Algorithms for Speech and Natural Language Processing

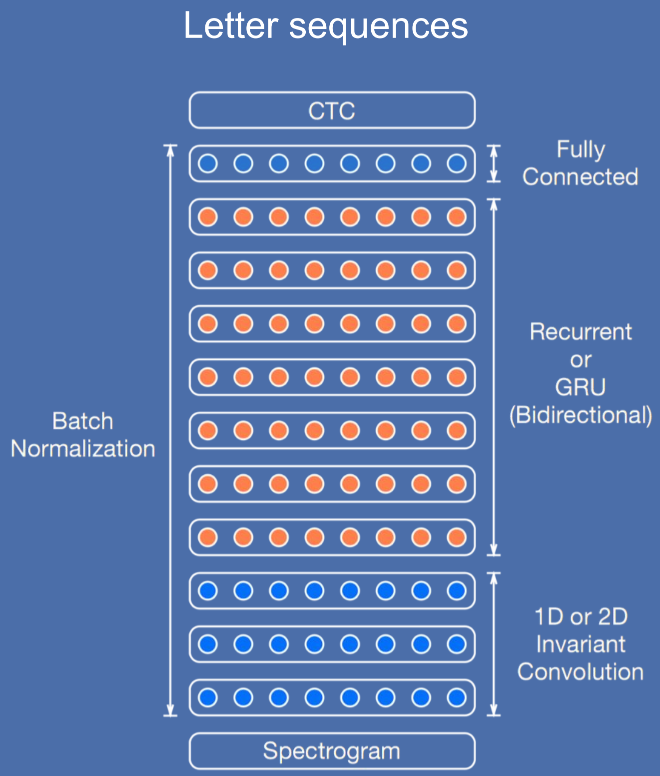
End-to-end speech recognition

Neil Zhegidour and Robin Algayres

CTC based ASR

- annotated speech: speech+unaligned transcription
- much simpler to set up than large vocabulary HMM
- much more parameters
- deepspeech SOTA in 2014

What then?

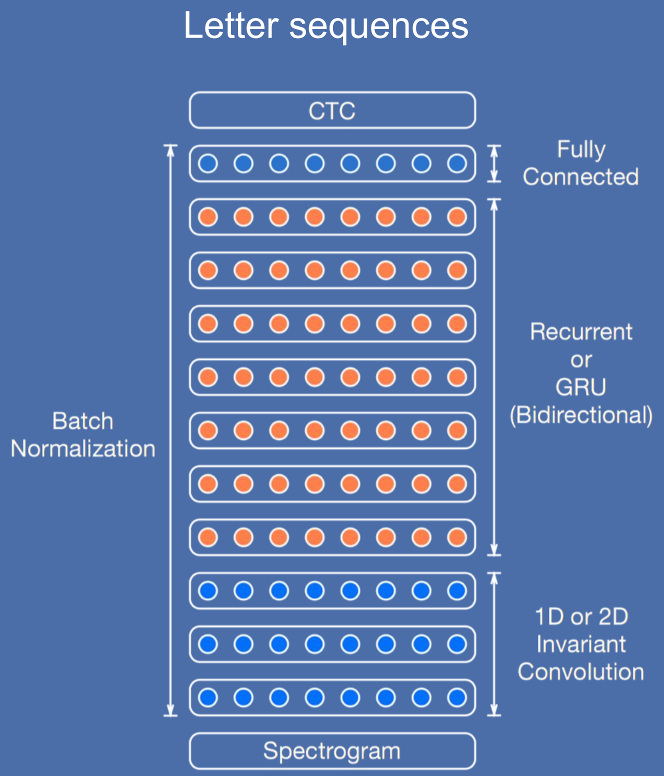


CTC based ASR

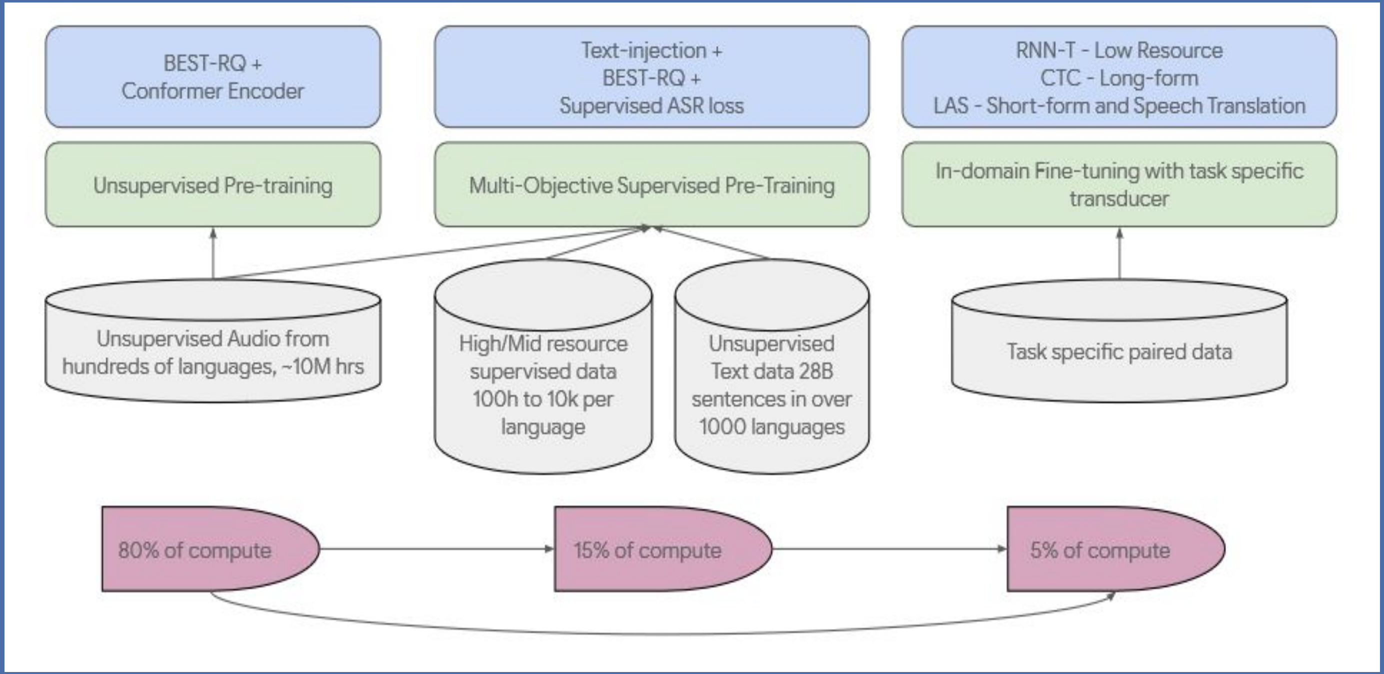
- annotated speech: speech+unaligned transcription
- much simpler to set up than large vocabulary HMM
- much more parameters
- deepspeech SOTA in 2014

What then?

most speech online is not annotated, can we still use them?



Sota ASR pipeline (Google USM 2023)

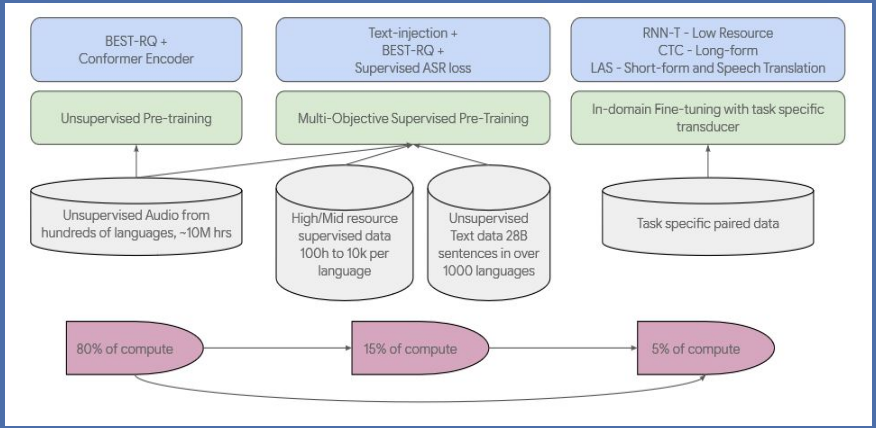


Current ASR pipeline
(figure from Google USM)

1- Semi-supervised learning:
- self-training
- pre-training
- SUPERB benchmark

2- Losses for ASR and more:
- CTC
- LAS
- RNN T

3- Dealing with long form



Semi-supervised learning:

methods to leverage unannotated speech to improve ASR models

Self training

1- Train an acoustic model (AM) on **small** corpus of annotated speech
2- Train a text-language model (LM) on **big** corpus of text

[1] « **Theoretical Analysis of Self-Training with Deep Networks on Unlabeled Data** », Wei et al., 2020

Self training

1- Train an acoustic model (AM) on **small** corpus of annotated speech
2- Train a text-language model (LM) on **big** corpus of text
3- Pseudo-transcriptions: use AM+LM to transcribe **big** corpus of unannotated speech
4- Discard pseudo-transcription with a heuristic

[1] « **Theoretical Analysis of Self-Training with Deep Networks on Unlabeled Data** », Wei et al., 2020

Self training

1- Train an acoustic model (AM) on **small** corpus of annotated speech
2- Train a text-language model (LM) on **big** corpus of text
3- Pseudo-transcriptions: use AM+LM to transcribe **big** corpus of unannotated speech
4- Discard pseudo-transcription with a heuristic
5- Retrain the model with both the true and pseudo-transcriptions (with data augmentation)
6- Iterate step 3,4,5 until WER on held out dataset stop decreasing

[1] « **Theoretical Analysis of Self-Training with Deep Networks on Unlabeled Data** », Wei et al., 2020

Self training

1- Train an acoustic model (AM) on **small** corpus of annotated speech
2- Train a text-language model (LM) on **big** corpus of text
3- Pseudo-transcriptions: use AM+LM to transcribe **big** corpus of unannotated speech
4- Discard pseudo-transcription with a heuristic
5- Retrain the model with both the true and pseudo-transcriptions (with data augmentation)
6- Iterate step 3,4,5 until WER on held out dataset stop decreasing

Note:

- it works even without LM, magic tool of deep learning (see theoretical understanding [1])
- Looks like Expectation Maximisation

[1] « **Theoretical Analysis of Self-Training with Deep Networks on Unlabeled Data** », Wei et al., 2020

Self training

- example of heuristic: evaluate the confidence of the model by using the predicted probabilities (low entropy = highly confident, high entropy = low confidence)

	460 hours labelled	100 hours labelled	100 hours labelled + 360 hours unlabelled
WER (%)	4.23	8.06	5.79

« Self-Training for End-to-End Speech Recognition », Kahn et al., 2020

Pre-training

what if we do not have annotated speech ?

Pre-training

what if we do not have annotated speech ?

semi-supervised learning using pre-training:

1- self-supervised learning to train a LM model on unannotated speech
2- continue training on the LM model with supervised learning with annotated speech

Cosine similarity and L2 norm

How to measure the similarity between two vectors ?

z_i and z_j two vectors in R^N

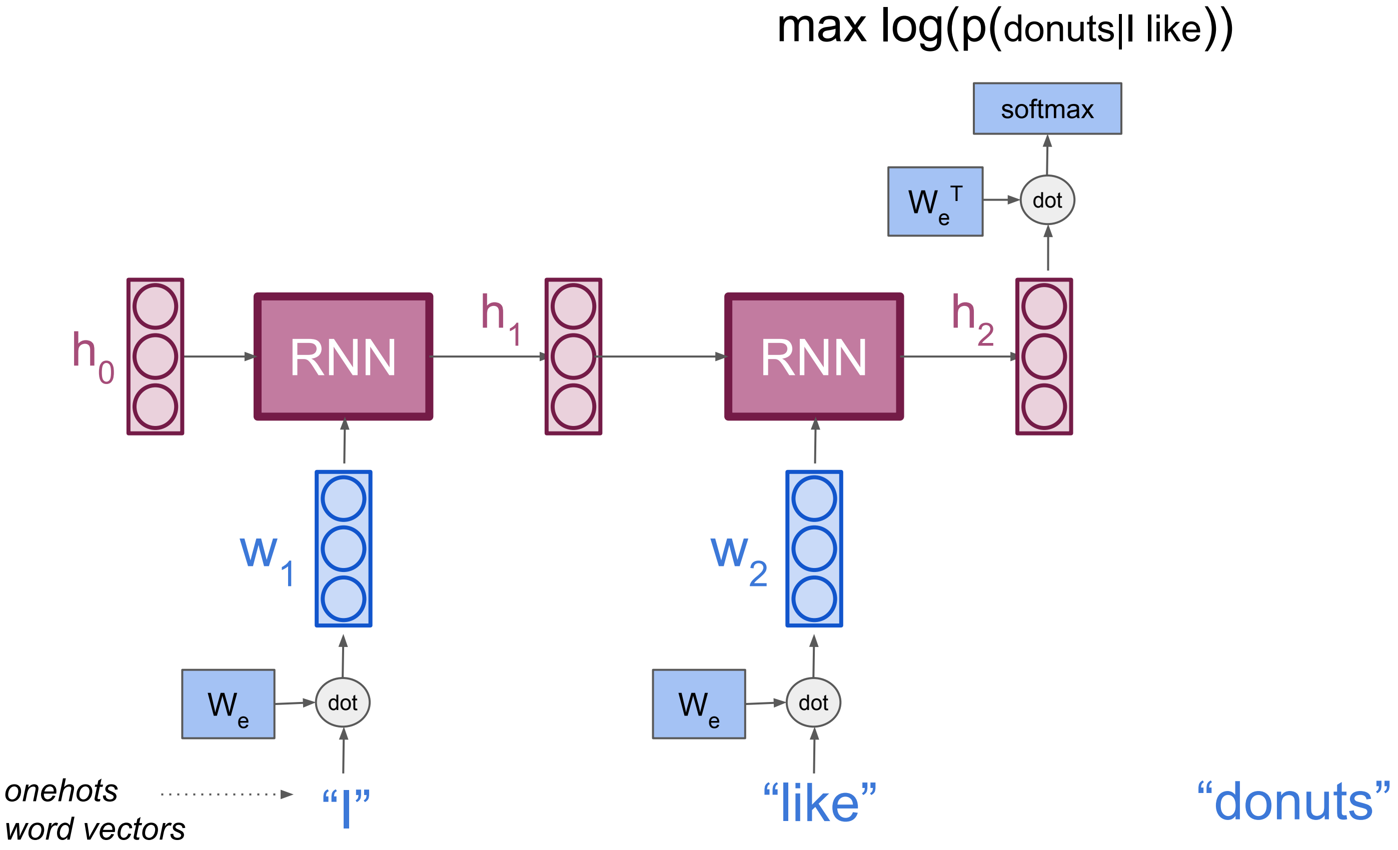
Cosine similarity: normalised dot product: $$sim(z_i, z_j) = \frac{z_i \cdot z_j}{||z_i|| ||z_j||}$$

L2 norm: $$||z_i - z_j||_2^2 = (z_i - z_j)^T (z_i - z_j) = ||z_i||_2^2 + ||z_j||_2^2 - 2z_i.z_j$$

