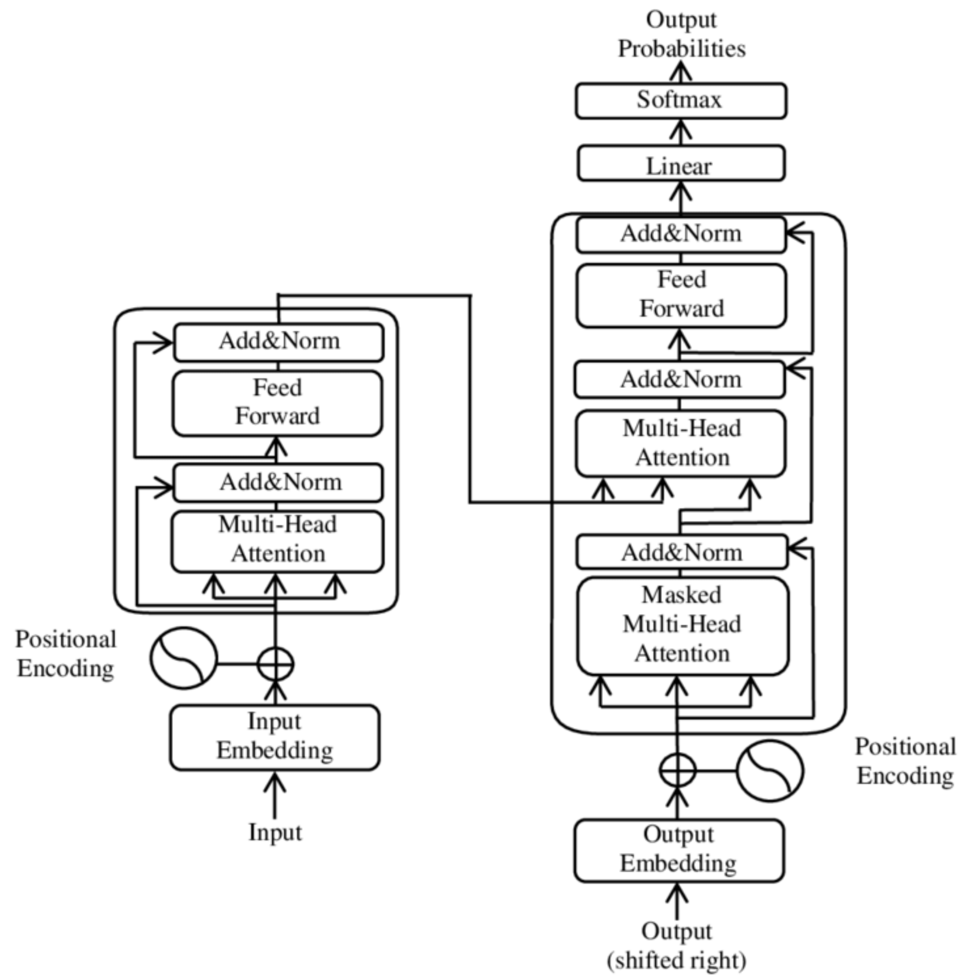
Lecture 24: Transformers

Mark Hasegawa-Johnson

3/2023



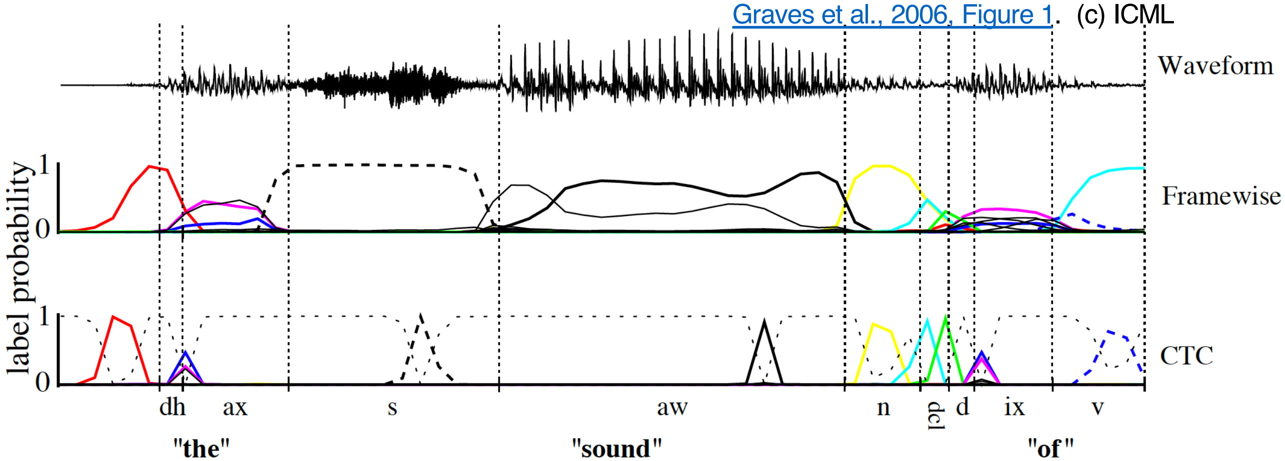
By Yuening Jia - DOI:10.1088/1742-6596/1314/1/012186, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=121340680

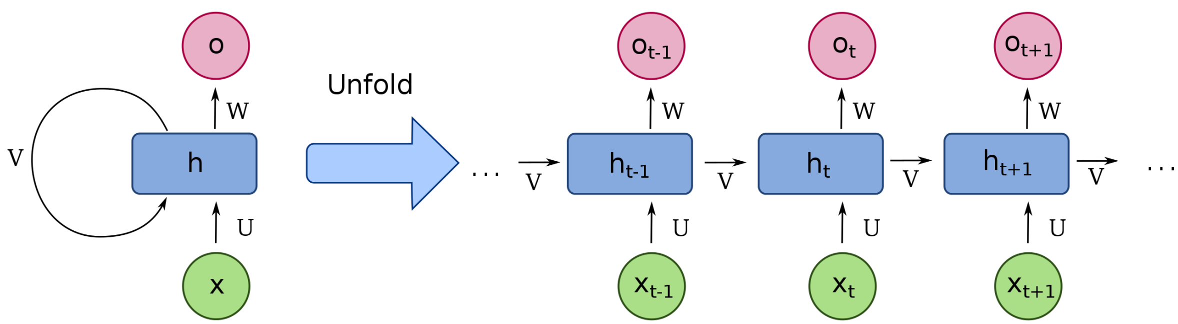
Outline

- Recurrent neural networks
- Attention
- Self-attention, Multi-headed attention, Cross-attention, and Masked attention
- Self-training

