ASR

Automatic Speech Recognition Part 3

ASR

- Wav2Vec2.0
- Multi model systems

Recap

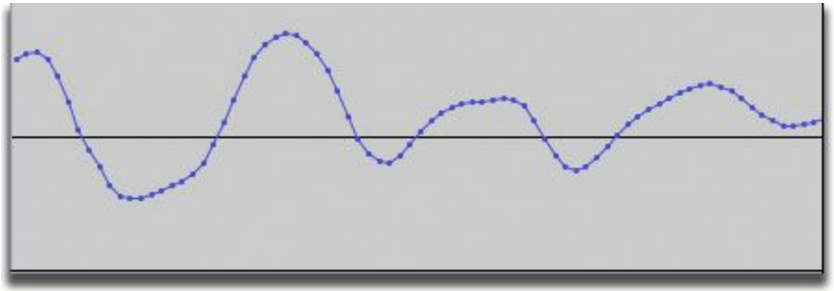
- Autoregressive Encoder-Decoder does not need actual alignment, alignment is created internally
- AED models are **autoregressive,** in order to make prediction the output X_{n} model needs output X_{n-1}.
- AED Outputs are not conditionally independent
- Needs full input sequence in order to make prediction
- RNN-T introduces audio-module and text-module
- Audio module encodes spectogram, text module encodes predicted text
- Aggregate prediction using joint network
- RNN-T has simpler rules than CTC
- RNN-T is autoregressive model
- RNN-T is better for streaming

Unsupervised learning

- NLP tasks successfully utilize raw data
- GPT-3-like language models, BERT language models are used as generative models or sequence embedders for downstream tasks
- Can we apply same logic for ASR?

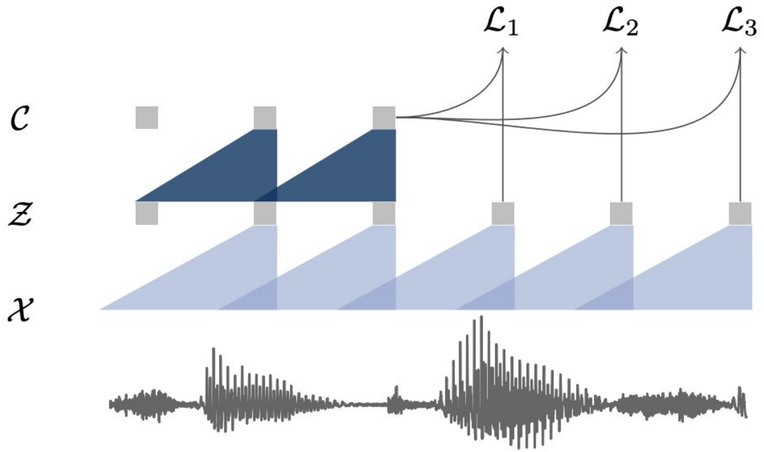
Unsupervised learning

- NLP language models have a task of predicting next token in sequence or predicting mask part of the text
- Tokens are mostly chars, BPEs or words
- All of them are from vocabulary size of W
- Predicting next token in waveform sequence is harder
- Usually 1 second contain 16000 tokens

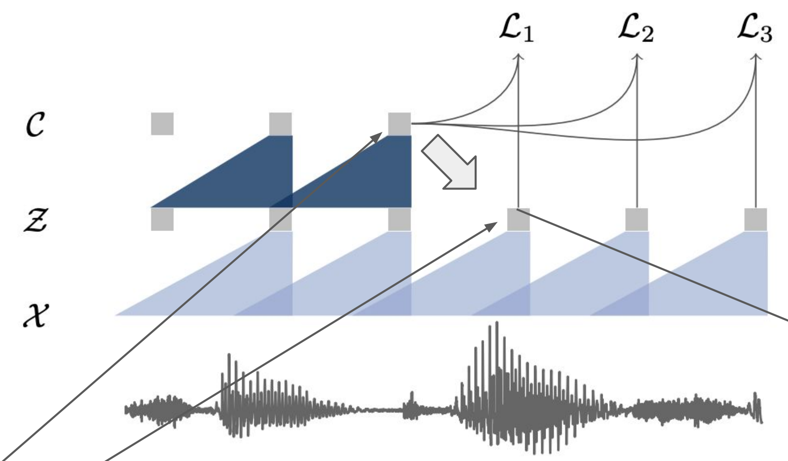


Contrastive Predictive Coding

- f: X -> Z – encoder network
- Use CNN:
  (10, 8, 4, 4, 4) kernels,
  (5, 4, 2, 2, 2) strides
- 30 ms encoding size, 10 ms stride
- g: Z-> C – context network
- Use CNN:
  9 layers, kernel = 3, stride=2
- receptive field ~210 ms



Contrastive Predictive Coding

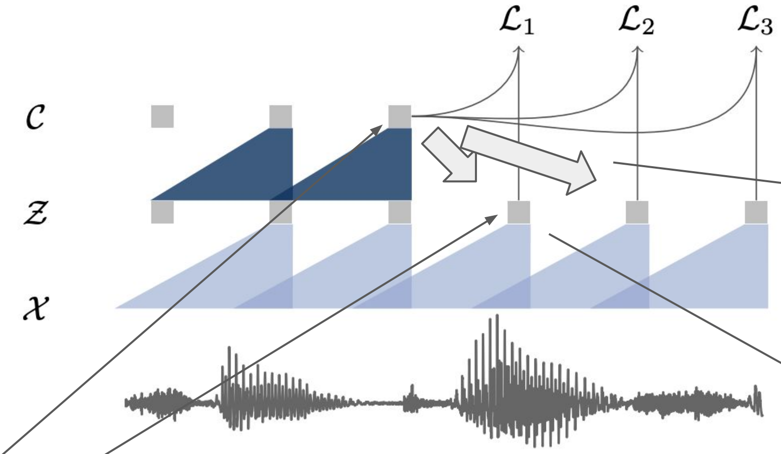


We want to c_i be able to predict z_{i+k}

$\mathbf{z}_{i+k}$ representation k steps into the future

$h_k(\mathbf{c}_i) = W_k \mathbf{c}_i + \mathbf{b}_k$ linear transformation of c_{i}

