CSC520 - Artificial Intelligence

Lecture 24

Dr. Scott N. Gerard

North Carolina State University

Apr 15, 2025

Language Model

- Language Model $=$ is a probability distribution over word sequences.
  - Sentence auto-completion
    "Bob drinks ..." {coffee:0.6, tea:0.1, sprite:0.05, ...}
  - Speech-to-text
    P("eyes awe of an") $<$ P("I saw a van")
  - Word correction
    P("Alice boarded a *ship* at the airport") $<$ P("Alice boarded a *plane* at the airport")

N-Gram Smoothing

Assume our vocabularly is

Word	Count	Add-0	Add-1	Add-2
Artificial	6	$\frac{6}{10} = 0.6$	$\frac{7}{14} = 0.5000$	$\frac{8}{18} = 0.4444$
Book	3	$\frac{3}{10} = 0.3$	$\frac{4}{14} = 0.2857$	$\frac{5}{18} = 0.2778$
Computer	1	$\frac{1}{10} = 0.1$	$\frac{2}{14} = 0.1429$	$\frac{3}{18} = 0.1667$
Decision	0	$\frac{0}{10} = 0.0$	$\frac{1}{14} = 0.0714$	$\frac{2}{18} = 0.1111$
Denominator	-	$10 + 0 = 10$	$10 + 4 = 14$	$10 + 8 = 18$

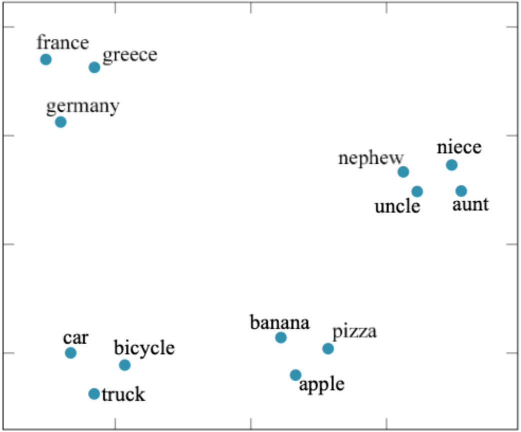
Agenda

- Word embeddings
- Neural network for POS tagging
- Recurrent neural networks
- RNN language model
- RNN-LSTM model for machine translation
- Transformers and self-attention

Word Embeddings

- Word embedding is a low-dimensional vector representation of a word
- Conceptually, we can think of each element in the vector as some feature
- Word embeddings are more efficient than one-hot encoding in most NLP tasks
- Pre-trained word embeddings are readily available for use
  - WORD2VEC, GloVe (Global Vectors), FASTTEXT
  - Form of transfer learning
- Word embeddings can also be trained as part of training a neural network

Word Embeddings - Clusters



Word Embeddings - Analogies

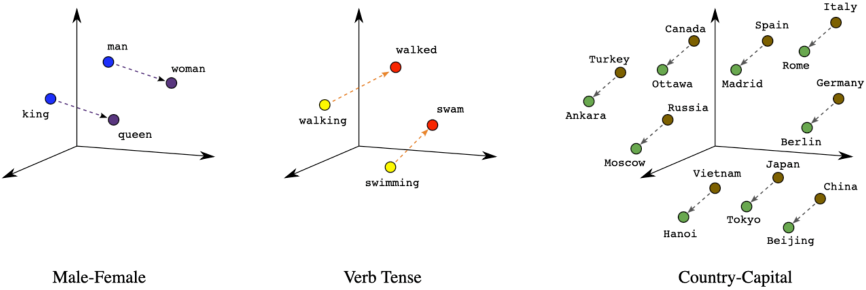
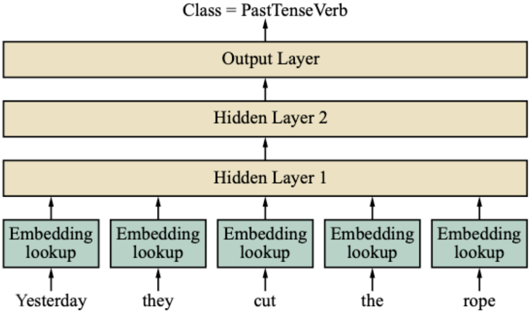


