OpenAI Vector Embeddings

Creating and Using Embeddings

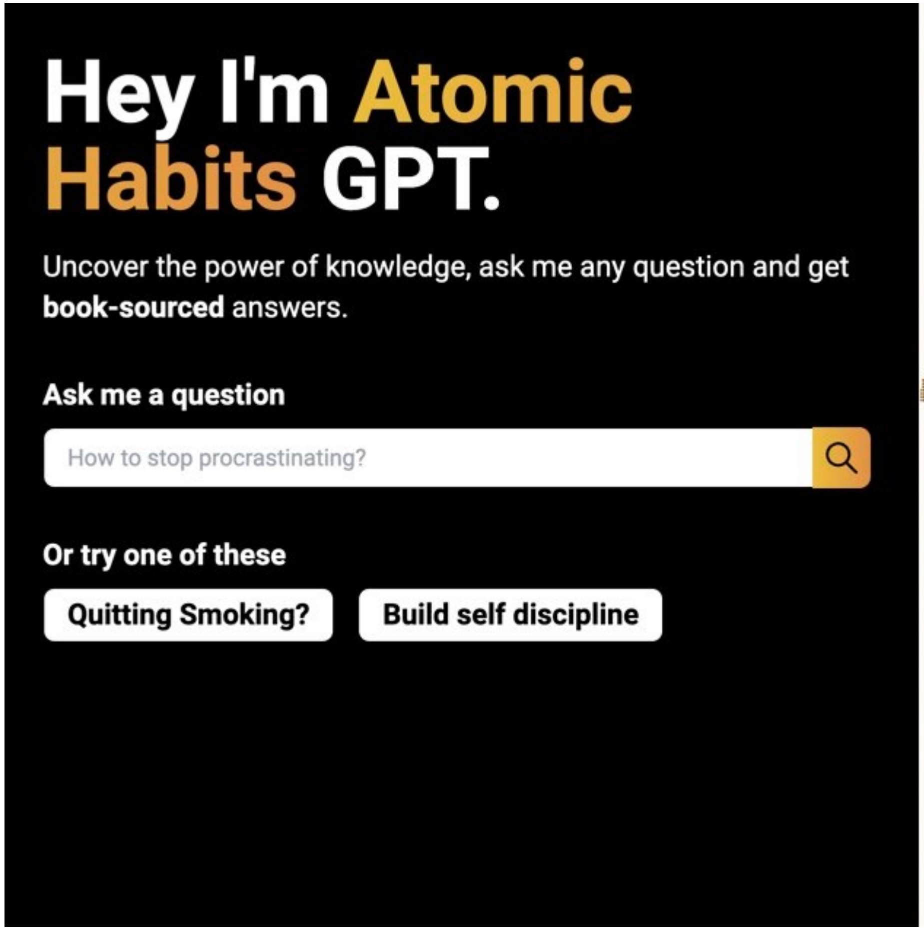
Presented by:

