TTIC 31190: Natural Language Processing

Lecture 7: Neural Networks and Sequence Labeling

Fall 2023

Announcement

- Assignment 1 due Thursday 11:59pm
- Assignment 2 will be out soon

Announcement

- TA session: review + quick introduction and discussion on papers 8-11
  - Houlsby et al. Parameter-Efficient Transfer Learning for NLP. ICML 2019
    (Parameter-efficient transfer learning)
  - Song et al. Score-Based Generative Modeling through Stochastic Differential Equations. ICLR 2021
    (Diffusion models and how they are used in NLP)
  - Borgeaud et al. Improving Language Models by Retrieving from Trillions of Tokens. ICML 2022
    (Retrieval augmented language models)
  - Meng et al. Locating and Editing Factual Associations in GPT. NeurIPS 2022
    (Model analysis and knowledge representation)

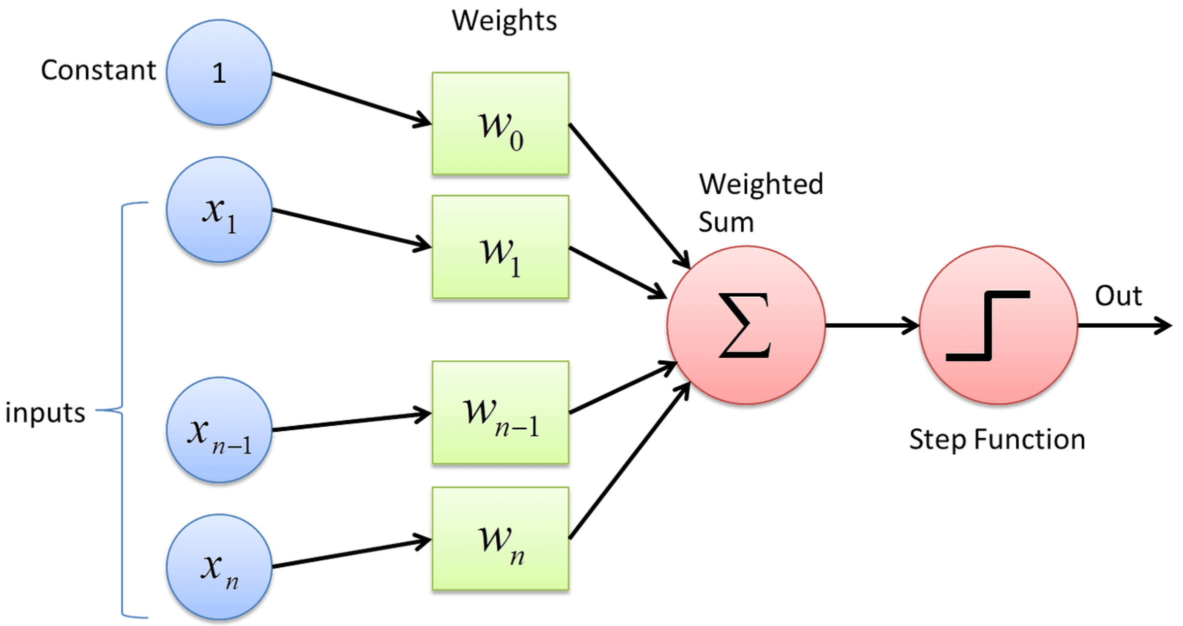
Recap

- Neural networks: perceptrons and multi-layer perceptrons (MLP)

$$\mathrm{perceptron}(\mathbf{x}) = \mathrm{step}\left(\mathbf{w}^\top \mathbf{x} + b\right)$$