Image from developers.google.com

Man is to woman as king is to ??

$$\vec{king} - \vec{man} + \vec{woman} \approx \vec{queen}$$

Neural Network for POS Tagging



$$\hat{y} = softmax(W_{out} z_2)$$
$$z_2 = g(W_2 z_1)$$
$$z_1 = g(W_1 x)$$

	$e1$	$e2$	...	e_n
yesterday	-0.24344	-0.60601	...	0.33112
they	0.70835	-0.57361	...	0.15375
cut	0.25322	-0.34355	...	0.79544
the	...			
rope	...			

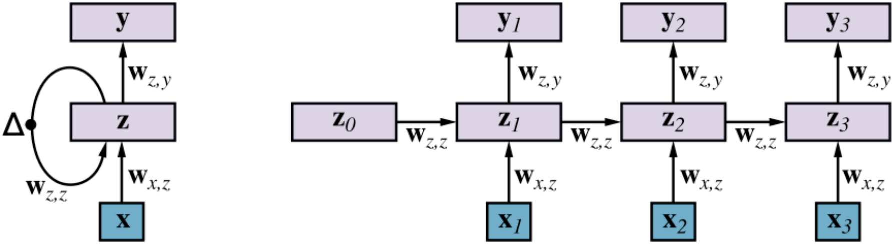
$$softmax(\vec{z}) = \frac{e^{z_i}}{\sum_i e^{z_i}}$$

z	softmax(z)
-1	0.00086
3	0.04723
6	0.94950
0	0.00235

Recurrent Neural Network

- Fully connected NN can be used for fixed length inputs and outputs
  - In NLP, training examples (sentences) can have different lengths
  - And the output lengths may also vary
- Fully connected NN learn different weights for each input
  - For a given word, we would like to share weights even if it appears in different positions
- RNN addresses these problems

Recurrent Neural Network



$$z_t = g(W_{z,z} z_{t-1} + W_{x,z} x_t + b_z)$$
$$\hat{y}_t = g(W_{z,y} z_t + b_y)$$

- if $W_{z,z} < 1$ then vanishing gradients problem
- if $W_{z,z} > 1$ then exploding gradients problem

Different Kinds of RNN Models

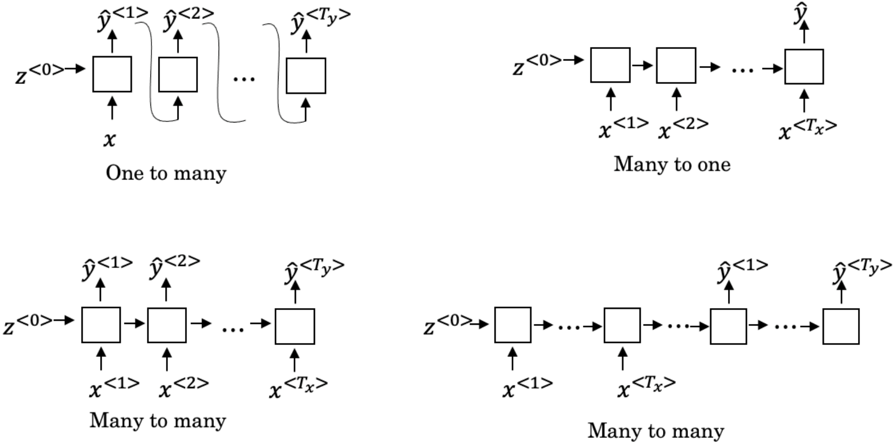


Image credit: Andrew Ng

LSTM

- RNN network cannot capture long range dependencies effectively

  The athletes, who all won their local qualifiers and advanced to the finals in Tokyo, now *compete or competes* . . .