Recurrent neural network

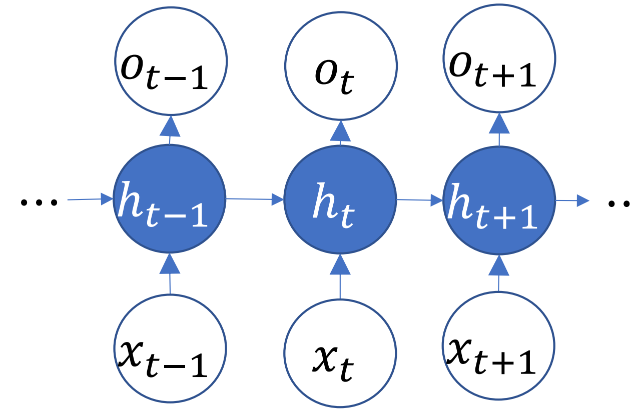
Graves et al., 2006, Figure 1. (c) ICML





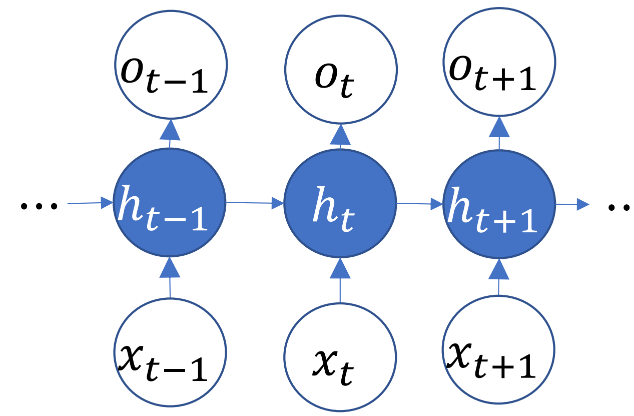
- In a recurrent neural network (RNN), the hidden node activation vector, h_t, depends on the value of the same vector at time $t-1$.
- From 2014-2017, the best speech recognition and machine translation used RNNs.
- The input is x_t=speech or input-language text
- The output is o_t=text in the target language

Example: Part of speech tagging



- x_t =vector representation of the t$^{\text{th}}$ word, e.g., trained using CBOW
- h_t =hidden state vector
- $o_t = \text{softmax}(h_t @ w) = [P(Y_t = \text{Noun} | X_1, \dots, X_t), P(Y_t = \text{Verb} | X_1, \dots, X_t), \dots]$

Training an RNN



An RNN is trained using gradient descent, just like any other neural network!

$$u_{j,i} \leftarrow u_{j,i} - \eta \frac{\partial \mathfrak{L}}{\partial u_{j,i}}$$

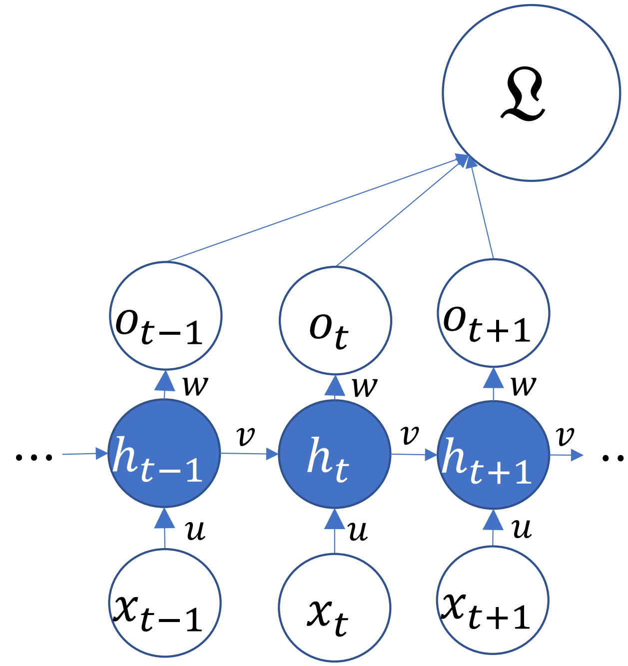
$$w_{j,k} \leftarrow w_{j,k} - \eta \frac{\partial \mathfrak{L}}{\partial w_{j,k}}$$

…where $\mathfrak{L}$ is the loss function, and η is a step size.

Training an RNN: Infinite recursion?

The big difference is that now the loss function depends on u, v and w in many different ways:

- The loss function depends on each of the state vectors h_t, which depends directly on u and v.
- But h_t also depends on h_{t-1}, which, in turn, depends on u and v.
- … and so on.

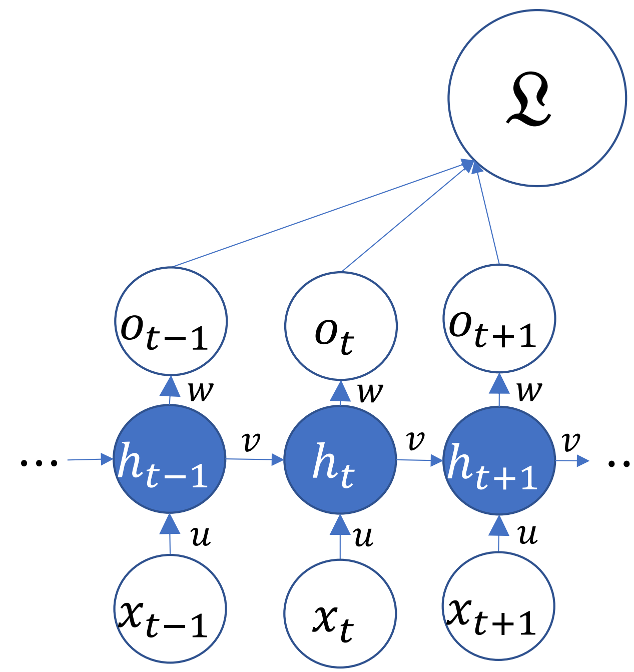


Back-propagation through time

The solution is something called back-propagation through time:

$$\frac{d\mathfrak{L}}{dh_{i,t}} = \frac{\partial\mathfrak{L}}{\partial h_{i,t}} + \sum_j \frac{d\mathfrak{L}}{dh_{j,t+1}} \frac{\partial h_{j,t+1}}{\partial h_{i,t}}$$

- The first term measures losses caused directly by $h_{i,t}$, for example, if $o_{i,t}$ is wrong.
- The second term measures losses caused indirectly, for example, because $h_{i,t}$ caused $h_{j,t+1}$ to be wrong.

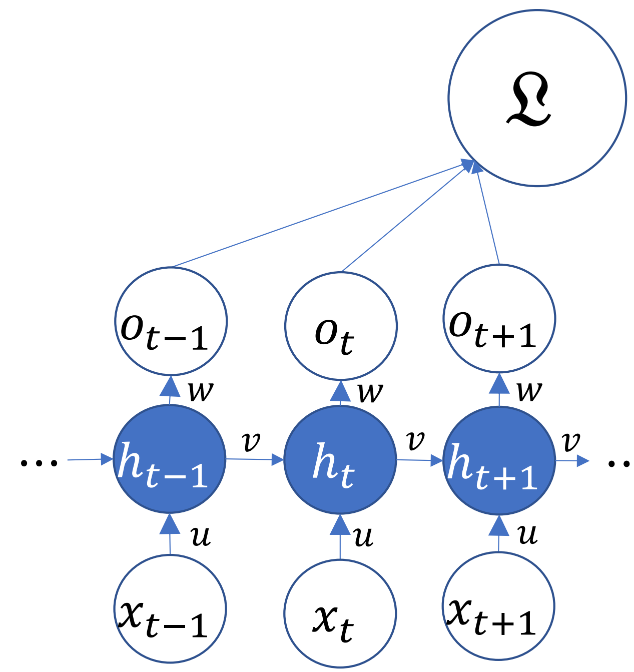


Back-propagation through time

Notice that this is just like training a very deep network!

- Back-propagation through time: back-propagate from time step $t + 1$ to time step t
- Back-propagation in a very deep network: back-propagate from layer $l + 1$ to layer l

Toolkits like PyTorch may use the same code in both cases.