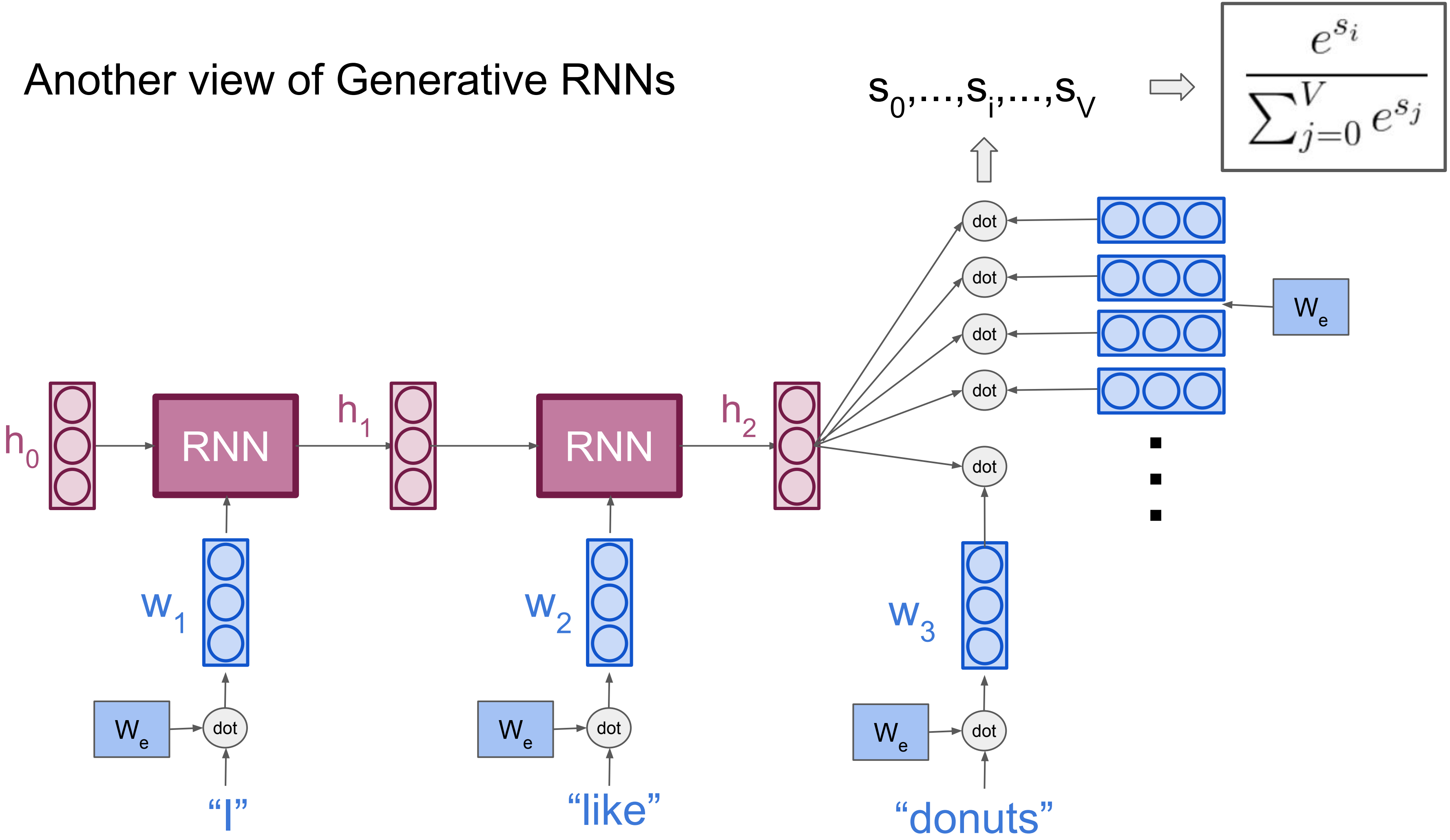
L2 norm if vectors are normalised: $$||z_i - z_j||_2^2 = 2 - 2sim(z_i, z_j)$$

The two metrics are equivalent for normalised vectors

Generative RNNs

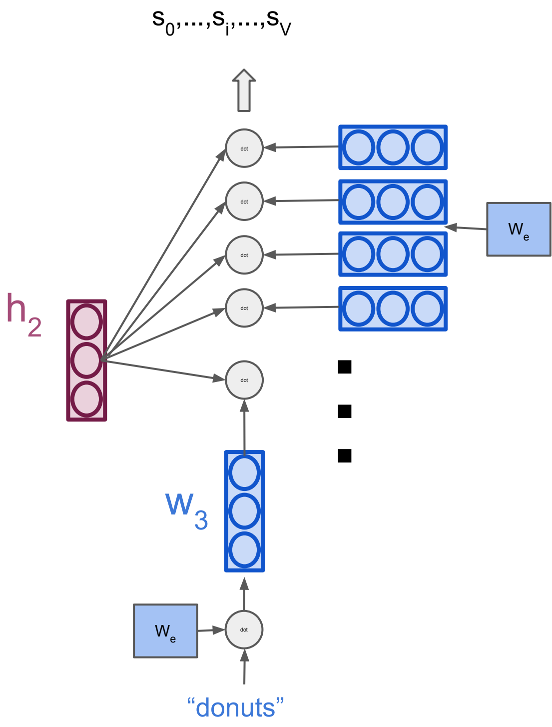


Another view of Generative RNNs



Another view of Generative RNNs

$$log(p('donuts'|'I\ like')) = log(\frac{e^{s_i}}{\sum_{j=0}^{V} e^{s_j}}) = \quad s_i - log(\sum_{j=0}^{V} e^{s_j})$$



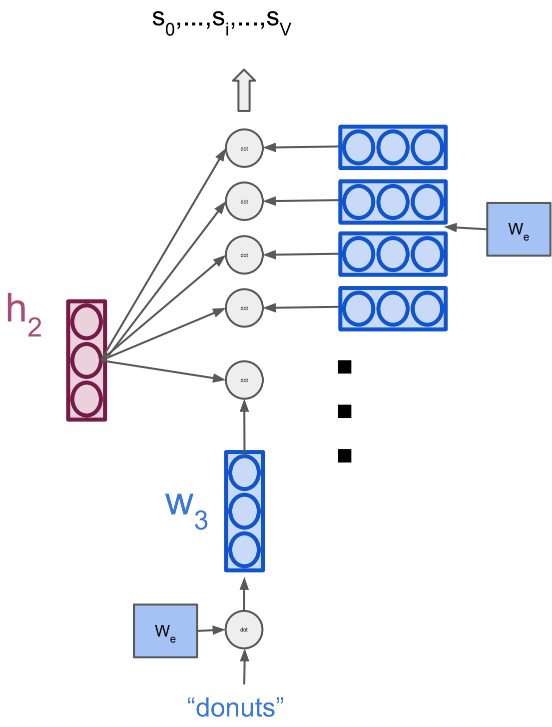
Another view of Generative RNNs

$$log(p('donuts'|'I\ like')) = log(\frac{e^{s_i}}{\sum_{j=0}^{V} e^{s_j}}) = \quad s_i - log(\sum_{j=0}^{V} e^{s_j})$$

h_2 is trained to be similar to w_3 and farther from all other words w_i

w_3 is the positive sample

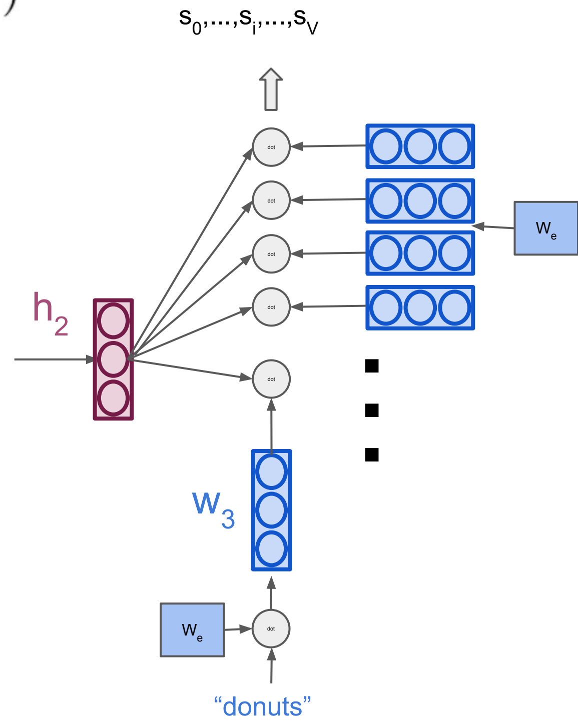
all w_i are the negative samples



Another view of Generative RNNs

$$log(p('donuts'|'I\ like')) = log(\frac{e^{s_i}}{\sum_{j=0}^{V} e^{s_j}}) = \quad s_i - log(\sum_{j=0}^{V} e^{s_j})$$

Do we need negative samples?



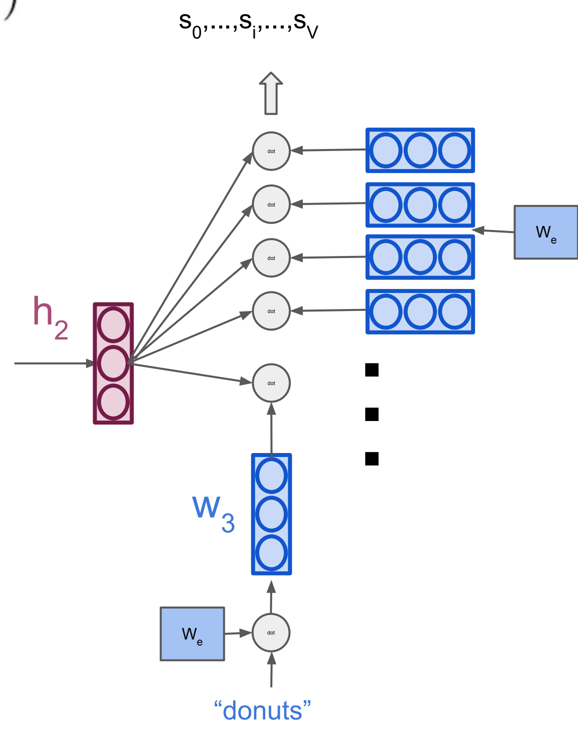
Another view of Generative RNNs

$$log(p('donuts'|'I\ like')) = log(\frac{e^{s_i}}{\sum_{j=0}^{V} e^{s_j}}) = \quad s_i - log(\sum_{j=0}^{V} e^{s_j})$$

Do we need negative samples?

yes, otherwise collapse on constant vector

Do we need all V negative samples?



Another view of Generative RNNs

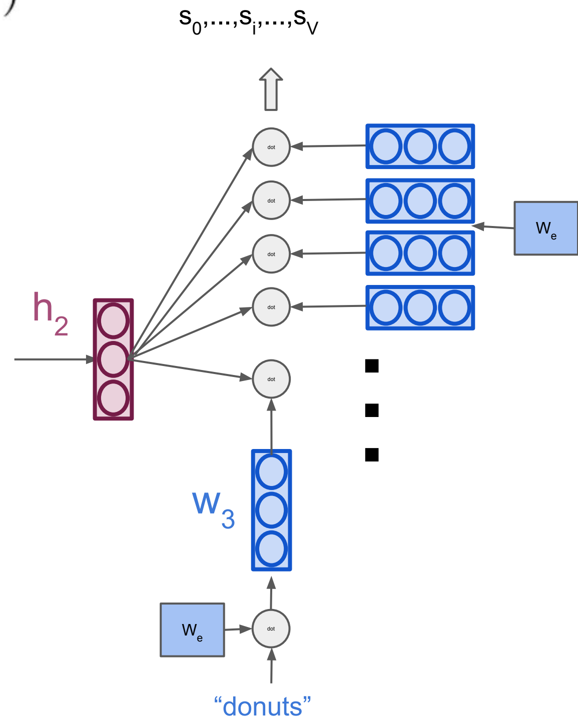
$$log(p('donuts'|'I\ like')) = log(\frac{e^{s_i}}{\sum_{j=0}^{V} e^{s_j}}) = \quad s_i - log(\sum_{j=0}^{V} e^{s_j})$$

Do we need negative samples?

yes, otherwise collapse on constant vector

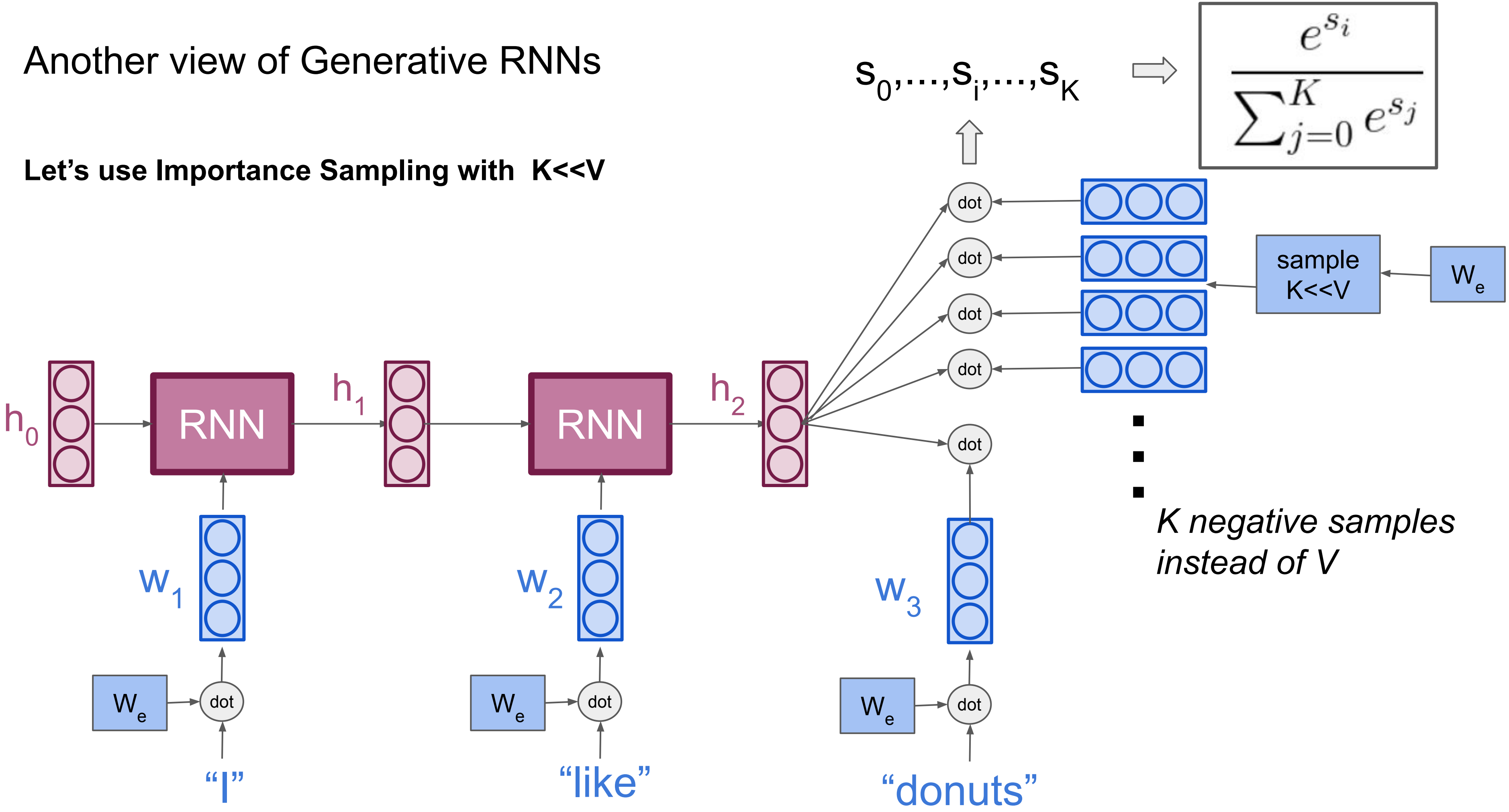
Do we need all V negative samples?

