Neural NLP I

HSS 510: NLP for HSS

Taegyoon Kim

May 22, 2024

Next Two Weeks

Week 14 (May 29): BERT and prompt-based labeling

- BERTopic (in-depth coverage, guided coding)
- Classification using BERT fine-tuning
- Classification with LLM-augmented labels
  - **Presentation by Jaehong (paper link)**

Week 15 (June 5)

- Promises and pitfalls of LLM
- Prompt-based relation inference
  - **Presentation by Prof. Woo Seokkyun**

Agenda

Things to be covered

- Static embeddings vs. contextual embeddings
- FNN, RNN, Attention/transformer
- BERT
- Usage of BERT in applied research

Static Embeddings

From contexts but not contextual

- Word2Vec, GloVe, Fast2Text
  - A word's embedding is derived from its context
- Different contexts do *not* lead to different embeddings
  $\rightarrow$ *static* embeddings
- Even if a word has the same form, its meaning differs depending on the context

Contextual Embeddings

Example

- Embeddings for the same word differ by context
- Sentence 1: *Open a* *bank* *account*
  $\rightarrow e_{bank_{s1}}$: [0.3, 0.9, …]
- Sentence 2: *On the river* *bank*
  $\rightarrow e_{bank_{s2}}$: [0.8, 0.1, …]

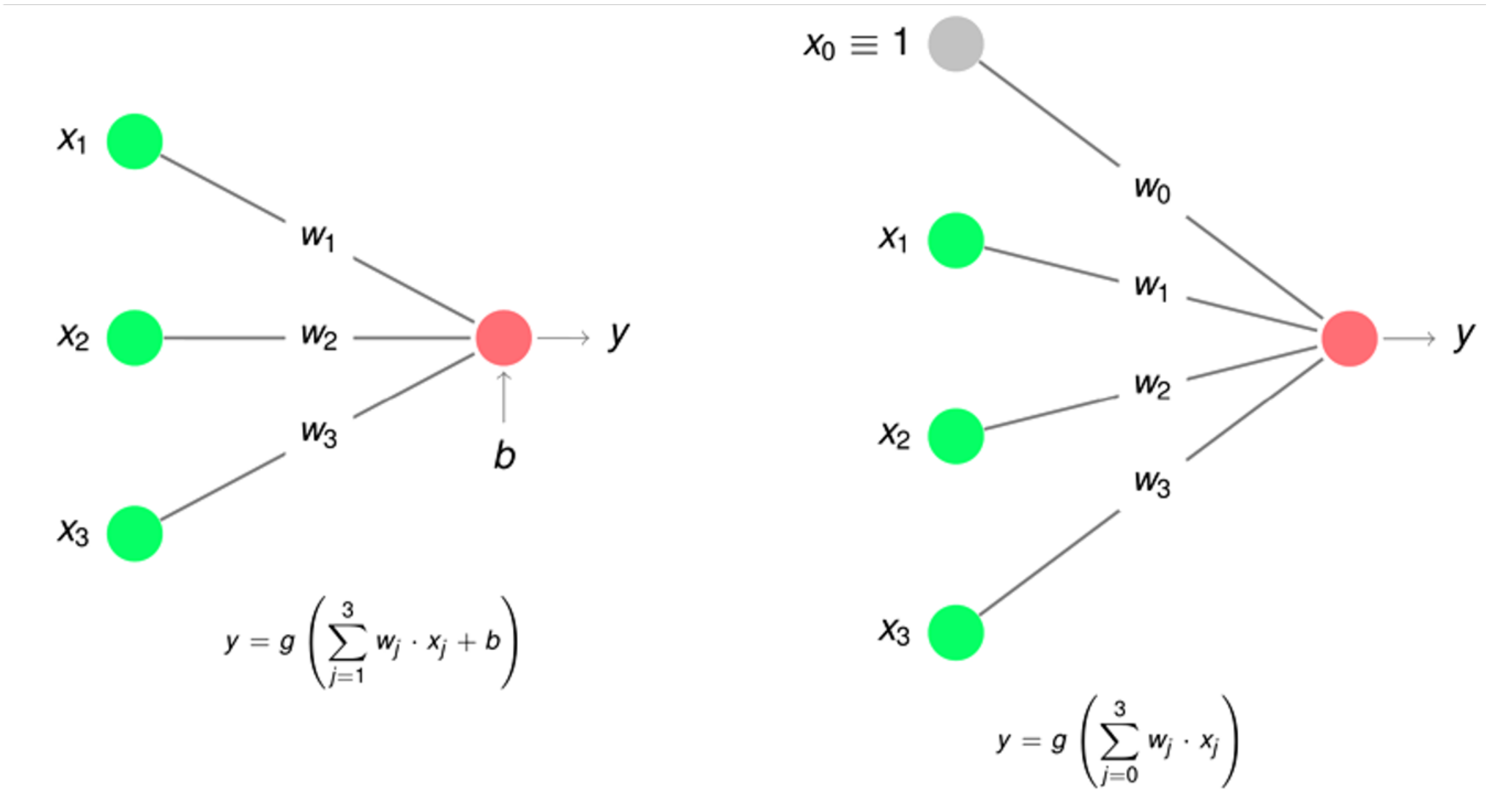
Contextual Information in Neural Networks

FNN, RNN, Attention/transformer

- Recent neural networks in NLP often encode contextual information
  - E.g., ELMo, BERT, GPT, Llama, etc.
- They are not always built to generate embeddings
  - E.g., predicting the next word (language model), question-answering, text summarization, etc.
- We will have a look at ELMo and BERT and some of the neural network models that preceded/motivated it (FNN, RNN)