Outline

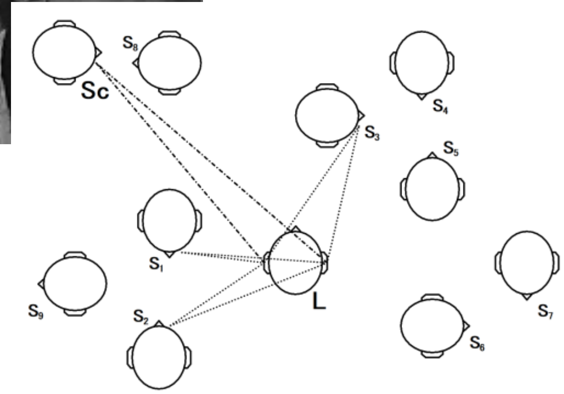
- Recurrent neural networks
- Attention
- Self-attention, Multi-headed attention, Cross-attention, and Masked attention
- Self-training

The Cocktail-Party Effect

- If you are focusing on one person's voice, but hear your name spoken by another person, your attention immediately shifts to the second voice.
- This "cocktail-party effect" suggests a model of hearing in which all sounds are processed preconsciously. Trigger sounds in an unattended source will cause attention to re-orient to that source.

https://commons.wikimedia.org/wiki/File:Cocktail_party_attendees_at_Fuller_Lodge,_1946.jpg

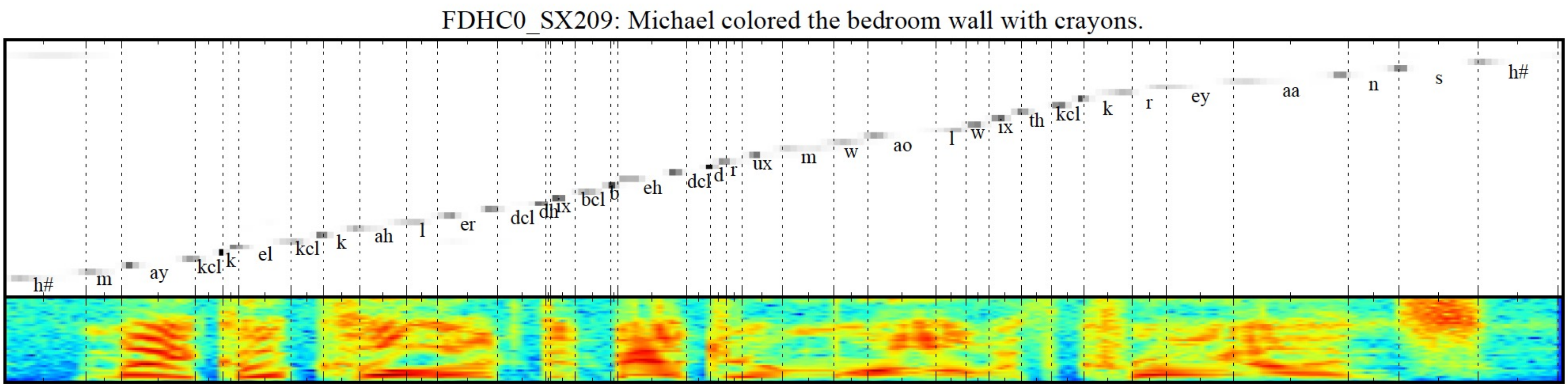
https://commons.wikimedia.org/wiki/File:Cocktail_Party_At_The_Imperial_Hotel_March_13,_1961_(Tokyo,_Japan)_(49 6610682).jpg



https://commons.wikimedia.org/wiki/File:Cocktail-party_effect.svg

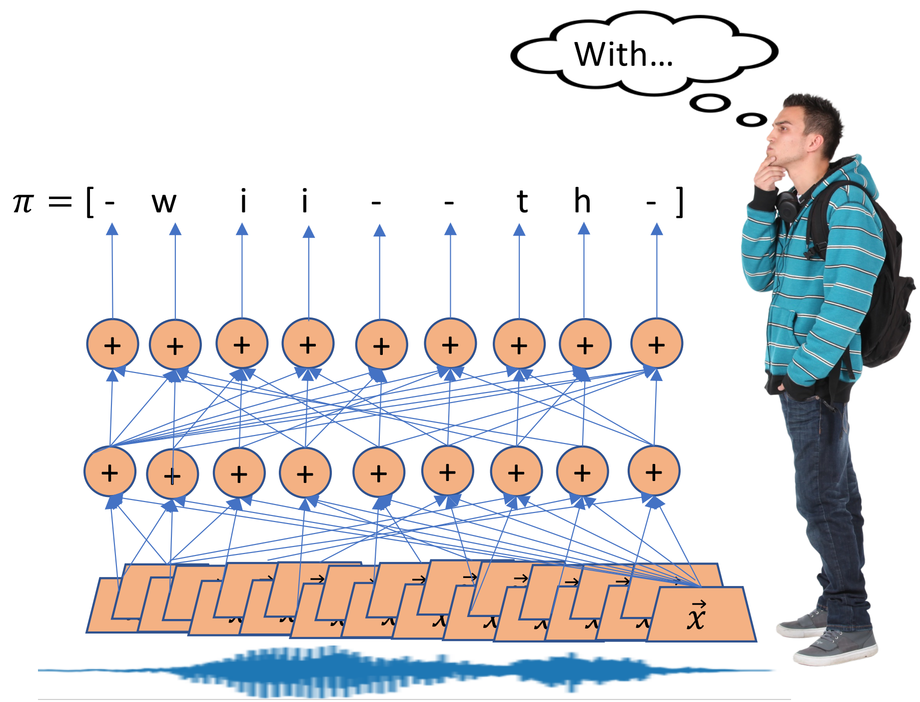
Bottom-up attention as a strategy for machine listening

- In 2014, researchers proposed that the past 200ms of RNN state vectors should be stored in a “short-term memory buffer”
- A speech recognizer can attend to several centiseconds, all at one time, to decide what words it thinks it is hearing



Chorowski, Bahdanau, Serdyk, Cho & Bengio, [Attention-Based Models for Speech Recognition](), Fig. 1

The Transformer: “Attention is all you need”



- In 2017, researchers proposed that the short-term memory buffer should contain raw signals, not processed signals.
- All processing is done using a model of bottom-up attention.

Attention: Key concepts

- The neural net needs to make a series of decisions, o_i
- Each decision needs to be based on some context, c_i
- Each context vector is a weighted sum of input values, $c_i = \sum_t \alpha_{i,t} v_t$
- $\alpha_{i,t}$ is the amount of attention that the output decision o_i is paying to the input value v_t. It is based on the similarity between a key vector, k_t, that describes the type of information available in v_t, and a query vector, q_i, that describes the type of information necessary in order to make the output decision

Inputs to an attention network

- Neural net inputs: a sequence of row vectors, x_t
- Neural net outputs: a sequence of row vectors, o_i
- Value: What type of information should x_t provide to the output? This may be just a linear transform of x_t, e.g.: $v_t = x_t @ w_V$
- Query: What type of information does o_i need? This may be just a linear transform of o_{i-1}, e.g.: $q_i = o_{i-1} @ w_Q$
- Key: The dot product $q_i @ k_t$ should be positive if v_t is useful, and negative if v_t is useless. This may be $k_t = x_t @ w_K$

Attention = a probability mass over time

- Attention is like probability: You only have a fixed amount of attention, so you need to decide how to distribute it.
- $\alpha_{i,t} = P(v_t|q_i)$ = the probability that v_t is the context that you need in order to make a decision related to the query vector q_i.

$$\sum_t \alpha_{i,t} = 1$$

- Each output context vector (c_i) is based on some input value vectors (h_t). But which ones? Answer: decide which inputs to pay attention to, then pay attention.