Contrastive Predictive Coding



And z_{i + k + 1}
All the way to i+K

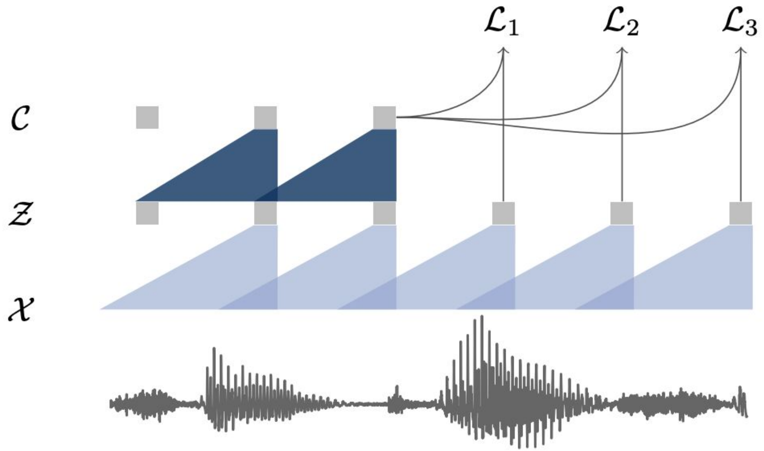
We want to c_i be
able to predict z_{i+k}

$\mathbf{z}_{i+k}$ representation k steps into the future

$h_k(\mathbf{c}_i) = W_k \mathbf{c}_i + \mathbf{b}_k$ linear transformation of c_{i}

Contrastive Loss

- Train to distinguish between Z_{i+k} and **distractors** from p_{n} distribution
- K - constant, maximum distance



Approximate expectation by sampling negative examples length of T from audio encoder

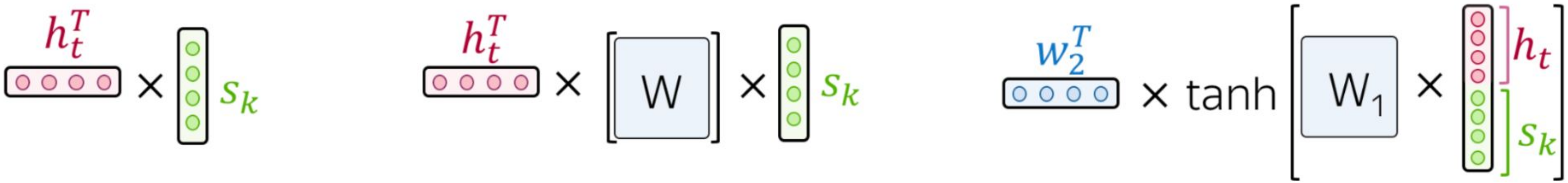
$$h_k(\mathbf{c}_i) = W_k\mathbf{c}_i + \mathbf{b}_k$$

$$\mathcal{L}_k = -\sum_{i=1}^{T-k}\left(\log\sigma(\mathbf{z}_{i+k}^\top h_k(\mathbf{c}_i)) + \lambda \mathop{\mathbb{E}}_{\tilde{\mathbf{z}}\sim p_n}\left[\log\sigma(-\tilde{\mathbf{z}}^\top h_k(\mathbf{c}_i))\right]\right)$$

output after decoder should be close with the true sample

Contrastive Loss

Dot-product | Bilinear | Multi-Layer Perceptron



$\text{score}(h_t, s_k) = h_t^T\, s_k$

$\text{score}(h_t, s_k) = h_t^T\, W s_k$

$\text{score}(h_t, s_k) = w_2^T \cdot \tanh(W_1[h_t, s_k])$

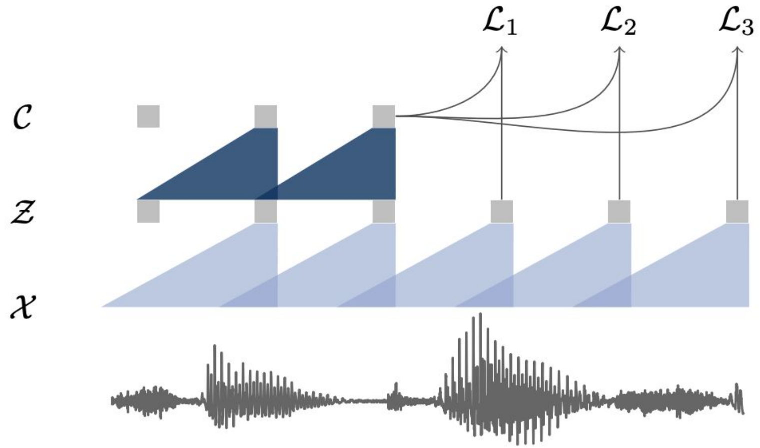
$$h_k(\mathbf{c}_i) = W_k\mathbf{c}_i + \mathbf{b}_k$$

$$\mathcal{L}_k = -\sum_{i=1}^{T-k}\left(\log\sigma(\mathbf{z}_{i+k}^\top h_k(\mathbf{c}_i)) + \lambda \mathop{\mathbb{E}}_{\tilde{\mathbf{z}}\sim p_n}\left[\log\sigma(-\tilde{\mathbf{z}}^\top h_k(\mathbf{c}_i))\right]\right)$$

output after decoder should be close with
the true sample

Contrastive Loss

- Train to distinguish between Z_{i+k} and **distractors** from p_{n} distribution
- T - constant, maximum distance
- In practice, we approximate the expectation by sampling ten negatives examples by uniformly choosing distractors from each audio sequence



$$\mathcal{L}_k = -\sum_{i=1}^{T-k} \left(\log \sigma(\mathbf{z}_{i+k}^\top h_k(\mathbf{c}_i)) + \lambda \mathop{\mathbb{E}}_{\tilde{\mathbf{z}} \sim p_n} \left[\log \sigma(-\tilde{\mathbf{z}}^\top h_k(\mathbf{c}_i)) \right] \right)$$