- Long-short term memory (LSTM) solves this problem

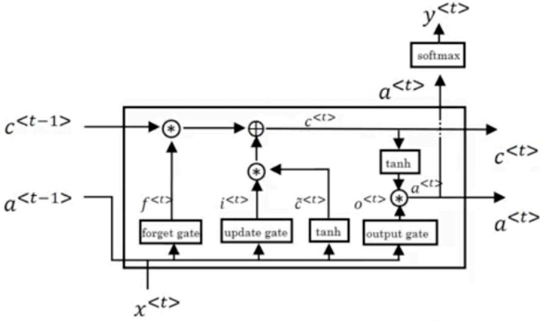
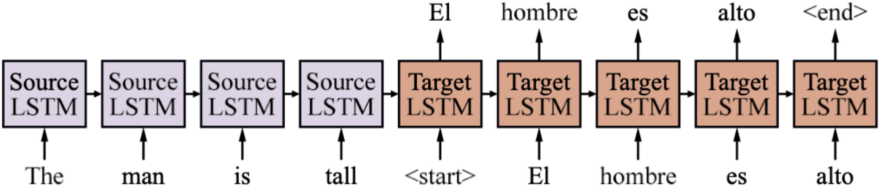


Image Credit: Andrew Ng

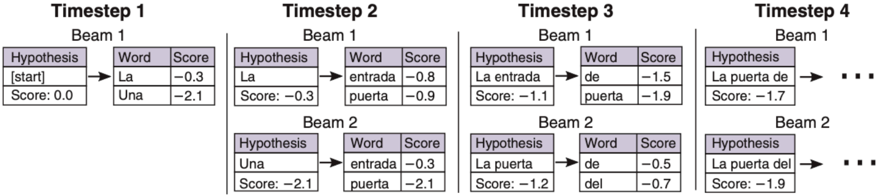
Sequence-to-sequence Models for Machine Translation

- Sequence-to-sequence models use two RNNs: encoder and decoder
- Sentence in source language is encoded using one RNN model
- Hidden state output of the source RNN is fed into a target RNN
- Target RNN generates sentence in target language that corresponds to the source language



Beam Search

- During decoding, a target word is generated one at a time and fed back as input to next step
- Greedy decoding: a word with highest probability is selected
  - Does not necessarily maximize the probability of the entire sentence
- Beam search: keep the top k possibilities at each stage



English: The door of entry is red
Spanish: La puerta de entrada es roja

Limitations of RNN Models

- Nearby context bias: hidden state has more information about closer words than farther words
- Fixed context size limit: entire source sentence is encoded in a fixed size hidden state vector
  - Vector may be insufficient to represent encoding of long sentences
  - Increasing the hidden vector size slows down training and causes overfitting
- Slower sequential processing: each timestep (word) needs to be processed sequentially

Transformers and Self-Attention

- Transformers address the problems in RNN models
- Introduced in a highly influential paper titled: “Attention is all you need” (Vaswani et.al, 2018)

Transformers and Self-Attention

- Transformers address the problems in RNN models
- Introduced in a highly influential paper titled: "Attention is all you need" (Vaswani et.al, 2018)
- Recurrence is replaced with a self-attention layer
  - Self-attention layer computes how much attention each word should pay to every other word in the sequence

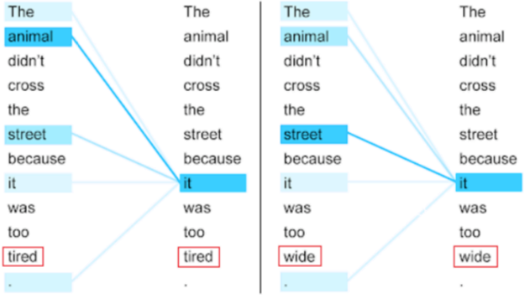


Image credit: John Bjorkman Nilsson

Self-Attention Layer

- Input is projected into three different representations
  - Query vector: $q_i = W_q x_i$
  - Key vector: $k_i = W_k x_i$
  - Value vector: $v_i = W_v x_i$

- Encoding of i^{th} word is calculated as follows.