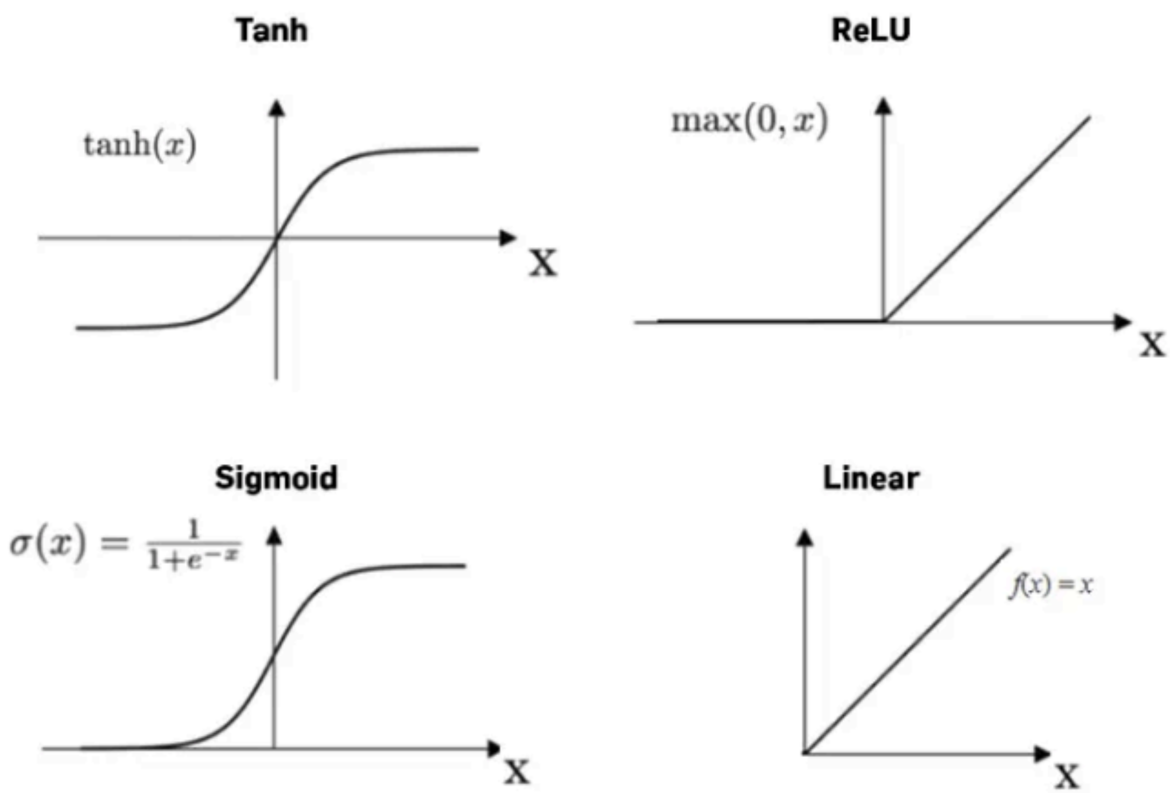
Feedforward Neural Network

Single neuron (alternative expressions for bias term)



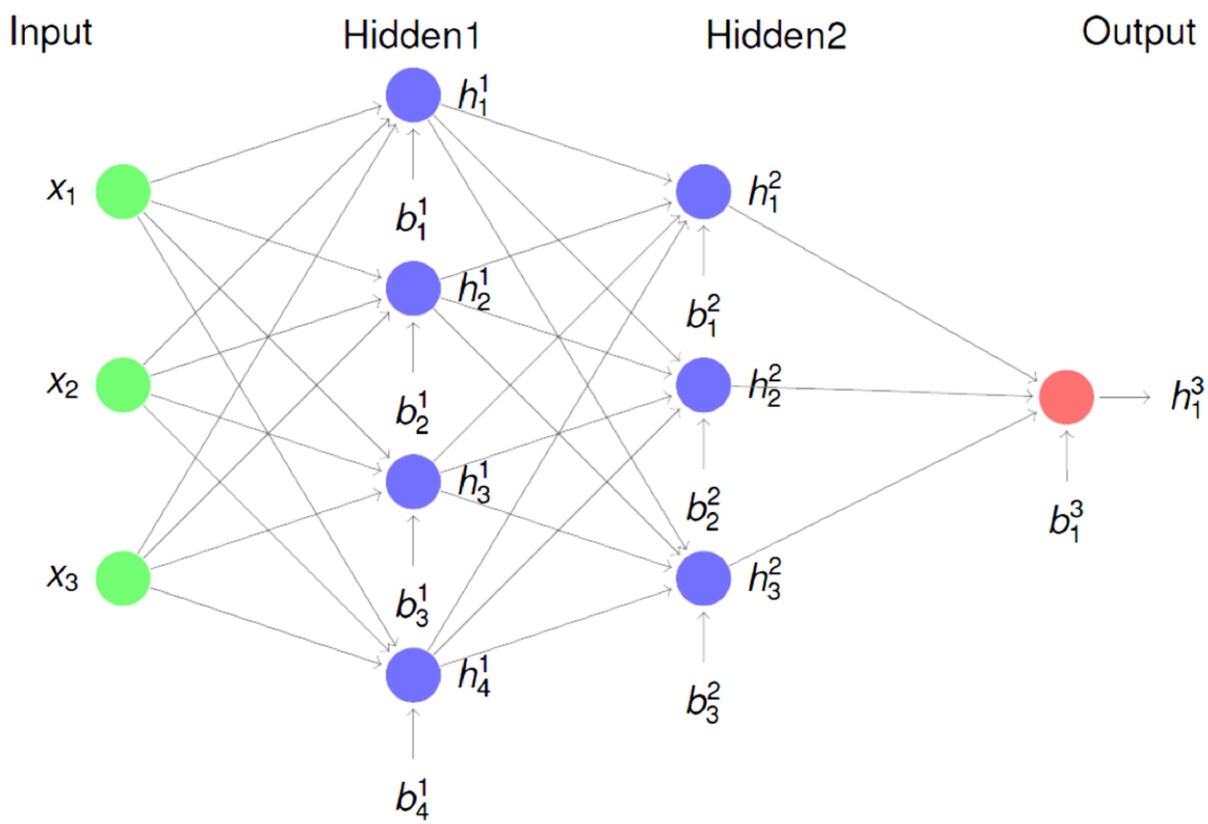
Feedforward Neural Network

Activation functions



Feedforward Neural Network

Stylized multi-layer network (L = 3)



Feedforward Neural Network

Matrix notation

Output of first hidden-layer:

$$\begin{pmatrix} h_1^1 \\ h_2^1 \\ h_3^1 \\ h_4^1 \end{pmatrix} = g\left(\begin{pmatrix} W_{11}^1 & W_{12}^1 & W_{13}^1 \\ W_{21}^1 & W_{22}^1 & W_{23}^1 \\ W_{31}^1 & W_{32}^1 & W_{33}^1 \\ W_{41}^1 & W_{42}^1 & W_{43}^1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} + \begin{pmatrix} b_1^1 \\ b_2^1 \\ b_3^1 \\ b_4^1 \end{pmatrix}\right)$$

Output of second hidden-layer:

$$\begin{pmatrix} h_1^2 \\ h_2^2 \\ h_3^2 \end{pmatrix} = g\left(\begin{pmatrix} W_{11}^2 & W_{12}^2 & W_{13}^2 & W_{14}^2 \\ W_{21}^2 & W_{22}^2 & W_{23}^2 & W_{24}^2 \\ W_{31}^2 & W_{32}^2 & W_{33}^2 & W_{34}^2 \end{pmatrix} \begin{pmatrix} h_1^1 \\ h_2^1 \\ h_3^1 \\ h_4^1 \end{pmatrix} + \begin{pmatrix} b_1^2 \\ b_2^2 \\ b_3^2 \end{pmatrix}\right)$$

Output of the network:

$$y = \begin{pmatrix} h_1^3 \end{pmatrix} = g\left(\begin{pmatrix} W_{11}^3 & W_{12}^3 & W_{13}^3 \end{pmatrix} \begin{pmatrix} h_1^2 \\ h_2^2 \\ h_3^2 \end{pmatrix} + \begin{pmatrix} b_1^3 \end{pmatrix}\right)$$