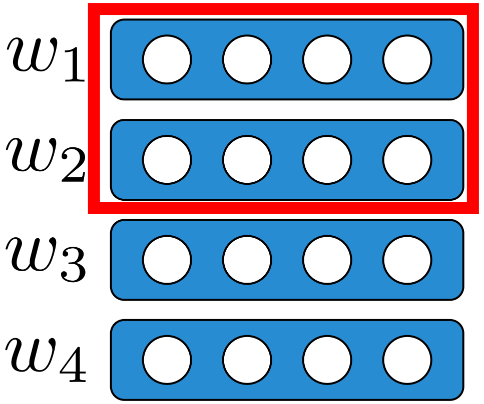
activation function

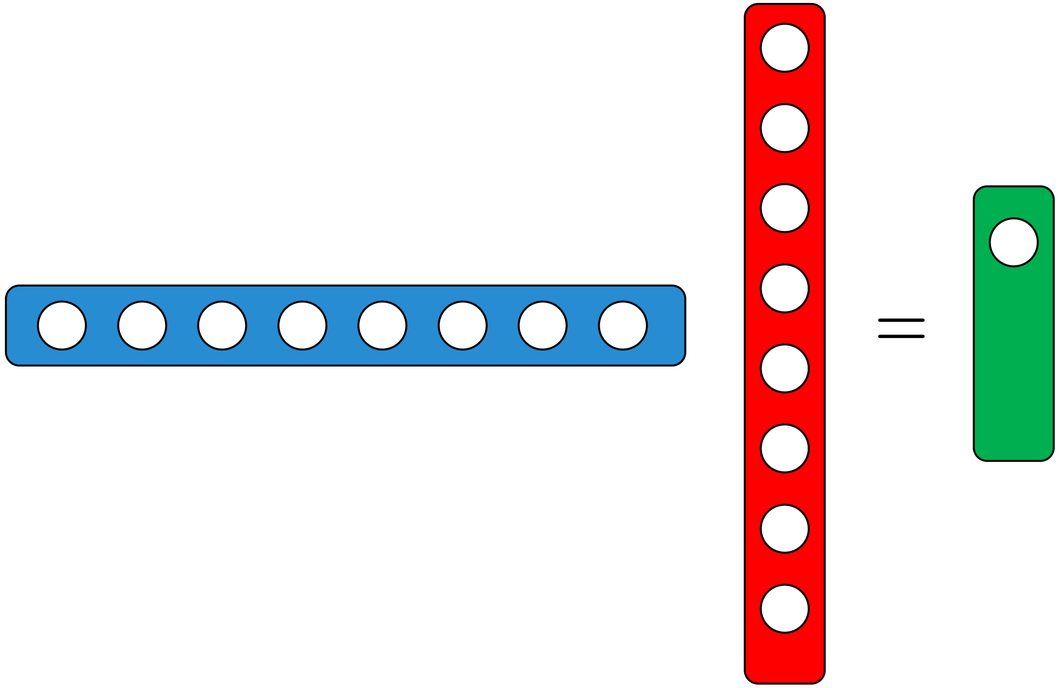
affine transform



Recap

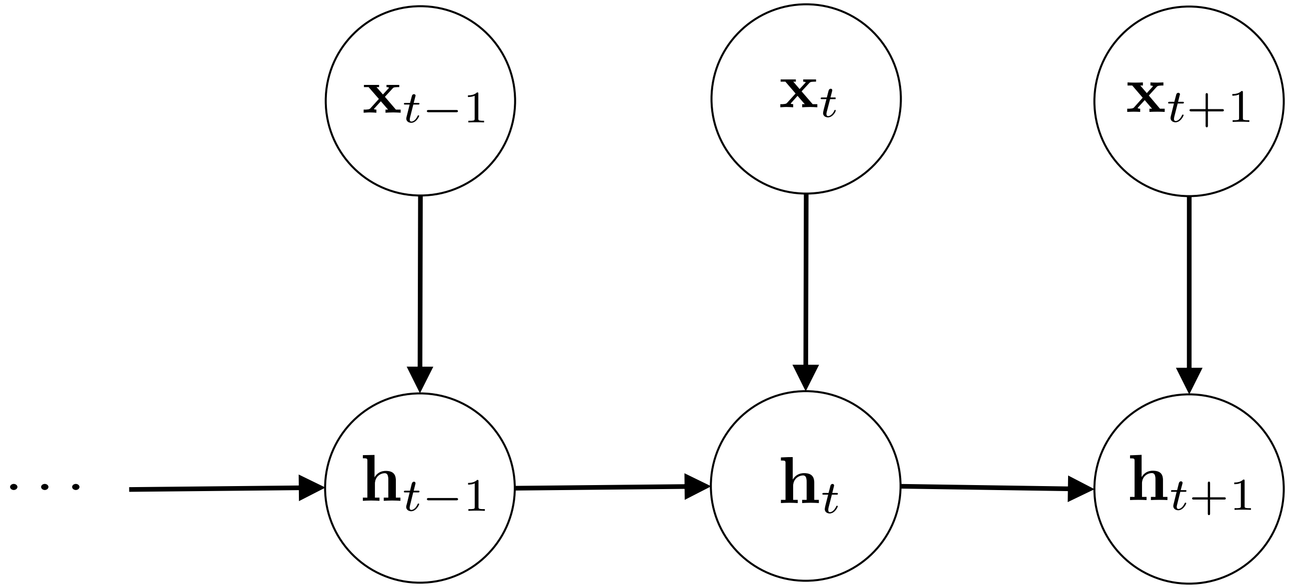
- Convolutional neural networks (CNNs)
  - Dot product between stretched kernel and word vectors
  - Pooling: convert a kernel's output to a scalar
  - Parallelize multiple kernels' output to get a fixed-dimensional representation





Recap

- Recurrent neural networks (RNNs)



$$\mathbf{h}_t = \mathbf{W}[\mathbf{x}_t; \mathbf{h}_{t-1}] + \mathbf{b}$$

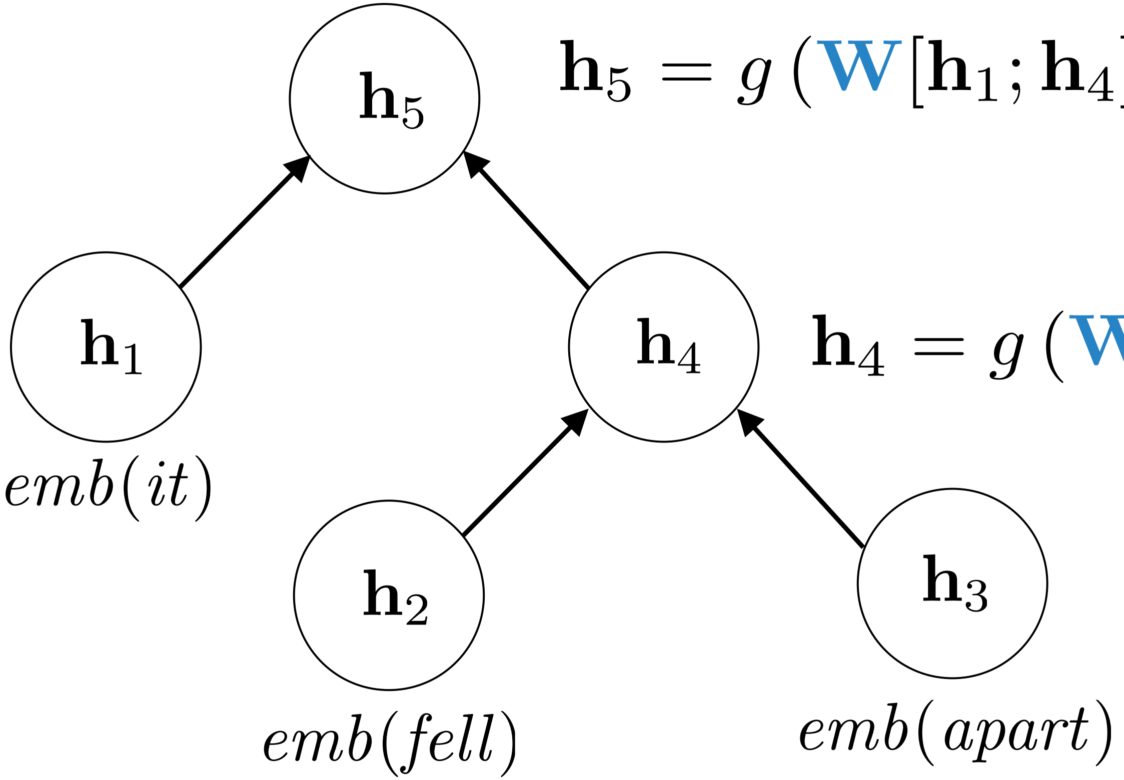
$$\mathbf{h}_{t+1} = \mathbf{W}[\mathbf{x}_{t+1}; \mathbf{h}_t] + \mathbf{b}$$

Recap

- RNN: absolute values can grow or vanish exponentially w.r.t. sequence length
- LSTM and GRU: gate mechanisms to preserve a stable value range

Recap

- Recursive neural networks (RvNN): apply same transformation at each node



$$\mathbf{h}_5 = g\left(\mathbf{W}[\mathbf{h}_1; \mathbf{h}_4] + \mathbf{b}\right)$$

$$\mathbf{h}_4 = g\left(\mathbf{W}[\mathbf{h}_2; \mathbf{h}_3] + \mathbf{b}\right)$$

This Lecture

- Neural networks
  - **Attention**
  - Transformers

- Sequence labeling
  - Tasks and problem formulation
  - Hidden Markov models (next lecture)
  - Conditional random fields (next lecture)