no, it works even with K<<V
Bengio et al (2003): **Importance Sampling**

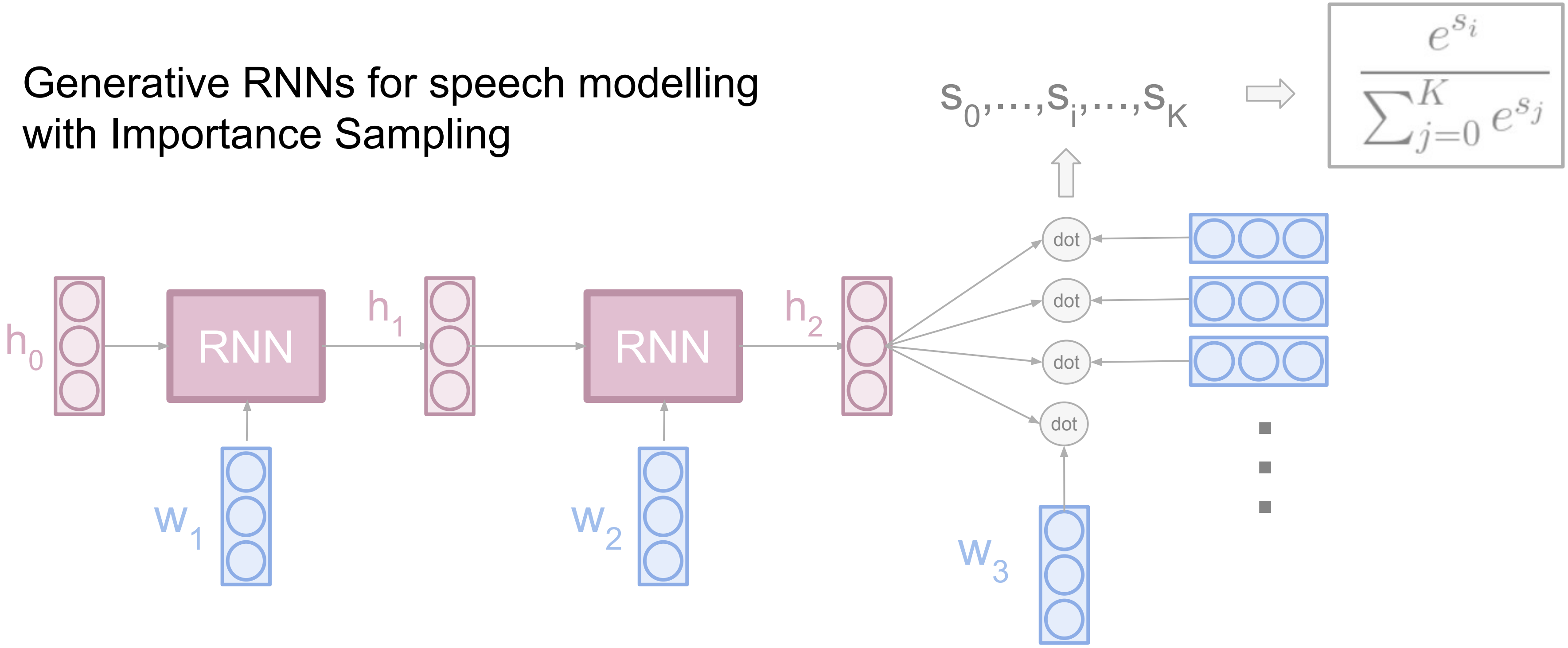


Another view of Generative RNNs

Let's use Importance Sampling with K<<V

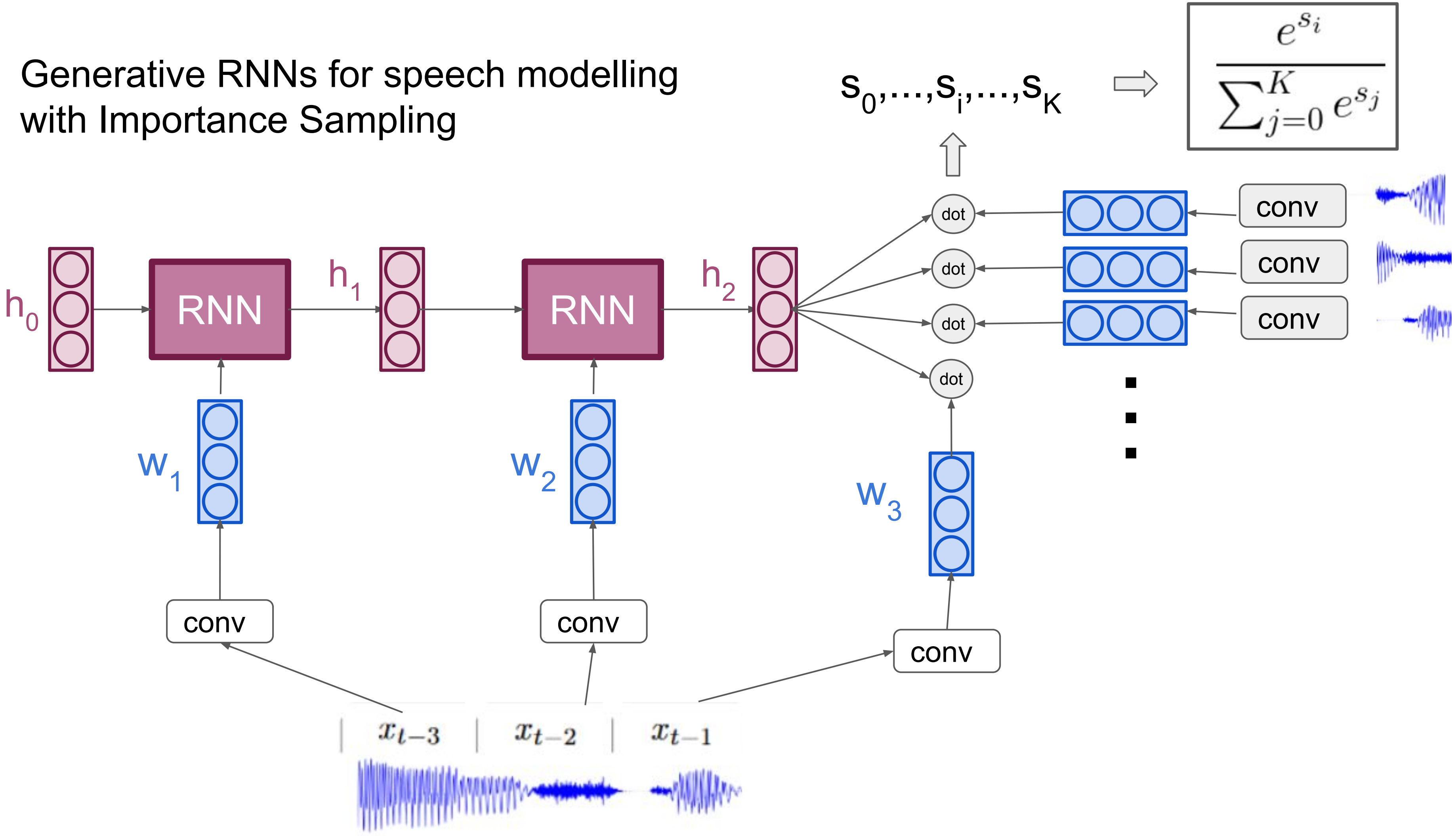


Generative RNNs for speech modelling with Importance Sampling



How to use speech instead of text in this pipeline ?

Generative RNNs for speech modelling with Importance Sampling



Contrastive loss: cross entropy on softmax and Importance sampling

Problem 1: w_1 , w_2 , w_3 are not independent random variables anymore

speaker identity and recording conditions interfere with the contrastive objective !

Contrastive loss: cross entropy on softmax and Importance sampling

Problem 1: w_1 , w_2 , w_3 are not independent random variables anymore

speaker identity and recording conditions interfere with the contrastive objective !

How to choose the negative samples to ensure semantic modelling ?

same speaker

same utterance (not that much used in practice)

Contrastive loss: cross entropy on softmax and Importance sampling

Problem 1: w_1 , w_2 , w_3 are not independent random variables anymore

speaker identity and recording conditions interfere with the contrastive objective !

How to choose the negative samples to ensure semantic modelling ?

same speaker

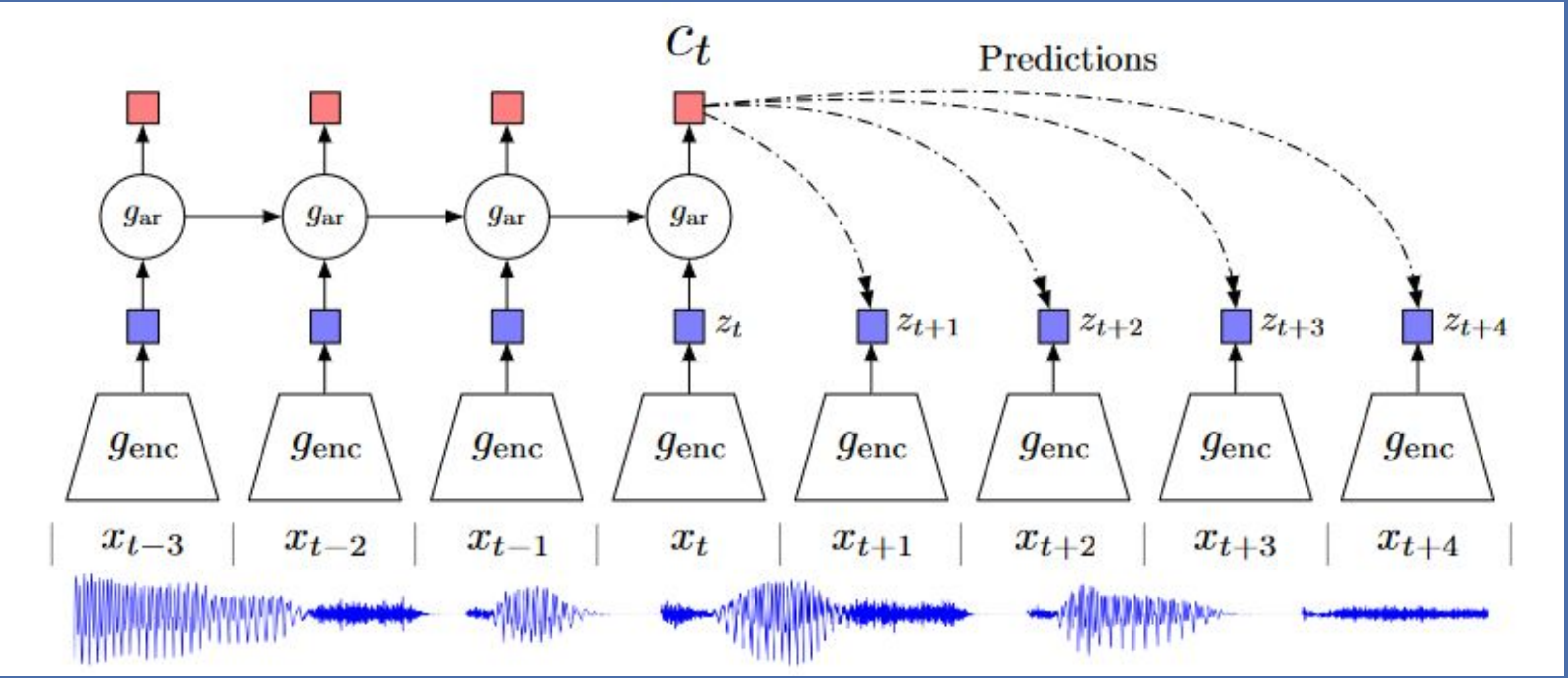
same utterance (not that much used in practice)

Problem 2: Some negative samples could be positives

not a problem, languages are rich enough

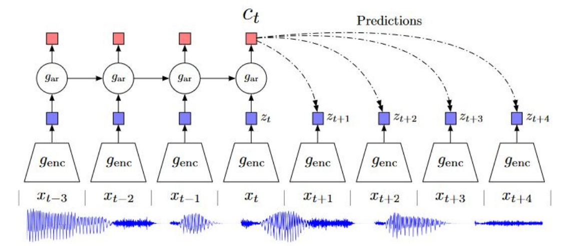
Contrastive Predicting Coding

(e.g with CPC)



Contrastive Learning with generative objective

(e.g with CPC)



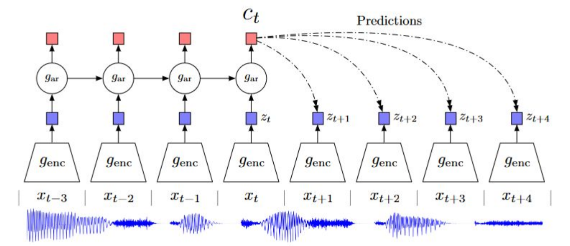
Multiple prediction heads

For each head:

    choose N negative same-speaker samples

Contrastive Learning with generative objective

(e.g with CPC)



Multiple prediction heads

For each head:

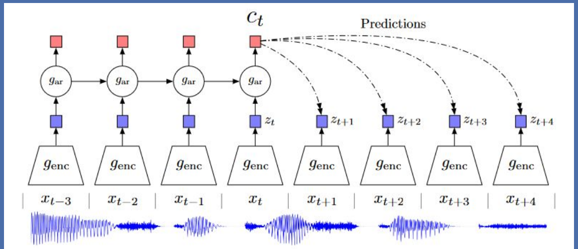
- choose N negative same-speaker samples
- compute the NT-Xent Loss (contrastive loss)

$$\mathcal{L}^{\mathrm{NT-Xent}} = -\frac{1}{N} \sum_{i,j \in \mathcal{MB}} \log \frac{\exp(sim(z_i, z_j)/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[k \neq i]} \exp(sim(z_i, z_k)/\tau)}$$

$$sim(z_i, z_j) = \frac{z_i \cdot z_j}{\|z_i\| \|z_j\|}$$

Contrastive Learning with generative objective

(e.g with CPC)



Multiple prediction heads

For each head:

- choose N negative same-speaker samples
- compute the NT-Xent Loss (contrastive loss)

$$\mathcal{L}^{\mathrm{NT-Xent}} = -\frac{1}{N} \sum_{i,j \in \mathcal{MB}} \log \frac{\exp(sim(z_i, z_j)/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[k \neq i]} \exp(sim(z_i, z_k)/\tau)}$$

$$sim(z_i, z_j) = \frac{z_i \cdot z_j}{\|z_i\| \|z_j\|}$$

normalisation is not necessary, it’s a safeguard, why ?

