Machine Learning, Artificial Intelligence, and Natural Language Processing

Marcelo C. Medeiros

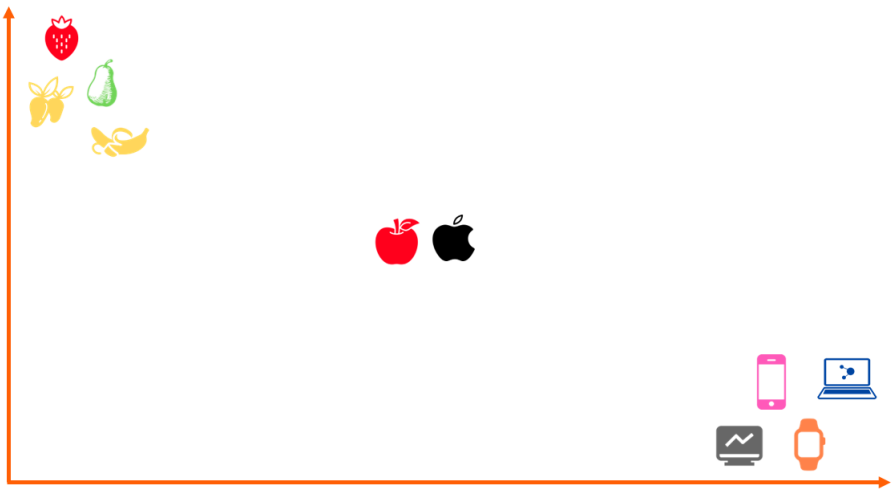
Department of Economics
The University of Illinois at Urbana-Champaign

Attention

Transformers

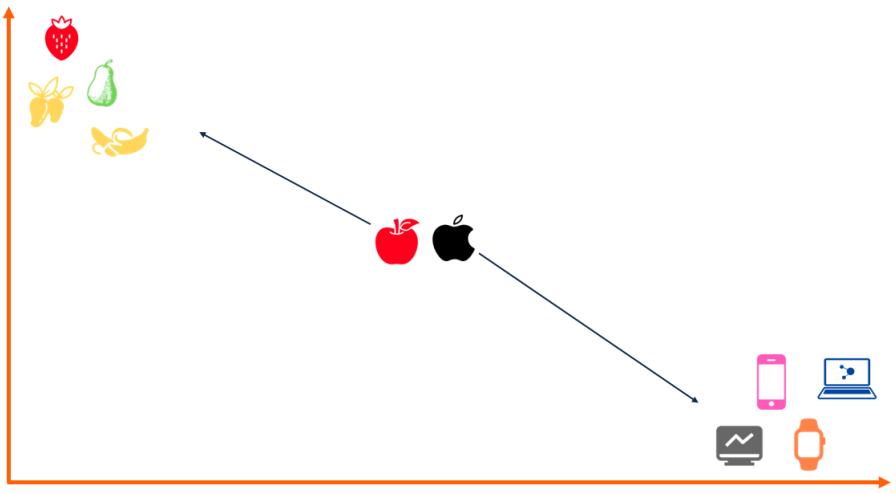
Attention Mechanism

Searching for a better word embedding



Attention Mechanism

Searching for a better word embedding



Attention Mechanism

Searching for a better word embedding

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

- They are static – The embedding for a word does not reflect how its meaning changes in context

Attention Mechanism

The problem of static embeddings - like Word2Vec

- They are static – The embedding for a word does not reflect how its meaning changes in context
- For example,

`The chicken didn't cross the road because it was too tired`

- They are static – The embedding for a word does not reflect how its meaning changes in context
- For example,

  `The chicken didn't cross the road because it was too tired`

  - What is the meaning represented in the static embedding for "it"?

Attention Mechanism

The problem of static embeddings - like Word2Vec

- They are static – The embedding for a word does not reflect how its meaning changes in context
- For example,

  `The chicken didn't cross the road because it was too tired`

  - What is the meaning represented in the static embedding for "it"?
- Consider now

  `The chicken didn't cross the road because it was too wide`

Attention Mechanism

The problem of static embeddings - like Word2Vec

- They are static – The embedding for a word does not reflect how its meaning changes in context
- For example,

  `The chicken didn't cross the road because it was too tired`

  * What is the meaning represented in the static embedding for "it"?
- Consider now

  `The chicken didn't cross the road because it was too wide`

  * Is the meaning the same?

- Problem: The representation of the meaning of a word should be different in different contexts
- Contextual Embedding: each word has a different vector that expresses different meanings depending on the surrounding words
- Solution: Attention mechanism.

`The chicken didn't cross the road because it ...`

- What should be the properties of "it"?

