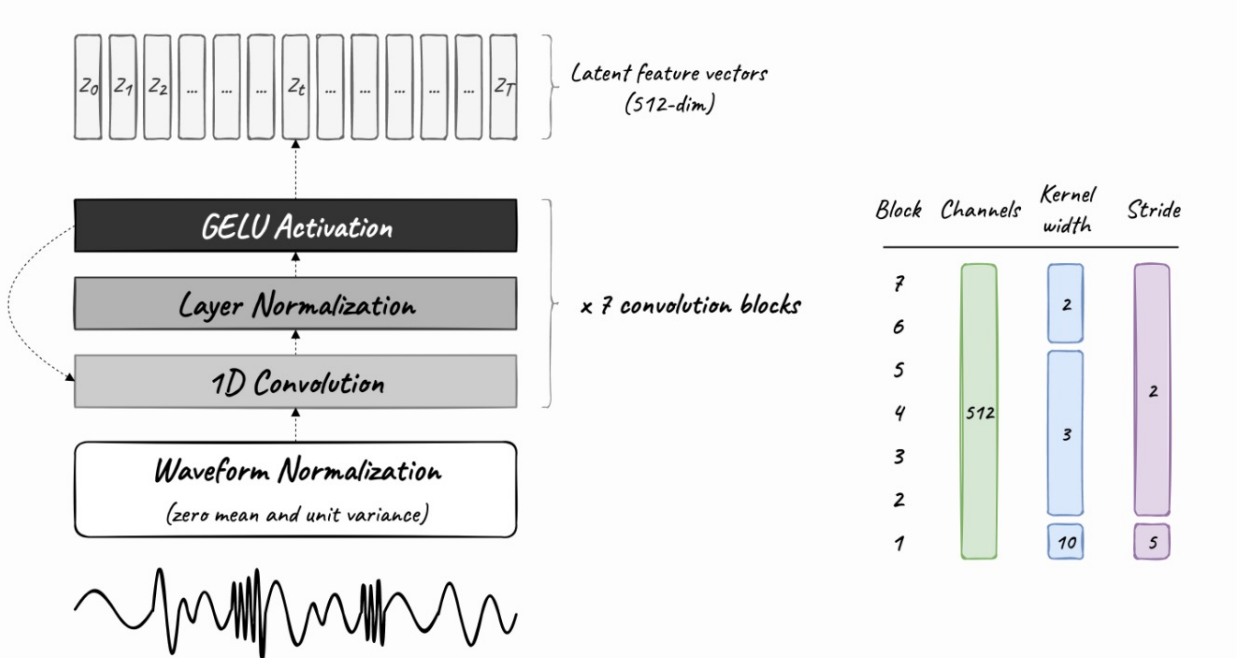
Wav2vec2 – Salient Points

- Architecture based on the Transformer's encoder
- Semi-supervised training: first, pre-train the model on a large quantity of unlabeled speech, then fine-tune on a smaller labeled dataset
- Four important elements
  - Feature Encoder, Context Network, Quantization Module, and the Contrastive Loss (pre-training objective)
- Automatically learn discrete speech units
- Contrastive task: Requires the model to identify the correct quantized speech units for the masked positions



Convolution: https://github.com/vdumoulin/conv_arithmetic/blob/master/README.md , https://e2eml.school/convolution_one_d.html