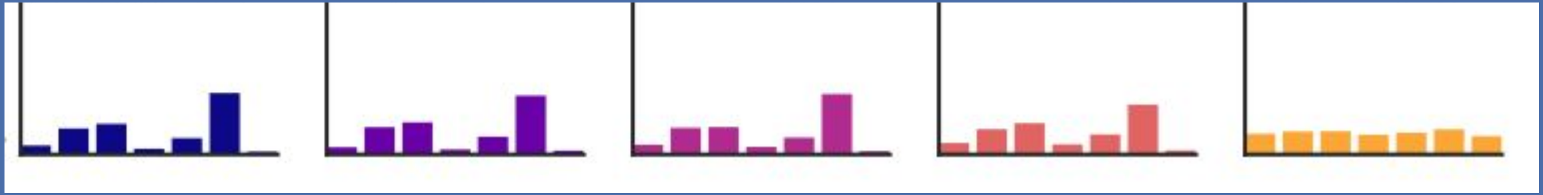
Softmax with Temperature

$$\frac{e^{\frac{y_i}{T}}}{\sum_{k=1}^{n} e^{\frac{y_k}{T}}}$$

Softmax is a smooth version of the categorical distribution

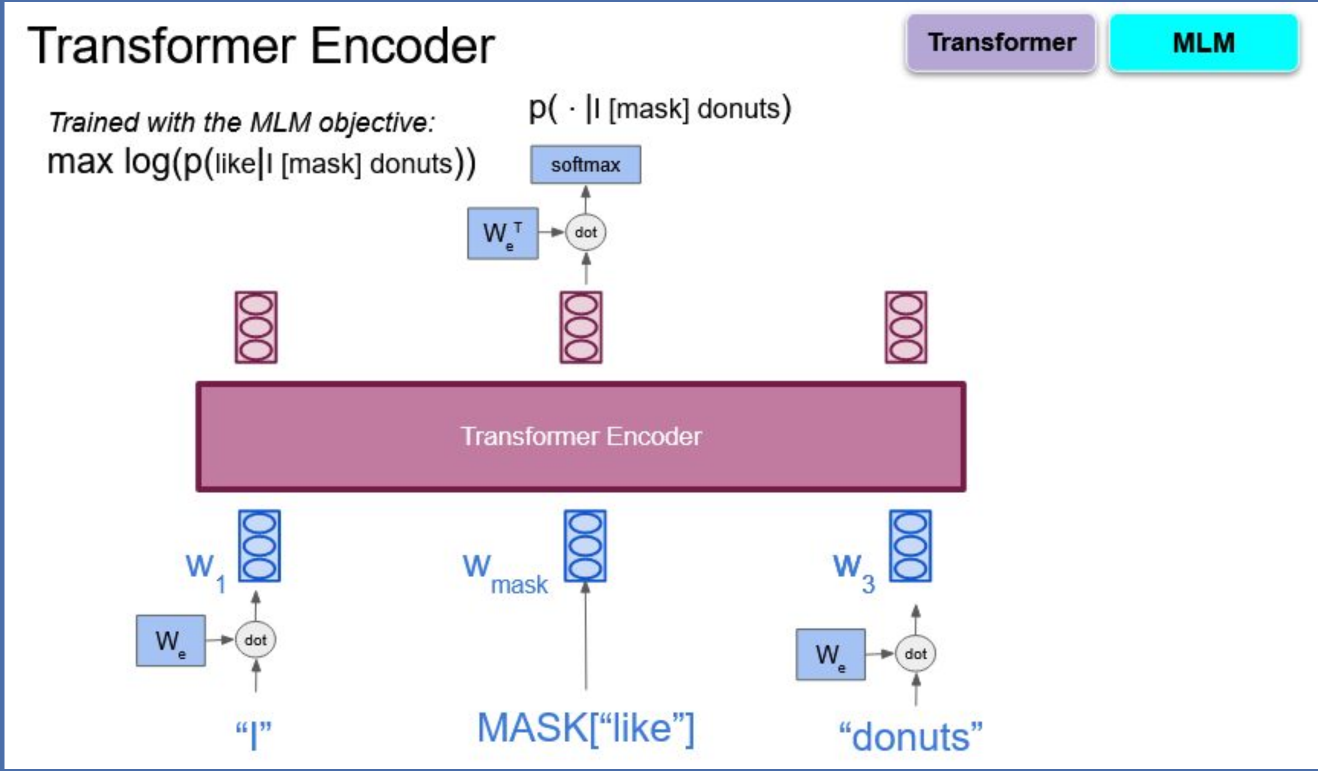
If T=0, softmax is categorical
If T=inf, softmax is uniform

Categorical T=0.1 T=0.5 T=1.0 T=10.0



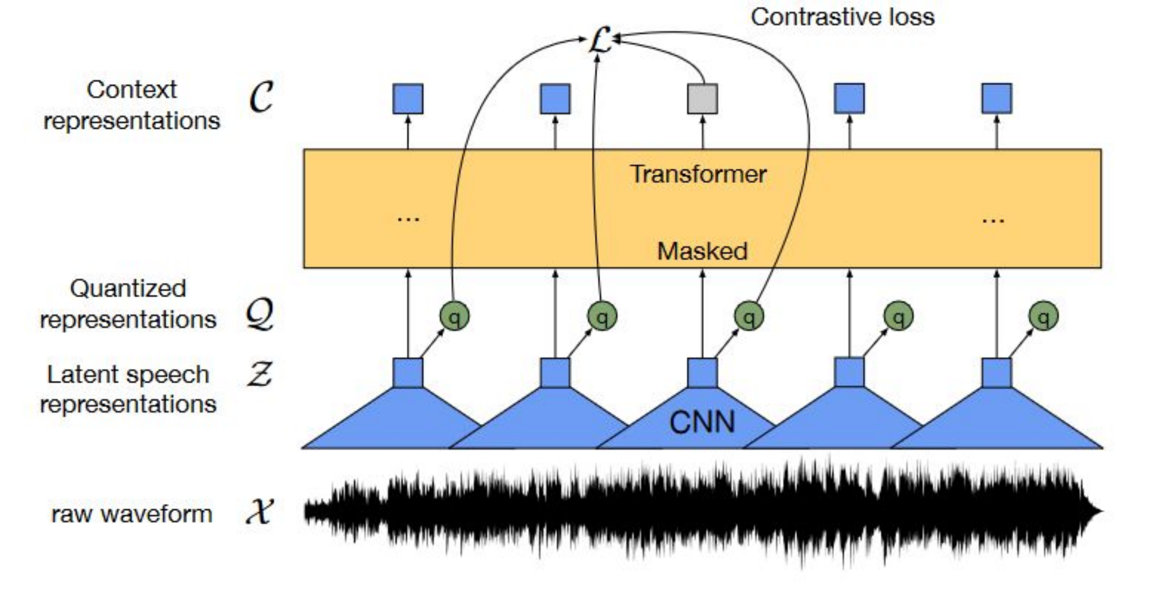
You can control the confidence of the model while training

From SLP class 3: MLM objective for Transformers



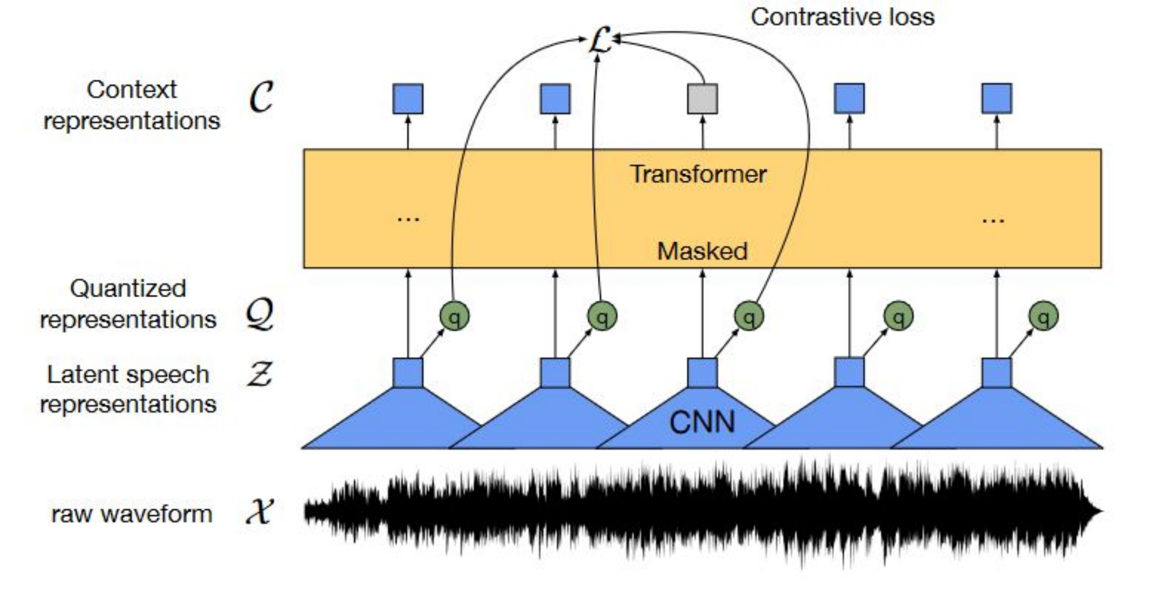
Contrastive Learning with mask language modelling

(e.g with wav2vec2)



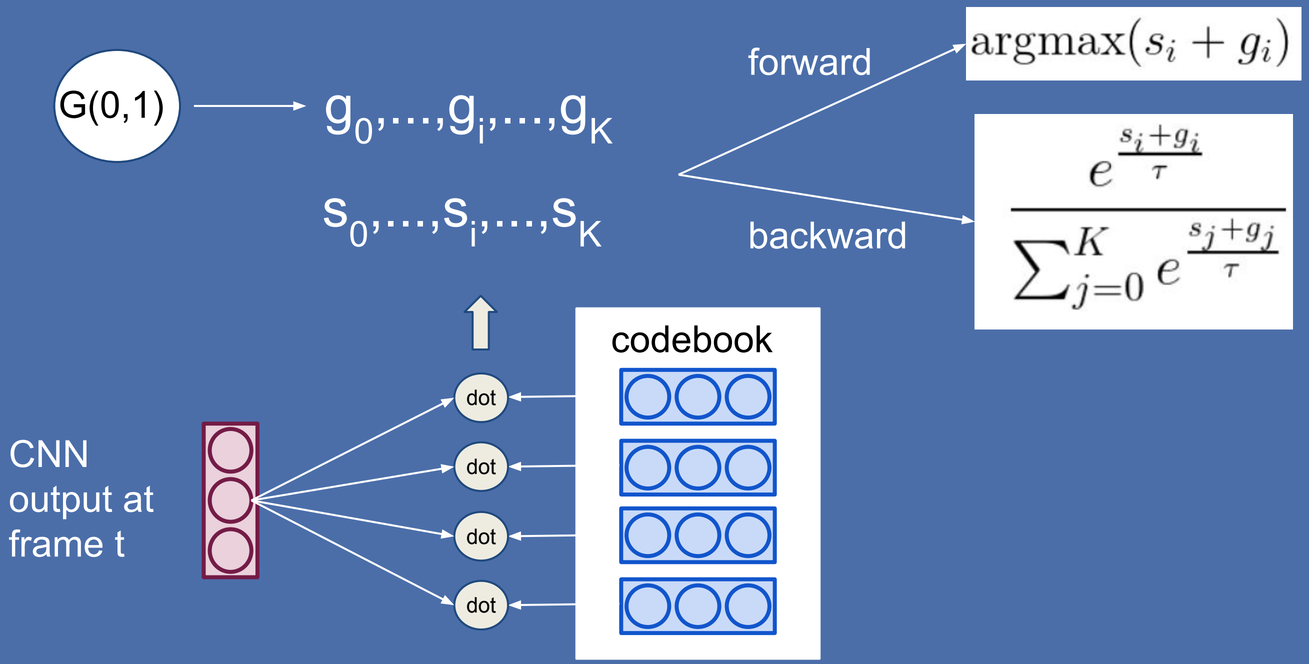
Contrastive Learning with mask language modelling

(e.g with wav2vec2)

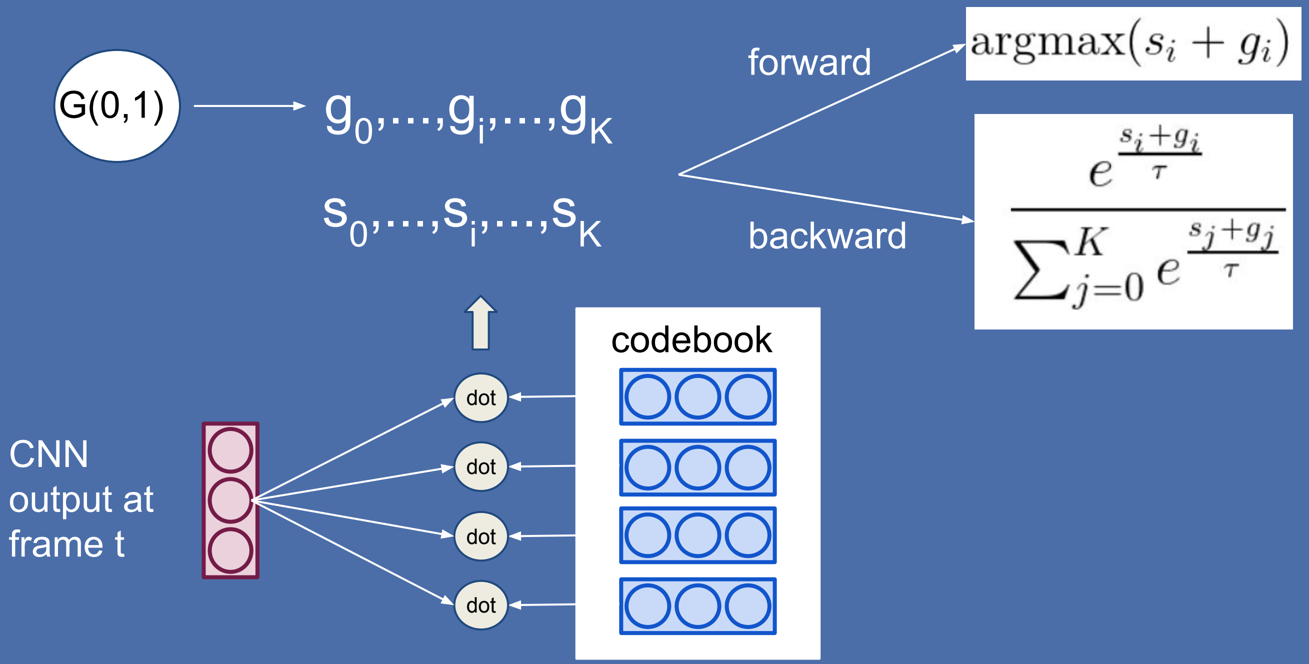


- masking is done with N(10,10)
- quantizer q: sample a class value for the output of CNN, classes are learned with Gumbel softmax
- q is not “necessary” but removes low level information that interfere with contrastive learning

Gumbel softmax: reparametrisation trick to sample from distribution and Straight Throught Estimator



Gumbel softmax: reparametrisation trick to sample from distribution and Straight Throught Estimator