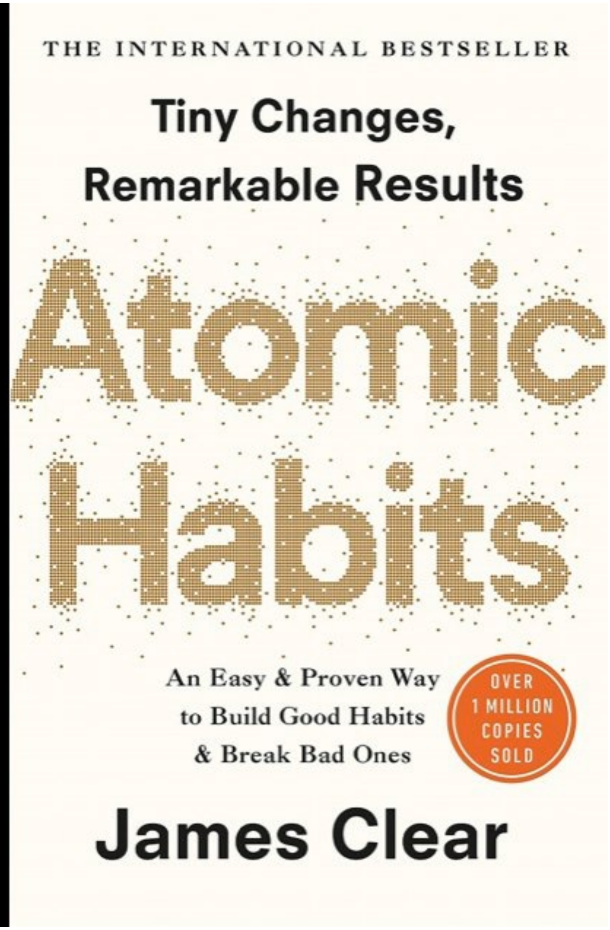
Manas Bam

John Tan Chong Min

Apps Showcase (Manas)

Manas Bam
NTU Freshman and Developer
Twitter:@bamitsmanas



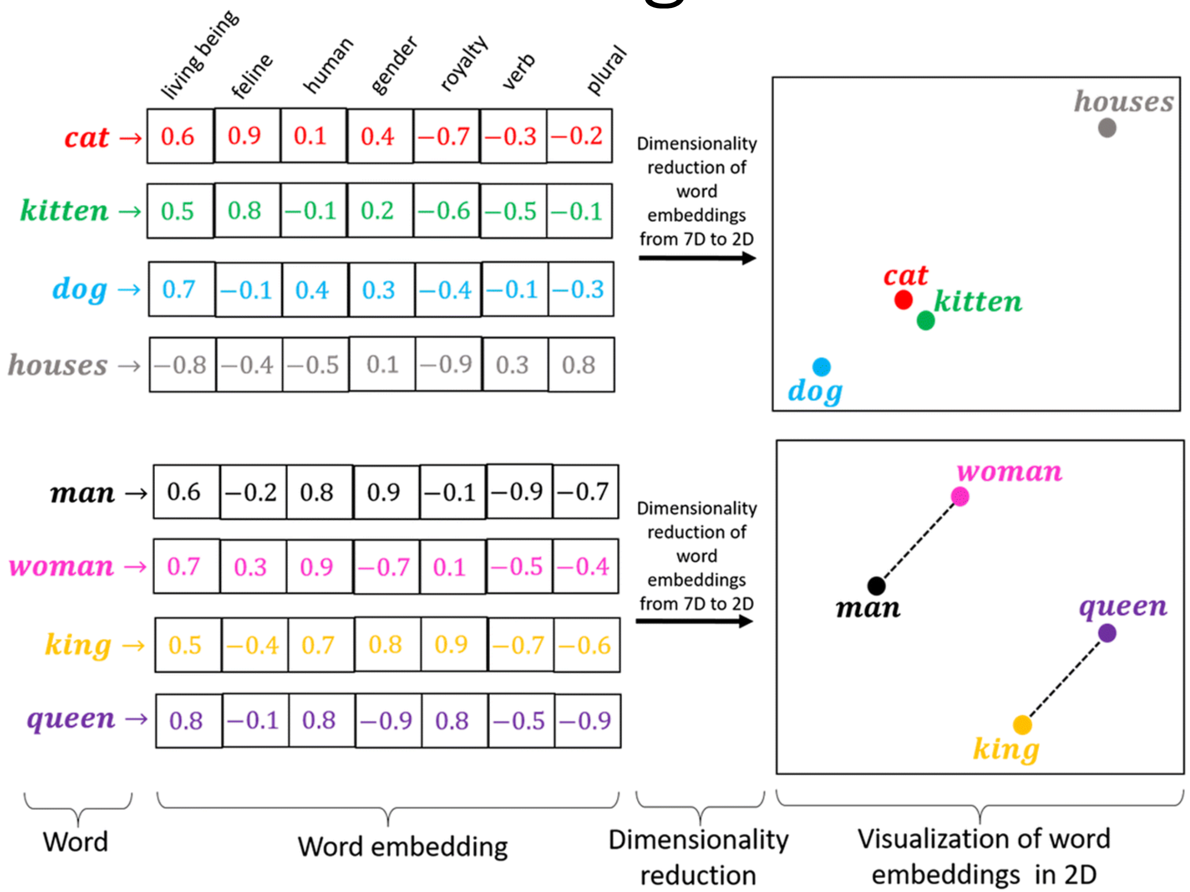


https://www.gptbook.club/atomic-habits

Embedding Space

Semantic Meaning

Word Embeddings



- Extracting semantic meaning in higher-dimensional space
- Number of dimensions depends on use case