Attention

- Can be thought of as weighted sum; each token receives a weight
- From (unweighted) bag of words to (weighted) bag of words
  - Each word receives a fixed weight
  - Normalize the weights with softmax

$$\alpha_{w_i} = \mathrm{softmax}_{i'=1}^{k}\left(weight_{w_i}\right) = \frac{e^{weight_{w_i}}}{\sum_{i'=1}^{k} e^{weight_{w_{i'}}}}$$

$$\mathbf{x} = \sum_{i=1}^{k} \alpha_{w_i} \cdot emb(w_i)$$

Parameterized Attention

- Word tokens with the same word type should probably receive different weights in different sentences
- Implement attention with an MLP (example below)

$$\bar{\mathbf{x}} = \frac{1}{k} \sum_{i=1}^{k} emb(w_i)$$

$$\alpha(w_i \mid \bar{\mathbf{x}}) = \text{softmax}_{i'=1}^{k} \left(\text{MLP}([emb(w_i); \bar{\mathbf{x}}]) \right) \in \mathbb{R}$$

$$\mathbf{x} = \sum_{i=1}^{k} \alpha(w_i \mid \bar{\mathbf{x}}) \cdot emb(w_i)$$

Self-Attentive RNNs

- The last hidden state of RNN could be bad feature. Why?
- At time step t, what matters to $\mathbf{h}_t$ is mostly $\mathbf{x}_{t'}$ where t' is close to t [Khandelwal et al., ACL 2018] (Lecture 06)

$$\alpha_i = \text{softmax}_{i'=1}^{k}\left(\text{MLP}(\mathbf{h}_i)\right) \in \mathbb{R}$$

$$\mathbf{x} = \sum_{i=1}^{k} \alpha_i \mathbf{h}_i$$

Trainable parameters,
Jointly trained w/ RNN parameters

Attention: Summary

- Attention: weighted sum over features
- Weights can be the output of some MLP, normalized by softmax

$$\alpha_i = \mathrm{softmax}_{i'=1}^{k}\left(\mathrm{MLP}(\mathbf{h}_i)\right) \in \mathbb{R}$$

$$\mathbf{x} = \sum_{i=1}^{k} \alpha_i \mathbf{h}_i$$

- Caveat: attention weights over RNN hidden states could be bad indicators on which token is more important

This Lecture

- Neural networks
  - Attention
  - **Transformers**

- Sequence labeling
  - Tasks and problem formulation
  - Hidden Markov models (next lecture)
  - Conditional random fields (next lecture)

Transformers

Attention Is All You Need

Ashish Vaswani*
Google Brain
avaswani@google.com

Noam Shazeer*
Google Brain
noam@google.com

Niki Parmar*
Google Research
nikip@google.com

Jakob Uszkoreit*
Google Research
usz@google.com

Llion Jones*
Google Research
llion@google.com

Aidan N. Gomez* †
University of Toronto
aidan@cs.toronto.edu

Łukasz Kaiser*
Google Brain
lukaszkaiser@google.com

Illia Polosukhin* ‡
illia.polosukhin@gmail.com

Transformer Encoder

- Transformer: attention-based sentence encoding, and optionally, decoding
- Idea: every token has attention to every other token
- For sentence with tokens $(w_1, \ldots, w_k)$

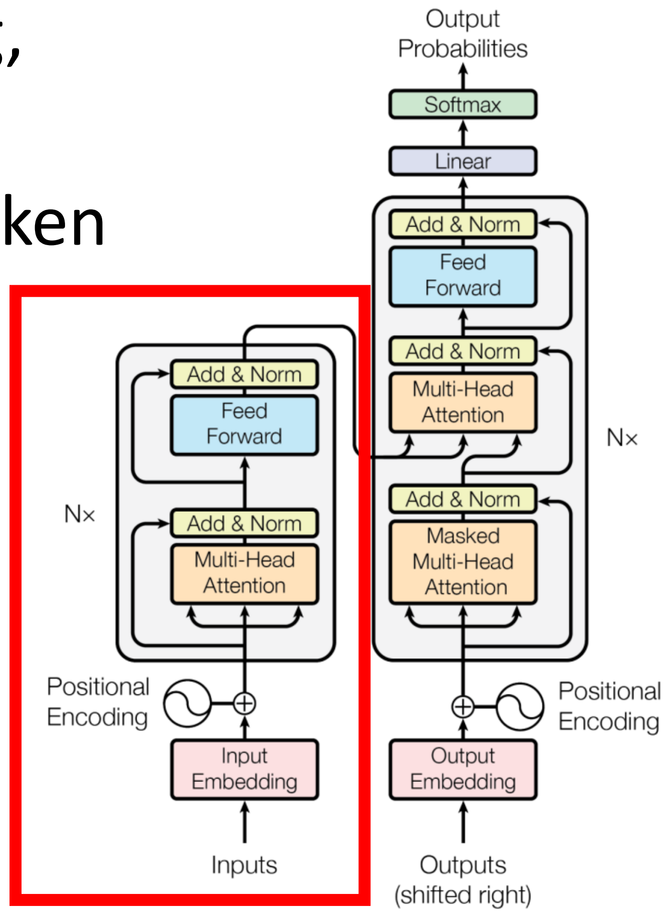
$$\mathbf{E} = (emb(w_1), \ldots, emb(w_k)) \in \mathbb{R}^{d_1 \times k}$$

$$\mathbf{K} = \mathbf{W}_k \mathbf{E} \quad \mathbf{W}_k \in \mathbb{R}^{d_2 \times d_1}$$

$$\mathbf{Q} = \mathbf{W}_q \mathbf{E} \quad \mathbf{W}_q \in \mathbb{R}^{d_2 \times d_1}$$

$$\mathbf{V} = \mathbf{W}_v \mathbf{E} \quad \mathbf{W}_v \in \mathbb{R}^{d_3 \times d_1}$$

Trainable parameters



Transformer Encoder

$$\mathbf{E} = (emb(w_1), \ldots, emb(w_k)) \in \mathbb{R}^{d_1 \times k}$$

$$\mathbf{K} = \mathbf{W}_k \mathbf{E} \quad \mathbf{W}_k \in \mathbb{R}^{d_2 \times d_1}, \mathbf{K} \in \mathbb{R}^{d_2 \times k}$$

$$\mathbf{Q} = \mathbf{W}_q \mathbf{E} \quad \mathbf{W}_q \in \mathbb{R}^{d_2 \times d_1}, \mathbf{Q} \in \mathbb{R}^{d_2 \times k}$$

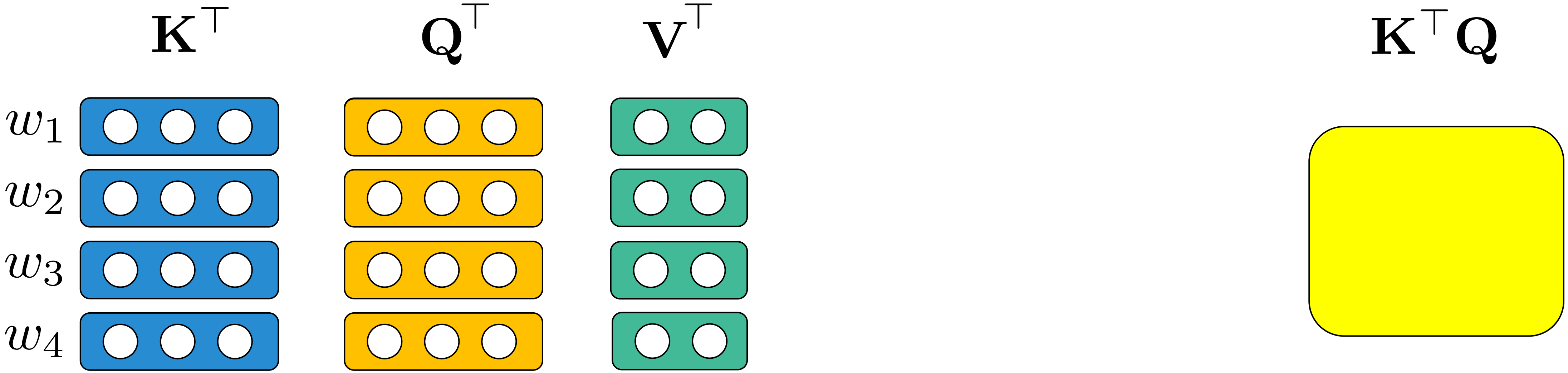
$$\mathbf{V} = \mathbf{W}_v \mathbf{E} \quad \mathbf{W}_v \in \mathbb{R}^{d_3 \times d_1}, \mathbf{V} \in \mathbb{R}^{d_3 \times k}$$