Feedforward Neural Network

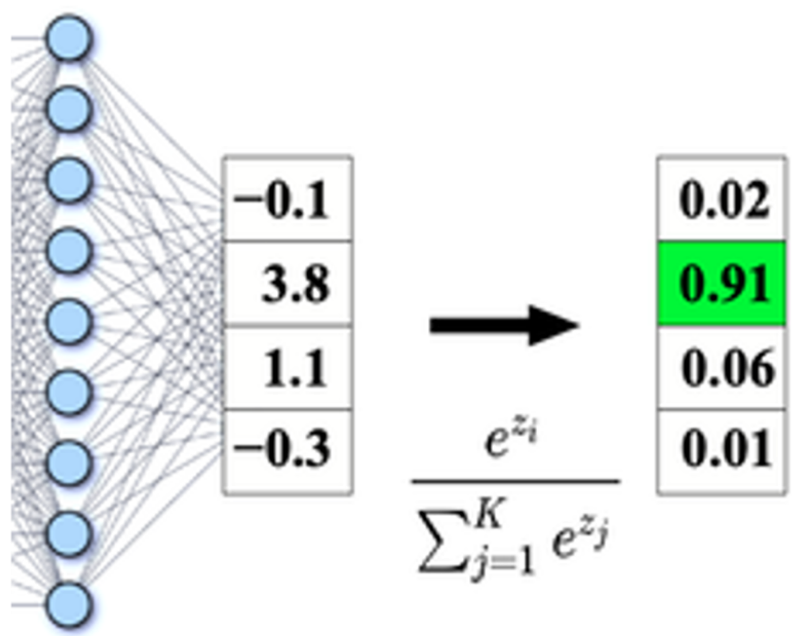
Neuron(s) at the output layer

- The number of neurons depends on the task
- For binary classification and regression: one neuron (e.g., spam detection)
- For multinomial classification (> 2), K neurons (e.g., language model)
  - Soft-max function converts a vector of K real numbers into a probability distribution of K possible outcomes
  - Generalization of the logistic function to multiple dimensions

Feedforward Neural Network

Soft-max function (see here for more discussion)

- E.g., multi-nomial classification: K = 4



Feedforward Neural Network

Training FNN

- Supervised machine learning
- Learn the weights and bias terms that minimize the differences between the predictions and the actual labels
- Use of the cross-entropy loss, gradient descent, and backpropagation
- See [JM] Section 5 in Chp.7 for details

Feedforward Neural Network

Input features: scalar (single number)

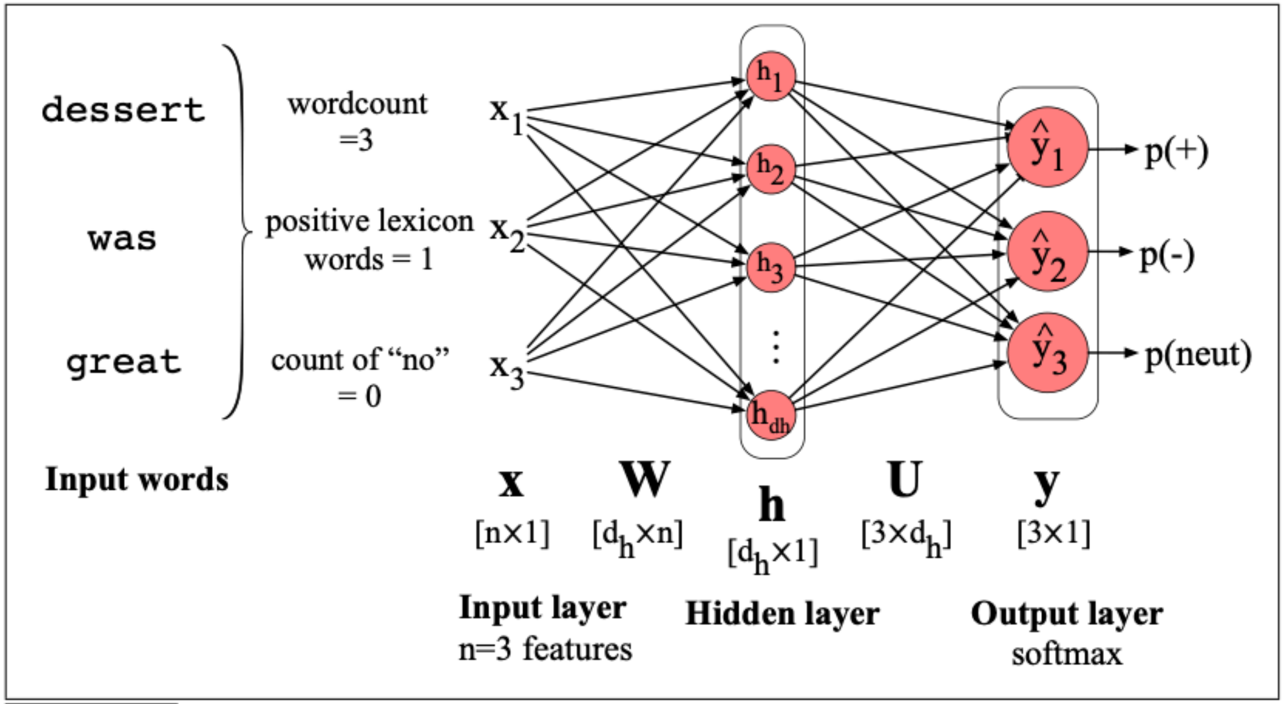


Figure 7.10 Feedforward network sentiment analysis using traditional hand-built features of the input text.