Taken from: https://medium.com/@hari4om/word-embedding-d816f643140

Traditional Approach:
TF-IDF (term frequency-inverse document frequency)

Term Frequency: TF of a term or word is the number of times the term appears in a document compared to the total number of words in the document.

$$TF = \frac{\text{number of times the term appears in the document}}{\text{total number of terms in the document}}$$

Importance of term in a document

Inverse Document Frequency: IDF of a term reflects the proportion of documents in the corpus that contain the term. Words unique to a small percentage of documents (e.g., technical jargon terms) receive higher importance values than words common across all documents (e.g., a, the, and).

$$IDF = log(\frac{\text{number of the documents in the corpus}}{\text{number of documents in the corpus contain the term}})$$

How common this term is in corpus

The TF-IDF of a term is calculated by multiplying TF and IDF scores.

$$TF\text{-}IDF = TF * IDF$$

https://www.learndatasci.com/glossary/tf-idf-term-frequency-inverse-document-frequency/

Issues of TF-IDF

- Only exact match of words allowed
- Ignores word order
- Does not take into account semantically similar words
- Need to calculate term counts across all documents

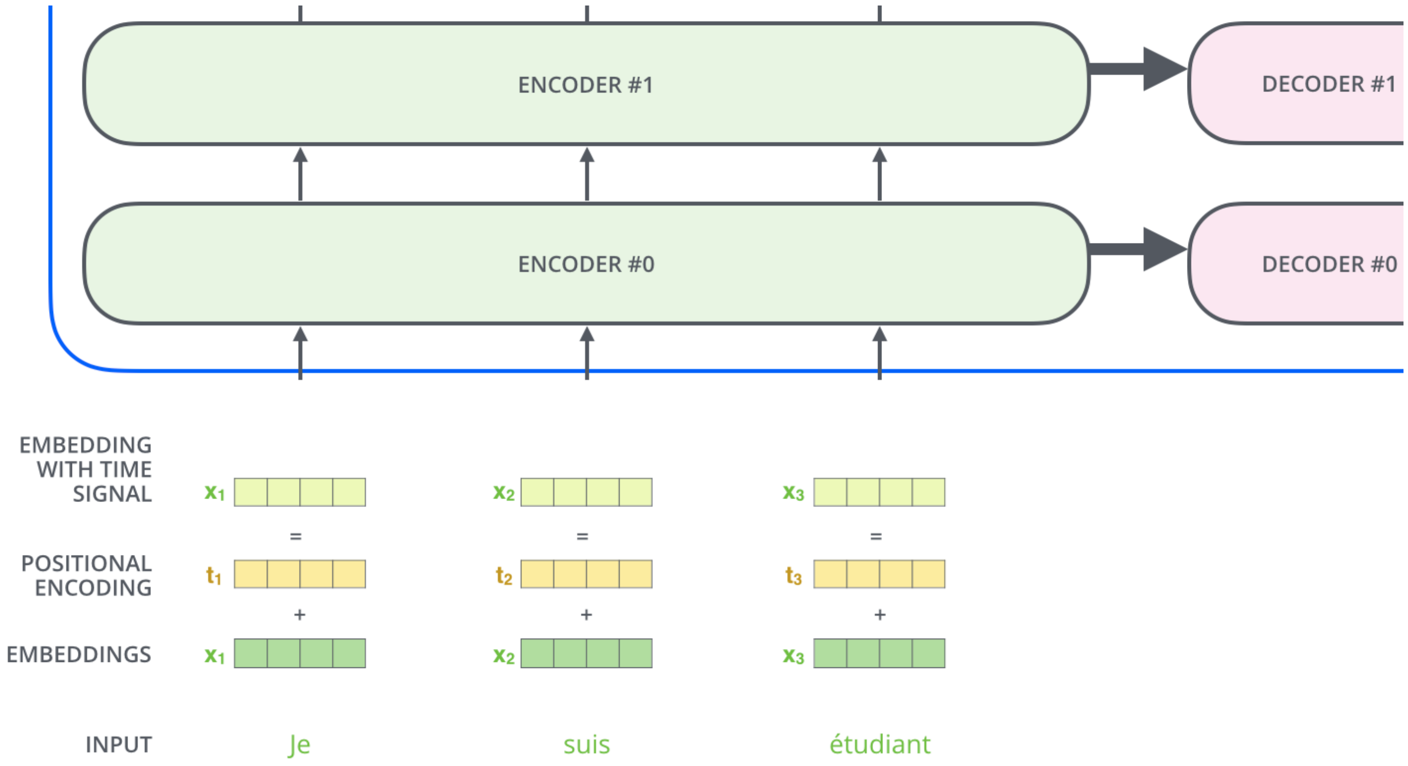
New Approach:
Vector-based lookup for sentences/paragraphs?