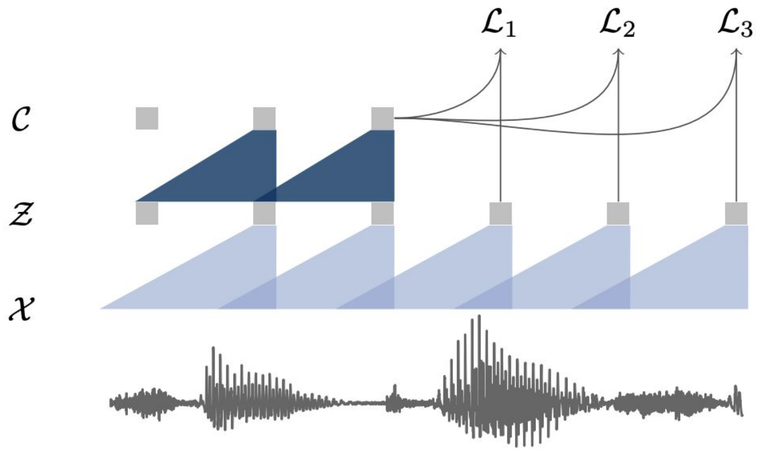
$$\mathcal{L} = \sum_{k=1}^{K} \mathcal{L}_k$$

be far away from distractor

Probability of z_{i+k} being true sample
== be close to true sample

Wav2Vec

- [Link to Wav2Vec paper]()
- Use self-supervised pretraining using CPC
- Pretrain on librispeech using no transcriptions
- Finetune on librispeech using CTC loss



Wav2Vec

Not good enough?

			nov93dev LER	nov93dev WER	nov92 LER	nov92 WER
Deep Speech 2 (12K h labeled speech; Amodei et al., 2016)			-	4.42	-	3.1
Trainable frontend (Zeghidour et al., 2018a)			-	6.8	-	3.5
Lattice-free MMI (Hadian et al., 2018)			-	5.66†	-	2.8†
Supervised transfer-learning (Ghahremani et al., 2017)			-	4.99†	-	2.53†
4-GRAM LM (Heafield et al., 2013)						
Baseline	–	–	3.32	8.57	2.19	5.64
wav2vec	Librispeech	80 h	3.71	9.11	2.17	5.55
wav2vec	Librispeech	960 h	2.85	7.40	1.76	4.57
wav2vec	Libri + WSJ	1,041 h	2.91	7.59	1.67	4.61
wav2vec large	Librispeech	960 h	2.73	6.96	1.57	4.32
WORD CONVLM (Zeghidour et al., 2018b)						
Baseline	–	–	2.57	6.27	1.51	3.60
wav2vec	Librispeech	960 h	2.22	5.39	1.25	2.87
wav2vec large	Librispeech	960 h	2.13	5.16	1.02	2.53
CHAR CONVLM (Likhomanenko et al., 2019)						
Baseline	–	–	2.77	6.67	1.53	3.46
wav2vec	Librispeech	960 h	2.14	5.31	1.15	2.78
wav2vec large	Librispeech	960 h	2.11	5.10	0.99	2.43

Table 1: Replacing log-mel filterbanks (Baseline) by pre-trained embeddings improves WSJ performance on test (nov92) and validation (nov93dev) in terms of both LER and WER. We evaluate pre-training on the acoustic data of part of clean and full Librispeech as well as the combination of all of them. † indicates results with phoneme-based models.

31	**wav2vec_wav2letter**	2.7	✓	Self-training and Pre-training are Complementary for Speech Recognition	2020

Recap

- Make use of unannotated data through unsupervised pretraining
- Use Contrastive Loss
- Contrastive Loss compares representation projected through context network against true one and distractors
- Get distractors from the batch
- Finetune on supervised data

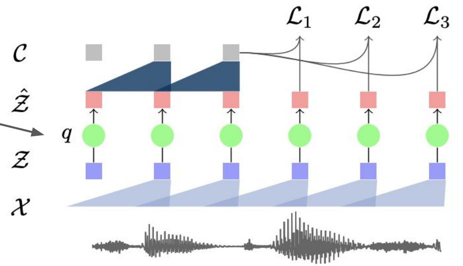
vq-Wav2Vec

Motivation:

- Bert-Like architectures using transformer and masking are quite good
- Cannot feed Z_{i} to bert - they are different every step, we need to quantize them

vq-Wav2Vec:

- Quantize Z output
- Train it using Gumbel-Softmax
  or online k-means clustering (both differentiable)



(a) vq-wav2vec

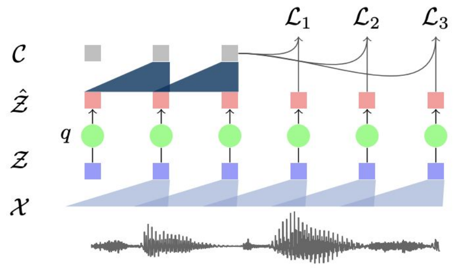
vq-Wav2Vec

Motivation:

- Bert-Like architectures using transformer and masking are quite good
- Cannot feed Z_{i} to bert - they are different every step, we need to quantize them

vq-Wav2Vec:

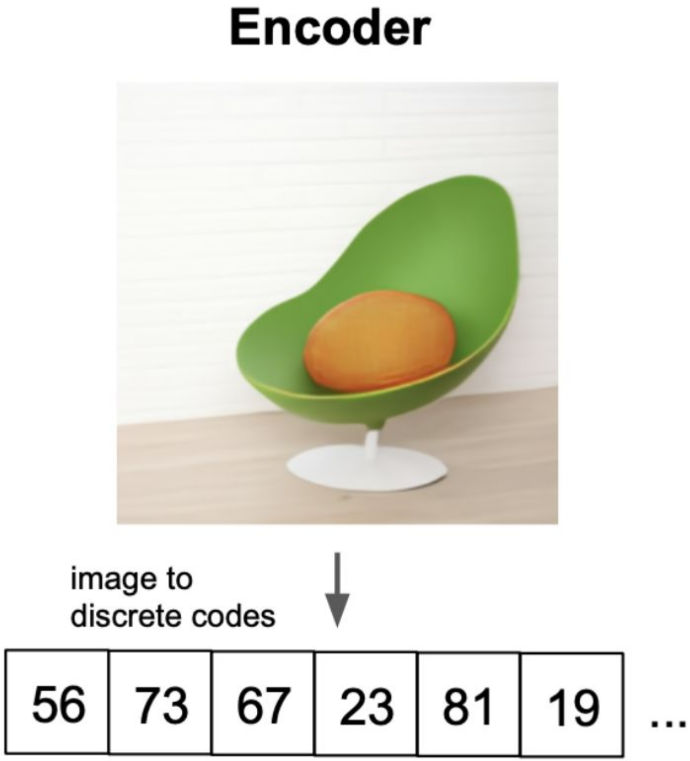
- Quantize Z output
- Train it using Gumbel-Softmax
  or online k-means clustering (both differentiable)