$$\tilde{\mathbf{E}} = \mathbf{V} \boxed{\mathrm{softmax}\left(\frac{\mathbf{K}^\top \mathbf{Q}}{\sqrt{d_2}}\right)} \in \mathbb{R}^{d_3 \times k}$$

$k \times k$ matrix, softmax over the first dimension

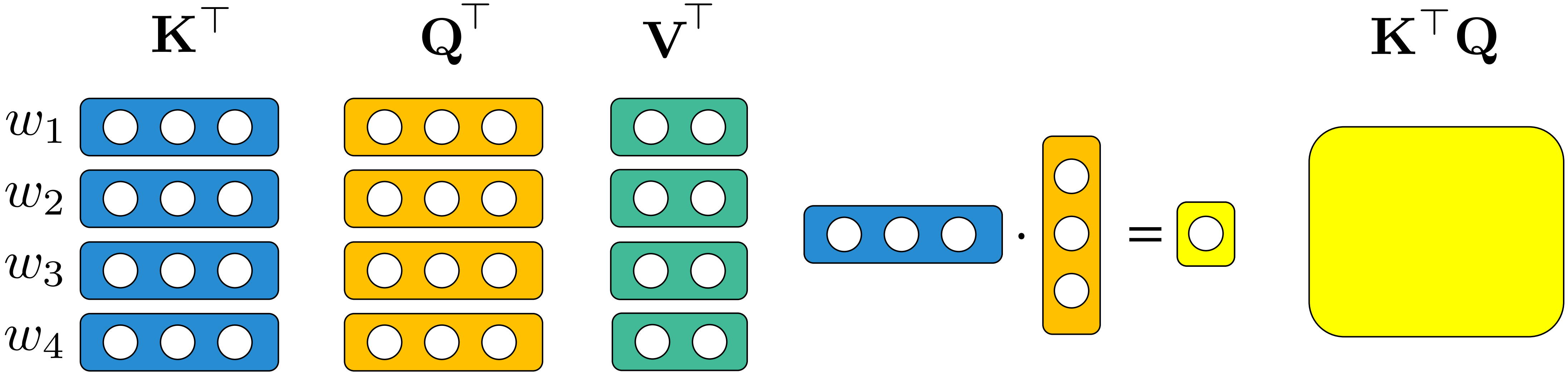
Transformer Encoder

$$\tilde{\mathbf{E}} = \mathbf{V}\mathrm{softmax}\left(\frac{\mathbf{K}^\top \mathbf{Q}}{\sqrt{d_2}}\right)$$



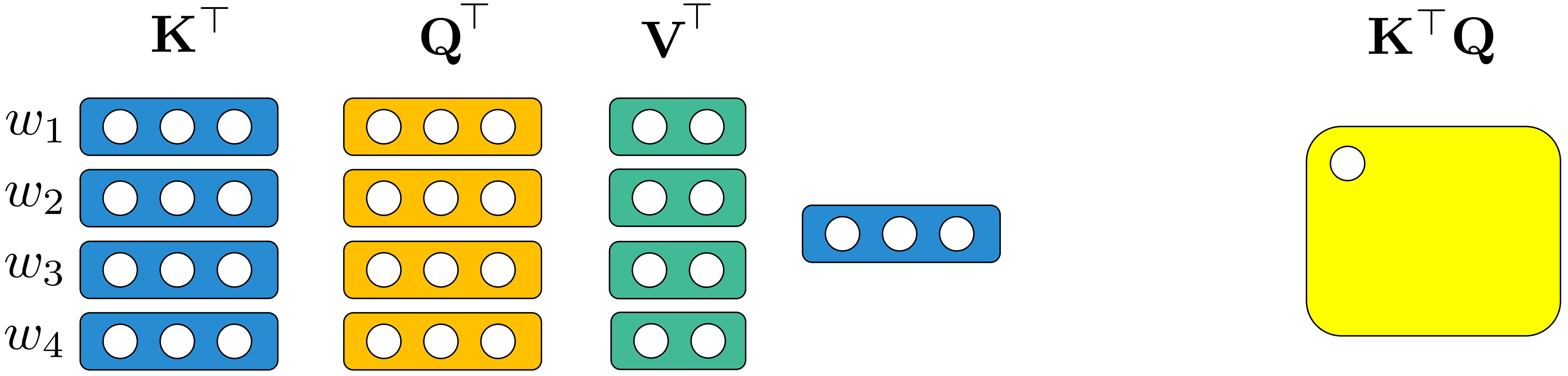
Transformer Encoder

$$\tilde{\mathbf{E}} = \mathbf{V}\mathrm{softmax}\left(\frac{\mathbf{K}^\top \mathbf{Q}}{\sqrt{d_2}}\right)$$



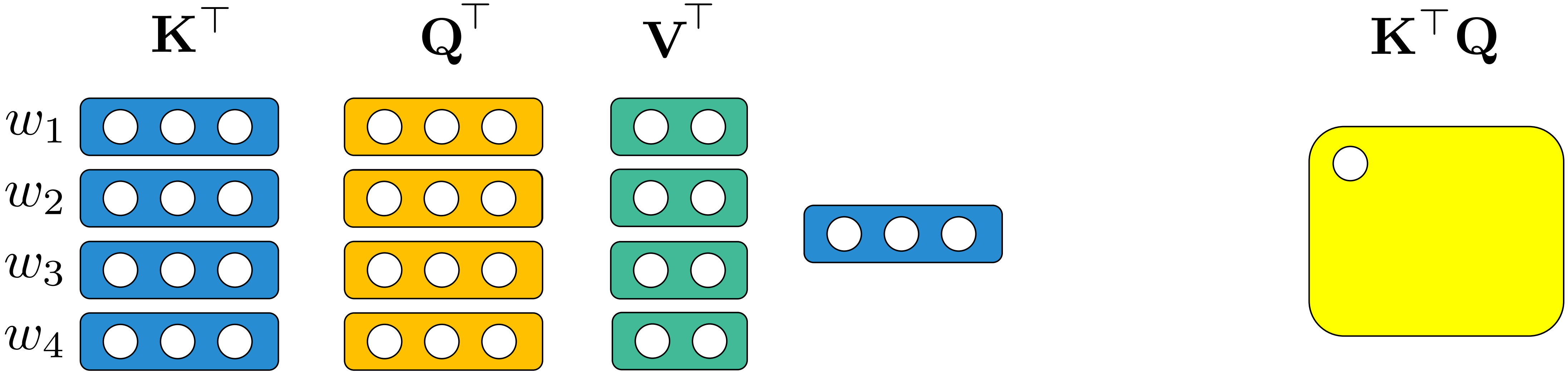
Transformer Encoder

$$\tilde{\mathbf{E}} = \mathbf{V}\text{softmax}\left(\frac{\mathbf{K}^\top \mathbf{Q}}{\sqrt{d_2}}\right)$$