`The chicken didn't cross the road because it was too tired`

`The chicken didn't cross the road because it was too wide`

- At this point in the sentence, it's probably referring to either the chicken or the street

- Build up the contextual embedding from a word by selectively integrating information from all the neighboring words

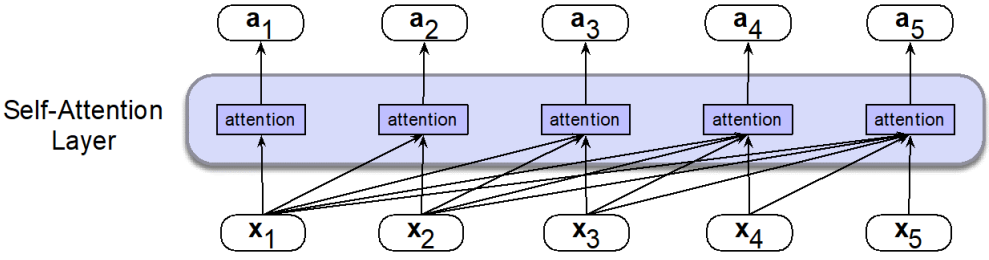
- A word "attends to" some neighboring words more than others

- Attention is a mechanism for helping compute the embedding for a token by selectively attending to and integrating information from surrounding tokens

- Mathematically: a method for computing a ("smart") weighted sum of vectors

Attention Mechanism

Left-to-right



Attention Mechanism

Simplified version

- Sequence of token embeddings: $\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3, \boldsymbol{x}_4, \boldsymbol{x}_5, \boldsymbol{x}_6, \boldsymbol{x}_7, \boldsymbol{x}_i$
- Measure distance between two tokens: inner product

$$\text{score}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{x}_i' \boldsymbol{x}_j$$

- Transform the score into weights between 0 and 1:

$$\alpha_{ij} = \text{softmax}(\text{score}(\boldsymbol{x}_i, \boldsymbol{x}_j)) = \frac{e^{\text{score}(\boldsymbol{x}_i, \boldsymbol{x}_j)}}{\sum_{k=1}^{8} e^{\text{score}(\boldsymbol{x}_i, \boldsymbol{x}_k)}}, \quad j \leq i$$

- Attention:

$$\boldsymbol{a}_i = \sum_{j \leq i} \alpha_{ij} \boldsymbol{x}_j$$

- High-level idea: instead of using embedding vectors directly, we will represent three separate roles each vector $\boldsymbol{x}_i$ plays:

1. query: element that is being compared to the preceding inputs
2. key: preceding input that is being compared to the current element to determine a similarity
3. value: a value of a preceding element that gets weighted and summed

- Matrices to project each vector $\boldsymbol{x}_i$ into a representation of its role as query, key, or value
  - Query: $\boldsymbol{Q}$
  - Key: $\boldsymbol{K}$
  - Value: $\boldsymbol{V}$
- Therefore,

$$\boldsymbol{q}_i = \boldsymbol{Q}\boldsymbol{x}_i \quad \boldsymbol{k}_i = \boldsymbol{K}\boldsymbol{x}_i \quad \boldsymbol{v}_i = \boldsymbol{V}\boldsymbol{x}_i$$

- The three matrices are learned from data

- Take the representation in terms of query, key, and value matrices

$$\boldsymbol{q}_i = \boldsymbol{Q}\boldsymbol{x}_i \quad \boldsymbol{k}_i = \boldsymbol{K}\boldsymbol{x}_i \quad \boldsymbol{v}_i = \boldsymbol{V}\boldsymbol{x}_i$$

- Similarity (score) between $\boldsymbol{x}_i$ and prior token $\boldsymbol{x}_j$: $\boldsymbol{q}_i'\boldsymbol{k}_j/\sqrt{d}$
- d is the dimension of the embeddings
- Therefore,

$$\alpha_{ij} = \text{softmax}(\boldsymbol{q}_i'\boldsymbol{k}_j), \quad j \leq i$$

- Attention:

$$\boldsymbol{a}_i = \sum_{j \leq i} \alpha_{ij}\boldsymbol{v}_j$$

- Instead of one attention head, we'll have lots of them
- Intuition: each head might be attending to the context for different purposes

- For each head h:

$$\boldsymbol{q}_i^h = \boldsymbol{Q}^h \boldsymbol{x}_i \quad \boldsymbol{k}_i^h = \boldsymbol{K}^h \boldsymbol{x}_i \quad \boldsymbol{v}_i^h = \boldsymbol{V}^h \boldsymbol{x}_i$$

- Similarity (score) between $\boldsymbol{x}_i^h$ and prior token $\boldsymbol{x}_j$: $\boldsymbol{q}_i^{h\prime} \boldsymbol{k}_j^h / \sqrt{d_h}$
- $d_h = d/H$, H is the number of heads
- Therefore,

$$\alpha_{ij}^h = \mathsf{softmax}(\boldsymbol{q}^{h\prime}{}_i \boldsymbol{k}_j^h), \quad \text{and} \quad \boldsymbol{a}_i^h = \sum_{j \le i} \alpha_{ij}^h \boldsymbol{v}_j^h$$

- Hence,

$$\boldsymbol{a}_i = \left(\boldsymbol{a}_i^1 \cdots \boldsymbol{a}_i^H \right) \boldsymbol{O}$$

Attention = contextual embedding

Attention

Transformers

Transformers

- Goal: given a sequence of words, predict which word will come next

- It is based on NN architectures plus attention mechanism

Transformers

The big picture