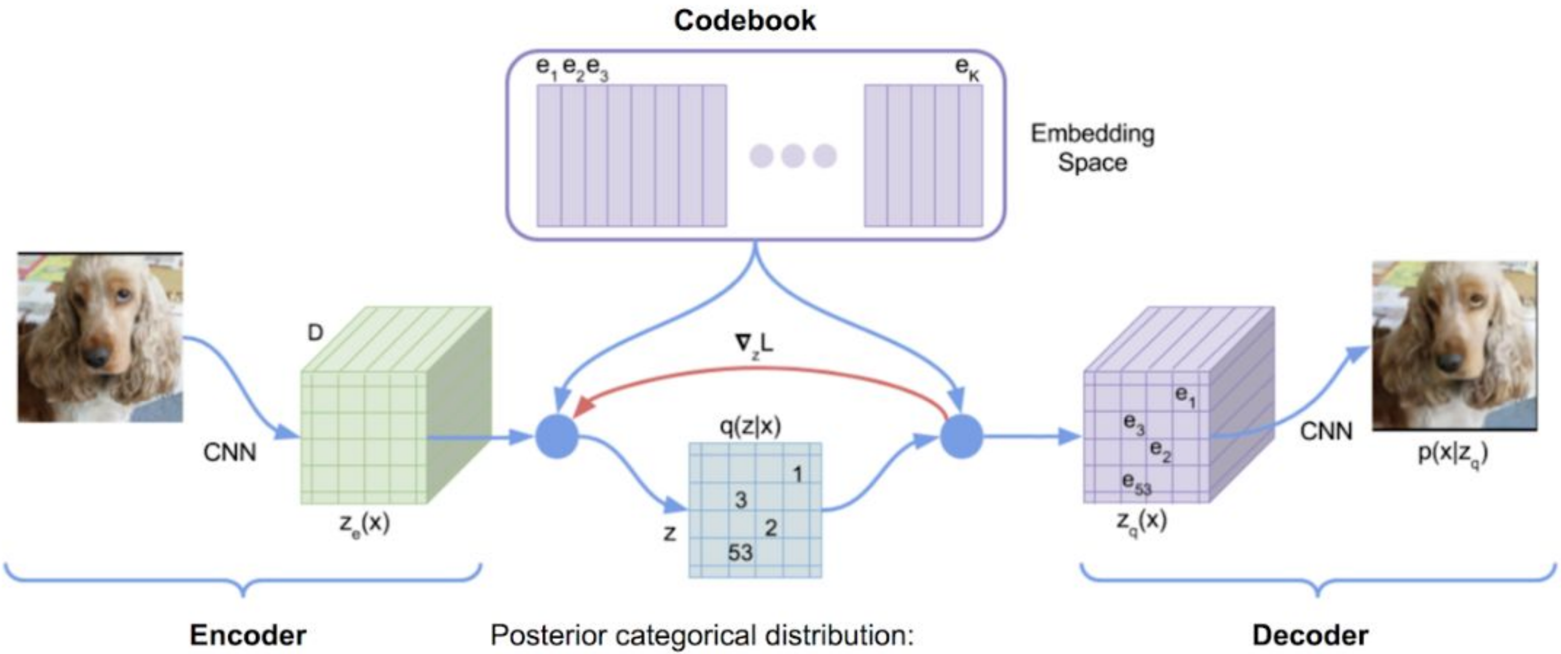
(a) vq-wav2vec

Quantization

- Goal is to replace original representation **z** with $\hat{\mathbf{z}} = \mathbf{e}_i$ from a fixed size codebook $\mathbf{e} \in \mathbb{R}^{V \times d}$
- $\mathbf{e} \in \mathbb{R}^{V \times d}$ contains V representation of size d



Quantization



Posterior categorical distribution:

$$q(\mathbf{z} = \mathbf{e}_k|\mathbf{x}) = \begin{cases} 1 & \text{if } k = \arg\min_i \|\mathbf{z}_e(\mathbf{x}) - \mathbf{e}_i\|_2 \\ 0 & \text{otherwise.} \end{cases}$$

Quantization

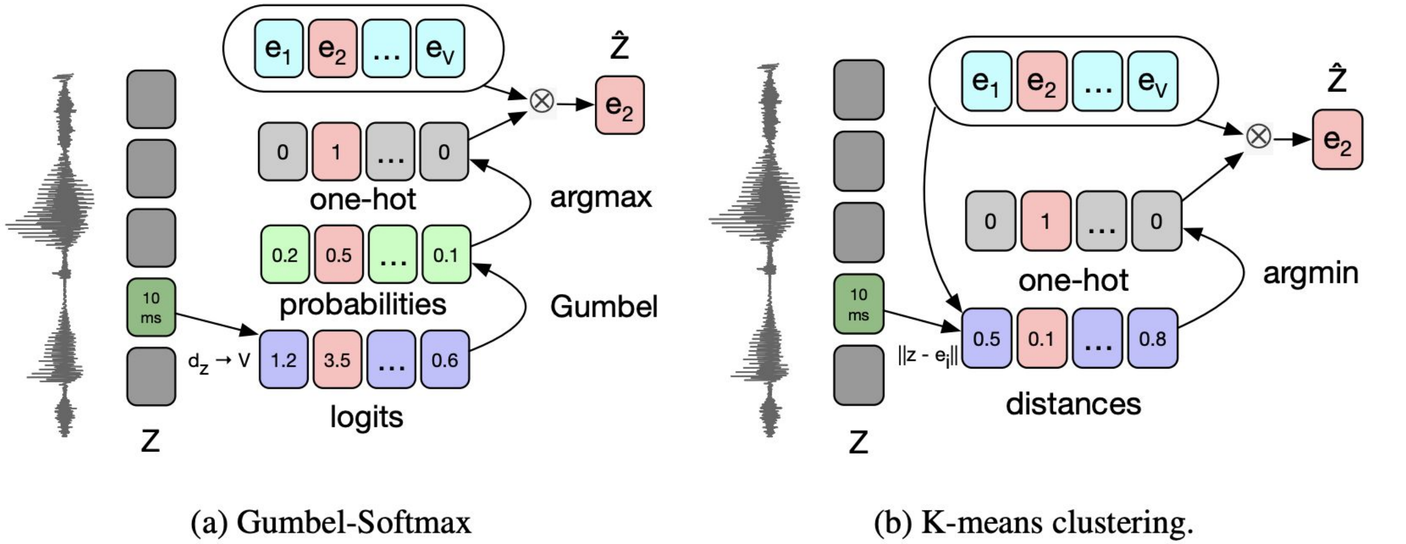
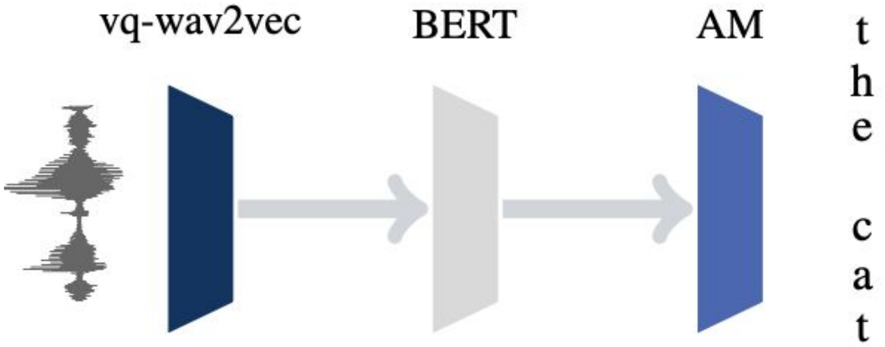