$$r_{ij} = (q_i . k_j)/\sqrt{d}$$

$$a_{ij} = e^{r_{ij}} / \sum_k e^{r_{ik}}$$

$$z_i = \sum_j a_{ij} . v_j$$

Self-Attention Layer

- Input is projected into three different representations
  - Query vector: $q_i = W_q x_i$
  - Key vector: $k_i = W_k x_i$
  - Value vector: $v_i = W_v x_i$
- Encoding of i^{th} word is calculated as follows.

$$r_{ij} = (q_i . k_j)/\sqrt{d}$$

$$a_{ij} = e^{r_{ij}} / \sum_k e^{r_{ik}}$$

$$z_i = \sum_j a_{ij} . v_j$$

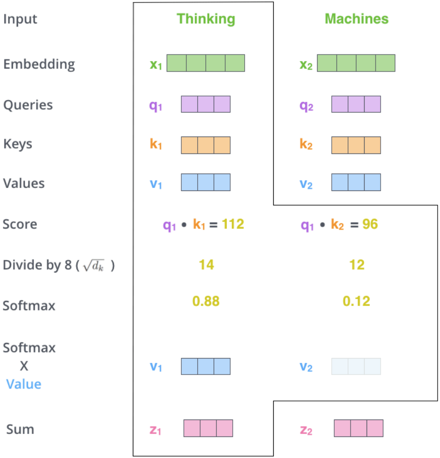
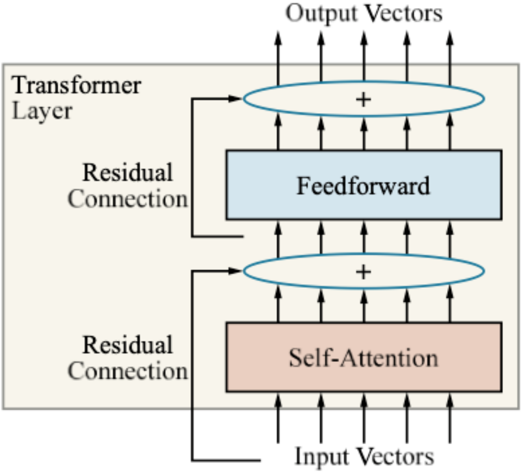


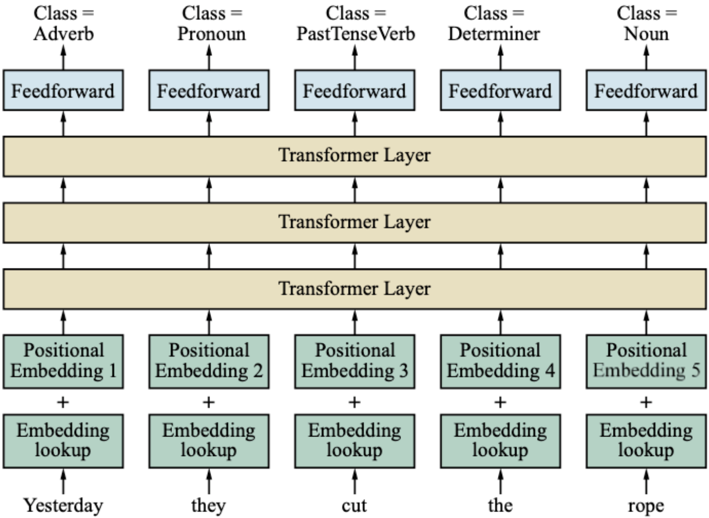
Image credit: Jay Alammar

Transformer

- Single-layer transformer has a self-attention, a feedforward layer and residual connections



Transfomer Model for POS tagging



State of the Art Models

- Large language models are used widely
- Pre-trained on a large text corpus and then fine-tuned for specific NLP tasks
  - Fine-tuning involves adding some layers and training the model using small amounts of labeled data
- Self-supervised training: model is trained to predict masked words, e.g. "the river *rose* five feet"
- Couple examples of such models are BERT and GPT3
  - BERT (Bidirectional encoder representations from transformers) is encoder-only bidirectional transformer with roughly 350M parameters
  - GPT3 (Generative Pre-trained Transformer-3) is the decoder-only transformer with 175B parameters (largest size)
  - GPT3 is trained on much larger corpus than BERT