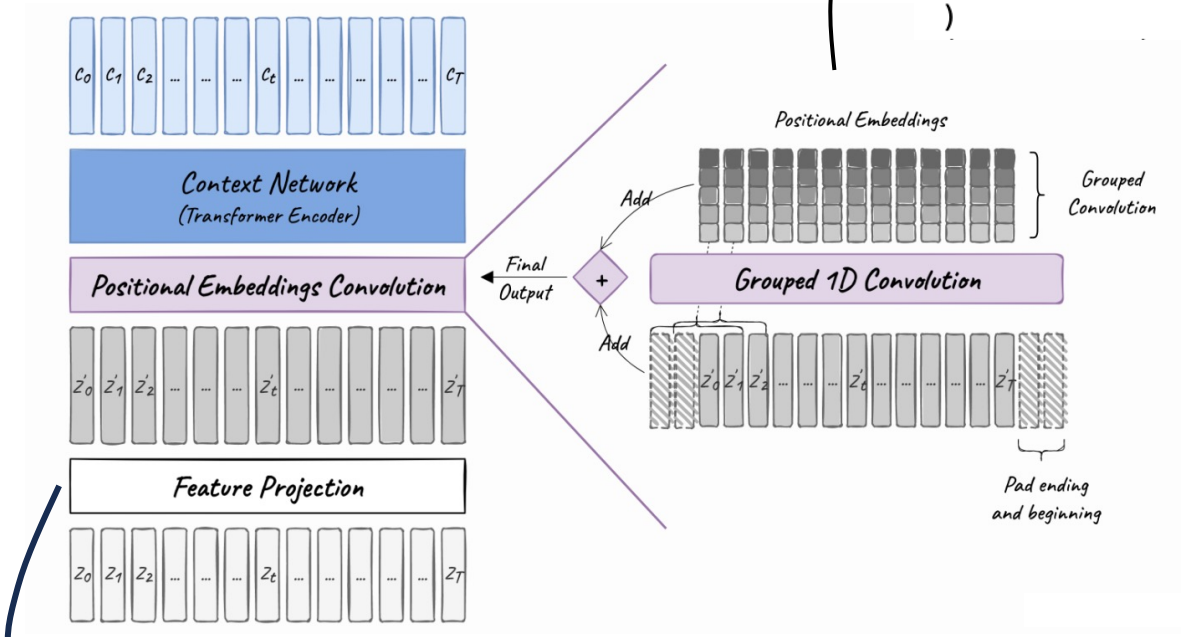
Feature Encoder




```
(feature_extractor): Wav2Vec2FeatureEncoder(
  (conv_layers): ModuleList(
    (0): Wav2Vec2GroupNormConvLayer(
      (conv): Conv1d(1, 512, kernel_size=(10,), stride=(5,), bias=False)
      (activation): GELUActivation()
      (layer_norm): GroupNorm(512, 512, eps=1e-05, affine=True)
    )
    (1-4): 4 x Wav2Vec2NoLayerNormConvLayer(
      (conv): Conv1d(512, 512, kernel_size=(3,), stride=(2,), bias=False)
      (activation): GELUActivation()
    )
    (5-6): 2 x Wav2Vec2NoLayerNormConvLayer(
      (conv): Conv1d(512, 512, kernel_size=(2,), stride=(2,), bias=False)
      (activation): GELUActivation()
    )
  )
)
```

- Has a receptive field of 400 samples or 25 ms of audio (audio data is encoded at a sample rate of 16 kHz)

Context Network




```
(pos_conv_embed): Wav2Vec2PositionalConvEmbedding(
  (conv): Conv1d(1024, 1024, kernel_size=(128,), stride=(1,), padding=(64,), groups=16)
  (padding): Wav2Vec2SamePadLayer()
  (activation): GELUActivation()
)
```

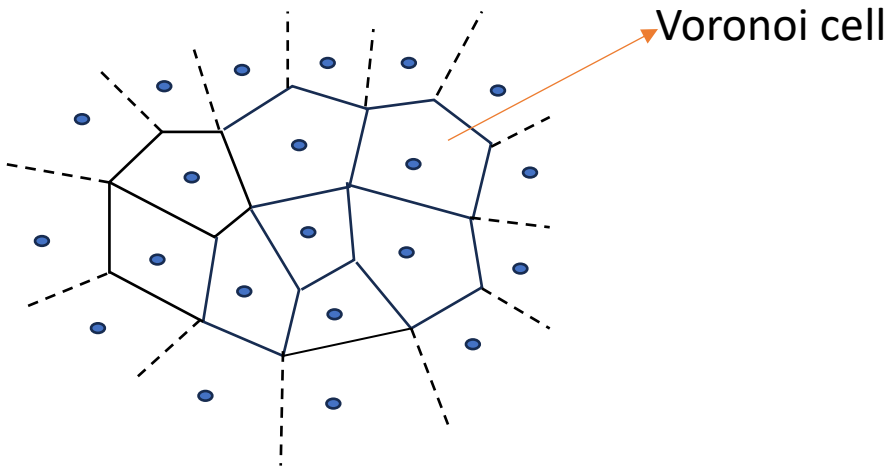
Context Network

```
(0-23): 24 x Wav2Vec2EncoderLayer(