Figure 2: (a) The Gumbel-Softmax quantization computes logits representing the codebook vectors (**e**). In the forward pass the argmax codeword ($\mathbf{e}_2$) is chosen and for backward (not shown) the exact probabilities are used. (b) K-means vector quantization computes the distance to all codeword vector and chooses the closest (argmin).

vq-Wav2Vec Inference

- Once training is over, use quantized outputs to train BERT
- use BERT embeddings as inputs to acoustic model



(b) Discretized speech training pipeline

vq-Wav2Vec

	nov93dev		nov92	
	LER	WER	LER	WER
Deep Speech 2 (12K h labeled speech; Amodei et al., 2016)	-	4.42	-	3.1
Trainable frontend (Zeghidour et al., 2018)	-	6.8	-	3.5
Lattice-free MMI (Hadian et al., 2018)	-	5.66†	-	2.8†
Supervised transfer-learning (Ghahremani et al., 2017)	-	4.99†	-	2.53†
NO LM				
Baseline (log-mel)	6.28	19.46	4.14	13.93
wav2vec (Schneider et al., 2019)	5.07	16.24	3.26	11.20
vq-wav2vec Gumbel	7.04	20.44	4.51	14.67
+ BERT base	**4.13**	**13.40**	**2.62**	**9.39**
4-GRAM LM (Heafield et al., 2013)				
Baseline (log-mel)	3.32	8.57	2.19	5.64
wav2vec (Schneider et al., 2019)	2.73	6.96	1.57	4.32
vq-wav2vec Gumbel	3.93	9.55	2.40	6.10
+ BERT base	**2.41**	**6.28**	**1.26**	**3.62**
CHAR CONVLM (Likhomanenko et al., 2019)				
Baseline (log-mel)	2.77	6.67	1.53	3.46
wav2vec (Schneider et al., 2019)	2.11	5.10	0.99	2.43
vq-wav2vec Gumbel + BERT base	**1.79**	**4.46**	**0.93**	**2.34**

Table 1: WSJ accuracy of vq-wav2vec on the development (nov93dev) and test set (nov92) in terms of letter error rate (LER) and word error rate (WER) without language modeling (No LM), a 4-gram LM and a character convolutional LM. vq-wav2vec with BERT pre-training improves over the best wav2vec model (Schneider et al., 2019).

Gumbel Softmax vs K-means

	nov93dev		nov92	
	LER	WER	LER	WER
No LM				
wav2vec (Schneider et al., 2019)	5.07	16.24	3.26	11.20
vq-wav2vec Gumbel	7.04	20.44	4.51	14.67
+ BERT small	4.52	14.14	2.81	9.69
vq-wav2vec k-means (39M codewords)	5.41	17.11	3.63	12.17
vq-wav2vec k-means	7.33	21.64	4.72	15.17
+ BERT small	4.31	13.87	2.70	9.62
4-gram LM (Heafield et al., 2013)				
wav2vec (Schneider et al., 2019)	2.73	6.96	1.57	4.32
vq-wav2vec Gumbel	3.93	9.55	2.40	6.10
+ BERT small	2.67	6.67	1.46	4.09
vq-wav2vec k-means (39M codewords)	3.05	7.74	1.71	4.82
vq-wav2vec k-means	4.37	10.26	2.28	5.71
+ BERT small	2.60	6.62	1.45	4.08

Table 2: Comparison of Gumbel-Softmax and k-means vector quantization on WSJ (cf. Table 1).

Recap

- vq-Wav2Vec produces quantized representations
- Use BERT after vq-wav2vec to get best quality
- Poor quality with no BERT

Wav2Vec2.0

Motivation:

- vq-wav2vec, bert, am training is too much training
- Need end2end model
- Poor performance without BERT

Wav2Vec2.0:

- End2end training
- Encoder network uses GELU + layernorm
- Context network is transformer
- Use product quantization + GumbelSoftmax
- Use quantization for contrastive loss directly

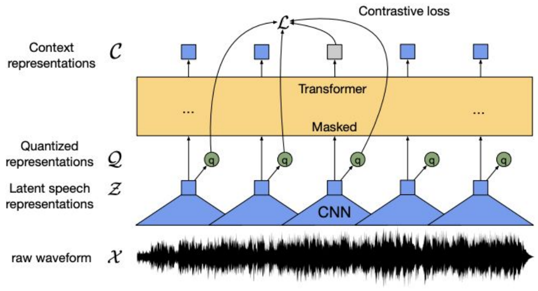


Figure 1: Illustration of our framework which jointly learns contextualized speech representations and an inventory of discretized speech units.