Feedforward Neural Network

Input features: pooled embeddings

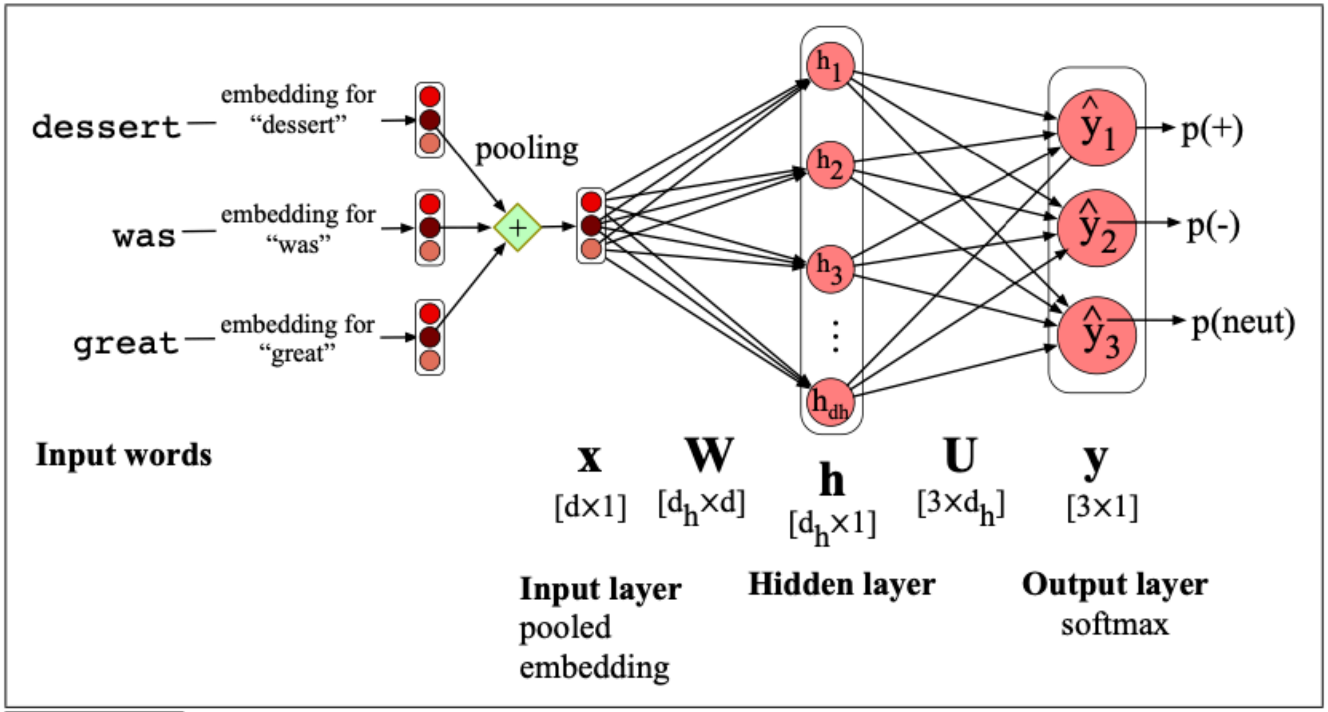


Figure 7.11 Feedforward network sentiment analysis using a pooled embedding of the input words.

Feedforward Neural Network

Input features: individual embeddings (task: LM)

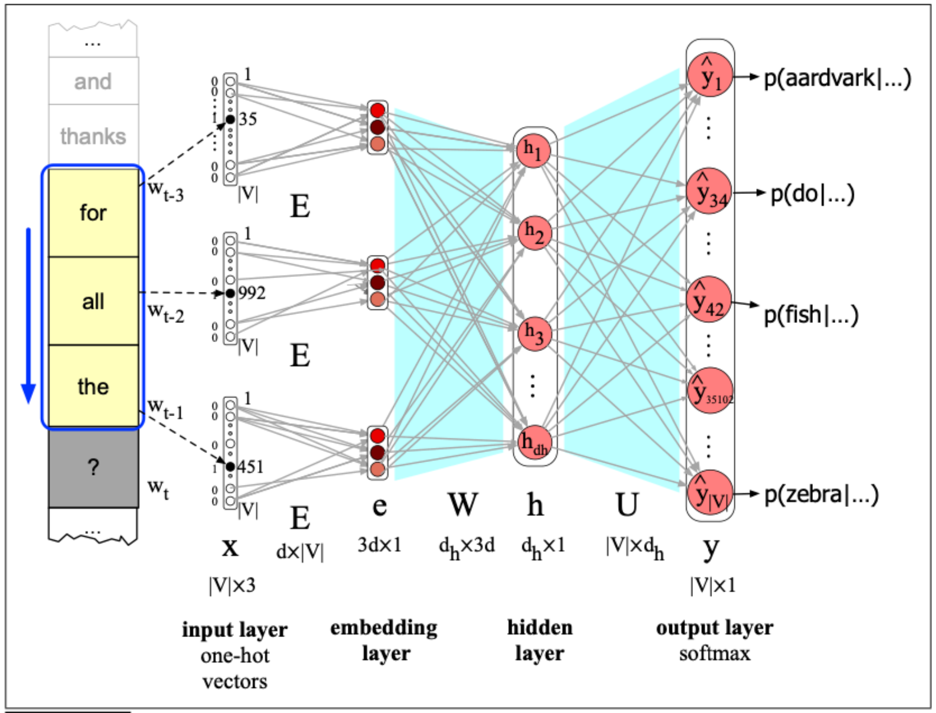


Figure 7.17 Forward inference in a feedforward neural language model. At each timestep t the network computes a d-dimensional embedding for each context word (by multiplying a one-hot vector by the embedding matrix **E**), and concatenates the 3 resulting embeddings to get the embedding layer **e**. The embedding vector **e** is multiplied by a weight matrix **W** and then an activation function is applied element-wise to produce the hidden layer **h**, which is then multiplied by another weight matrix **U**. Finally, a softmax output layer predicts at each node i the probability that the next word w_t will be vocabulary word V_i.

Recurrent Neural Network

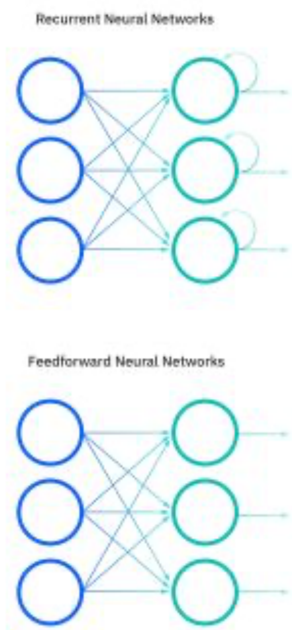
Recurring connections sequentially capture context

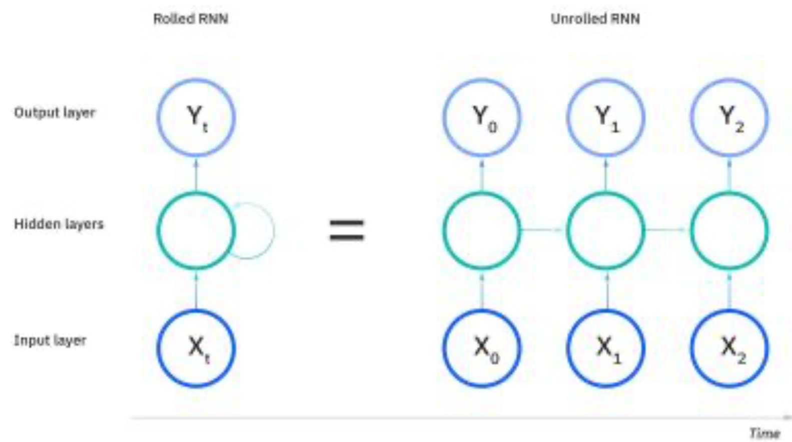
- FNN propagates signals only in one direction: from the input towards the output layers
  - The output of layer i can be passed to the input of neurons in layer j, if and only if $i < j$
- In an RNN, the output of neurons in layer j can be passed to the input of neurons in the same layer or to neurons in layer i ($< j$)

→ RNN can handle contextual information through their sequential processing

Recurrent Neural Network

Hidden layer in FNN (L) vs. RNN (R) (*unrolled*)