- OpenAI's text embeddings measure the relatedness of text strings. Embeddings are commonly used for:
  - **Search** (where results are ranked by relevance to a query string)
  - **Clustering** (where text strings are grouped by similarity)
  - **Recommendations** (where items with related text strings are recommended)
  - **Anomaly detection** (where outliers with little relatedness are identified)
  - **Diversity measurement** (where similarity distributions are analyzed)
  - **Classification** (where text strings are classified by their most similar label)

https://platform.openai.com/docs/guides/embeddings/what-are-embeddings

How are embeddings created?

Token Embeddings Learned by Backprop



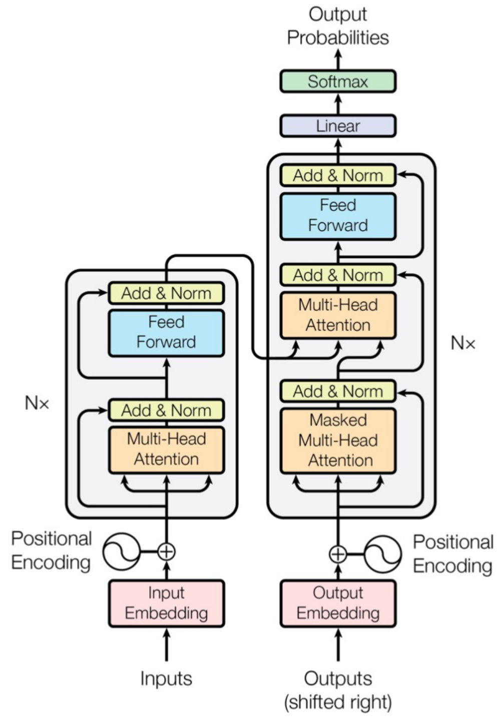


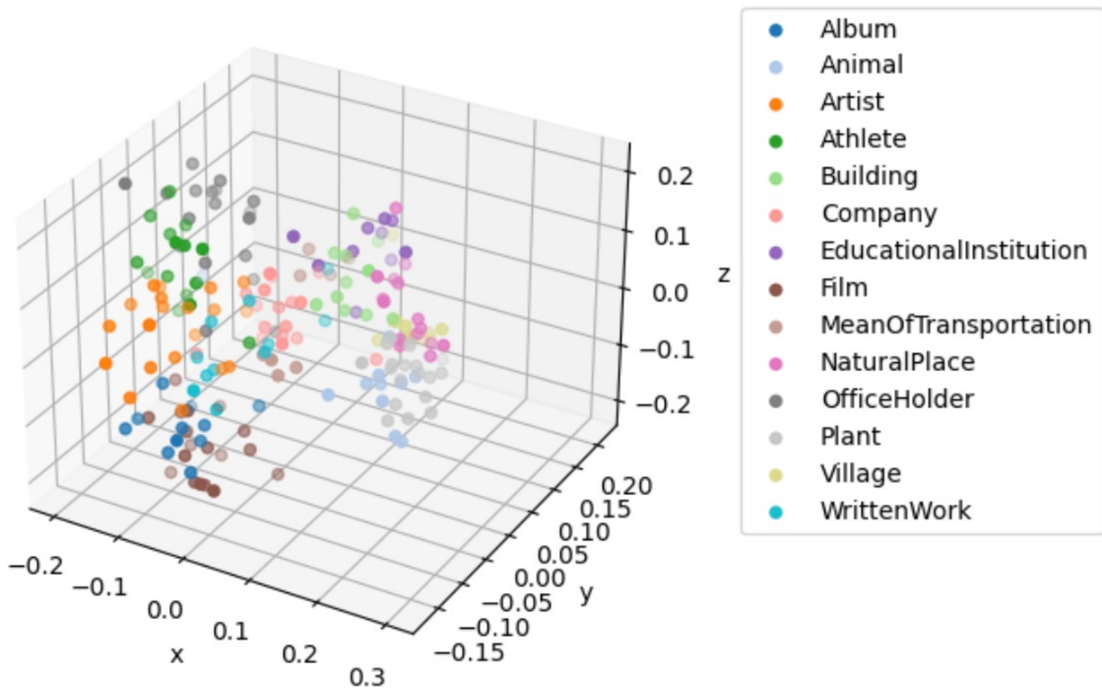
Figure 1: The Transformer - model architecture.

https://jalammar.github.io/illustrated-transformer/

DBpedia 3D Embeddings Visualization

- DBpedia dataset extracts structured information from Wikipedia

- Principal Component Analysis (PCA) to reduce the dimensionality of the embeddings from 1536 to 3



https://github.com/openai/openai-cookbook/blob/main/examples/Visualizing_embeddings_in_3D.ipynb

Contrastive Learning

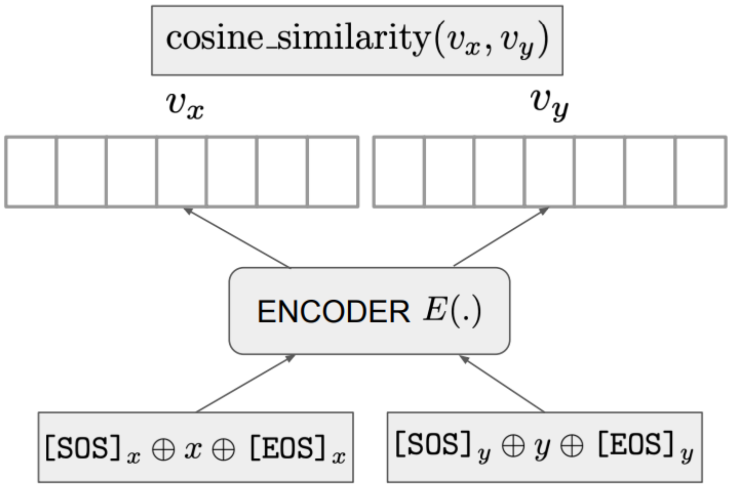


Figure 3. The encoder E maps inputs x and y, to embeddings, v_x and v_y independently. The similarity score between x and y is defined as the cosine similarity between these two embedding vectors.