  (attention): Wav2Vec2Attention(
    (k_proj): Linear(in_features=1024, out_features=1024, bias=True)
    (v_proj): Linear(in_features=1024, out_features=1024, bias=True)
    (q_proj): Linear(in_features=1024, out_features=1024, bias=True)
    (out_proj): Linear(in_features=1024, out_features=1024, bias=True)
  )
  (dropout): Dropout(p=0.1, inplace=False)
  (layer_norm): LayerNorm((1024,), eps=1e-05, elementwise_affine=True)
  (feed_forward): Wav2Vec2FeedForward(
    (intermediate_dropout): Dropout(p=0.1, inplace=False)
    (intermediate_dense): Linear(in_features=1024, out_features=4096, bias=True)
    (intermediate_act_fn): GELUActivation()
    (output_dense): Linear(in_features=4096, out_features=1024, bias=True)
    (output_dropout): Dropout(p=0.1, inplace=False)
  )
  (final_layer_norm): LayerNorm((1024,), eps=1e-05, elementwise_affine=True)
)
```

```
(feature_projection): Wav2Vec2FeatureProjection(
  (layer_norm): LayerNorm((512,), eps=1e-05, elementwise_affine=True)
  (projection): Linear(in_features=512, out_features=1024, bias=True)
  (dropout): Dropout(p=0.0, inplace=False)
)
```