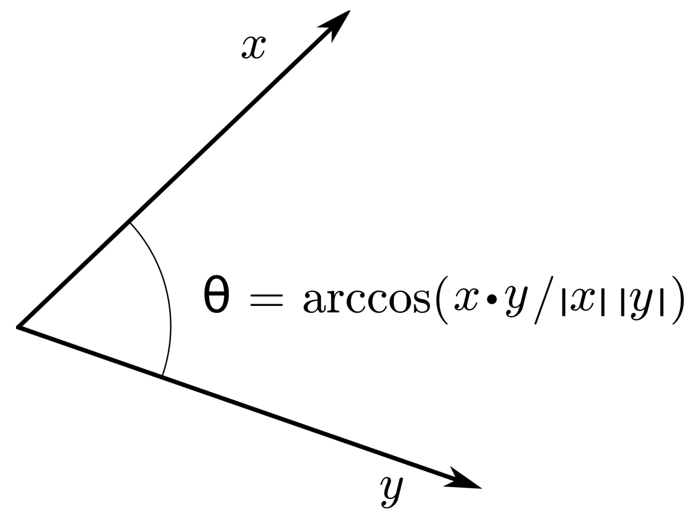
$$c_i = \sum_t \alpha_{i,t} v_t$$

Dot-product attention

How can you decide which value vectors, v_t are most relevant to a particular query? Answer:

1. Create a key vector, k_t, such that $q_i@k_t > 0$ if v_t is relevant to q_i, otherwise $q_i@k_t < 0$.
2. Convert the similarity measures into a probability distribution using softmax:

$$\alpha_{i,t} = \frac{\exp(q_i@k_t)}{\sum_\tau \exp(q_i@k_\tau)}$$



Putting it all together

- Stack up v_t, k_t, and q_i into matrices:

$$v = \begin{bmatrix} v_1 \\ \vdots \\ v_n \end{bmatrix}, k = \begin{bmatrix} k_1 \\ \vdots \\ k_n \end{bmatrix}, q = \begin{bmatrix} q_1 \\ \vdots \\ q_m \end{bmatrix}$$

- $\alpha_{i,t}$ is the t^th output of a softmax whose input vector is $q_i@k^T$:

$$\alpha_{i,t} = \mathrm{softmax}_t(q_i@k^T) = \frac{\exp(q_i@k_t)}{\sum_\tau \exp(q_i@k_\tau)}$$

- c_i is the product of the vector $\mathrm{softmax}(q_i@k^T)$ times the v matrix:

$$c_i = \mathrm{softmax}(q_i@k^T)@v = \sum_t \alpha_{i,t}\, v_t$$

Quiz!

- Try the quiz! https://us.prairielearn.com/pl/course_instance/129874/assessment/2337906

$$\exp(q_i @ k^T) = [0,0,0,1,0,0,1]$$
$$\alpha_{i,t} = \mathrm{softmax}(q_i @ k^T) = [0,0,0,0.5,0,0,0.5]$$

Outline

- Recurrent neural networks
- Attention
- Self-attention, Multi-headed attention, Cross-attention, and Masked attention
- Self-training

Self-attention

Self-attention (literally!) adds context to each input vector:

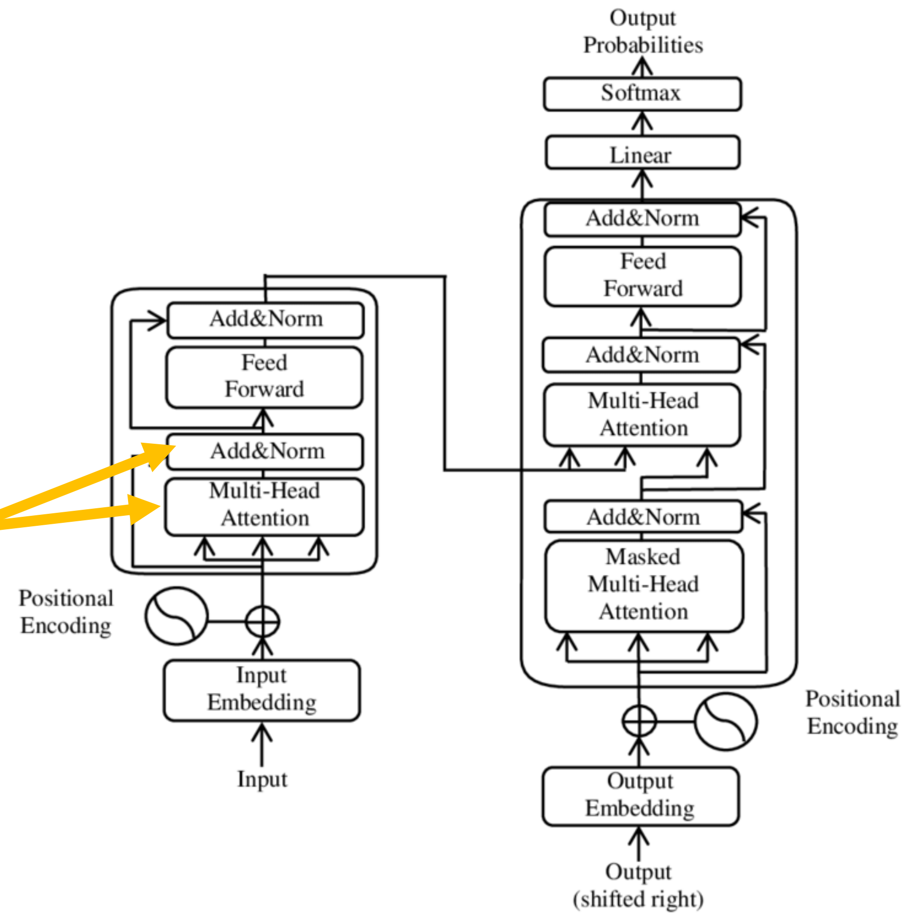
$$q_i = x_i @ w_Q$$

$$k_t = x_t @ w_K$$

$$v_t = x_t @ w_V$$

$$c_i = \mathrm{softmax}(q_i @ k^T) @ v$$

$$y_i = x_i + c_i$$



Multi-headed-attention

Multi-headed-attention uses 8 different w_Q, w_K, and w_V matrices, in order to get 8 different views of the input data:

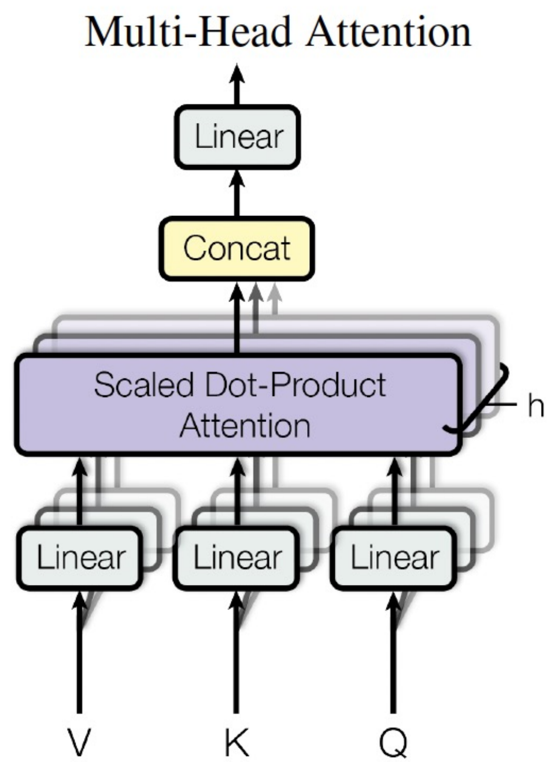
$$q_{j,i} = x_i @ w_{j,Q}, \qquad 1 \leq j \leq 8$$