Recurrent Neural Network

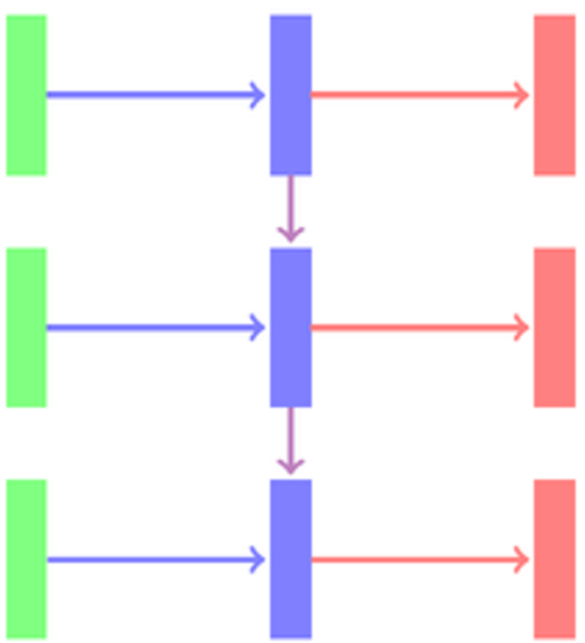
Stylized matrix notation

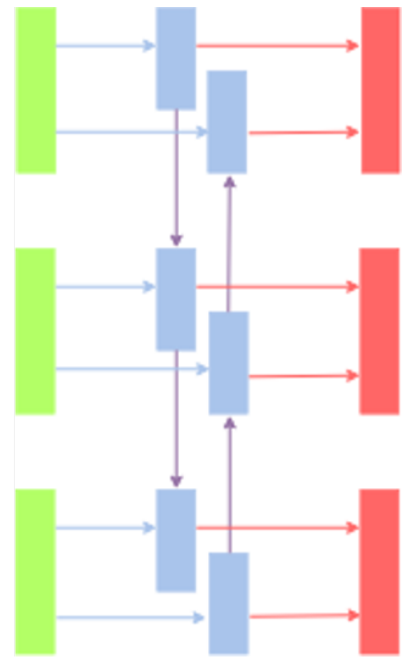
$$\mathbf{h}^1(t) = g\left(W^1\mathbf{x}(t) + R^1\mathbf{h}^1(t-1) + \mathbf{b}^1\right)$$

$$= g\left(\left(W^1 \mid R^1\right)\begin{pmatrix}\mathbf{x}(t)\\ \mathbf{h}^1(t-1)\end{pmatrix} + \mathbf{b}^1\right)$$

Recurrent Neural Network

Uni-directional RNN (L) vs. Bi-directional RNN (R)





Recurrent Neural Network

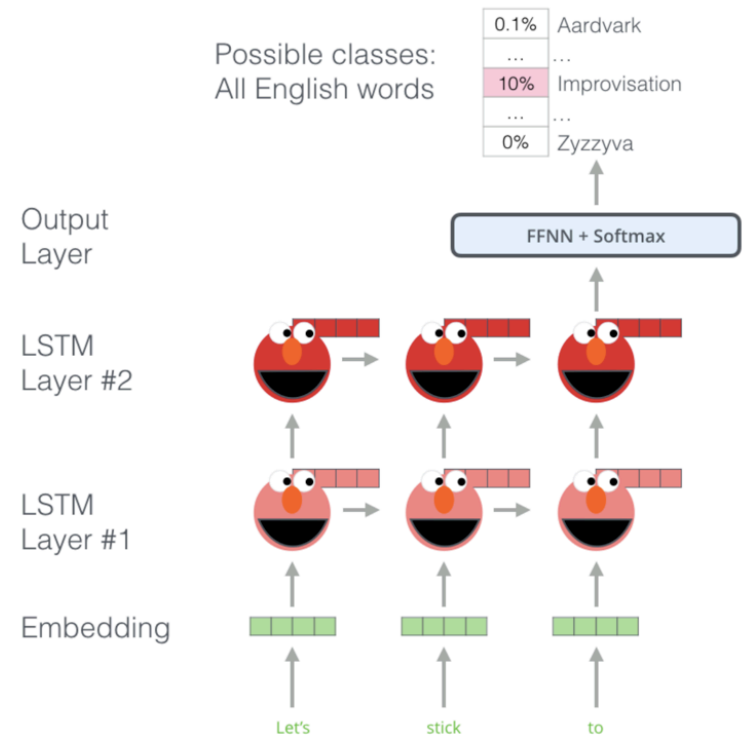
Extensions of simple RNN (see this for a detailed treatment)

- LSTM (Long Short-term Memory)
- GRU (Gated Recurrent Unit)
- ELMo (Embeddings from Language Model)
  - Based on bi-dirctional LSTM