Quantization Module

- Learns a set of speech units
- Chooses a speech unit for the latent audio representation from an inventory of learned units
- Discretize the output of the feature encoder **z** to a finite set of speech representations via product quantization*
- Uses 2 Codebooks each with 320 entries



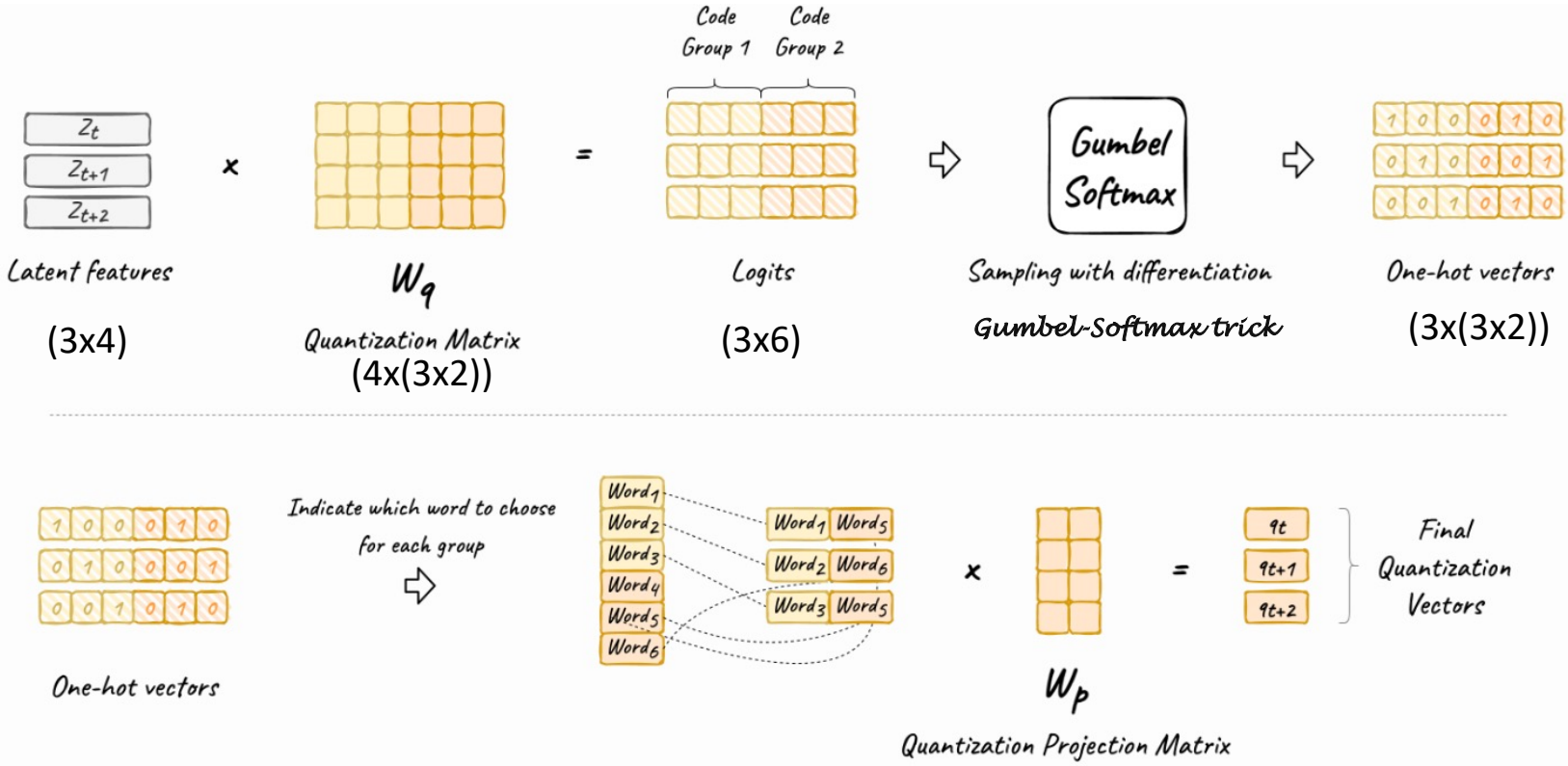
A quantizer is a function q mapping a D-dimensional vector $x \in \mathbb{R}^D$ to a vector $q(x) \in \mathcal{C} = \{c_i; i \in \mathcal{I}\}$, where the index set is $\mathcal{I} = 0, \ldots, v-1$. The reproduction values c_i are called centroids. The set of reproduction values $\mathcal{C}$ is the codebook of size v.