$$k_{j,t} = x_t @ w_{j,K}, \qquad 1 \leq j \leq 8$$

$$v_{j,t} = x_t @ w_{j,V}, \qquad 1 \leq j \leq 8$$

$$h_{j,i} = \mathrm{softmax}\left(q_{j,i} @ k_j^T\right) @ v_j$$

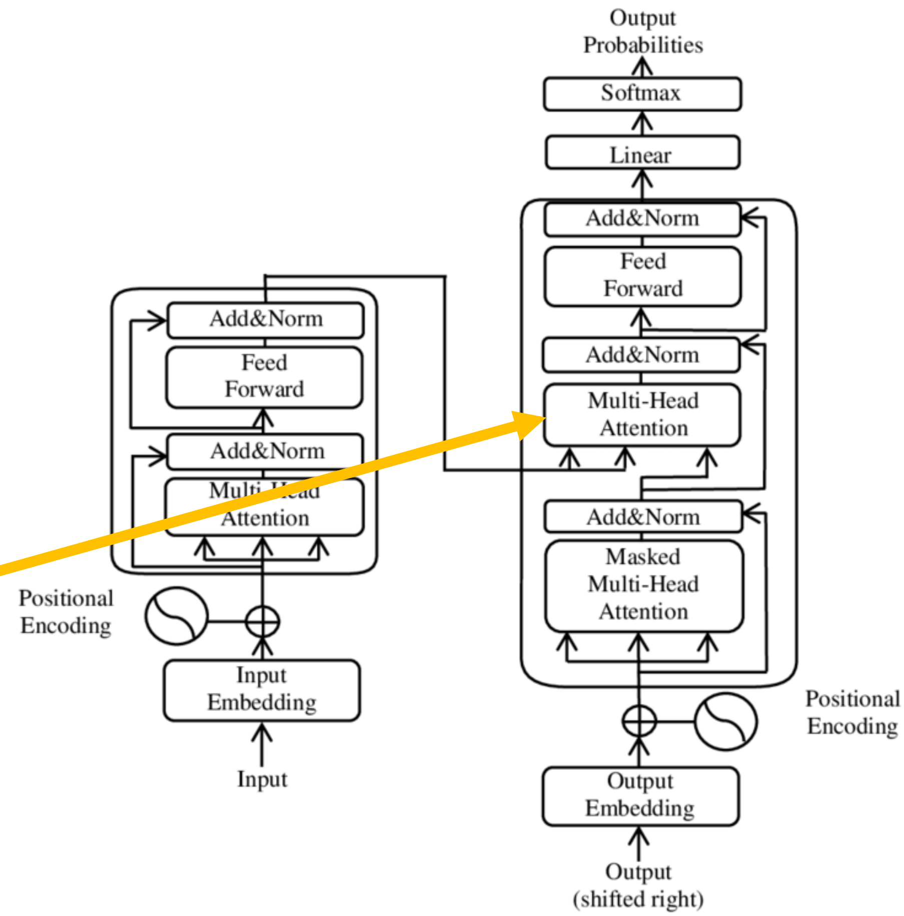
$$c_i = \left[h_{1,i}, \dots, h_{8,i}\right] @ w_O$$



Cross-attention

Cross-attention: query depends on preceding output, key and value depend on input:

$$q_i = o_{i-1} @ w_Q$$
$$k_t = x_t @ w_K$$
$$v_t = x_t @ w_V$$
$$c_i = \text{softmax}(q_i @ k^T) @ v$$



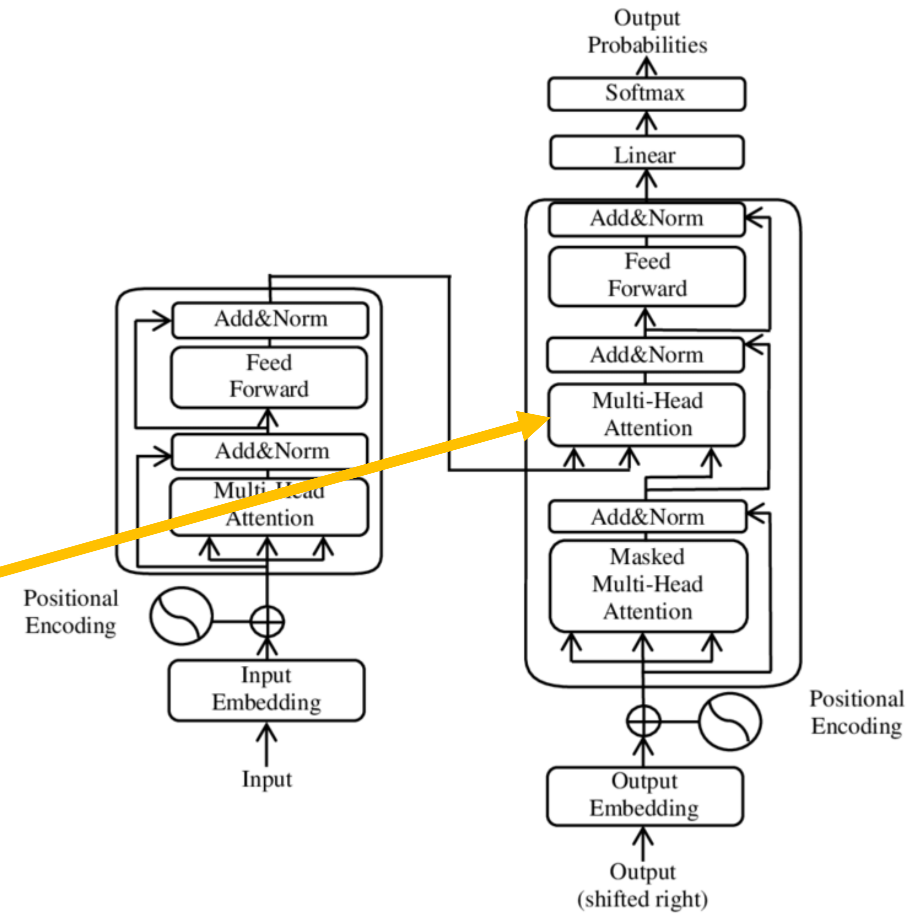
Masked attention

Masked attention forces c_i to pay attention to value vectors v_t only if $t < i$:

$$s(q_i, k_t) = \begin{cases} q_i @ k_t & t < i \\ -\infty & t \geq i \end{cases}$$

$$\alpha_{i,t} = \frac{\exp\left(s(q_i, k_t)\right)}{\sum_\tau \exp\left(s(q_i, k_\tau)\right)}$$

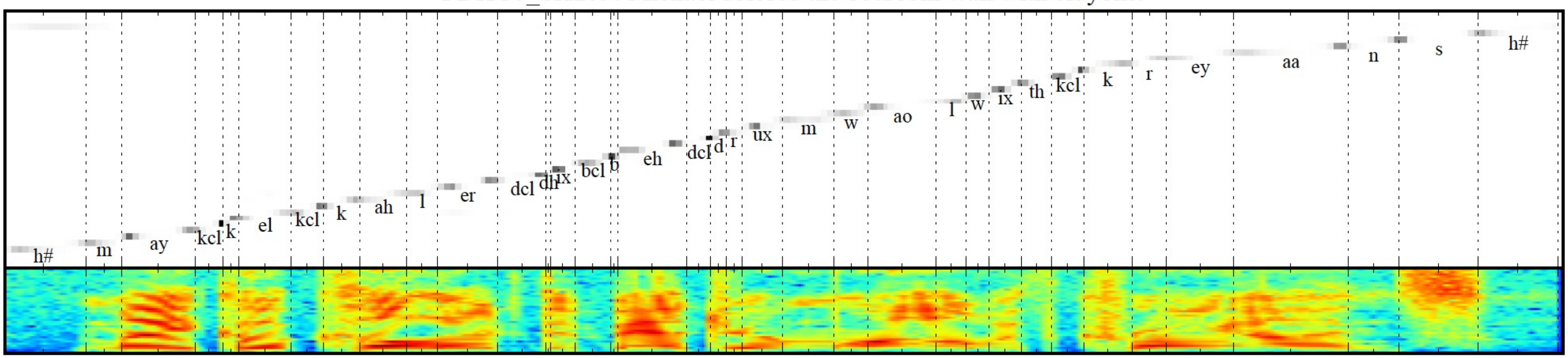
$$= \begin{cases} \text{softmax}(q_i @ k^T) & t < i \\ 0 & t \geq i \end{cases}$$



Cross-attention visualization

This plot shows $\alpha_{i,t}$ where i = output character, and t = input spectrum

FDHC0_SX209: Michael colored the bedroom wall with crayons.