ELMo

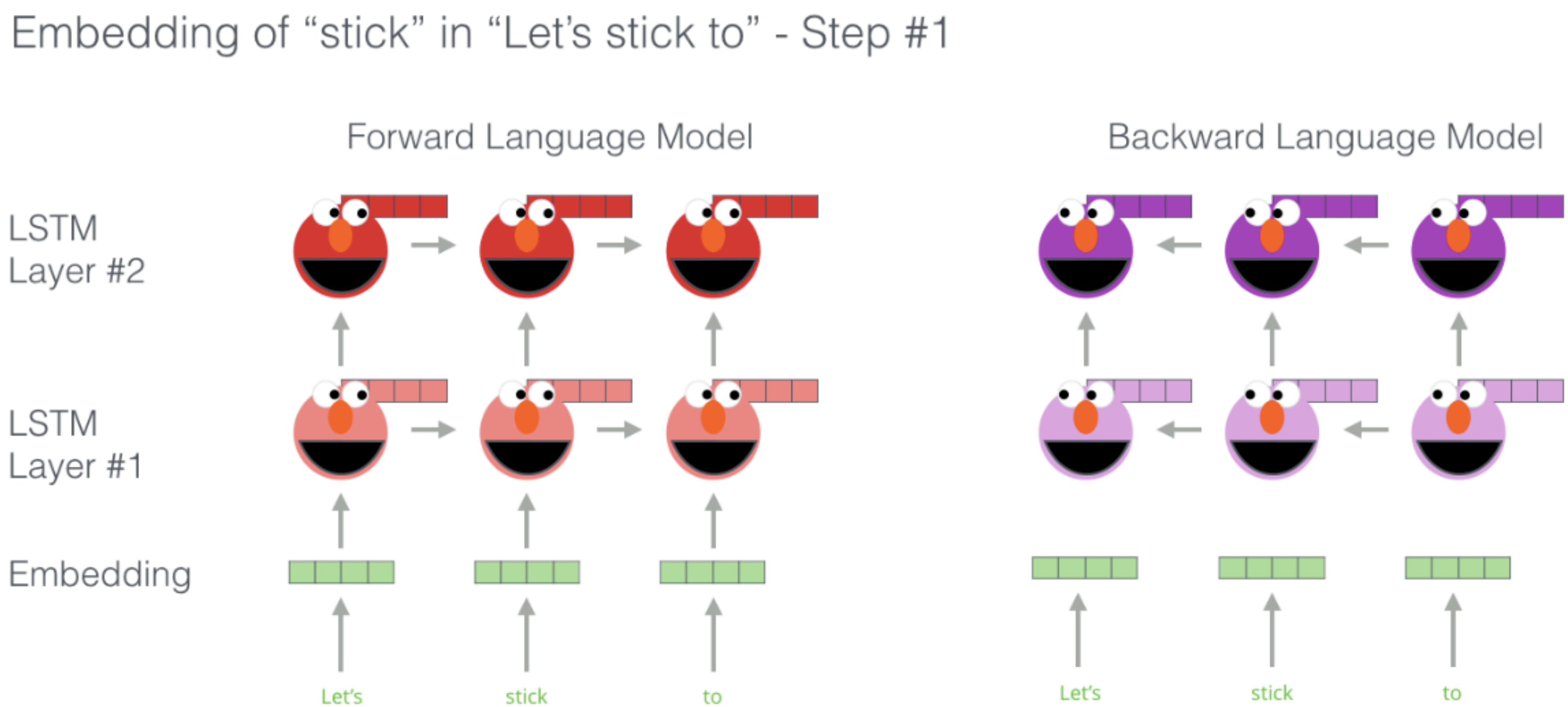
ELMo is a language model

- The output layer predicts the probability of the next word
- A sentence (sequence consisting of tokens) is input
- Each token is represented as an initial embedding
- Hidden states are used as updated embeddings



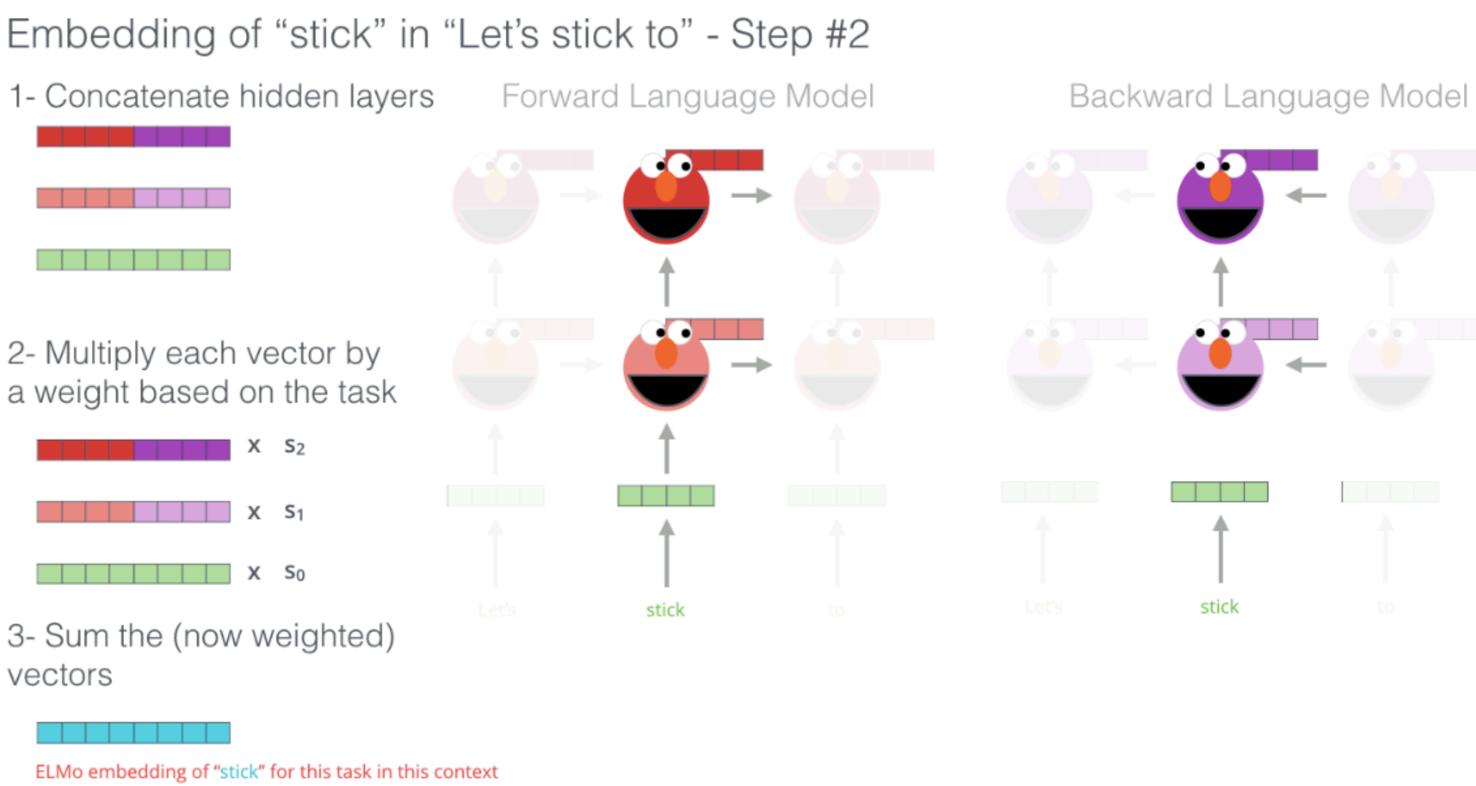
ELMo

ELMo is based on bi-directional LSTMs



ELMo

ELMo is based on bi-directional LSTMs



Limitations of RNN framework

Difficulty with dealing with long sequences

- Capturing and maintaining context over extended sequences
- Long-range dependencies
- Long training time

Attention and Transformer

Self-attention

- The primary goal of self-attention is to generate the representations of the words in a sequence
- Self-attention allows each token of the input sequence to ‘attend’ to (or reference) all other parts of the sequence (including self)
- Attention is quantified as weights that indicate how much focus should be put on other tokens of the sequence in generating the representing of a given token
- E.g., “I took an `umbrella` because it was `raining`”

Attention and Transformer

Previous example

- *Open a* *bank* *account* $\rightarrow e_{bank_{s1}}$: [0.3, 0.9, …]
- *On the river* *bank* $\rightarrow e_{bank_{s2}}$: [0.8, 0.1, …]

To capture contextual meanings, the embeddings (i.e., representations) of the tokens are updated based on the relationships between tokens

- $e_{bank_{s1}}$ should be similar to $e_{account_{s1}}$, and $e_{bank_{s2}}$ should be similar to $e_{river_{s2}}$