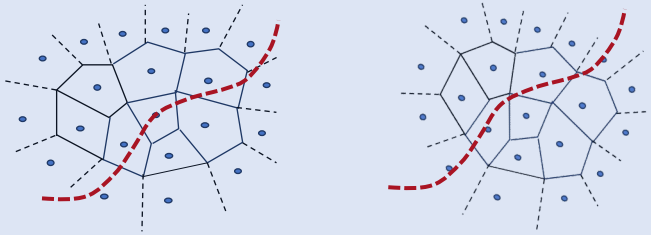
The v cells of a quantizer form a partition of $\mathbb{R}^D$.

*Product Quantization for Nearest Neighbor Search, Jegou et.al., 2011
Categorical Reparameterization with Gumbel-softmax, Jang et.al., 2016
https://sassafras13.github.io/GumbelSoftmax/

Quantization Process Example

Dim $\boldsymbol{D}$=4, Codebooks $\boldsymbol{g}$=2, Codewords $\boldsymbol{v} = \mathbf{3}$, No. of entries=$\boldsymbol{v}^{\boldsymbol{g}}$



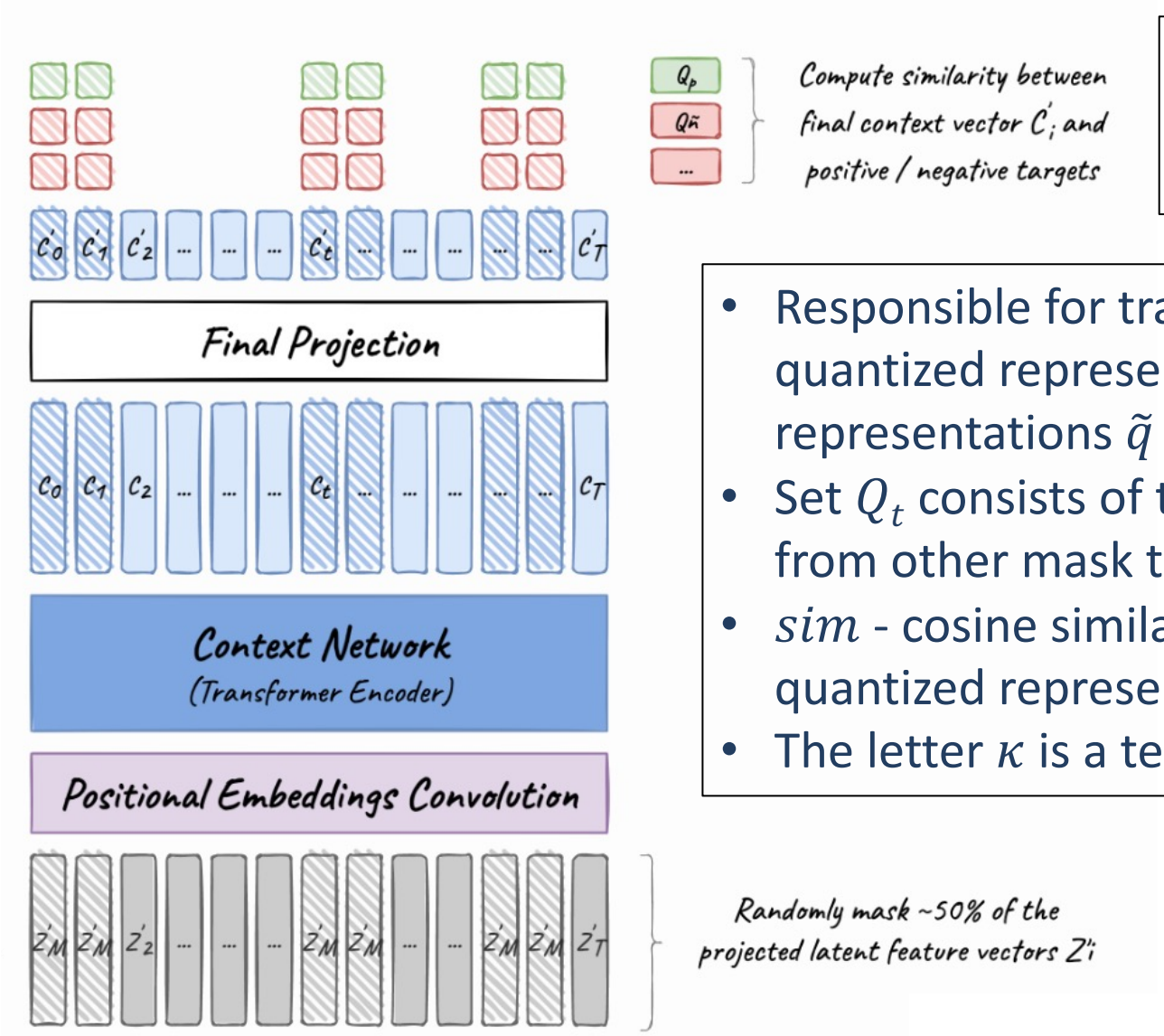


$$p_{g,v} = \frac{\exp(l_{g,v} + n_v)/\tau}{\sum_{k=1}^{V} \exp(l_{g,k} + n_k)/\tau}$$

- $l \in \mathbb{R}^{GxV}$ – logits calculated from z,
- $n_k = -log(-log(u_k))$,
- u_k is sampled from the uniform distribution $U(0, 1)$,
- τ – temperature.

- The Gumbel-Softmax trick allows sampling a single codeword from each codebook, after converting these logits into probabilities $p_{g,v}$
- During the forward pass, codeword i is chosen by $i = \ argmax_j p_{g,j}$ and in the backward pass, the true gradient of the Gumbel softmax outputs is used. Here, $p_{g,j}$ is the probability for choosing the j-th codebook entry for group g.

Masking and Contrastive Loss



$$\mathcal{L}_m = -\log \frac{\exp(sim(\mathbf{c}_t, \mathbf{q}_t)/\kappa)}{\sum_{\tilde{\mathbf{q}} \sim \mathbf{Q}_t} \exp(sim(\mathbf{c}_t, \tilde{\mathbf{q}})/\kappa)}$$

- Responsible for training the model to predict the correct quantized representation q_t among $K + 1$ quantized candidate representations $\tilde{q} \in Q_t$.
- Set Q_t consists of target q_t and *K* distractors sampled uniformly from other mask time steps.
- sim - cosine similarities between context representation c_t and quantized representations q_t
- The letter κ is a temperature which is constant during training.