Chorowski, Bahdanau, Serdyk, Cho & Bengio, [Attention-Based Models for Speech Recognition](), Fig. 1

Word Error Rates using Transformers

By 9/2020, transformers had error rates of:

- 2%: English, quiet recording conditions
- 4%: Chinese or Japanese, quiet recording conditions
- 5-7%: if the reference transcript has errors
- 14%: 2-talker mixtures, synthetic reverberation
- 38%: actual in-home recordings in noisy households

Table 1. CER/WER results on various open source ASR corpora. Both Transformer and Conformer models are implemented based on ESPnet toolkit. ∗ marks ESPnet2 results. † and ‡ indicate only w/ speed or only w/ SpecAugment, respectively. § denotes w/o any data augmentation.

Dataset	Vocab	Metric	Evaluation Sets	Transformer	Conformer
AIDATATANG	Char	CER	dev / test	(†) 5.9 / 6.7	**4.3 / 5.0**
AISHELL-1	Char	CER	dev / test	(†) 6.0 / 6.7	(*) **4.4 / 4.7**
AISHELL-2	Char	CER	android / ios / mic	(†) 8.9 / 7.5 / 8.6	**7.6 / 6.8 / 7.4**
AURORA4	Char	WER	dev_0330 (A / B / C / D)	**3.3 / 6.0 / 4.5** / 10.6	4.3 / 6.0 / 5.4 / **9.3**
CSJ	Char	CER	eval{1, 2, 3}	(∗) 4.7 / 3.7 / 3.9	(∗) **4.5 / 3.3 / 3.6**
CHiME4	Char	WER	{dt05, et05}_{simu, real}	(†) 9.6 / 8.2 / 15.7 / 14.5	**9.1 / 7.9 / 14.2 / 13.4**
Fisher-CallHome	BPE	WER	dev / dev2 / test / devtest / evltest	22.1 / 21.5 / 19.9 / 38.1 / 38.2	**21.5 / 21.1 / 19.4 / 37.4 / 37.5**
HKUST	Char	CER	dev	(†) 23.5	(†) **22.2**
JSUT	Char	CER	our split	(†) 18.7	**14.5**
LibriSpeech	BPE	WER	{dev, test}_{clean, other}	2.1 / 5.3 / 2.5 / 5.5	**1.9 / 4.9 / 2.1 / 4.9**
REVERB	Char	WER	et_{near, far}	(†) 13.1 / 15.4	(†) **10.5 / 13.9**
Switchboard	BPE	WER	eval2000 (callhm / swbd)	17.2 / 8.2	**14.0 / 6.8**
TEDLIUM2	BPE	WER	dev / test	9.3 / 8.1	**8.6 / 7.2**
TEDLIUM3	BPE	WER	dev / test	10.8 / 8.4	**9.6 / 7.6**
VoxForge	Char	CER	our split	(§) 9.4 / 9.1	(§) **8.7 / 8.2**
WSJ	BPE	WER	dev93/ eval92	(‡) **7.4 / 4.9**	(‡) 7.7 / 5.3
WSJ-2mix	Char	WER	tt	(§) 12.6	(§) **11.7**

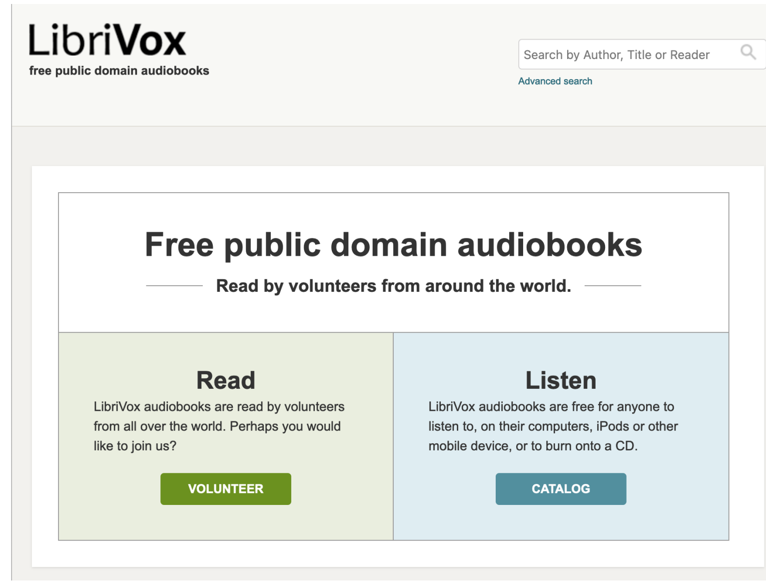
Guo, Boyer, Chang, Hayashi, Higuchi et al., ICASSP 2021,

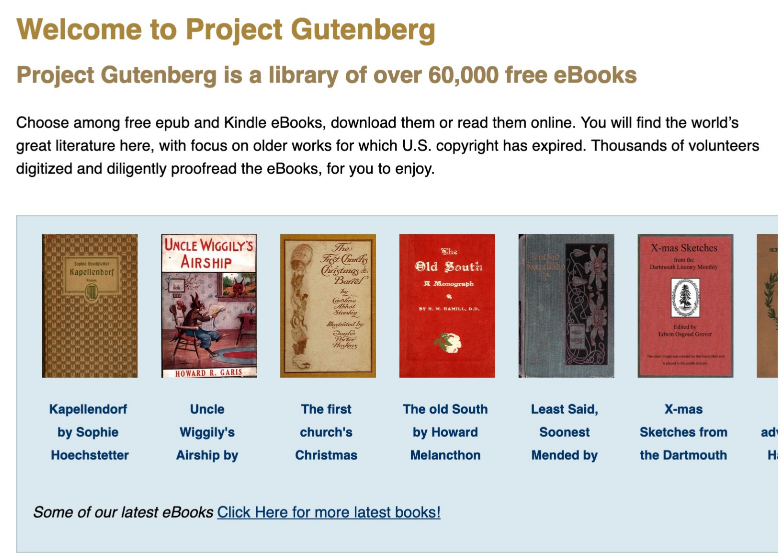
Outline

- Recurrent neural networks
- Attention
- Self-attention, Multi-headed attention, Cross-attention, and Masked attention
- Self-training

Label quality

- Audiobooks on librivox.org are readings of texts from Gutenberg.org.
- Often, readers make mistakes, or read different versions, or change the title enough to make it hard to know which Gutenberg text they read.
- Until 2020, speech technology was trained using “labeled data:”
  - 400 hours of librivox books with verified transcripts (“Librispeech Clean”)
  - 600 hours of librivox books with acoustic noise or transcript errors (“Librispeech Other”)





What do babies hear?

How much unlabeled speech does a baby hear?