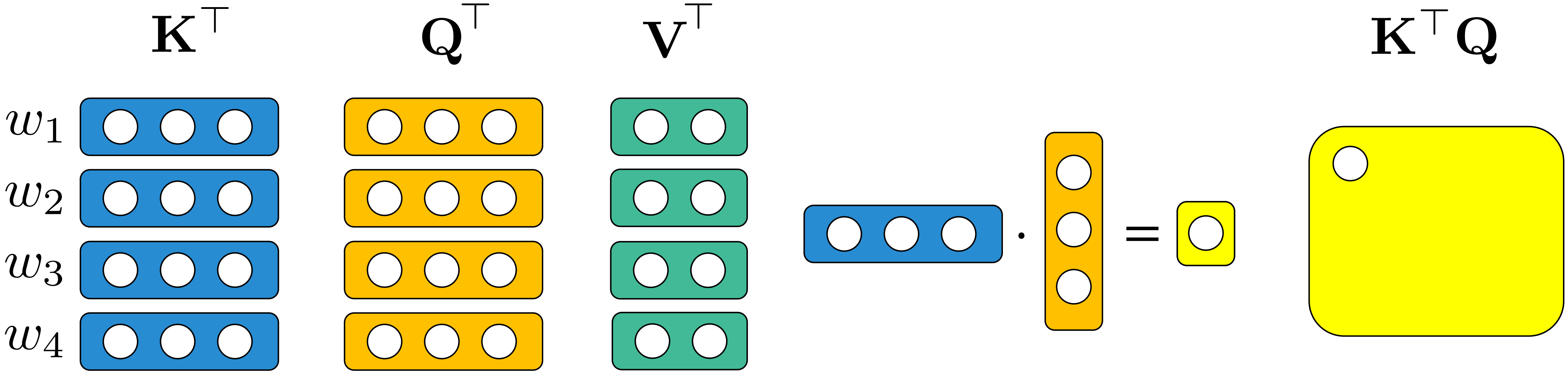
Transformer Encoder

$$\tilde{\mathbf{E}} = \mathbf{V}\mathrm{softmax}\left(\frac{\mathbf{K}^\top \mathbf{Q}}{\sqrt{d_2}}\right)$$



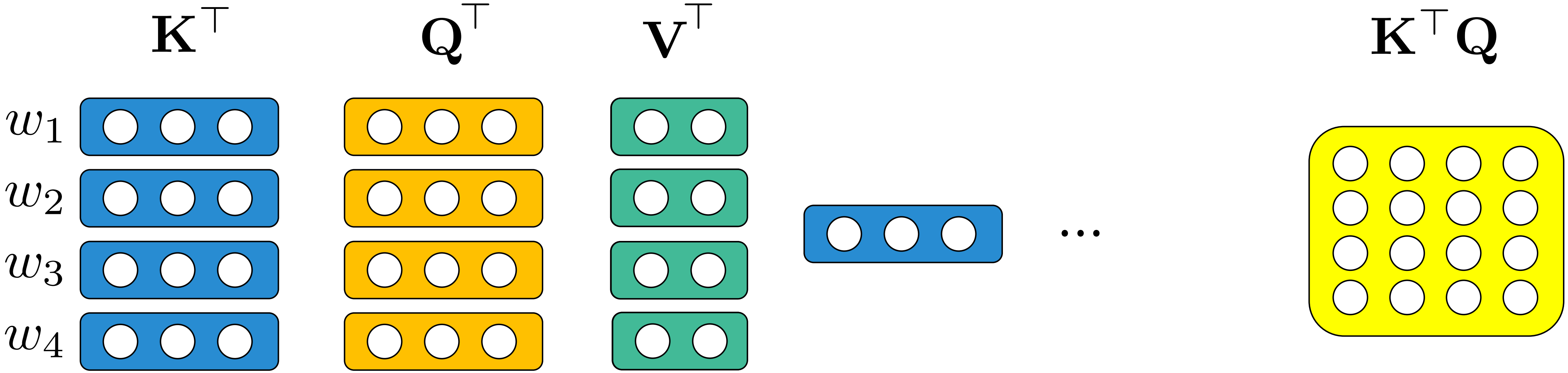
Transformer Encoder

$$\tilde{\mathbf{E}} = \mathbf{V}\text{softmax}\left(\frac{\mathbf{K}^\top \mathbf{Q}}{\sqrt{d_2}}\right)$$



Transformer Encoder

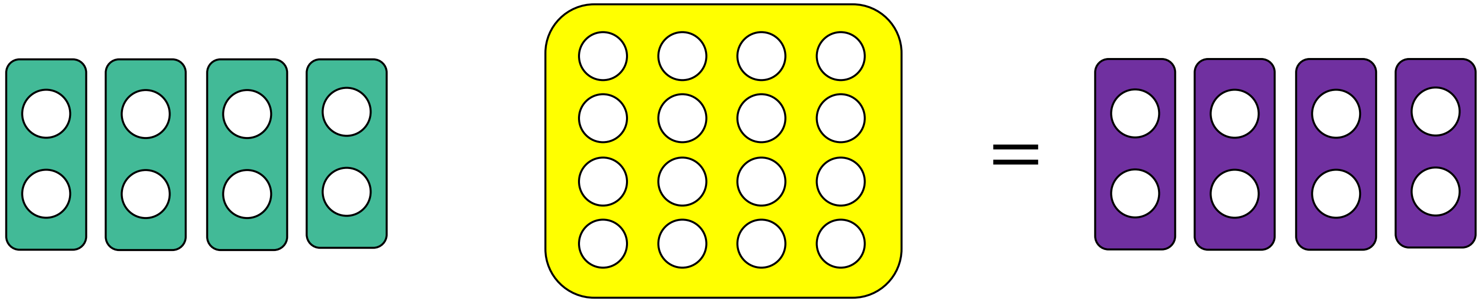
$$\tilde{\mathbf{E}} = \mathbf{V}\mathrm{softmax}\left(\frac{\mathbf{K}^\top\mathbf{Q}}{\sqrt{d_2}}\right)$$



Transformer Encoder

$$\tilde{\mathbf{E}} = \mathbf{V}\mathrm{softmax}\left(\frac{\mathbf{K}^\top\mathbf{Q}}{\sqrt{d_2}}\right)$$

$\mathbf{V}$ $\quad$ $\mathrm{softmax}\left(\frac{\mathbf{K}^\top\mathbf{Q}}{\sqrt{d_2}}\right)$