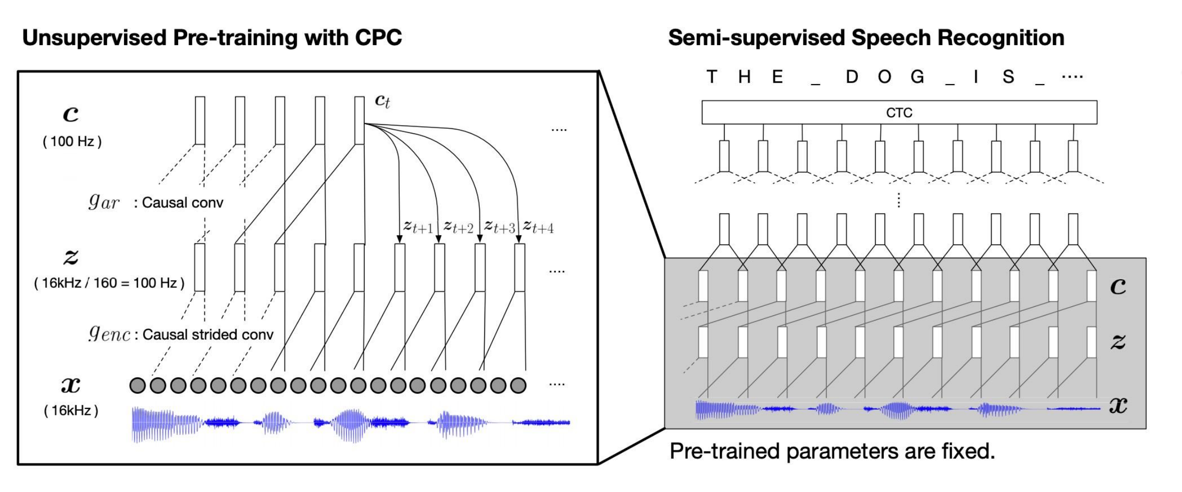
-reparametrisation trick: the sampling value is held constant during differentiation of the parameters

-diminish the temperature along training (low temperature=using a softmax)

Semi-supervised learning for ASR

- After pre-training, add a small network train with CTC on a low-resource language with only few annotated speech

ENGLISH



AMHARIC

« Unsupervised Learning of Efficient and Robust Speech Representations », Kawakami et al., 2019

Semi-supervised learning for ASR

- Word Error Rate (%) on CPC

MODEL	AMHARIC	FONGBE	SWAHILI	WOLOF
NO PRE-TRAINING	78.85	65.34	77.18	69.93
CPC (8K HOURS OF ENGLISH)	**66.10**	**57.20**	**69.23**	**55.41**

« **Unsupervised Learning of Efficient and Robust Speech Representations** », Kawakami et al., 2019

Wav2Vec2

finetune the transformer stack (freeze convolutions)
+ new linear layer

Model	Unlabeled data	LM	dev clean	dev other	test clean	test other
10 min labeled						
BASE	LS-960	None	46.1	51.5	46.9	50.9
		4-gram	8.9	15.7	9.1	15.6
		Transf.	6.6	13.2	6.9	12.9
LARGE	LS-960	None	43.0	46.3	43.5	45.3
		4-gram	8.6	12.9	8.9	13.1
		Transf.	6.6	10.6	6.8	10.8
LARGE	LV-60k	None	38.3	41.0	40.2	38.7
		4-gram	6.3	9.8	6.6	10.3
		Transf.	4.6	7.9	4.8	8.2
1h labeled						
BASE	LS-960	None	24.1	29.6	24.5	29.7
		4-gram	5.0	10.8	5.5	11.3
		Transf.	3.8	9.0	4.0	9.3
LARGE	LS-960	None	21.6	25.3	22.1	25.3
		4-gram	4.8	8.5	5.1	9.4
		Transf.	3.8	7.1	3.9	7.6
LARGE	LV-60k	None	17.3	20.6	17.2	20.3
		4-gram	3.6	6.5	3.8	7.1
		Transf.	2.9	5.4	2.9	5.8
10h labeled						
BASE	LS-960	None	10.9	17.4	11.1	17.6
		4-gram	3.8	9.1	4.3	9.5
		Transf.	2.9	7.4	3.2	7.8
LARGE	LS-960	None	8.1	12.0	8.0	12.1
		4-gram	3.4	6.9	3.8	7.3
		Transf.	2.9	5.7	3.2	6.1
LARGE	LV-60k	None	6.3	9.8	6.3	10.0
		4-gram	2.6	5.5	3.0	5.8
		Transf.	2.4	4.8	2.6	4.9

Wav2Vec2

finetune the transformer stack (freeze convolutions)
+ new linear layer

the bigger model the better

the bigger pretraining the better

the more annotated speech, the less LM are useful

Model	Unlabeled data	LM	dev clean	dev other	test clean	test other
10 min labeled						
BASE	LS-960	None	46.1	51.5	46.9	50.9
		4-gram	8.9	15.7	9.1	15.6
		Transf.	6.6	13.2	6.9	12.9
LARGE	LS-960	None	43.0	46.3	43.5	45.3
		4-gram	8.6	12.9	8.9	13.1
		Transf.	6.6	10.6	6.8	10.8
LARGE	LV-60k	None	38.3	41.0	40.2	38.7
		4-gram	6.3	9.8	6.6	10.3
		Transf.	4.6	7.9	4.8	8.2
1h labeled						
BASE	LS-960	None	24.1	29.6	24.5	29.7
		4-gram	5.0	10.8	5.5	11.3
		Transf.	3.8	9.0	4.0	9.3
LARGE	LS-960	None	21.6	25.3	22.1	25.3
		4-gram	4.8	8.5	5.1	9.4
		Transf.	3.8	7.1	3.9	7.6
LARGE	LV-60k	None	17.3	20.6	17.2	20.3
		4-gram	3.6	6.5	3.8	7.1
		Transf.	2.9	5.4	2.9	5.8
10h labeled						
BASE	LS-960	None	10.9	17.4	11.1	17.6
		4-gram	3.8	9.1	4.3	9.5
		Transf.	2.9	7.4	3.2	7.8
LARGE	LS-960	None	8.1	12.0	8.0	12.1
		4-gram	3.4	6.9	3.8	7.3
		Transf.	2.9	5.7	3.2	6.1
LARGE	LV-60k	None	6.3	9.8	6.3	10.0
		4-gram	2.6	5.5	3.0	5.8
		Transf.	2.4	4.8	2.6	4.9

SUPERB: a benchmark for pre-trained SSL speech models

Wav2vec2, CPC: Self-supervised-Learning (SSL) models

SSL models can do more than ASR:

keyword spotting, intent classification, slot-filling, emotion recognition, speaker identification, diarization…

SUPERB: a benchmark for pre-trained SSL speech models

Wav2vec2, CPC: Self-supervised-Learning (SSL) models

SSL models can do more than ASR:

keyword spotting, intent classification, slot-filling, emotion recognition, speaker identification, diarization…

one benchmark for all ?

SUPERB: train a SSL, freeze, add a linear layer (or small RNN), finetune on a task

SUPERB: a benchmark for pre-trained SSL speech models

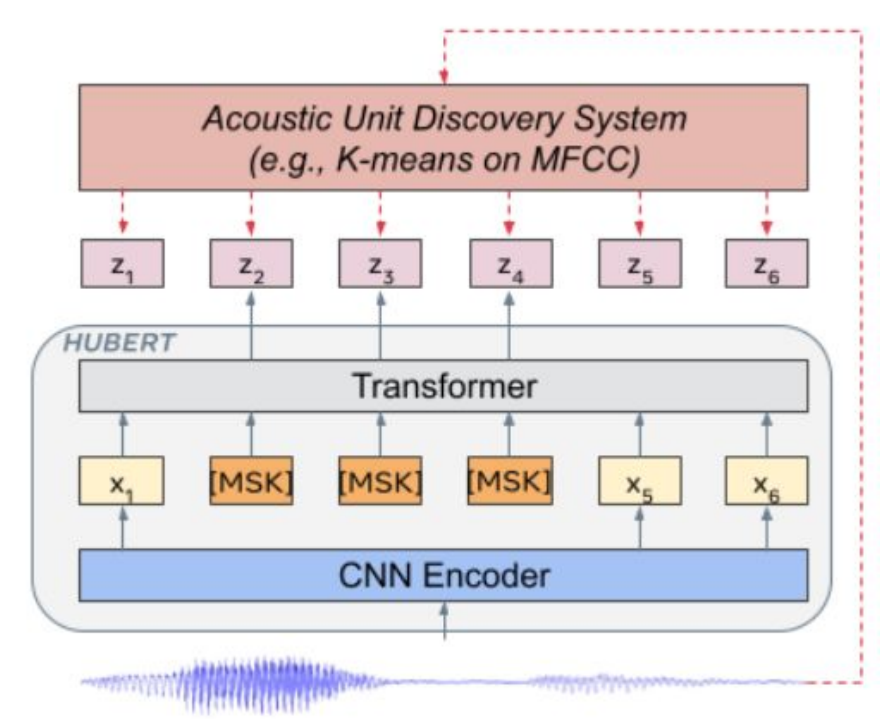
	PR	KS	IC	SID	ER	ASR (WER)		QbE	SF		ASV	SD
	PER ↓	Acc ↑	Acc ↑	Acc ↑	Acc ↑	w/o ↓	w/ LM ↓	MTWV ↑	F1 ↑	CER ↓	EER ↓	DER ↓
FBANK	82.01	8.63	9.10	8.5E-4	35.39	23.18	15.21	0.0058	69.64	52.94	9.56	10.05
PASE+ [16]	58.87	82.54	29.82	37.99	57.86	25.11	16.62	0.0072	62.14	60.17	11.61	8.68
APC [7]	41.98	91.01	74.69	60.42	59.33	21.28	14.74	0.0310	70.46	50.89	8.56	10.53
VQ-APC [32]	41.08	91.11	74.48	60.15	59.66	21.20	15.21	0.0251	68.53	52.91	8.72	10.45
NPC [33]	43.81	88.96	69.44	55.92	59.08	20.20	13.91	0.0246	72.79	48.44	9.4	9.34
Mockingjay [8]	70.19	83.67	34.33	32.29	50.28	22.82	15.48	6.6E-04	61.59	58.89	11.66	10.54
TERA [9]	49.17	89.48	58.42	57.57	56.27	18.17	12.16	0.0013	67.50	54.17	15.89	9.96
DeCoAR 2.0 [10]	14.93	94.48	90.80	74.42	62.47	13.02	9.07	0.0406	83.28	34.73	7.16	6.59
modified CPC [34]	42.54	91.88	64.09	39.63	60.96	20.18	13.53	0.0326	71.19	49.91	12.86	10.38
wav2vec [12]	31.58	95.59	84.92	56.56	59.79	15.86	11.00	0.0485	76.37	43.71	7.99	9.9
vq-wav2vec [13]	33.48	93.38	85.68	38.80	58.24	17.71	12.80	0.0410	77.68	41.54	10.38	9.93
wav2vec 2.0 Base [14]	5.74	96.23	92.35	75.18	63.43	6.43	4.79	0.0233	88.30	24.77	6.02	6.08
wav2vec 2.0 Large [14]	4.75	**96.66**	95.28	86.14	65.64	3.75	3.10	0.0489	87.11	27.31	5.65	**5.62**
HuBERT Base [35]	5.41	96.30	98.34	81.42	64.92	6.42	4.79	**0.0736**	88.53	25.20	**5.11**	5.88
HuBERT Large [35]	**3.53**	95.29	**98.76**	**90.33**	**67.62**	**3.62**	**2.94**	0.0353	**89.81**	**21.76**	5.98	5.75

HuBERT: learning to predict MFCC+kmeans

Not a contrastive loss, just a
cross entropy on the pseudo-labels

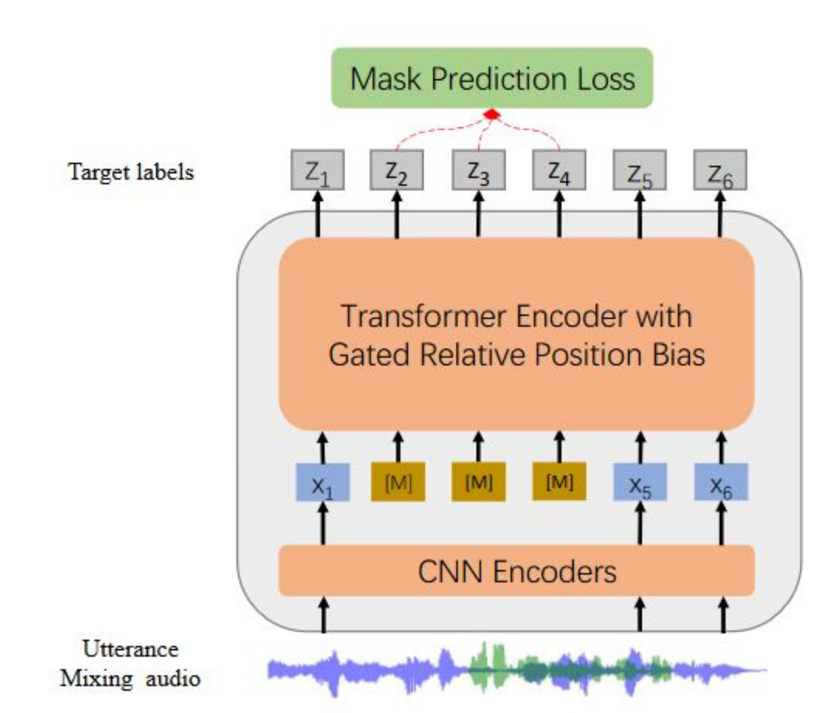
Iterative learning

Works better than Wav2vec2.0 and CPC



WavLM: HuBERT + mixing audio

better pre-training objective than HuBERT:
- -learn to do denoising in addition
- -better performances than HuBERT on many SUPERB tasks



Is it necessary to learn the pseudo-labels with k-means?

No…

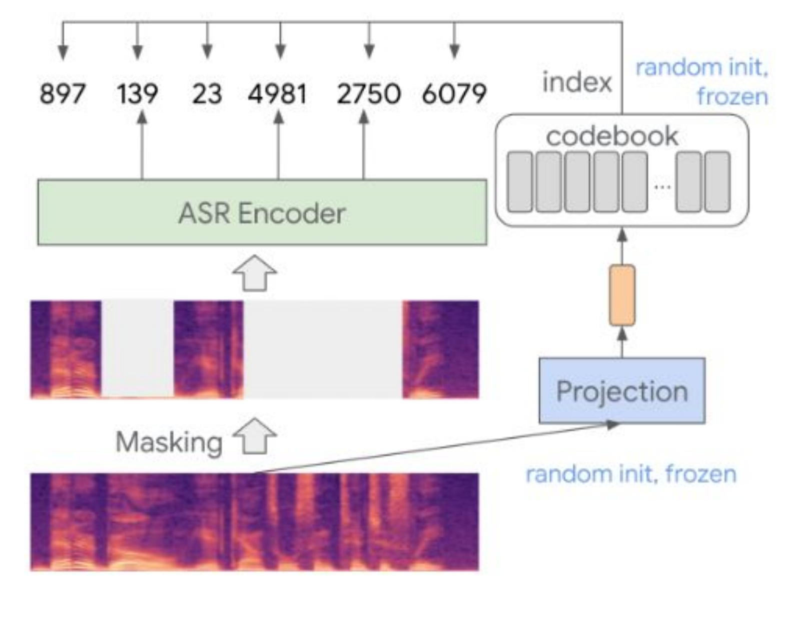
Random projection and random quantisation

Same as model as HuBERT and Wav2vec2

Random projection:
-dimensionality reduction method
-a matrix of gaussian noise with **unit norm columns**

followed by Random quantisation:
like an untrained k-means

This is not random labelling !



Random Projection and Random quantisation

same performances than HuBERT and Wav2vec.2.0

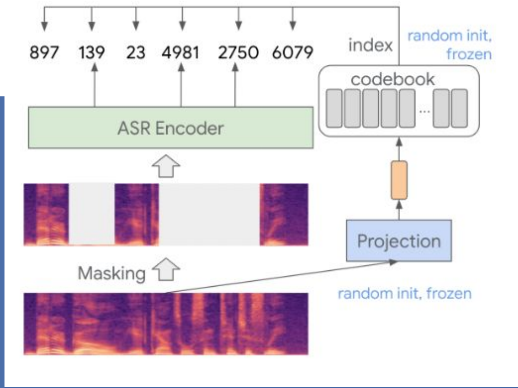
Table 1. LibriSpeech results with non-streaming models. The LM used in our experiment is a Transfomer LM with model size 0.1B.