Loss Functions

- Contrastive Loss

$$\mathcal{L}_m = -\log \frac{\exp(sim(\mathbf{c}_t, \mathbf{q}_t)/\kappa)}{\sum_{\tilde{\mathbf{q}} \sim \mathbf{Q}_t} \exp(sim(\mathbf{c}_t, \tilde{\mathbf{q}})/\kappa)}$$

- Pre-train Datasets - unlabeled data
  - Librispeech corpus without transcriptions containing 960 hours of audio
  - LibriVox preprocessed for 53.2k hours of audio
  - Trained on 128 V100 GPUs

- Diversity Loss

$$\mathcal{L}_d = \frac{1}{GV} \sum_{g=1}^{G} -H(\bar{p}_g) = \frac{1}{GV} \sum_{g=1}^{G} \sum_{v=1}^{V} \bar{p}_{g,v} \log \bar{p}_{g,v}$$

- Entropy assumes the maximum value when the data distribution is uniform
- Maximization entropy will encourage the model to take advantage of all code words
- The maximization equals minimization of negative entropy which is the diversity loss.

- Total Loss

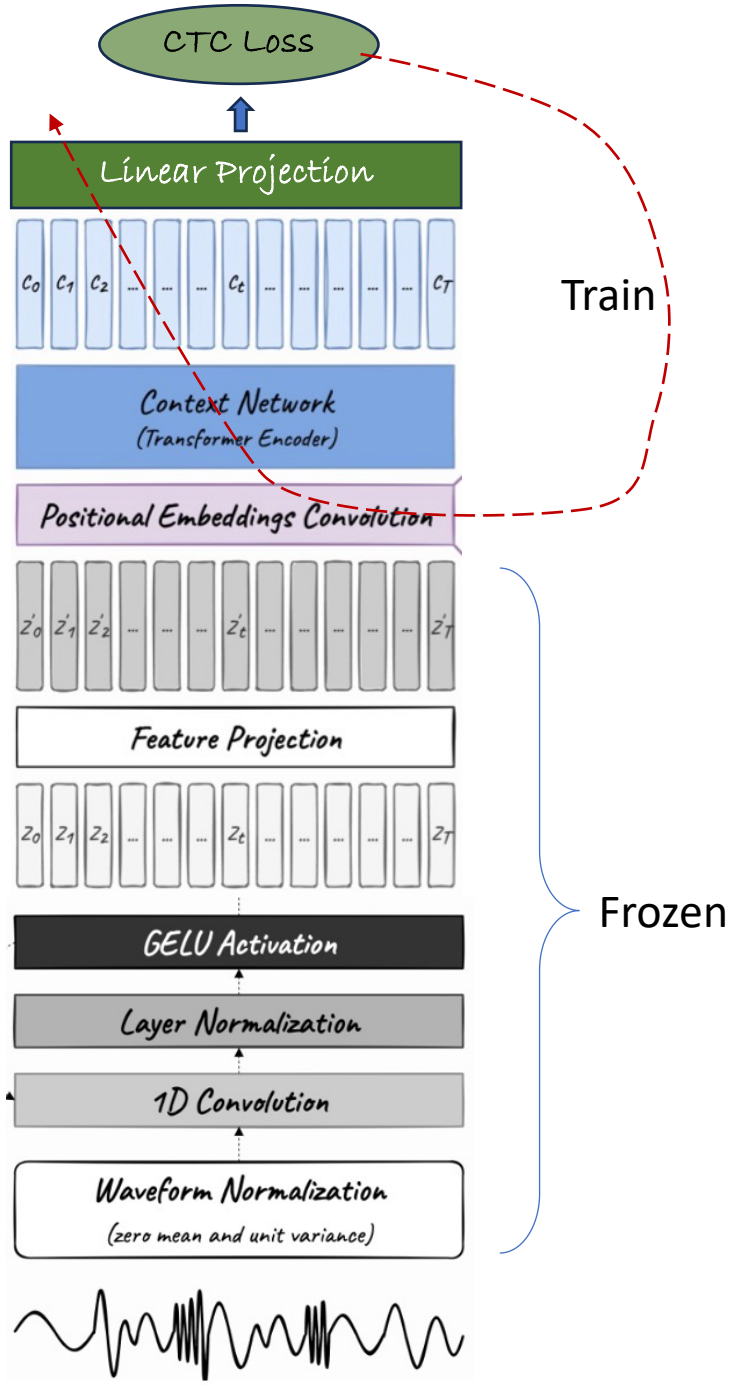
$$\mathcal{L} = \mathcal{L}_m + \alpha \mathcal{L}_d$$

α is a tuned hyperparameter.