Transformer Encoder

$$\tilde{\mathbf{E}} = \mathbf{V}\mathrm{softmax}\left(\frac{\mathbf{K}^\top\mathbf{Q}}{\sqrt{d_2}}\right)$$

- What is $\sqrt{d_2}$ for?
- Consider $\langle \mathbf{a}, \mathbf{b} \rangle$: if each entry in both vector is drawn from a distribution with zero mean and unit variance, what would happen if the dimensionality grows?
- The variance of dot product grows.

$$\mathrm{softmax}([1, -1]) = [.8808, .1192]$$

$$\mathrm{softmax}([10, -10]) = [1, 2.0612 \times 10^{-9}]$$

Recap: Variance and Covariance

For independent zero-mean, unit-variance random variables X and Y

$$
\begin{aligned}
Var[XY] &= \mathbb{E}[X^2Y^2] - \mathbb{E}^2[XY] \\
&= (Cov[X^2, Y^2] + \mathbb{E}[X^2][Y^2]) - (Cov[X, Y] + \mathbb{E}[X]\mathbb{E}[Y])^2 \\
&= \mathbb{E}[X^2]\mathbb{E}[Y^2] - \mathbb{E}^2[X]\mathbb{E}^2[Y] \\
&= \ Var[X]\,Var[Y] + \ Var[X]\mathbb{E}^2[Y] + \ Var[Y]\mathbb{E}^2[X] \\
&= 1
\end{aligned}
$$

Recap: Variance and Covariance

For independent zero-mean, unit-variance random variables X and Y

$$Var[XY] = 1$$

If we have $2n$ independent zero-mean, unit variance variables $X_1, Y_1, X_2, Y_2, \ldots, X_n, Y_n$

$$Var[\sum_{i=1}^{n} X_i Y_i] = \sum_{i=1}^{n} Var[X_i Y_i] = n$$

$$Var[\sum_{i=1}^{n} \frac{X_i Y_i}{\sqrt{n}}] = \sum_{i=1}^{n} Var[\frac{X_i Y_i}{\sqrt{n}}] = \sum_{i=1}^{n} \frac{1}{n} Var[X_i Y_i] = 1$$

Transformer Encoder

$$Var[\sum_{i=1}^{n}\frac{X_iY_i}{\sqrt{n}}] = \sum_{i=1}^{n} Var[\frac{X_iY_i}{\sqrt{n}}] = \sum_{i=1}^{n}\frac{1}{n} Var[X_iY_i] = 1$$

$$\tilde{\mathbf{E}} = \mathbf{V}\mathrm{softmax}\left(\frac{\mathbf{K}^\top\mathbf{Q}}{\sqrt{d_2}}\right)$$

The application of $\sqrt{d_2}$ is theoretically motivated.

See also Xavier initialization: initialize a dot product parameter vector

with values drawn from $U\left(-\sqrt{\frac{3}{d}}, \sqrt{\frac{3}{d}}\right)$

Positional Encoding

$$\mathbf{E} = (emb(w_1), \ldots, emb(w_k)) \in \mathbb{R}^{d_1 \times k}$$

$$\mathbf{K} = \mathbf{W}_k \mathbf{E} \quad \mathbf{W}_k \in \mathbb{R}^{d_2 \times d_1}$$

$$\mathbf{Q} = \mathbf{W}_q \mathbf{E} \quad \mathbf{W}_q \in \mathbb{R}^{d_2 \times d_1}$$

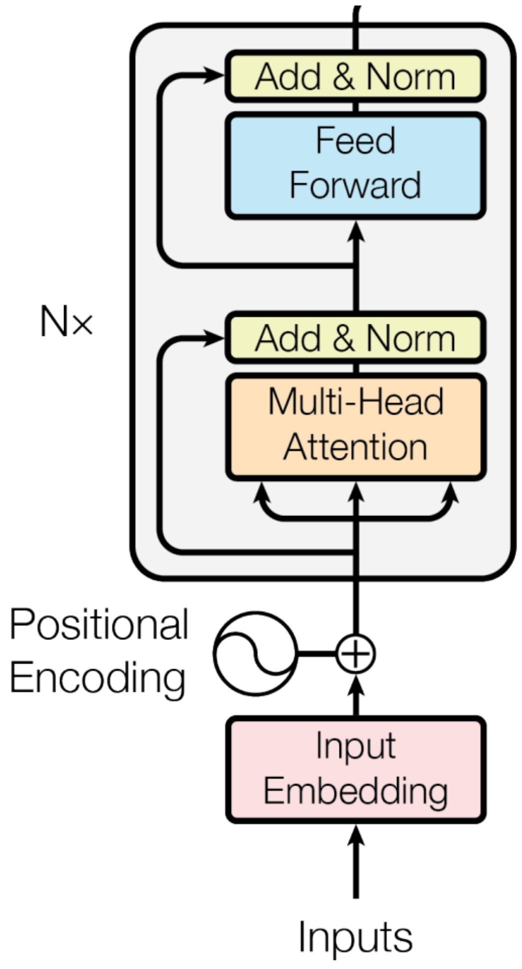
$$\mathbf{V} = \mathbf{W}_v \mathbf{E} \quad \mathbf{W}_v \in \mathbb{R}^{d_3 \times d_1}$$

$$\tilde{\mathbf{E}} = \mathbf{V}\mathrm{softmax}\left(\frac{\mathbf{K}^\top \mathbf{Q}}{\sqrt{d_2}}\right)$$

This is just complicated bag of words…

Columns of $\tilde{\mathbf{E}}$ for “a cat”

= permutation of columns of $\tilde{\mathbf{E}}$ for “cat a”



Positional Encoding

$$\mathbf{p}_{p,2i} = \sin\left(\frac{p}{10000^{\frac{2i}{d}}}\right), \mathbf{p}_{p,2i+1} = \cos\left(\frac{p}{10000^{\frac{2i}{d}}}\right)$$

- The choice of $n = 10{,}000$ is somewhat arbitrary, but it's overall theoretically motivated: The positional add-δ relation can be represented by a linear transformation.

$$\forall\delta, \exists\mathbf{M}_\delta, \text{s.t. } \mathbf{p}_{p+\delta} = \mathbf{M}_\delta \mathbf{p}_p \quad (\forall p)$$