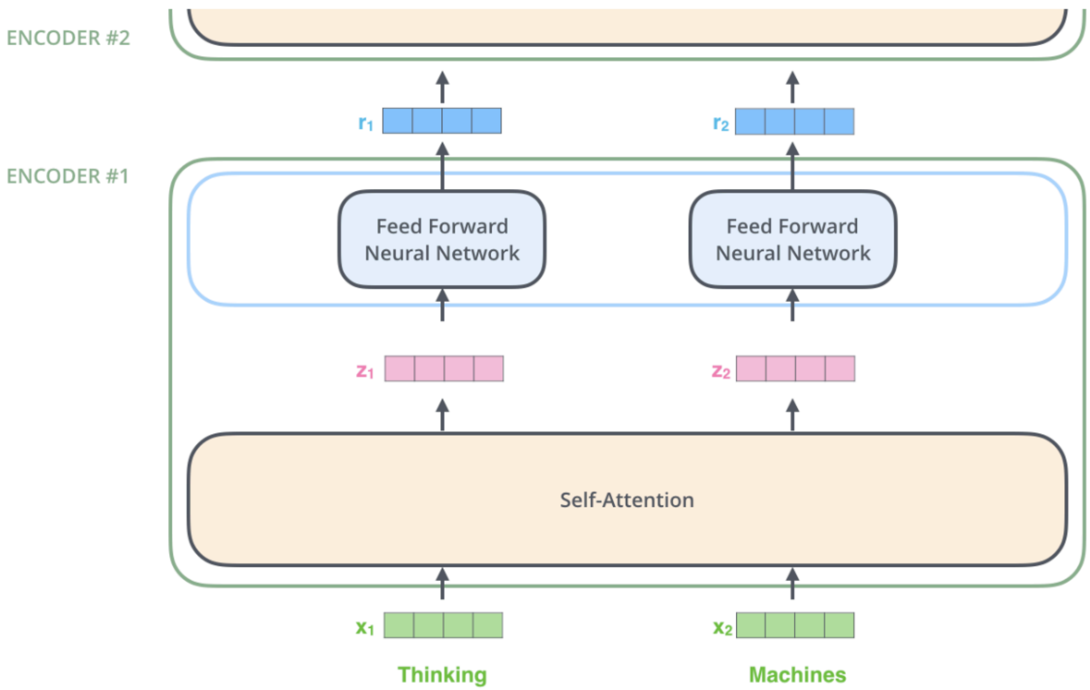
Attention and Transformer

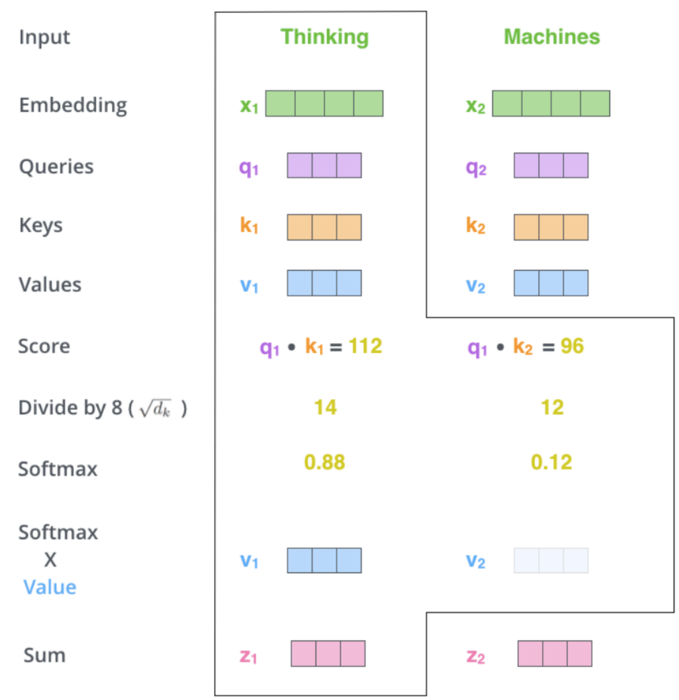
What is Transformer

- Transformer is a type of neural network architecture with self-attention mechanisms
    - Introduced in 2017: link to the paper
    - Cited 120,165 times as of May 14 2024
- Transformers consist of a stack of (encoder/decoder) layers with self-attention
- Transformer dispenses with recurrence and instead relies entirely on attention mechanisms

Attention and Transformer

Transformer encoder





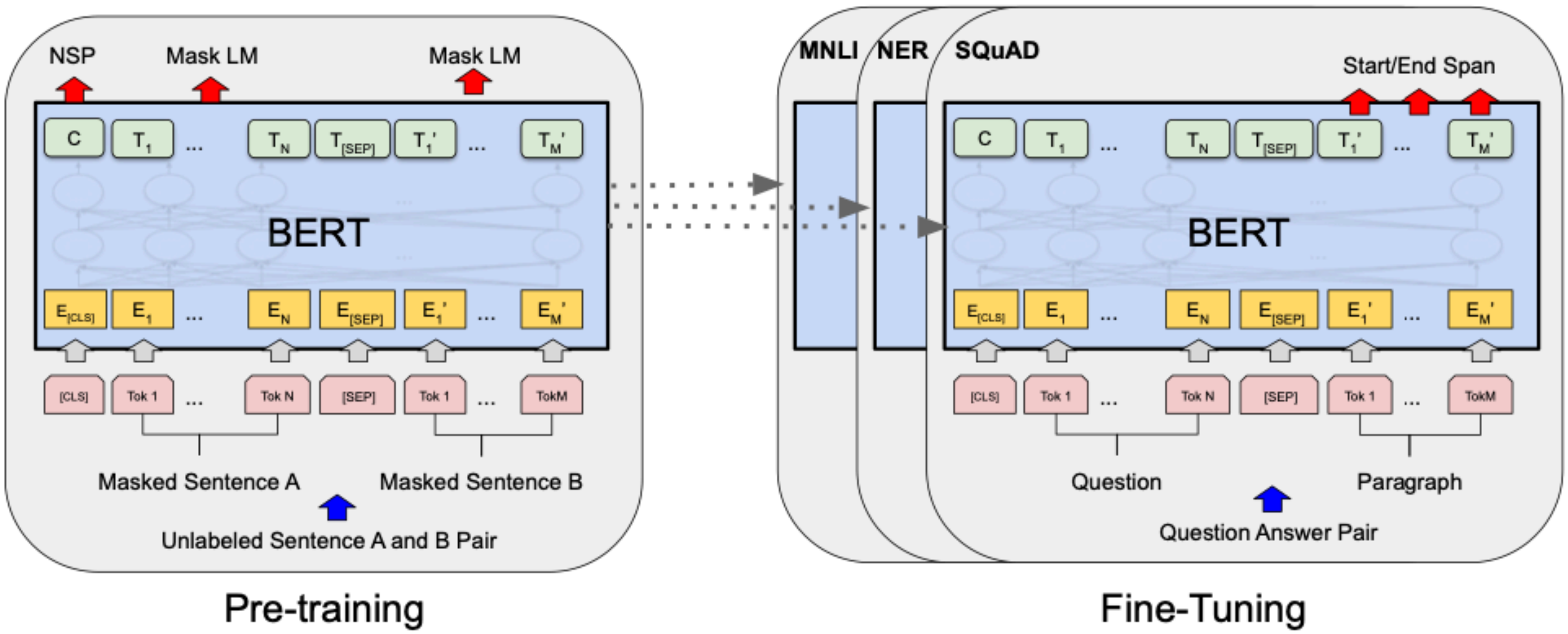
BERT

Bi-directional **E**ncoder **R**epresentations from **T**ransformers

- A form of transformer trained as a language model
  - Two tasks: masked token prediction, next sentence prediction
- Introduced in 2018: link to the paper
- Cited 99,718 times as of May 14 2024
- Pre-trained on huge data sets (BookCorpus and Wikipedia)
- Two versions of the model introduced in the paper
  - BERT BASE (12 encoder stacks)
  - BERT LARGE (24 encoder stacks)