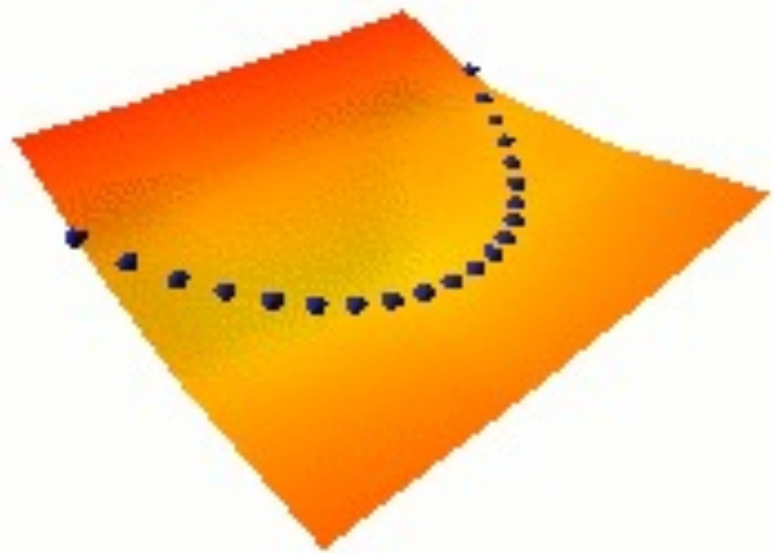
Given a training pair (x, y), a Transformer (Vaswani et al., 2017) encoder E is used to process x and y independently. The encoder maps the input to a dense vector representation or embedding (Figure 2). We insert two special token delimiters, `[SOS]` and `[EOS]`, to the start and end of the input sequence respectively. The hidden state from the last layer corresponding to the special token `[EOS]` is considered as the embedding of the input sequence.

Text and Code Embeddings by Contrastive Pre-training. Neelakantan et al. 2022.

Finding the manifold

- We seek to find the right manifold to represent the data (e.g. input sequence)

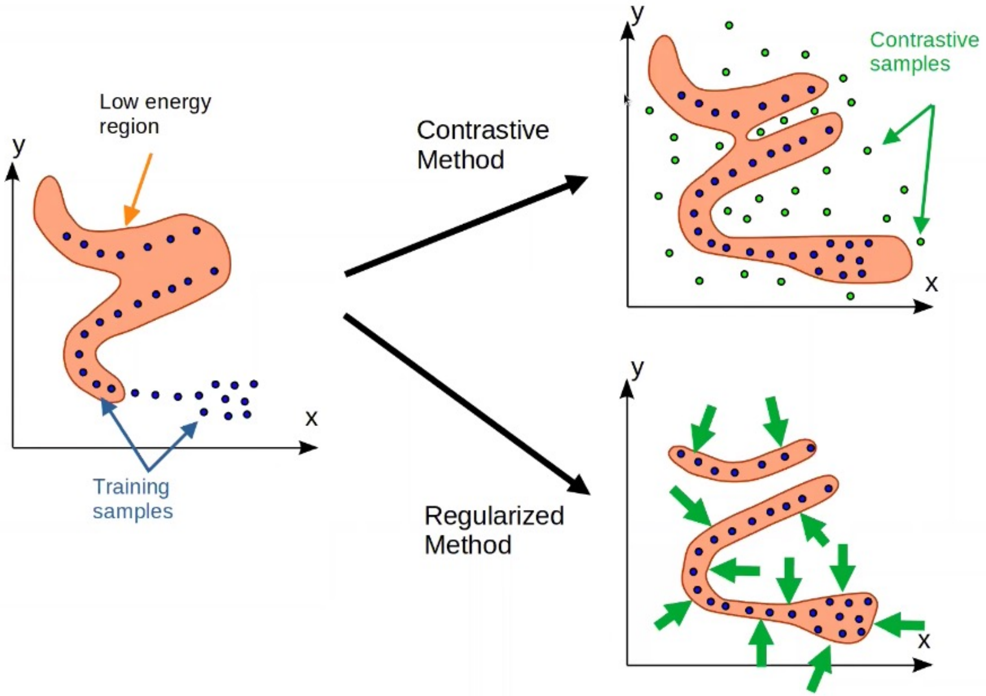
- Contrastive methods can scale exponentially in the number of dimensions



Extracted from: Yann Lecun's slides

Contrastive vs Regularized methods

- **Contrastive methods**
  - Push down on energy of training samples
  - Pull up on energy of suitably-generated contrastive samples
  - Scales very badly with dimension
- **Regularized Methods**
  - Regularizer minimizes the volume of space that can take low energy



Extracted from: Yann Lecun's slides

My thoughts: Why not just take final embedding of the last token?