- Proof idea: use the addition theorems on trigonometric functions

$$\sin(\alpha+\beta) = \sin\alpha\cos\beta + \cos\alpha\sin\beta$$

$$\cos(\alpha+\beta) = \cos\alpha\cos\beta - \sin\alpha\sin\beta$$

Positional Encoding

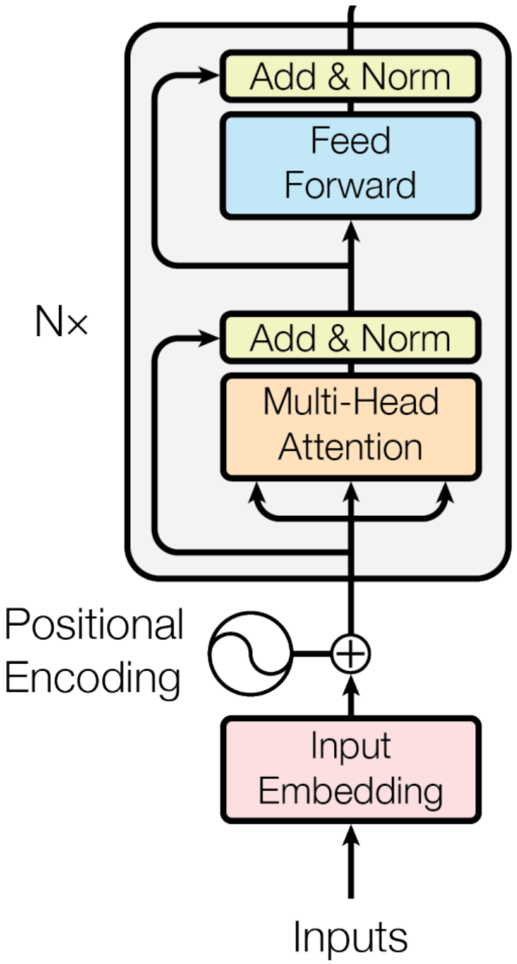
$$\mathbf{E} = (emb(w_1), \ldots, emb(w_k)) + \mathbf{P} \in \mathbb{R}^{d_1 \times k}$$

$$\mathbf{K} = \mathbf{W}_k\mathbf{E} \quad \mathbf{W}_k \in \mathbb{R}^{d_2 \times d_1}$$

$$\mathbf{Q} = \mathbf{W}_q\mathbf{E} \quad \mathbf{W}_q \in \mathbb{R}^{d_2 \times d_1}$$

$$\mathbf{V} = \mathbf{W}_v\mathbf{E} \quad \mathbf{W}_v \in \mathbb{R}^{d_3 \times d_1}$$

$$\tilde{\mathbf{E}} = \mathbf{V}\text{softmax}\left(\frac{\mathbf{K}^\top\mathbf{Q}}{\sqrt{d_2}}\right)$$



- Limitation: only fixed number of positions available
- Another option: learnable positional encoding

Multi-Head Attention

$$\mathbf{E} = (emb(w_1), \ldots, emb(w_k)) + \mathbf{P} \in \mathbb{R}^{d_1 \times k}$$

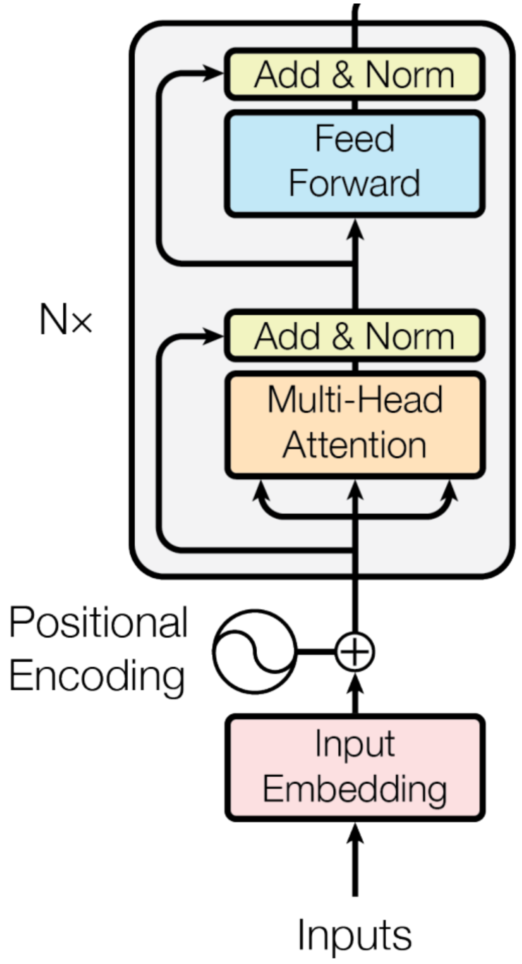
$$\mathbf{K} = \mathbf{W}_k \mathbf{E} \quad \mathbf{W}_k \in \mathbb{R}^{d_2 \times d_1}$$

$$\mathbf{Q} = \mathbf{W}_q \mathbf{E} \quad \mathbf{W}_q \in \mathbb{R}^{d_2 \times d_1}$$

$$\mathbf{V} = \mathbf{W}_v \mathbf{E} \quad \mathbf{W}_v \in \mathbb{R}^{d_3 \times d_1}$$

$$\tilde{\mathbf{E}} = \mathbf{V}\text{softmax}\left(\frac{\mathbf{K}^\top \mathbf{Q}}{\sqrt{d_2}}\right)$$

- We can parallelize multiple $\mathbf{W}_k, \mathbf{W}_q, \mathbf{W}_v$ with different random initialization (and hope they learn different ways to attend tokens.



Stacking Transformer Layers

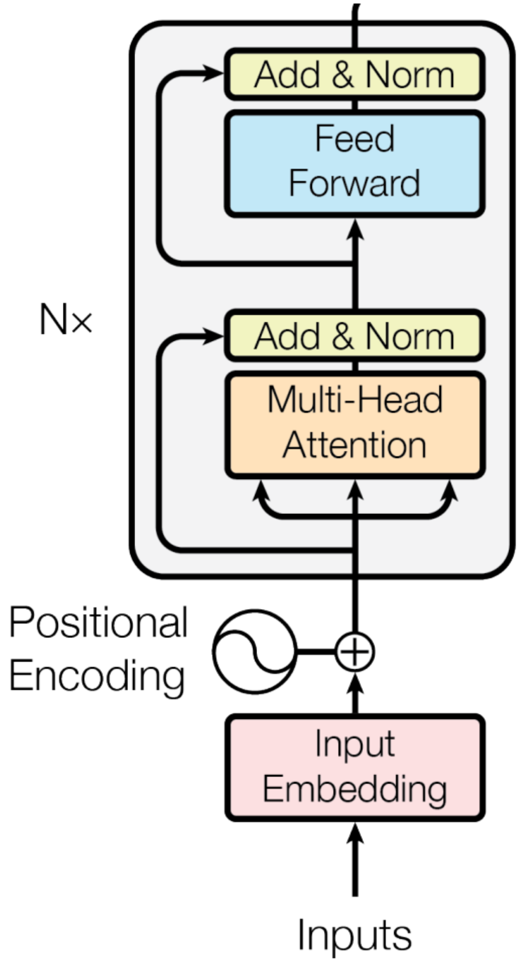
$$\mathbf{E} = (emb(w_1), \ldots, emb(w_k)) + \mathbf{P} \in \mathbb{R}^{d_1 \times k}$$

$$\mathbf{K} = \mathbf{W}_k\mathbf{E} \quad \mathbf{W}_k \in \mathbb{R}^{d_2 \times d_1}$$

$$\mathbf{Q} = \mathbf{W}_q\mathbf{E} \quad \mathbf{W}_q \in \mathbb{R}^{d_2 \times d_1}$$

$$\mathbf{V} = \mathbf{W}_v\mathbf{E} \quad \mathbf{W}_v \in \mathbb{R}^{d_3 \times d_1}$$

$$\tilde{\mathbf{E}} = \mathbf{V}\mathrm{softmax}\left(\frac{\mathbf{K}^\top\mathbf{Q}}{\sqrt{d_2}}\right)$$