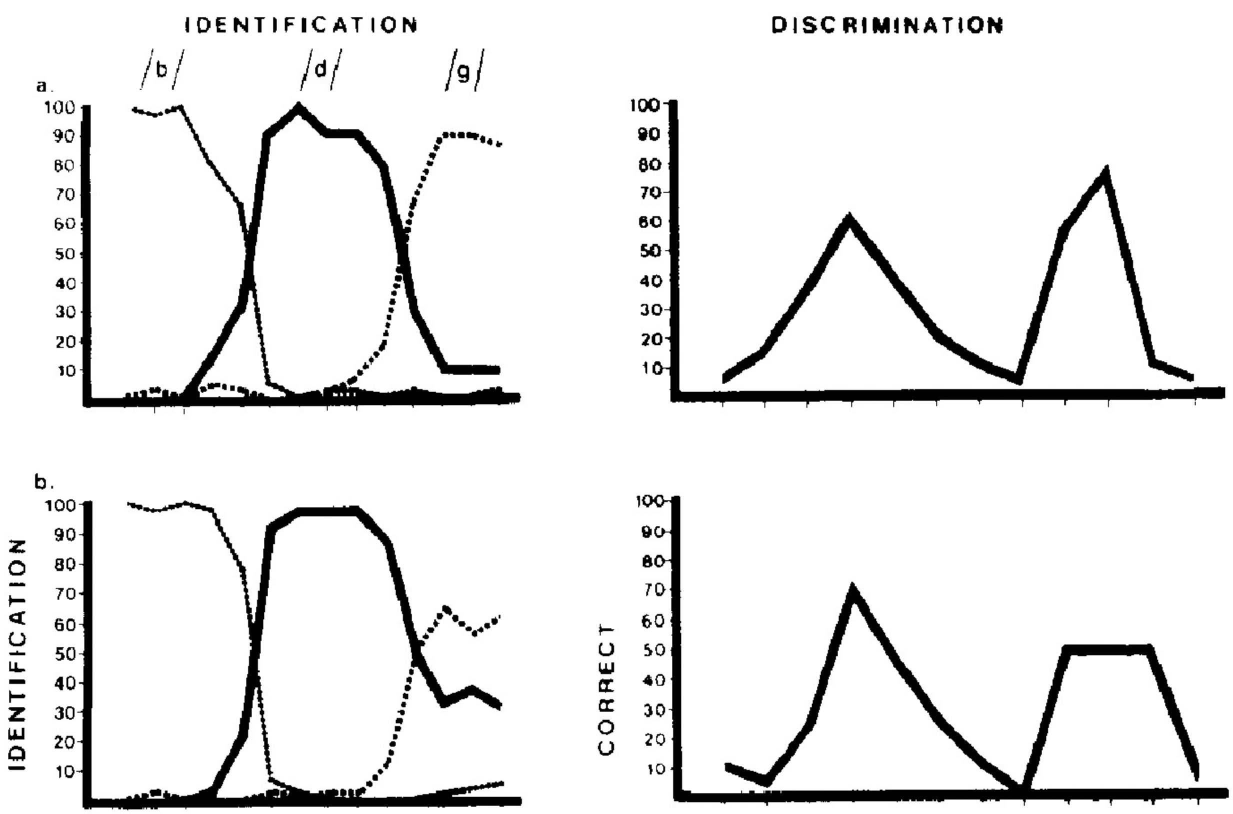
- 2000-15000 words/day = 600-4500 hours of speech by age 6 (Weisleder & Fernald, 2013)

How much labeled speech does a baby hear?

- 30 (?) words/day accompanied by referential gestures = 9.1 hours of speech by age 6

By Steve Jurvetson from Los Altos, USA - A Proper Space Book for Babies, CC BY 2.0, https://commons.wikimedia.org/w/index.php?curid=105132804

Is speech learned, or innate?
(Hint: it's a trick question)

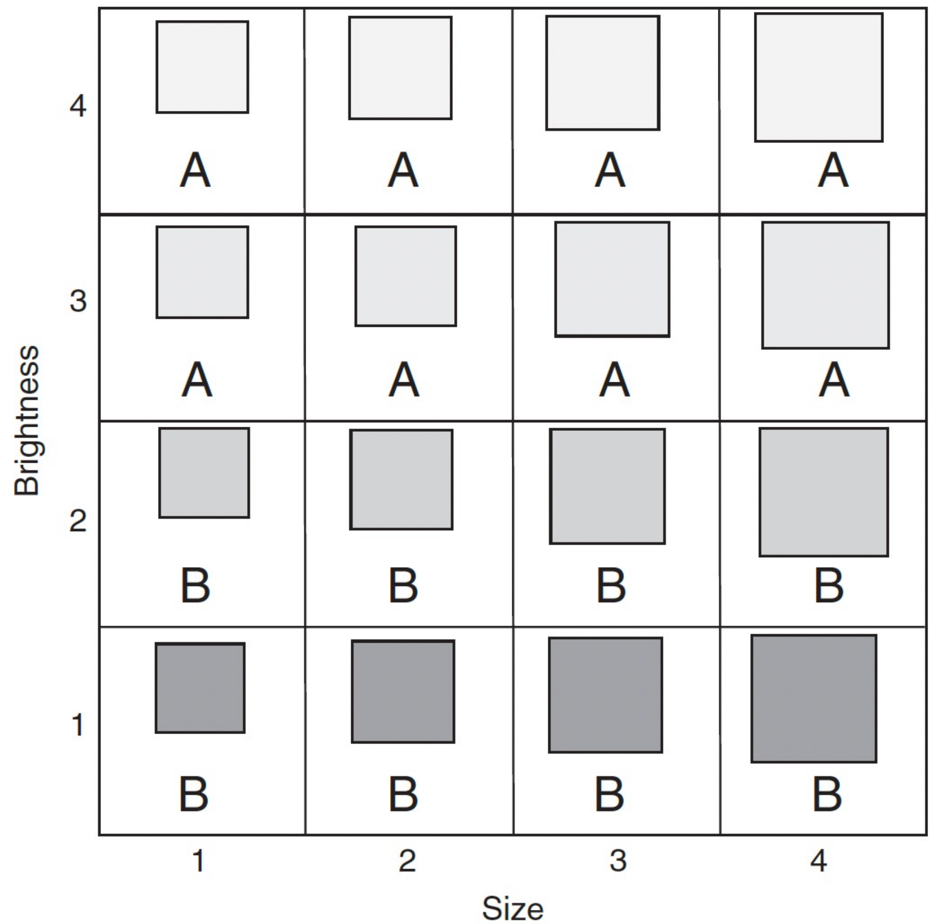


Brandt & Rosen, 1980

- 15 synthetic syllables, continuous from /ba/ to /da/ to /ga/
- same label ⇒ hard to tell if they are same sound or different sounds
- different labels ⇒ heard as obviously different

Categorical perception can be learned!

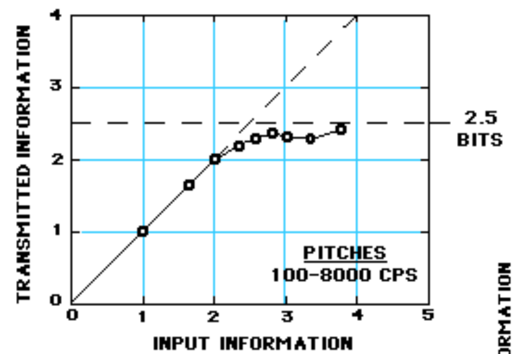
- People trained to categorize based on ***brightness*** show reduced within-category perceptual memory, and greater across-category perceptual memory, for ***brightness***, but ***size*** is perceived on a continuum.
- People trained to categorize based on ***size*** show reduced within-category perceptual memory, and greater across-category perceptual memory, for ***size***, but ***brightness*** is perceived on a continuum.