BERT

Pretraining vs. Fine-tuning



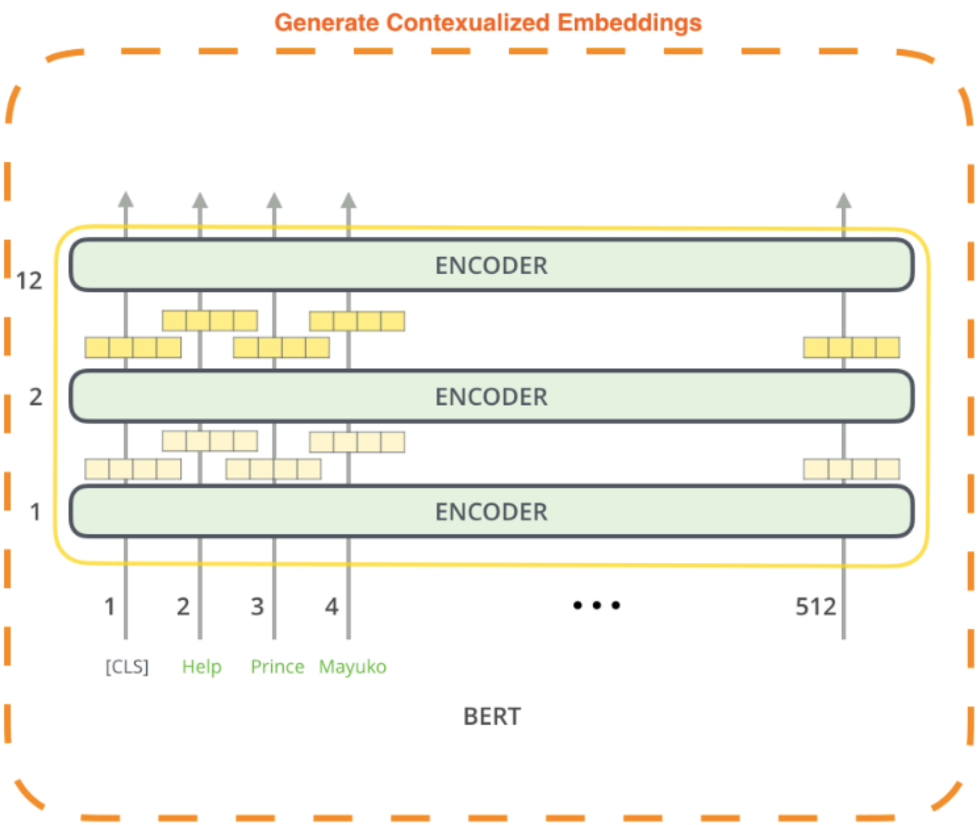
BERT

Peak at under the hood

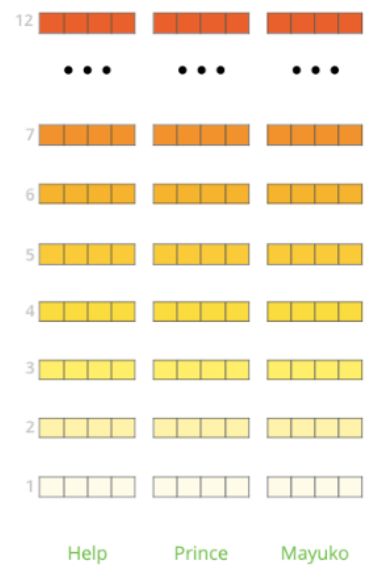
- BERT generates its own embeddings (from scratch) as part of its training process
- Stack of Transformer encoder layers involving “multi-head” self-attention
- Each layer passes its results through a feed-forward network, and then hands it off to the next encoder in the stack
- Each position outputs a vector of size 768 (BASE) or 1024 (LARGE)

BERT

Extracting embeddings



The output of each encoder layer along each token's path can be used as a feature representing that token.



But which one should we use?

BERT

(Some of) practical usage in applied research

- Fine-turning for text classification
  - E.g., idenfiying statements and stance on immigration (p. 7 *Classification* in Card et al. 2022)
- The pre-trained model (and embeddings) is of interest
  - E.g., identifying dehumanizing metaphors against immigrants (p. 8 *Measuring Dehumanization* in Card et al. 2022)
- Topic models
  - E.g., BERTopic

Topic modeling with contextual embeddings

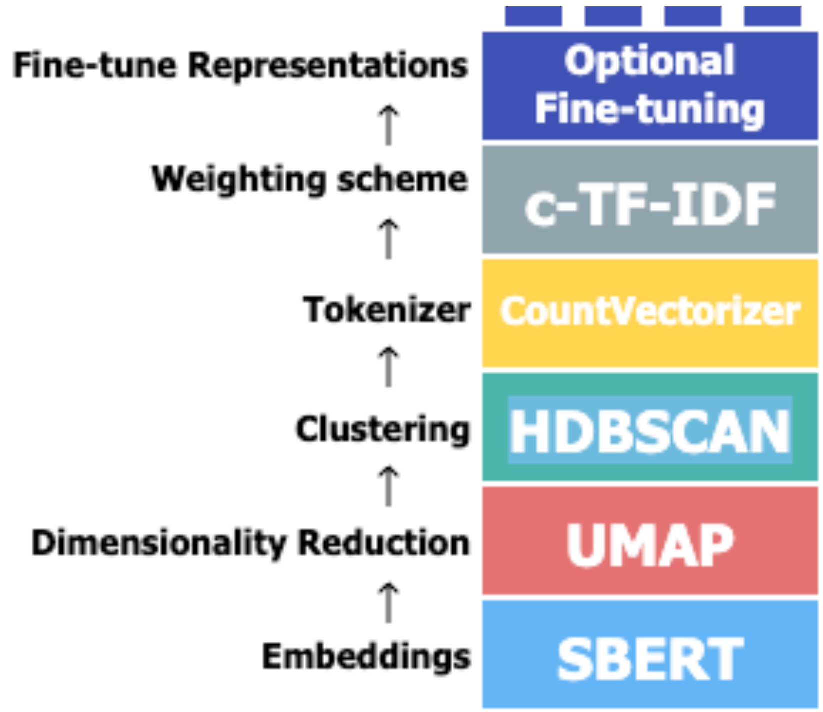
Drawbacks for conventional models

- LDA, CTM, STM, DTM, etc.
- Bag-of-Words model
- Lack of encoding contextual information

Topic modeling with contextual embeddings

Major steps in BERTopic

- Document embeddings: SBERT
- Dimensionality reduction: UMAP
- Clustering: HDBSCAN
- Topic representations: class-based TF-IDF



Summary

- Modern NLP models based on neural networks excel at capturing contextual information
- They are not only useful for the tasks they were originally designed for, but also for their internal representations as contextual embeddings
  - The embeddings themselves are of interest
  - Modeling thematic strcutre of a corpus