This Lecture

- Neural networks
  - Attention
  - Transformers

- Sequence labeling
  - **Tasks and problem formulation**
  - Hidden Markov models (next lecture)
  - Conditional random fields (next lecture)

Linguistic Phenomena

- Words have structure (stems and affixes)
- Words have multiple meanings (senses) → word sense ambiguity
  - Senses of a word can be homonymous or polysemous
  - Senses have relationships:
    - Synonymy, hyponymy ("is a"), meronymy ("part of", "member of")
- Variability/flexibility of linguistic expression
  - many ways to express the same meaning
  - word embeddings tell us when two words are similar
- Today: **part-of-speech**

Part-of-Speech Tagging

determiner	verb (past)	prep.	proper noun	proper noun	poss.	adj.	noun
Some	questioned	if	Tim	Cook	’s	first	product

modal	verb	det.	adjective	noun	prep.	proper noun	punc.
would	be	a	breakaway	hit	for	Apple	.

Part-of-Speech Tagging

- Functional category of a word:
  - noun, verb, adjective, etc.

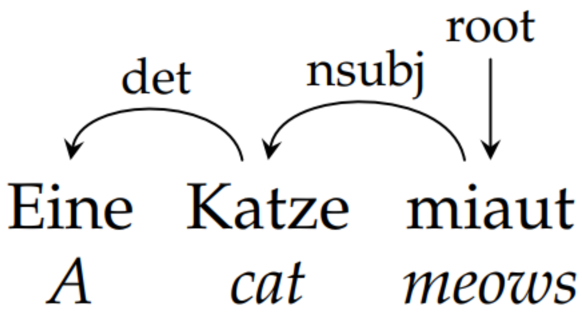
- Dependent on context like word sense, but different from sense:
  - Sense represents word meaning, POS represents word function
  - Sense uses a distinct category of senses per word, POS uses same set of categories for all words

- Arguably the simplest type of syntactic information

Universal Tag Set

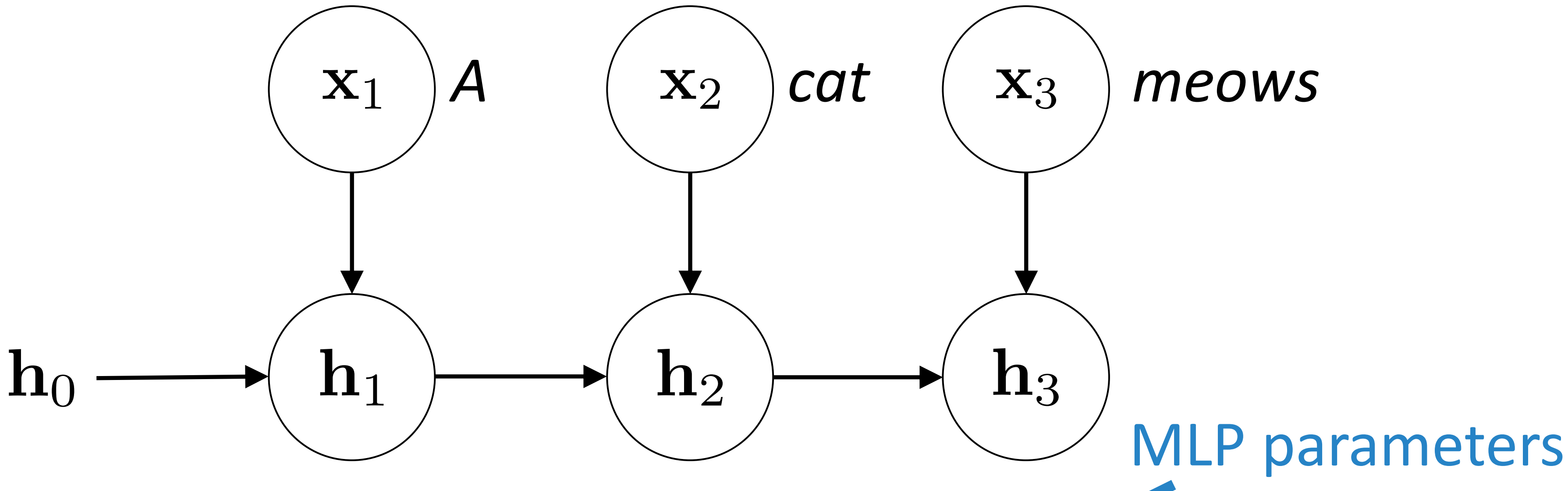
- 12 categories: Noun, verb, adjective, adverb, pronoun, determiner/article, adposition (preposition or postposition), numeral, conjunction, particle, punctuation, other

- Foundation of the universal dependency hypothesis



Part-of-Speech Tagging with an RNN

- Idea: breaking it down into k individual classification problems
  Collect hidden states, then pass them into an MLP classifier



$$\mathrm{POS}(\mathbf{x}_i) = \arg\max_y \mathrm{score}(\mathbf{x}_{1:i}, y; \mathbf{w})$$

Span Extraction as Sequence Tagging

- Named entity recognition: recognizing names of real-world objects from a sentence

O	O	O	B-PERSON	I-PERSON	O	O	O
Some	questioned	if	Tim	Cook	’s	first	product

O	O	O	O	O	O	B-ORGANIZATION	O
would	be	a	breakaway	hit	for	Apple	.

B=beginning, I=inside, O=outside

Span Extraction as Sequence Tagging

- Named entity recognition: recognizing names of real-world objects from a sentence
- Alternative option: simple B-I-O tags, then predict fine grained labels with span features

O	O	O	B	I	O	O	O
Some	questioned	if	Tim	Cook	’s	first	product

O	O	O	O	O	O	B	O
would	be	a	breakaway	hit	for	Apple	.

B=beginning, I=inside, O=outside

Sequence Tagging

- Feature vector can be produced by any model architecture, as long as it’s a reasonable representation of the corresponding token

- What’s the limitation?
- It doesn’t explicitly consider the underlying dependency among tags
    - Example: it’s (nearly) impossible to have a determiner followed by another determiner
    - Example: model shouldn’t have an “O” tag followed by an “I”

Sequence Tagging: Problem Formulation

$$\mathrm{POS}(\mathbf{x}_i) = \arg\max_{y} \mathrm{score}(\mathbf{x}, i, y; \mathbf{w})$$

$$\mathrm{POS}(\mathbf{x}) = \arg\max_{\boldsymbol{y}} \mathrm{score}(\mathbf{x}, \boldsymbol{y}; \mathbf{w})$$

structured object
(sequence of tags)

Next Lecture

- Neural networks
  - Attention
  - Transformers

- Sequence labeling
  - Part-of-speech tagging with neural networks
  - **Hidden Markov models**
  - **Conditional random fields**