- Final layer embedding of the final token of the sequence should already contain the semantic information required for the next token
- Just do next-token prediction to learn the initial embedding space
- Run through transformer network to get final embedding

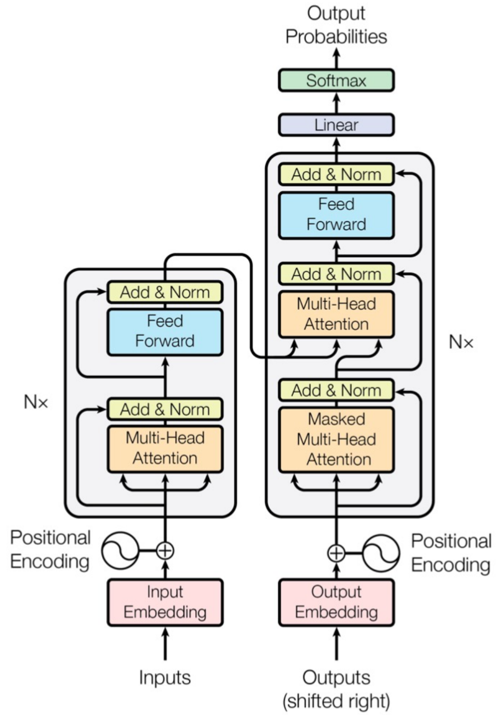
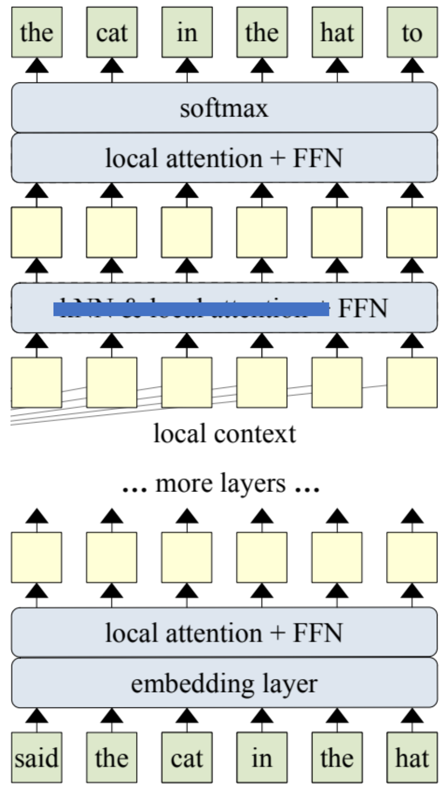


Figure 1: The Transformer - model architecture.

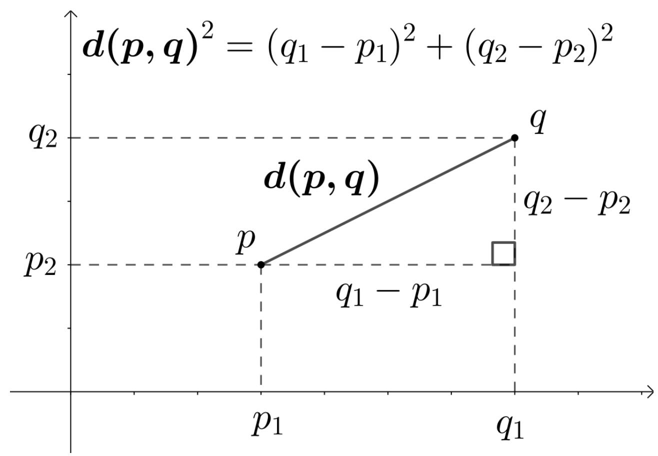


Attention is all you need. Vaswani et al. 2017.
Memorizing Transformers. Wu et al. 2022.