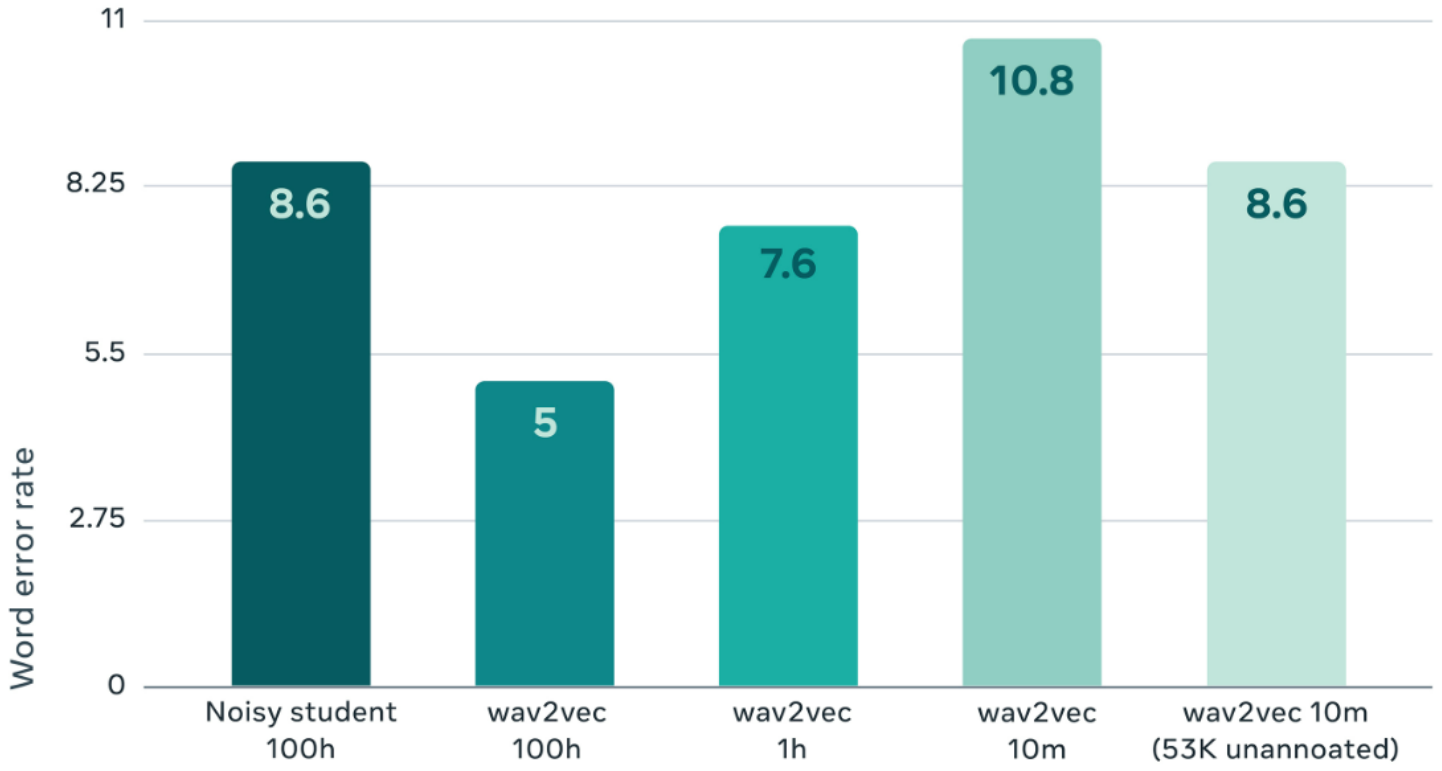
https://neurosys.com/blog/wav2vec-2-0-framework

Fine Tuning

- Add a randomly initialized linear projection on top of the context network into C classes representing the vocabulary of the task
- The feature encoder is not trained during fine-tuning.
- Quantization is not used
- Models are optimized by minimizing a CTC loss
- A modified version of *SpecAugment* is used to augment data



Key Results



- Using only 10 minutes of labeled training data, or 48 recordings of 12.5 seconds on average, a WER of 10.8 on test-other of Librispeech was reported
- On the clean 100 hour Librispeech setup, wav2vec 2.0 outperforms the previous best result while using 100 times less labeled data
- Demonstrates the feasibility of speech recognition with limited amounts of labeled data

https://ai.meta.com/blog/wav2vec-20-learning-the-structure-of-speech-from-raw-audio/

Thanks