Improving Efficiency and Robustness of Transformer-based Information Retrieval Systems

Tutorial presented by Edmon Begoli, PhD, Sudarshan Srinivasan, PhD and Maria Mahbub, SIGIR 2022

Oak Ridge National Laboratory

Outline

- Motivations, Introduction, and Background
- Transformers Use Cases
- Transformers for Information Retrieval
- Break
- Transformer Architecture
- Optimization and Efficiency Improvement Techniques
- Robustness
- Q&A Session

Preliminaries

Background - Presenter(s)/Co-Authors

Edmon Begoli, PhD

Section Head and Distinguished Scientist, ORNL

Joint Faculty, EECS, The University of Tennessee, Knoxville

Principal Investigator

Sudarshan Srinivasan, PhD

Research Associate in NLP

ML/NLP Scientist on Suicide Risk Assessment and Prevention Project

Maria Mahbub, M.S.

PhD Student, EECS, The University of Tennessee, Knoxville

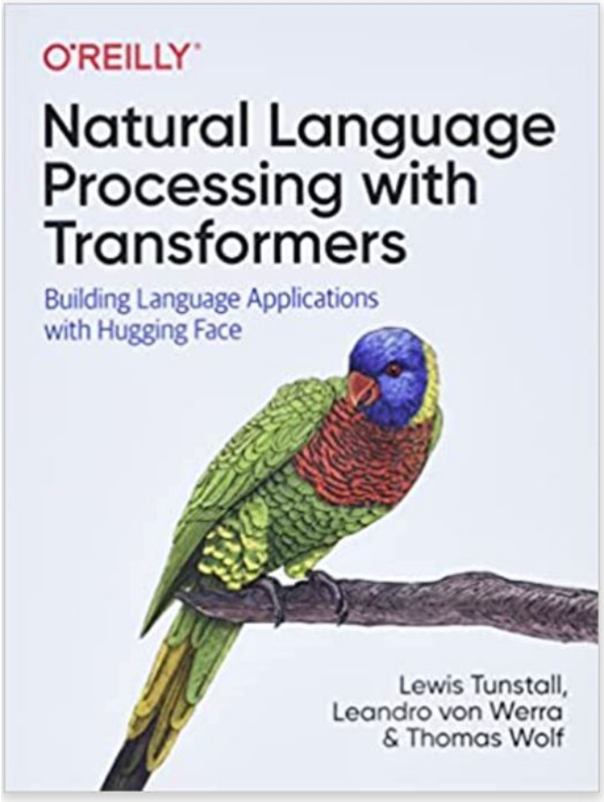
Graduate Researcher, ML/NLP MRC on Suicide Risk Assessment and Prevention Project