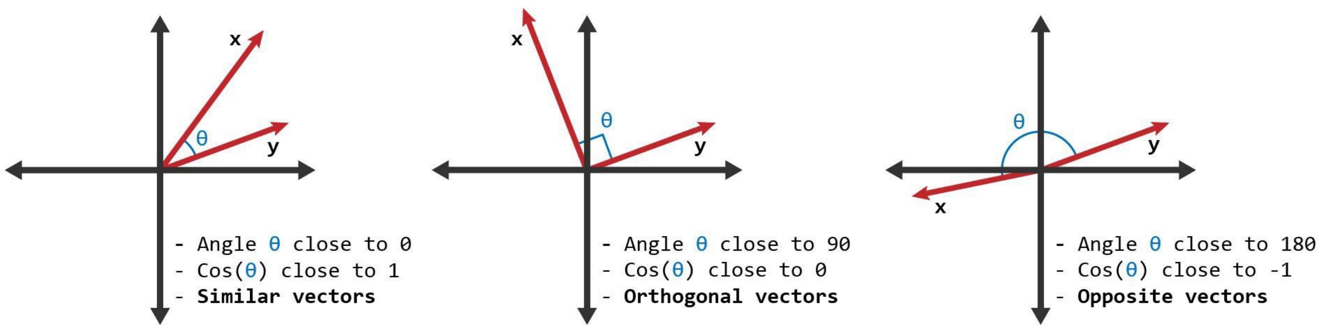
How to measure embedding similarity?

Distance metrices

- L2 Distance
  - $||a-b||_2^2$
    $= \sum_{i=1}^{n}(a_i - b_i)^2$



- Cosine similarity
  - $cos\theta = \frac{a \cdot b}{||a||||b||}$
  - Equal to dot product if vector magnitudes are 1

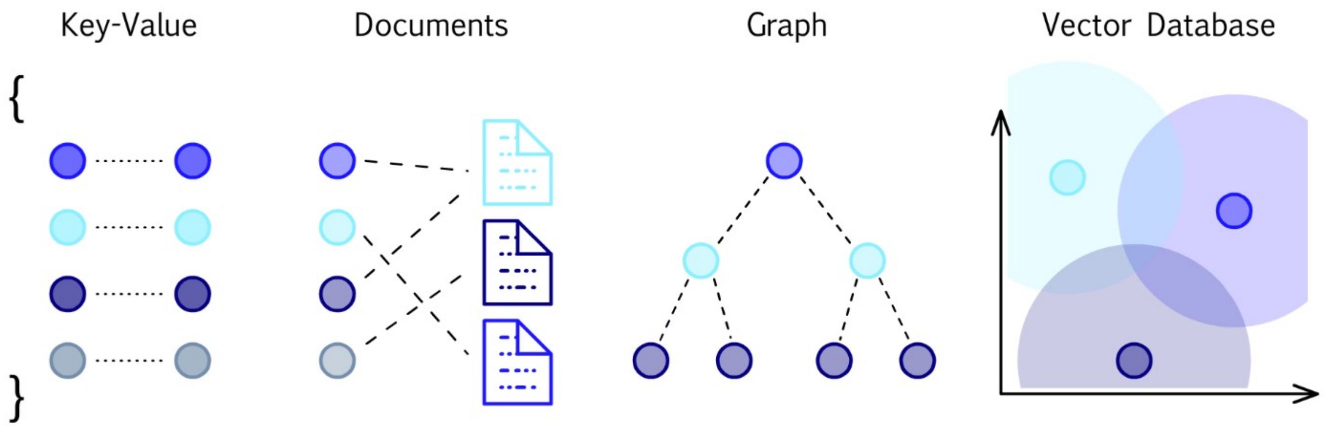


https://www.learndatasci.com/glossary/cosine-similarity/

Vector Retrieval/Storage

External Tools

Vector Databases (e.g. Pinecone)



- Stores embedding values on an online storage

- Able to perform fast calculation of top-k neighbours for a given query
  - Approximate Nearest Neighbours

https://www.pinecone.io/learn/vector-database/

Use Cases

- Talk to document as though you are talking to a person
- Q&A with reference document
- Memory-augmented query

Questions to Ponder

- What are the limitations of cosine similarity to find out embedding similarity?
- What are the benefits and drawbacks of using embeddings for encoding semantics compared to a scalar value?
- What are the benefits and drawbacks of using contrastive learning to learn embeddings? Are there other ways?
- How can we ensure that the embedding space for the training set is generalizable to those outside of the training set