Wav2Vec2.0

Contrastive task

Quantization diversity

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

Identify **quantized** state, not Z state as in wav2vec

Distractors

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

Wav2Vec2.0

Quantization diversity

- Encourage the equal use of the V entries in each of the G codebooks by maximizing the entropy of the averaged softmax distribution over the codebook entries for each 3 codebook across a batch of utterances

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

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

Wav2Vec2.0

Not bad

Table 1: WER on the Librispeech dev/test sets when training on the Libri-light low-resource labeled data setups of 10 min, 1 hour, 10 hours and the clean 100h subset of Librispeech. Models use either the audio of Librispeech (LS-960) or the larger LibriVox (LV-60k) as unlabeled data. We consider two model sizes: BASE (95m parameters) and LARGE (317m parameters). Prior work used 860 unlabeled hours (LS-860) but the total with labeled data is 960 hours and comparable to our setup.

Model	Unlabeled data	LM	dev		test	
			clean	other	clean	other
10 min labeled						
Discrete BERT [4]	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 [4]	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 [4]	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

7	**wav2vec 2.0 with Libri-Light**	1.8	✓	wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations		2020

Recap

- Wav2Vec2.0 uses end2end training with vq-wav2vec idea
- Use contrastive loss and diversity loss
- Diversity loss is used to increase the use of codebook’s representations
- Finetune on supervised data
- Get extremely good results only on 10 minutes of speech

What could be done better?

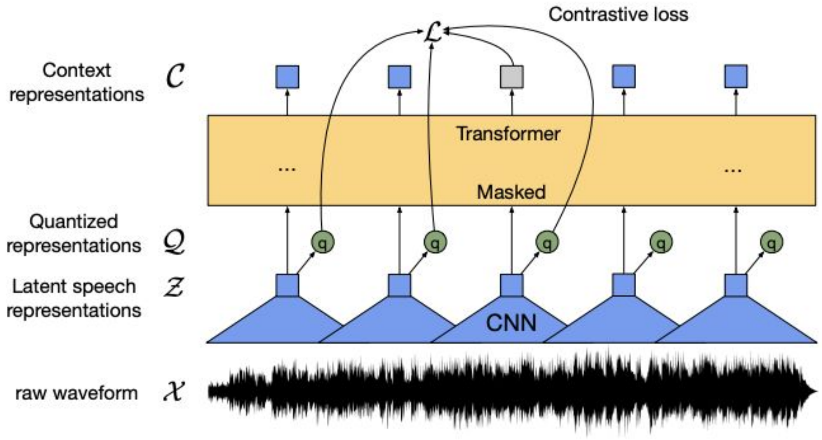


Figure 1: Illustration of our framework which jointly learns contextualized speech representations and an inventory of discretized speech units.

What could be done better?

1	**Conformer + Wav2vec 2.0 + SpecAugment-based Noisy Student Training with Libri-Light**	1.4	✓	Pushing the Limits of Semi-Supervised Learning for Automatic Speech Recognition			2020	Conformer
2	**w2v-BERT XXL**	1.4	✓	W2v-BERT: Combining Contrastive Learning and Masked Language Modeling for Self-Supervised Speech Pre-Training			2021	
3	**Conv + Transformer + wav2vec2.0 + pseudo labeling**	1.5	✓	Self-training and Pre-training are Complementary for Speech Recognition			2020	Transformer

Wav2Vec as pretraining technique

- Pretrain conformer
- Get rid of quantization

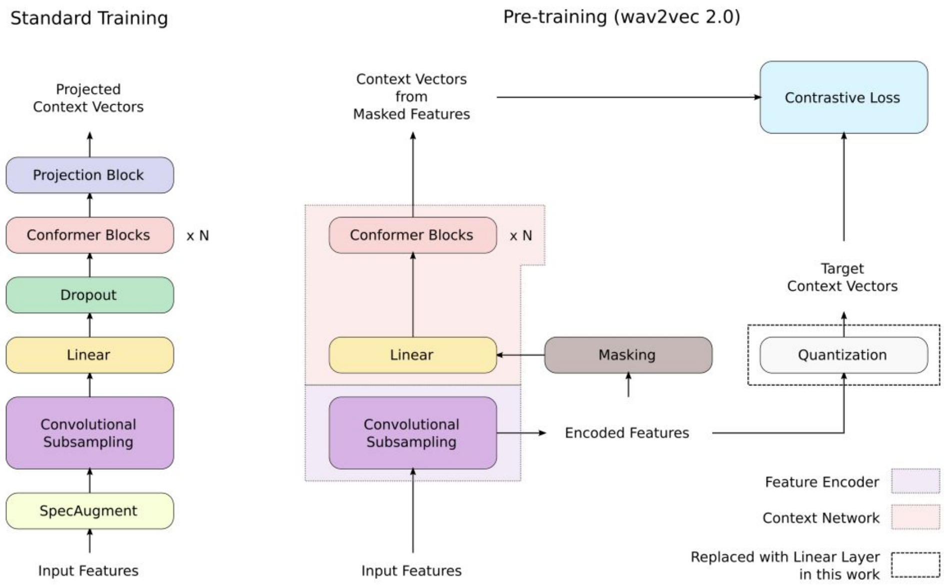
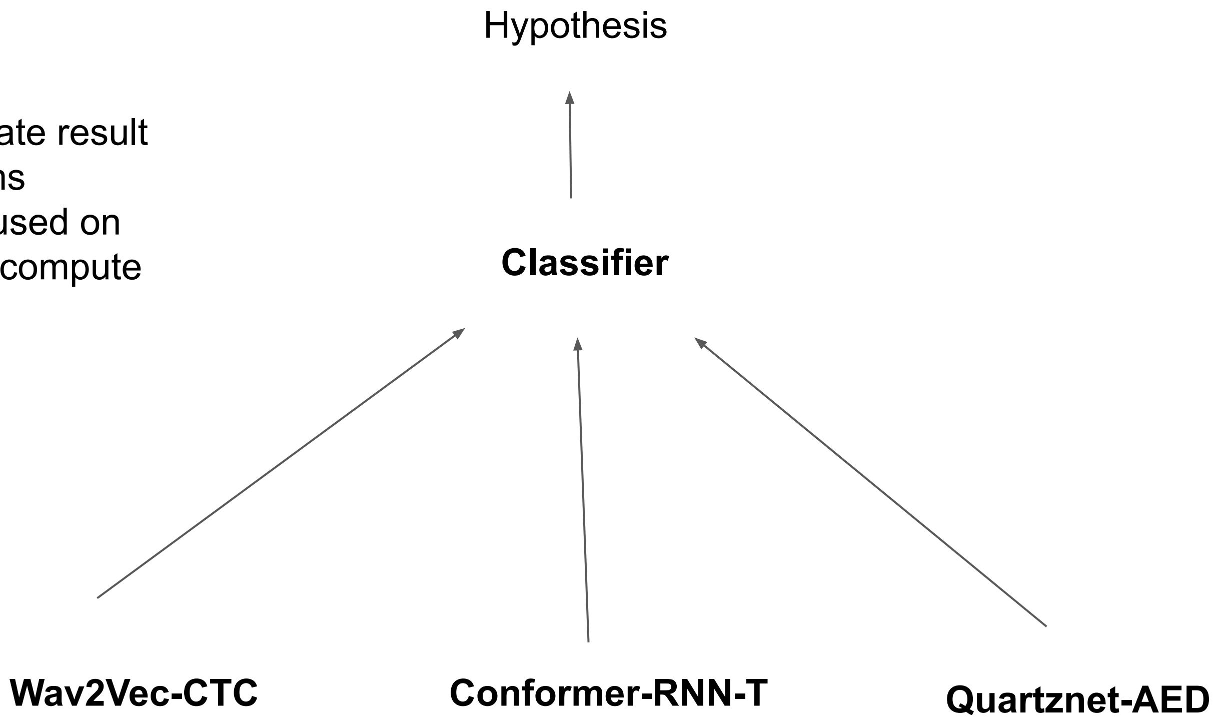