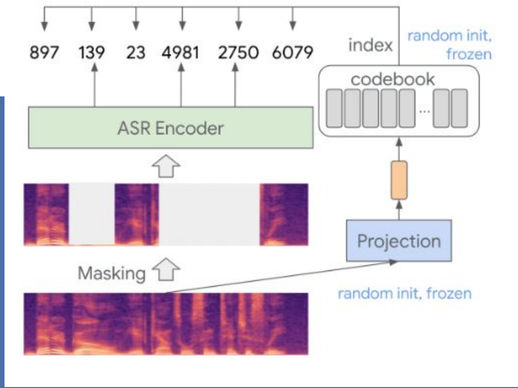
Method	Size (B)	No LM				With LM			
		dev	dev-other	test	test-other	dev	dev-other	test	test-other
wav2vec 2.0 (Baevski et al., 2020b)	0.3	2.1	4.5	2.2	4.5	1.6	3.0	1.8	3.3
HuBERT Large (Hsu et al., 2021)	0.3	–	–	–	–	1.5	3.0	1.9	3.3
HuBERT X-Large (Hsu et al., 2021)	1.0	–	–	–	–	1.5	**2.5**	1.8	2.9
w2v-Conformer XL (Zhang et al., 2020)	0.6	1.7	3.5	1.7	3.5	1.6	3.2	**1.5**	3.2
w2v-BERT XL (Chung et al., 2021)	0.6	**1.5**	2.9	**1.5**	**2.9**	**1.4**	2.8	**1.5**	2.8
BEST-RQ (Ours)	0.6	**1.5**	**2.8**	1.6	**2.9**	**1.4**	2.6	**1.5**	**2.7**

Random Projection

It is **not** random labelling: random projection uses the input

what is a good dimensionality reduction method?

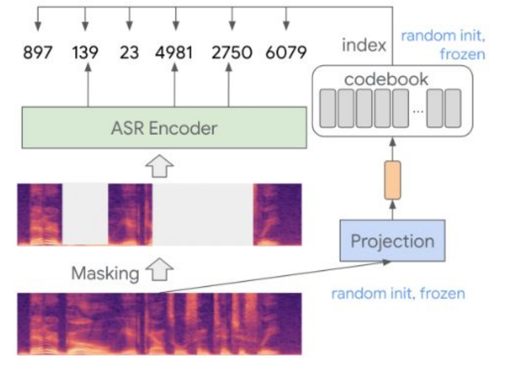




Random Projection

It is **not** random labelling: random projection uses the input

what is a good dimensionality reduction method?
    the axis do not collapse on each other: orthogonal basis
    preserves distances: correlation (or equality) between dot product in input and output space



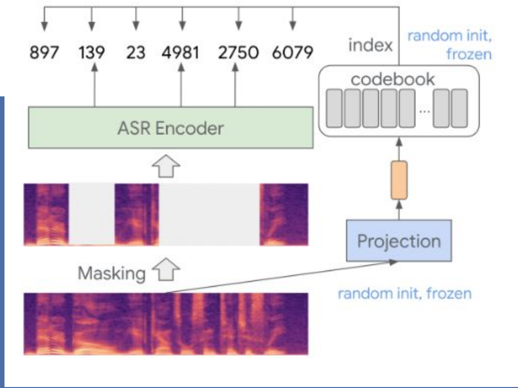
Random Projection

It is **not** random labelling: random projection uses the input

what is a good dimensionality reduction method?
the axis do not collapse on each other: orthogonal basis
preserves distances: correlation (or equality) between dot product in input and output space

It is the case of PCA, LDA,... but they are learnt

Random proj is an orthogonal basis that preserve the dot product. but why ?



Random Projection: why does it work?

Orthogonal basis?

Sample N vector unit normalised vector x_i in R^d with each dimension in N(0,1)

$y_{ij}=\mathrm{dot}(x_i , x_j)$

we can prove:

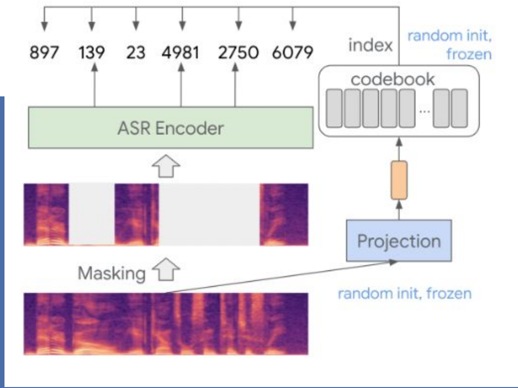
$E(y_{ii})=1$

if $i \neq j$ $E(y_{ij})=0$ and $\mathrm{Var}(y_{ij})=1/d$

therefore:

if d is big enough, the axis are quasiorthogonal to each other

Random Projection: why does it work?



Preserved distances ?

The distortion in dot product brought by the random proj is 0 in average and its variance is at most 1/d

formal proof in :

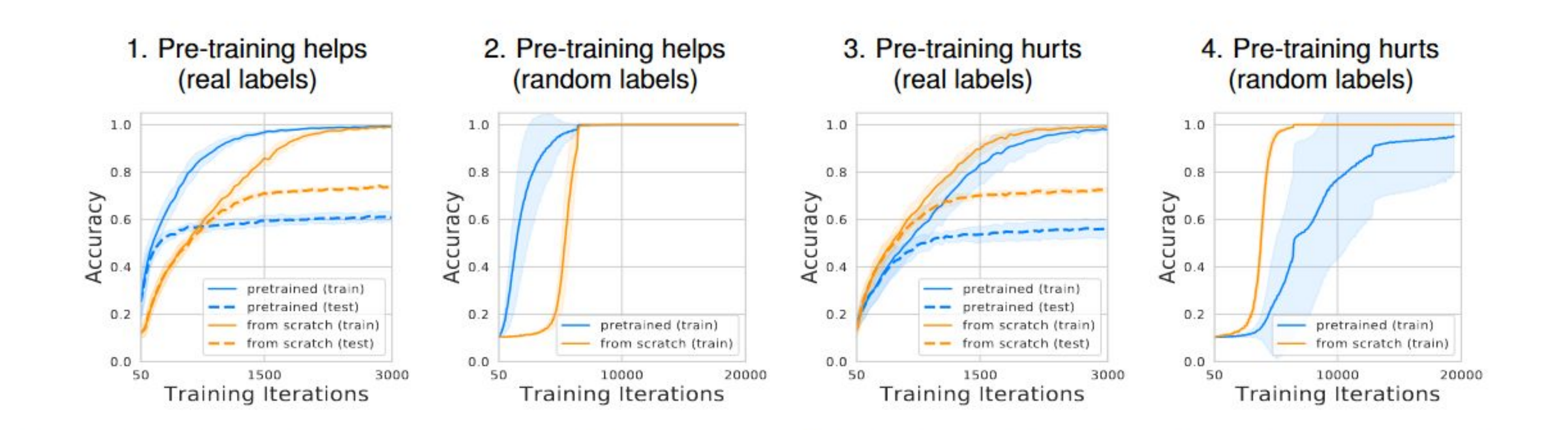
Dimensionality reduction by random mapping: Fast similarity computation for clustering

Note:

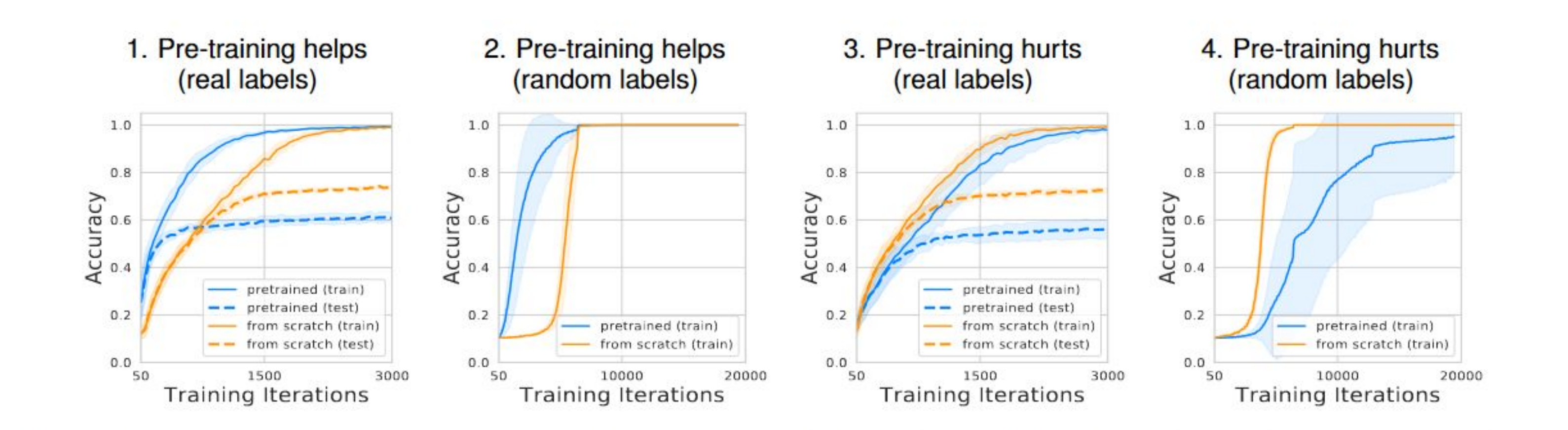
Random Projection is the default init of a fully connected layer in Pytorch

Real random labels ?

Real random labelling for pre-training in Image Classification (CIFAR10, Maennel et al 2006)

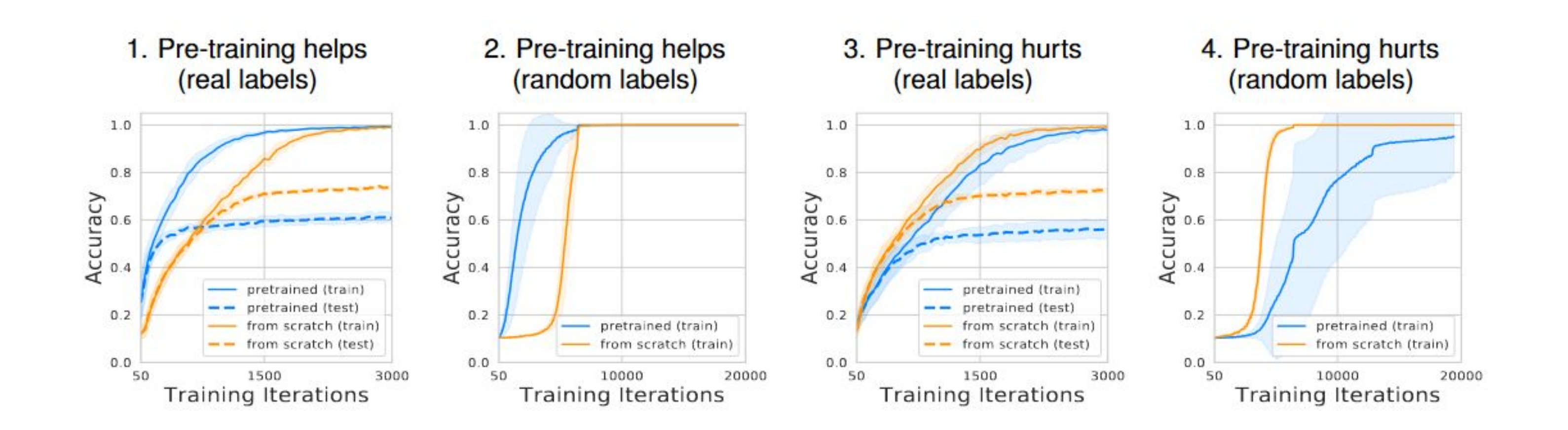


Real random labelling for pre-training in Image Classification (CIFAR10, Maennel et al 2006)



- after pre-training: “the principal components of weights at the first layer are aligned with the principal components of data”
- reproduce results without pre-training by sampling weights along the eigenvectors of the covariance matrix

Real random labelling for pre-training in Image Classification (CIFAR10, Maennel et al 2006)



- after pre-training: “the principal components of weights at the first layer are aligned with the principal components of data”
- reproduce results without pre-training by sampling weights along the eigenvectors of the covariance matrix
- don’t know why this happen, the bigger the model, the less pre-training hurts
- Not applied to speech yet

A family of SSL models:

- Wav2vec2.0
- HuBERT
- WavLM
- BERT-RQ (random projection)

Same encoder model but with different training losses on raw speech

Zero-shot performances of speech SSL models ?

ABX test on synonyms and POS tagging:

three recorded words: A, B and X

- A and X : similar meaning / same POS tags
- B has : different meaning / POS tags

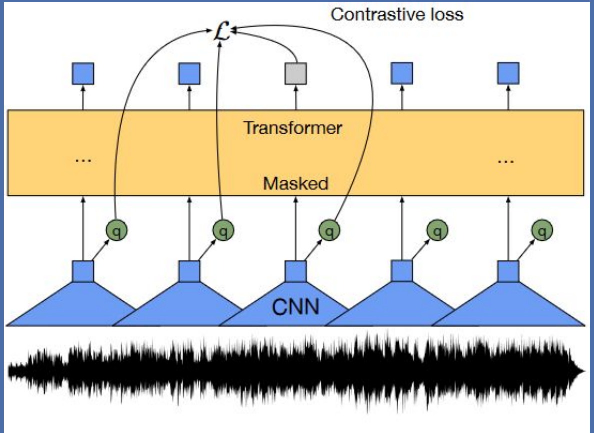
Zero-shot performances of speech SSL models ?

ABX test on synonyms and POS tagging:

three recorded words: A, B and X
- A and X : similar meaning / same POS tags
- B has : different meaning / POS tags

Compute the embedding of A B and X (mean pool the frames out of the transformer stack)

Correct if dist(A,X)<dist(B,X)



Zero-shot performances of speech SSL models ?