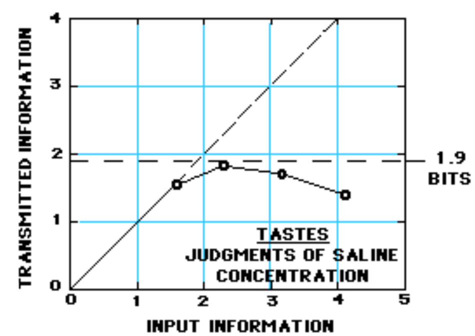


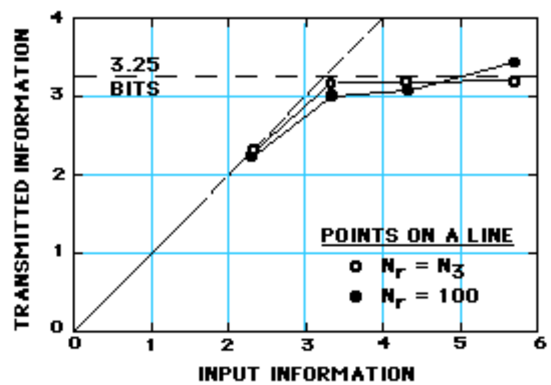
Goldstone & Hendrickson, 2009. © Wiley & Sons

Categorical perception as a cognitive bias

- If we categorize things, maybe we can remember them longer.
- In “The Magic Number Seven,” Miller argued that people can be taught to categorize any continuum (pitch, taste, position, size) into seven categories, but not more.

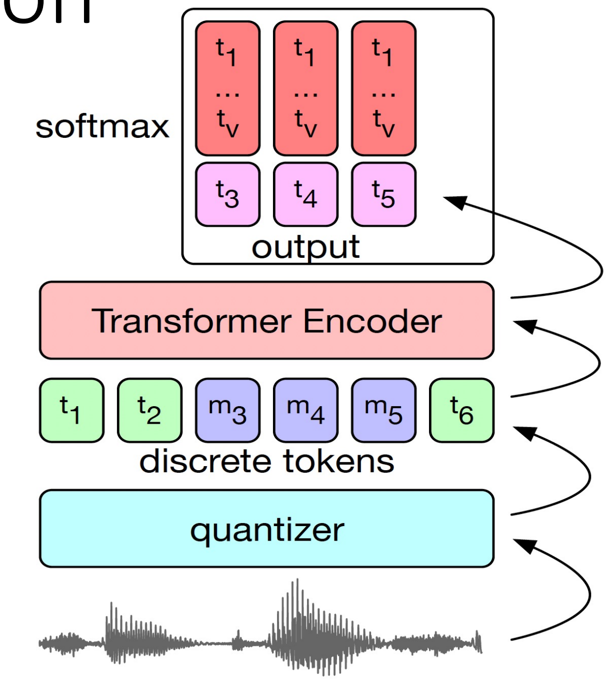






Unsupervised pre-training of transformers based on categorical perception

- Given: 60,000 hours of speech, with no associated text.
- Suppose we train the neural network to form its own categories. What would make those categories speech-like?
- **Context-Predictable Speech Categories**: given the context (the quantized units t_1, t_2, and t_6), it should be possible to figure out what phonemes were masked (t_3, t_4, t_5).



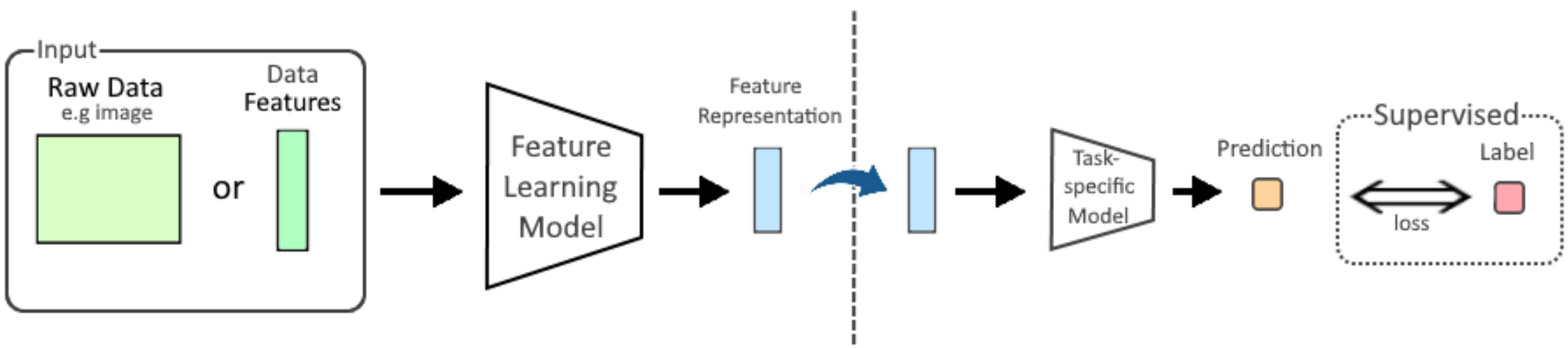
Baevski, Auli & Mohamed, 2019

Pre-training and Fine-tuning

- A transformer is pre-trained to create its own context-predictable speech categories using, say, 60,000 hours of speech
- Then it is fine-tuned using a few hours, or a few hundred hours, or labeled speech

Pre-training

Fine-tuning



Word Error Rates using Pre-Training

Pre-training makes it possible to achieve error rates of

- 4.4% using only 10 minutes of labeled data
- 2.6% using only 1 hour of labeled data

Model	Unlabeled Data	LM	dev-clean	dev-other	test-clean	test-other
		10-min labeled				
DiscreteBERT [52]	LS-960	4-gram	15.7	24.1	16.3	25.2
wav2vec 2.0 BASE [7]	LS-960	4-gram	8.9	15.7	9.1	15.6
wav2vec 2.0 LARGE [7]	LL-60k	4-gram	6.3	9.8	6.6	10.3
wav2vec 2.0 LARGE [7]	LL-60k	Transformer	4.6	7.9	4.8	8.2
HUBERT BASE	LS-960	4-gram	9.1	15.0	9.7	15.3
HUBERT LARGE	LL-60k	4-gram	6.1	9.4	6.6	10.1
HUBERT LARGE	LL-60k	Transformer	**4.3**	7.0	4.7	7.6
HUBERT X-LARGE	LL-60k	Transformer	4.4	**6.1**	**4.6**	**6.8**
		1-hour labeled				
DeCoAR 2.0 [51]	LS-960	4-gram	-	-	13.8	29.1
DiscreteBERT [52]	LS-960	4-gram	8.5	16.4	9.0	17.6
wav2vec 2.0 BASE [7]	LS-960	4-gram	5.0	10.8	5.5	11.3
wav2vec 2.0 LARGE [7]	LL-60k	Transformer	2.9	5.4	2.9	5.8
HUBERT BASE	LS-960	4-gram	5.6	10.9	6.1	11.3
HUBERT LARGE	LL-60k	Transformer	**2.6**	4.9	2.9	5.4
HUBERT X-LARGE	LL-60k	Transformer	**2.6**	**4.2**	**2.8**	**4.8**

Hsu, Boldt, Tsai, Lakhotia, Salakhutdinov & Mohamed,

Outline

- Recurrent neural networks
- Attention

$$c_i = \mathrm{softmax}(q_i @ k^T) @ v$$

- Self-attention, Multi-headed attention, Cross-attention, and Masked attention
- Self-training