Figure 2: The Conformer encoder. In wav2vec 2.0 pre-training, the features generated by the convolutional sampling block are masked and passed into the rest of the network to yield context vectors, and also quantized to yield target context vectors. The contrastive loss between the context vectors obtained from masked features and the quantization unit is optimized. In our work, we replace the quantization layer with a linear layer. During fine-tuning, an additional projection block is added to produce features to be passed to the transducer.

Multi model systems

- Goal is to get the most accurate result possible using N ASR systems
- System combination can be used on backend If you have enough compute



Rover

- Simple yet effective algorithm to choose best transcript
- [Link to ROVER paper]()

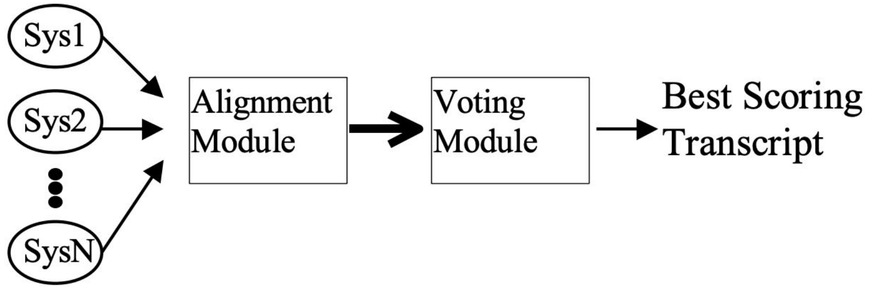


Figure 1 Rover System Architecture

Rover

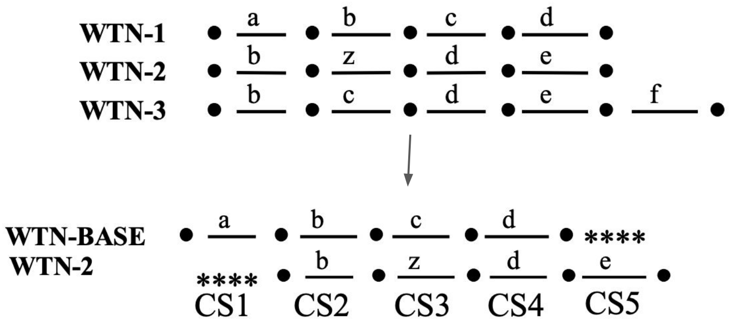


Figure 3 Aligned WTNs and correspondence set labels

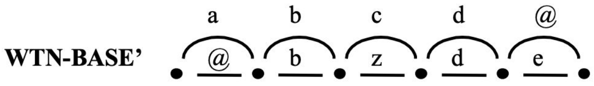


Figure 4 Composite WTN made from WTN-1 and WTN-2

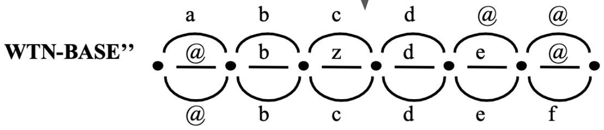


Figure 5 Final composite WTN

Rover

- Search WTN using scoring formula

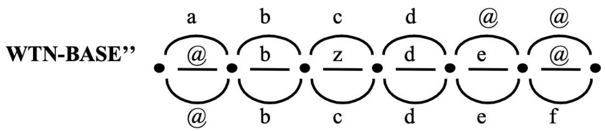


Figure 5 Final composite WTN

grid search it

$$Score(w) = \alpha(N(w,i)/Ns) + (1-\alpha)C(w,i)$$

Number of occurrences of word w

Confidence score of word

Number of combined systems

Rover

Ансамбли без ML, WERR

Датасет / Модель	QuartzNet Small (CPU)	QuartzNet Big (GPU)	Conformer-CTC	Conformer-LAS	Wav2Vec-1B-RNN-T	ROVER (без QuartzNet Small)
Callcenter #1	**+27%**	**0%**	**-19%**	**-36%**	**-20%**	**-40%**
Callcenter #2	**+17%**	**0%**	**-23%**	**-38%**	**-28%**	**-41%**
IVR	**+28%**	**0%**	**-5%**	**-31%**	**-30%**	**-44%**
Автоответчик	**+22%**	**0%**	**-25%**	**-35%**	**-28%**	**-39%**

*Word Error Rate Reduction – относительное изменение WER по сравнению с бейзлайном

SBER DEVICES

9-10 сентября 2022

Improved Rover

- Since ROVER algorithm is very easy There are many attempts to improve it
- Use LM information
- LM Rover

number of combined systems:	2	3	4	5
arbitrary ties:				
word error:	18.9%	14.3%	14.1%	14.1%
sentence error:	80.9%	74.1%	73.4%	72.9%
arbitrary ties + LM:				
word error:	15.2%	**13.6%**	13.8%	14.0%
rel. improvement:	-11.1%	**-20.5%**	-19.3%	-18.1%
sentence error:	75.8%	73.4%	72.5%	73.0%
rel. improvement:	-1.8%	-4.9%	-6.1%	-5.4%

Table 5: 1999 broadcast news test set word and sentence error rates when using LM information compared to breaking ties arbitrarily. The relative improvement is indicated with respect to the best single recognizer (17.1% werr, 77.2% serr).