Input Representations	Segments	SSE	BERT	Dev Set $ABX_{sem} \uparrow$	Dev Set $ABX_{POS} \uparrow$	Test Set $ABX_{sem} \uparrow$	Test Set $ABX_{POS} \uparrow$	Dev Set sSIMI ↑	Test Set sSIMI ↑
	No supervision, frame based								
CPC+	–	–	–	51.22 [2]	53.67 [2]	53.86 [2]	54.68 [2]	**6.14 [2]**	4.34 [2]
W2V2 base×	–	–	–	54.96 [8]	55.78 [8]	56.20 [8]	56.86 [8]	3.85 [9]	5.21 [9]
HuBERT base‡	–	–	–	54.65 [8]	55.65 [8]	55.12 [8]	56.77 [8]	3.53 [8]	0.68 [8]
W2V2 large×	–	–	–	57.69 [13]	**59.86 [13]**	58.88 [13]	**60.93 [13]**	2.04 [11]	**6.82 [11]**
HuBERT large‡	–	–	–	**58.04 [23]**	57.77 [23]	**58.95 [23]**	57.19 [23]	4.14 [7]	0.55 [7]
	Full supervision, segment based								
text	gold words	1-hot	✓	**83.23 [3]**	**81.07 [3]**	**84.12 [3]**	**80.09 [3]**	**41.78 [1]**	**36.83 [1]**

SSL models are not just acoustic models, they are speech language models

more on this next time…

DP-Parse: Finding Word Boundaries from Raw Speech with an Instance Lexicon. Algayres et al

That was “pre-training” and “self-training” for semi-supervised ASR

Both are complementary

Vocabulary:

Self-Supervised Learning (SSL) and Unsupervised Learning:
    learning on raw data without labels

Self-training:
    supervised learning, then pseudo-labelling then supervised training

Pre-training:
    training a model on unannotated data with the idea of doing finetuning later

Contrastive loss:
    cross entropy with softmax and importance sampling

Semi-supervised learning:
    methods to leverage unannotated data for ASR

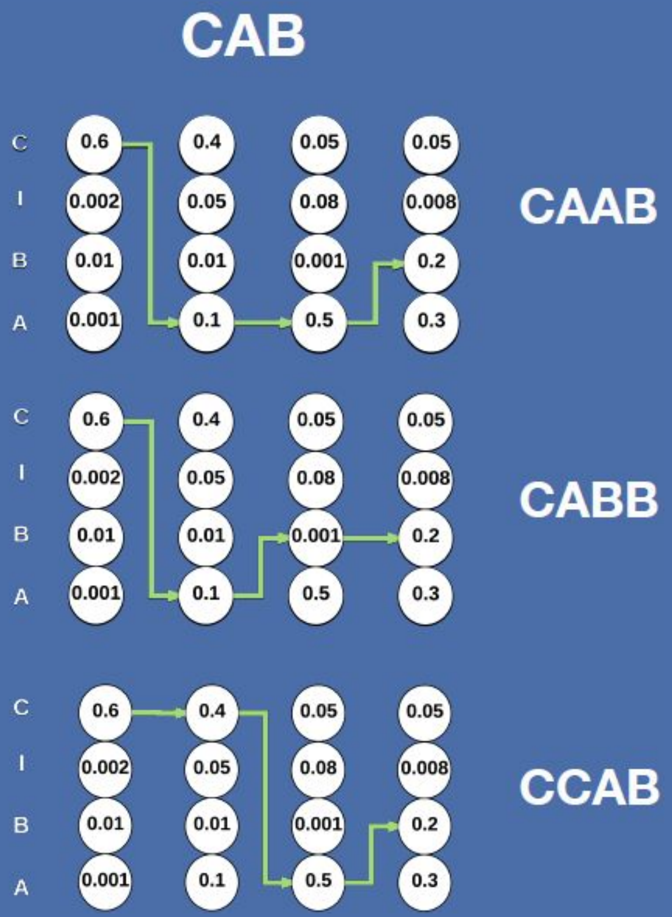
2 - Three losses for ASR:

Connectionist Temporal Classification (CTC)
Listen Attend and Spell (LAS)
RNN Transducer (RNN-T)

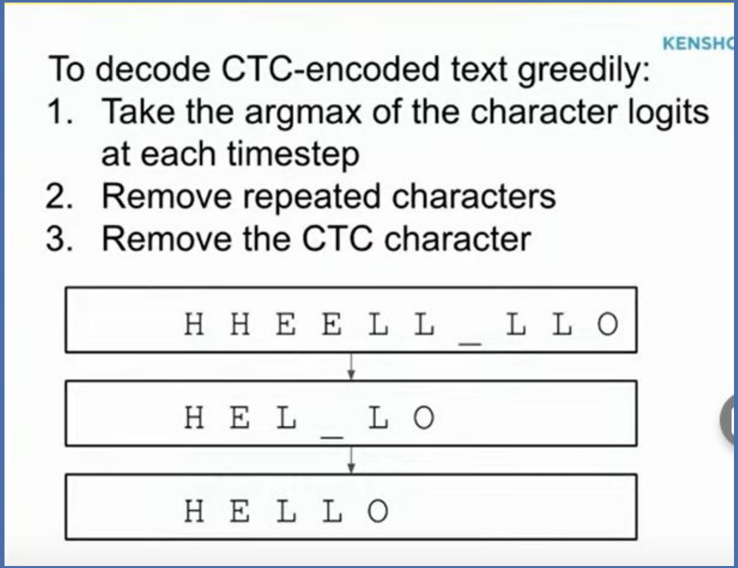
CTC Training

- Loss function:

$$-\sum_{\pi \in \text{Valid paths}} P(\pi|x)$$



CTC with greedy-decoding



But we want to use a LM to improve decoding

CTC decoding with LM and exact Search:

1- get the probability of all possible paths
2- multiply each of them by their probability under the LM
3- get the best one

Good but too long

N frames with K output symbols: $O(K^N)$

Beam Search: approximation of Exact Search

Idea: exact search but keep only the B best paths

If B=1, Greedy Search
if $B=K^N$, Exact Search

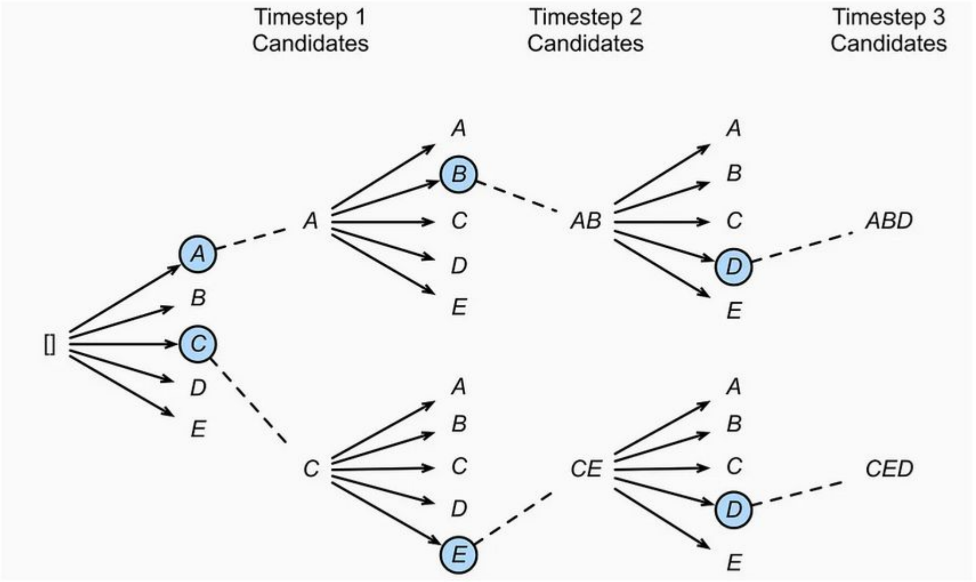


Figure 2: Beam Search with BeamWidth=2 (Source)

Beam Search: approximation of Exact Search

Idea: exact search but keep only the B best paths

If B=1, Greedy Search
if $B=K^N$, Exact Search

At time t, holding B paths in memory:
for each path:

> compute the probability of including each letter in the current path under both CTC and the LM

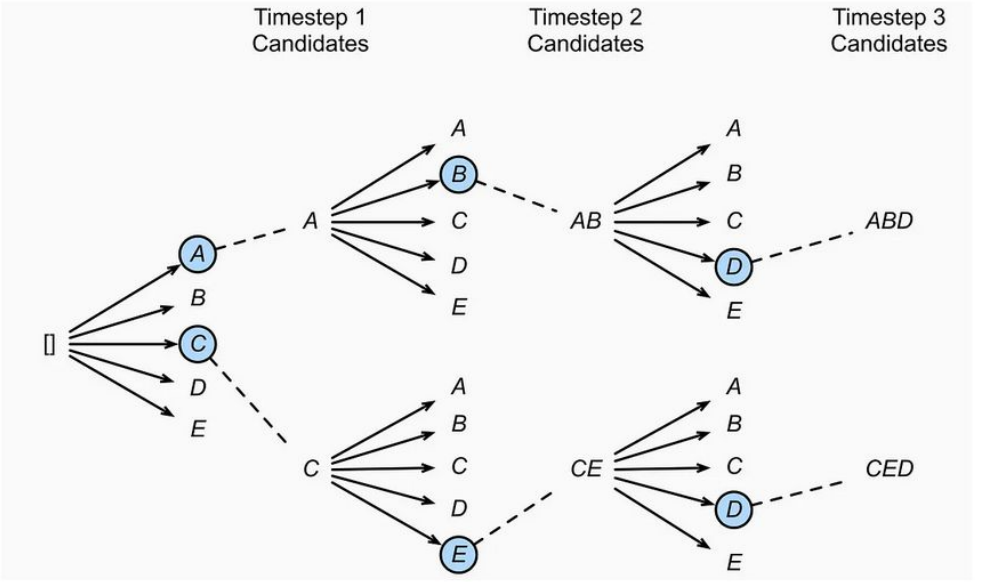


Figure 2: Beam Search with BeamWidth=2 (Source)

Beam Search: approximation of Exact Search

Idea: exact search but keep only the B best paths

If B=1, Greedy Search
if $B=K^N$, Exact Search

At time t, holding B paths in memory:
for each path:

> compute the probability of including each letter in the current path under both CTC and the LM

Sort the path by probability and keep only the B best paths

Complexity: O(N*B*K*log(B*K))

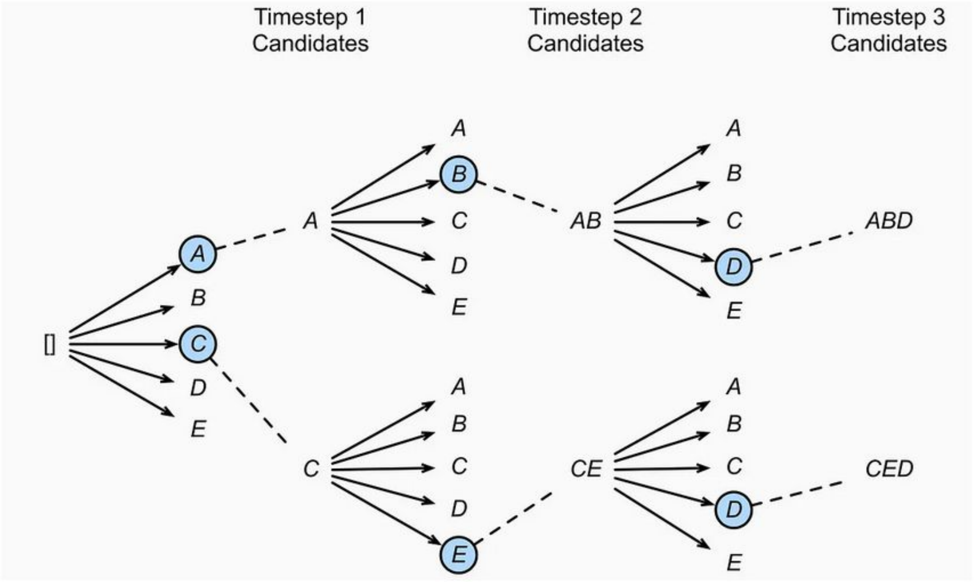


Figure 2: Beam Search with BeamWidth=2 (Source)

Other methods than CTC

CTC needs an external LM

Two other methods LAS and RNN-T:

- integrate the LM concept into the model
- better than CTC without LM
- good for low resource languages when both speech and text are scarce

Listen Attend and Spell (LAS)

Pros: Good for speech translation
Cons: hallucination

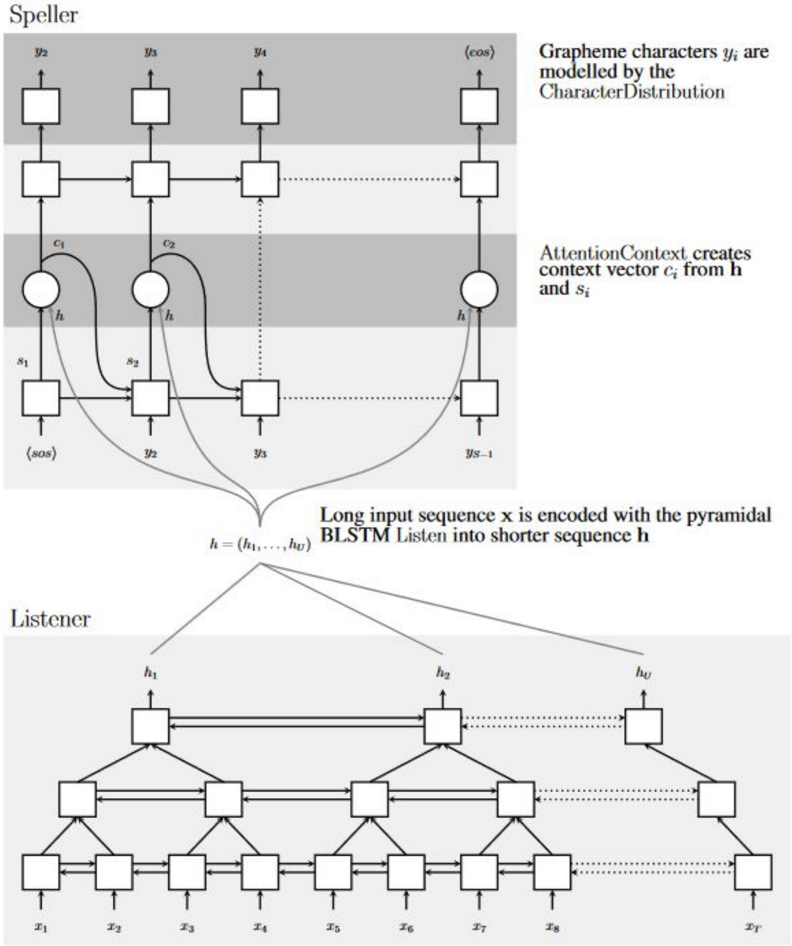
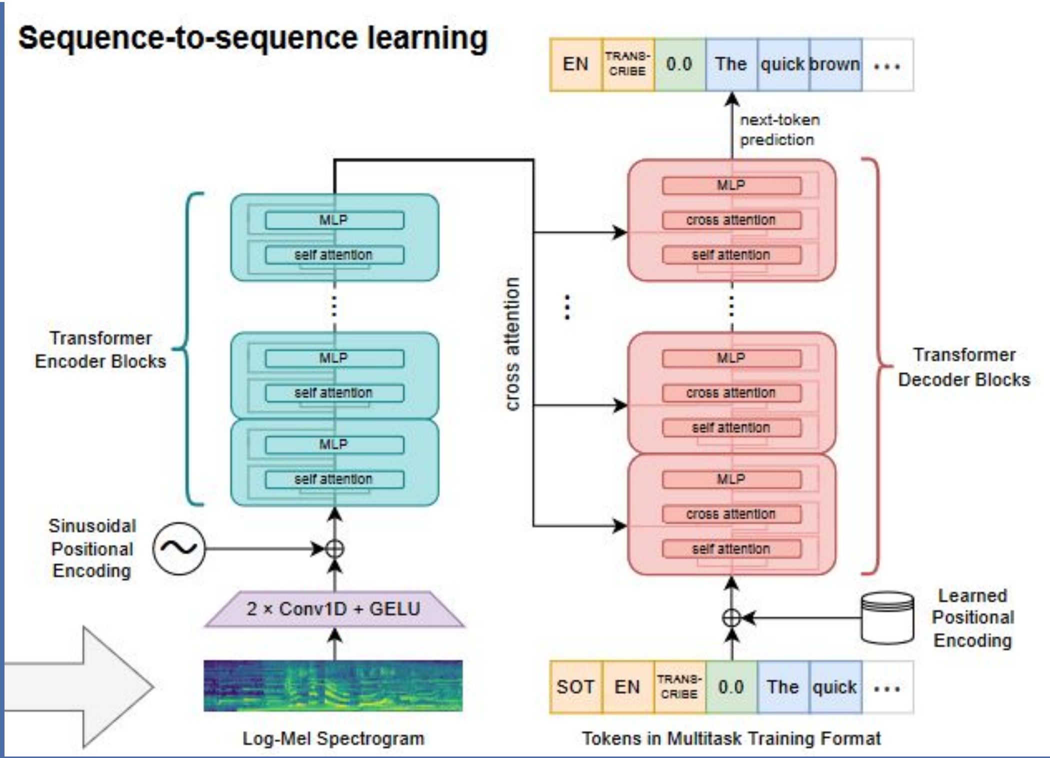


Figure 1: Listen, Attend and Spell (LAS) model: the listener is a pyramidal BLSTM encoding our input sequence **x** into high level features **h**, the speller is an attention-based decoder generating the **y** characters from **h**.

Whisper: recent LAS model

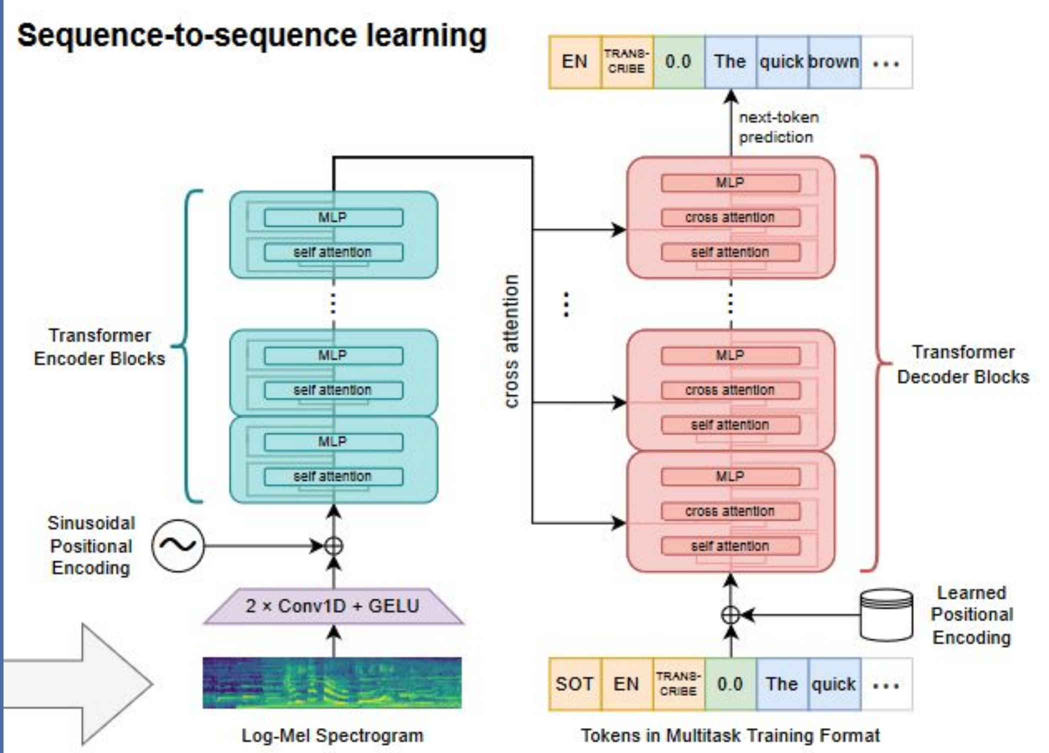
-Trained with word-level LAS
-no external LM ! simplest architecture
-no unsupervised pretraining/self-training
-use task-specific tokens
-multi lingual ASR
-any-to-english translation



Whisper: recent LAS model

-database collection:
- -600k hours of multi-lingual (96 languages) annotated speech
- -removed automatically generated transcripts

(general rule of recent large scale models: it doesn't matter if the labels are noisy just get a lot of it and apply coarse filter on it)



Whisper: very high ASR performances

Dataset	wav2vec 2.0 Large (no LM)	Whisper Large V2	RER (%)
LibriSpeech Clean	**2.7**	**2.7**	0.0
Artie	24.5	**6.2**	74.7
Common Voice	29.9	**9.0**	69.9
Fleurs En	14.6	**4.4**	69.9
Tedlium	10.5	**4.0**	61.9
CHiME6	65.8	**25.5**	61.2
VoxPopuli En	17.9	**7.3**	59.2
CORAAL	35.6	**16.2**	54.5
AMI IHM	37.0	**16.9**	54.3
Switchboard	28.3	**13.8**	51.2
CallHome	34.8	**17.6**	49.4
WSJ	7.7	**3.9**	49.4
AMI SDM1	67.6	**36.4**	46.2
LibriSpeech Other	6.2	**5.2**	16.1
Average	29.3	**12.8**	55.2

Strong hallucination exemple

WHISPER doesn't really know what to do with silence…

Human transcribed
I'm ok
and maybe I xxx
yes
ta-da
right leg right
xxx lizard
my xxx hers
dinosaur

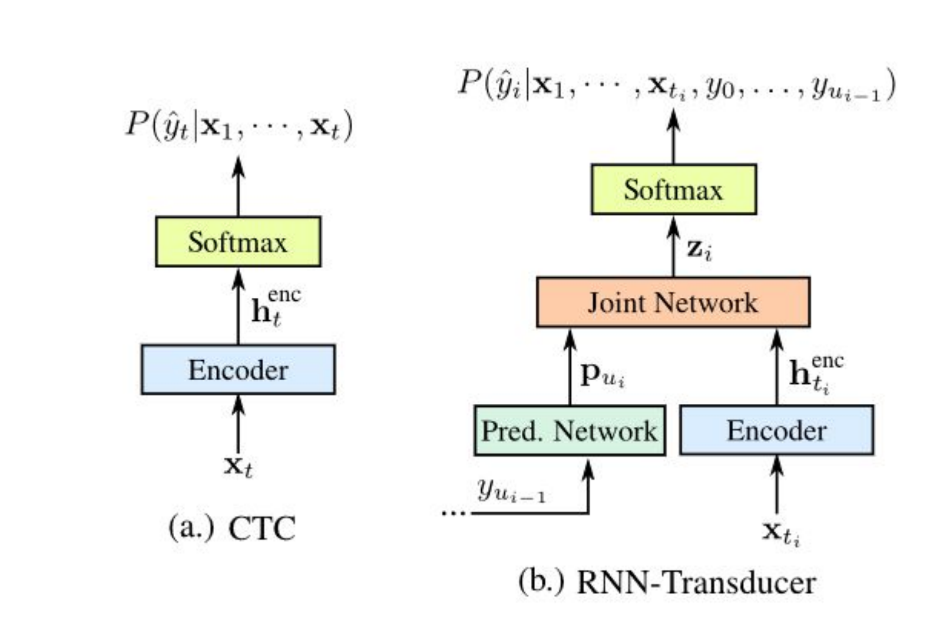
whisper transcribed
Now you're making fun of me? It's me, the dinosaur, who's going to get the prize. Whoa! What in the world? You're the dinosaur who's going to get the prize? Hey, hey, hey, hey, hey. What are you talking about? He can do just about anything he wants. Oh, yeah? Did you hear what Mr. Rabbit told us today? That dinosaurs are the collectibles of the universe. Now, don't you dare to tell me that only dinosaurs can win the prize. You know what you might learn? Dinosaur are not dinosaurs. Me and Mr. Rabbit are dinosaurs. We can do anything we want, Doctor. And he knows we can do everything we want. Oh, yeah? We can do anything we want. Oh, yeah? Yeah, that's why you can pretty much give up right now. Yeah, but I'm still a dinosaur. You can't prevent me from winning the prize. Hey, Mr. Rabbit. He's a cheater. Hey! Why don't you just ignore his manhood and I'm going to win the most prize? What do you know? We can do anything we want. Hey!

slide by Marvin Lavechin

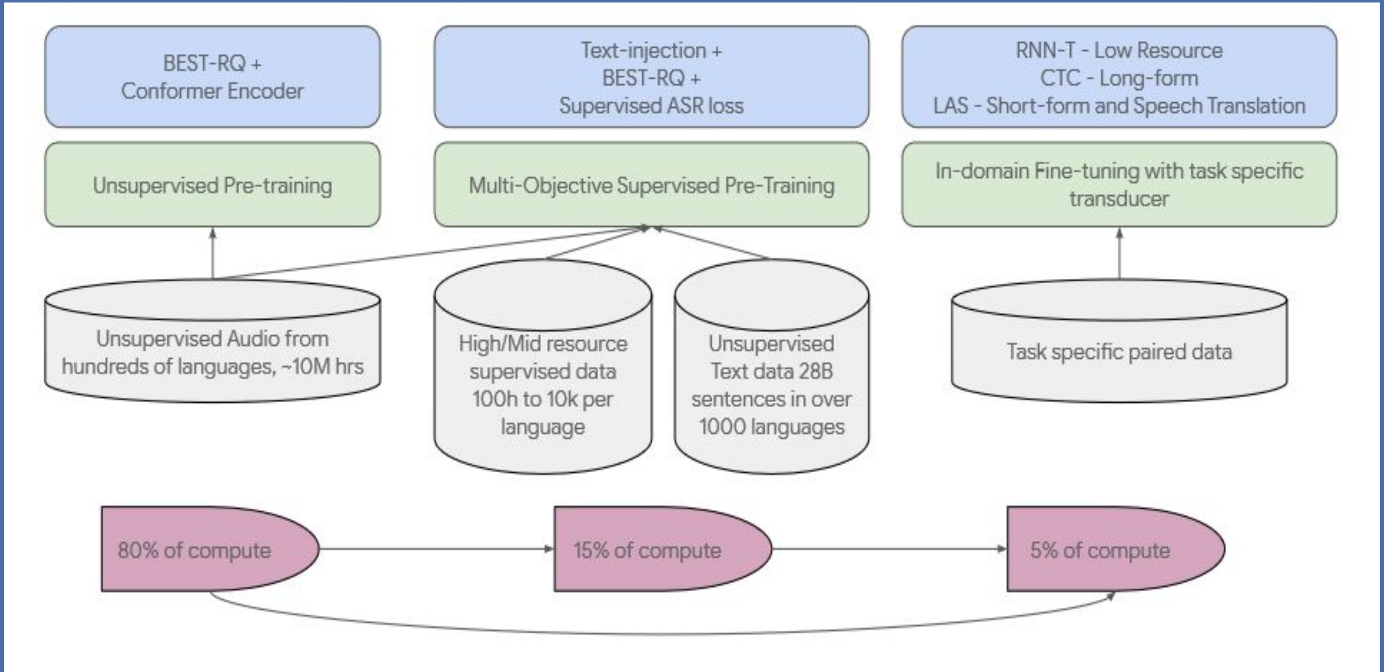
RNN Transducer (RNN-T)

trained to predict letters conditioning on all previously predicted letters
blanks and repetitions are removed before being fed into the pred. network

less hallucination problem



Google USM: *everything everywhere all at once*



Google USM: performances

Database:

-12 M hours of youtube over 300 languages (pretraining)

-100k of annotated speech over 100 languages

Language specific heads (encoder is frozen during finetuning)

Multi-lingual ASR and Speech Translation (not only towards english!)

Overall better performances than Whisper

3 - Long form prediction

Most models (Whisper, Google USM,Wav2vec2,...) are trained on short speech clips (<30seconds)

what about a 1 hour long speech clip?

3 - Long form prediction

Most models (Whisper, Google USM,Wav2vec2,...) are trained on short speech clips (<30seconds)

what about a 1 hour long speech clip?

Attention mechanics in Transformer:

- quadratic with input length
- performs badly on sequences longer than seen during training

3 - Long form prediction

Most models (Whisper, Google USM,Wav2vec2,...) are trained on short speech clips (<30seconds)

what about a 1 hour long speech clip?

Attention mechanics in Transformer:

- quadratic with input length
- performs badly on sequences longer than seen during training

Main idea: reduce the work of attention to a smaller window

Google USM: long form prediction with chunk attention

local attention:

- restrict each attention to its k neighbors
- bad because the receptive increases through the layers

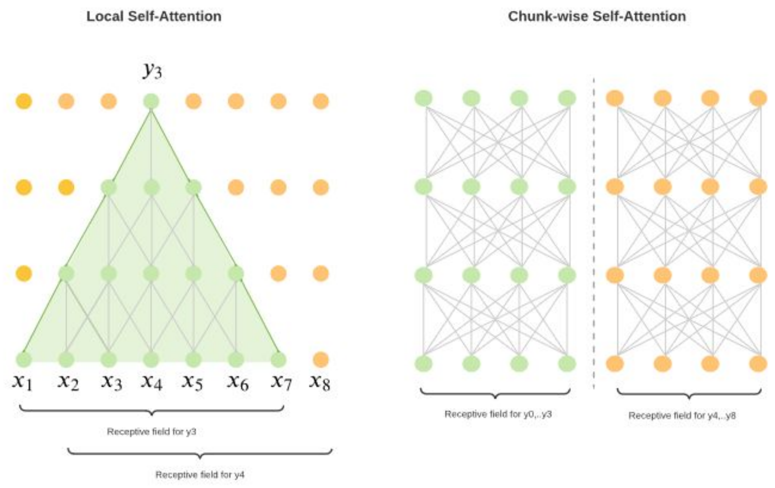


Figure 4: Comparing receptive fields of two networks with 4 layers of local self attention and chunk-wise attention.

Google USM: long form prediction with chunk attention

local attention:

- restrict each attention to its k neighbors
- bad because the receptive increases through the layers

chunk-wise attention:

- attention can attend only 8s long chunks of speech
- not block processing: other layers have access to the whole sequence

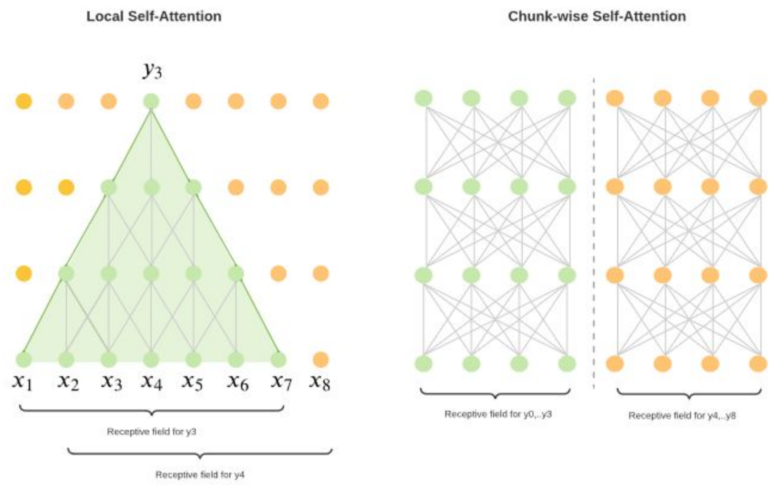


Figure 4: Comparing receptive fields of two networks with 4 layers of local self attention and chunk-wise attention.

Whisper: long form prediction with sliding window

Sliding 30s windows with adaptive sliding

…

Conclusion

HMM-GMM and HMM-DNN:

small annotated speech and lot of human knowledge

Deepspeech with CTC:

large annotated speech

Semi supervised learning:

large unannotated speech, small annotated speech

Zero ressource:

no annotated speech ?