Other techniques

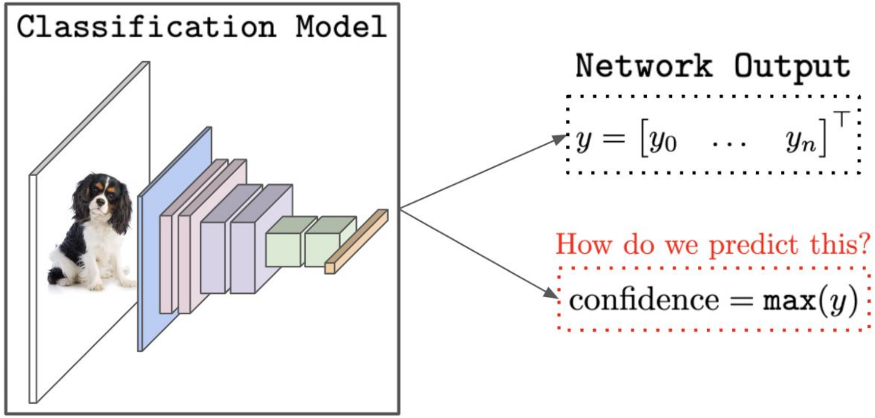
- More advanced lattice decoding techniques can be used
- Minimum Bayes Risk decoding is one of them
- [MBR post]

Table 5: *Method of combining hybrid, LAS, and RNN-T models*

Combination method	WER (%)
1-best of merged N-best	7.59
ROVER	7.33
MBR	6.89

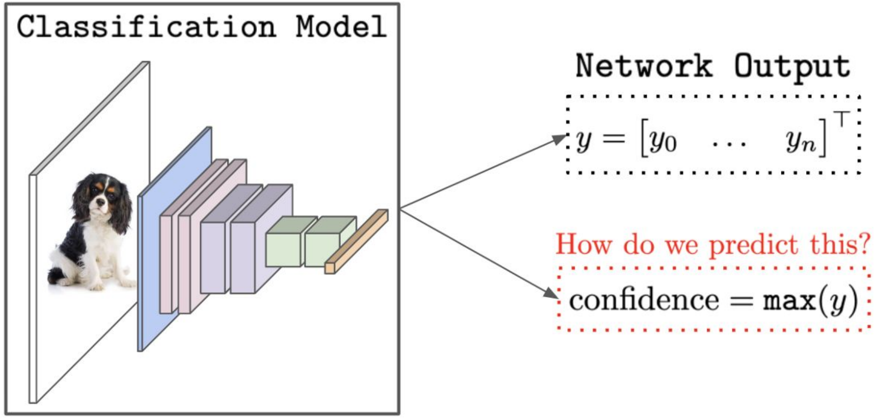
Confidence estimation

- Given hypothesis from ASR system
- Calculate confidence score from 0 to 1 approximating probability of error



Confidence estimation

- Logprobs of models
- Models disagreement
- External classifiers



Cascade models

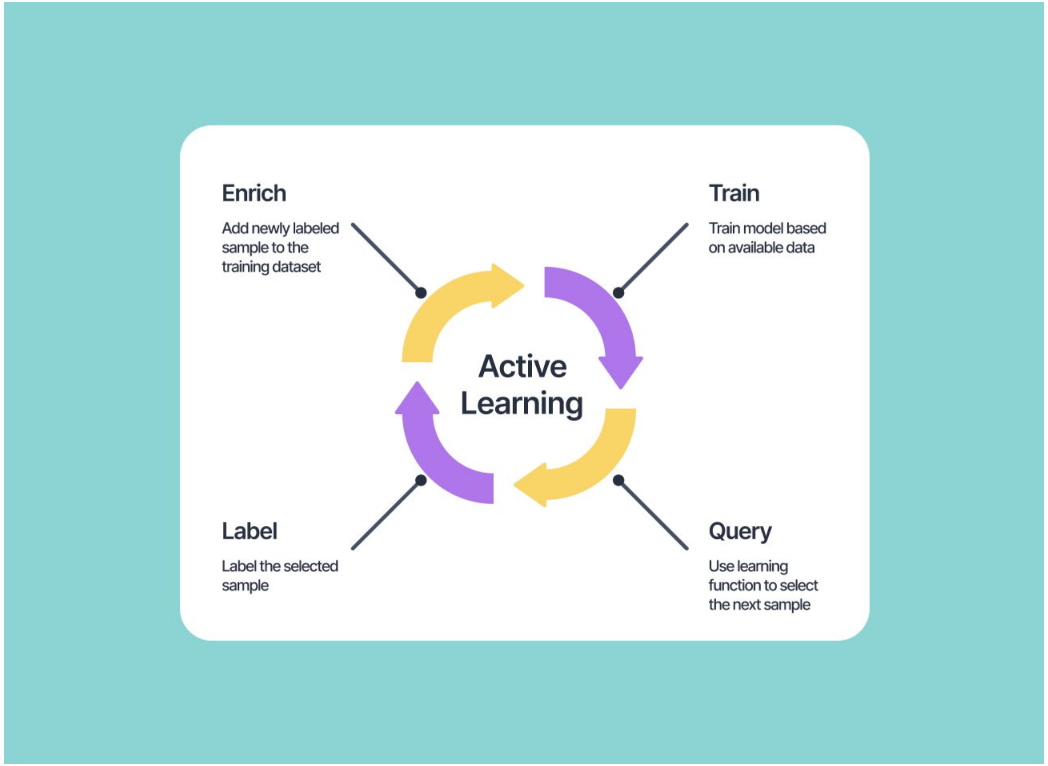
Small ASR
QuartzNet

Confidence Estimation

Big ASR rescoring
Conformer XXL

Active learning

- Get rich examples for annotations
- Filter 90% easy onces
- Use N models disagreement



Recap

- Use ROVER or MBR to combine hypotheses from multiple systems
- Those techniques perform better than one system
- Create confidence estimation models to get cascade systems
- Active learning relies on confidence estimation and leads to better performance