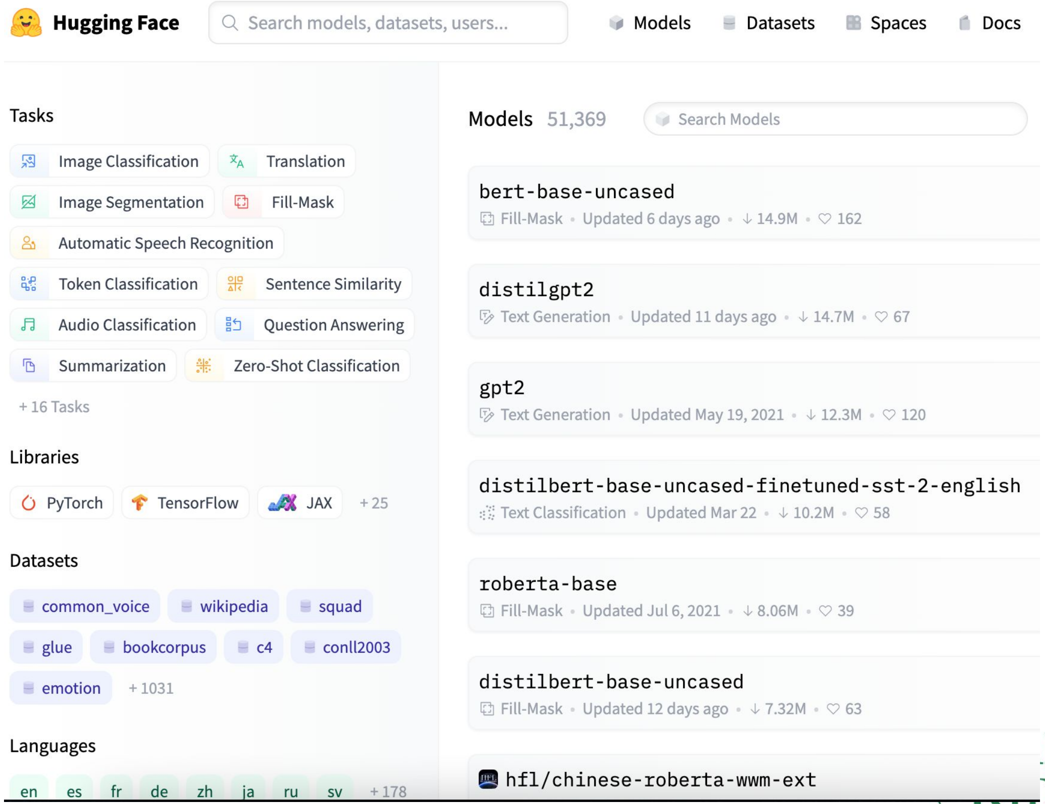
OAK RIDGE National Laboratory

Motivation

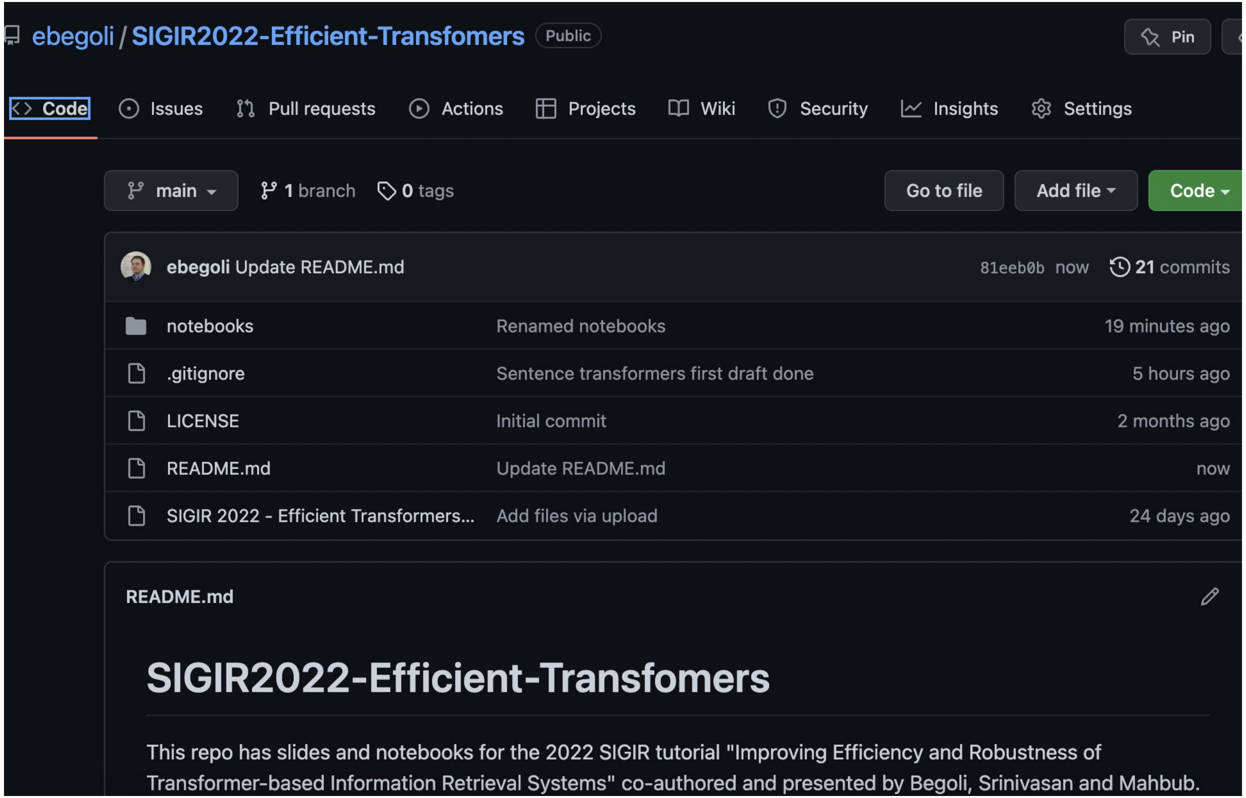
- Superior performance of attention-based models on IR-related tasks and our experience
- Computational cost and complexity of common tasks:
  - Training
  - Fine-tuning
- Inference performance
- Relevant project experience

Credits and Resources





Source Material



Source code (Jupyter NB)

This presentation

Background

Transformers and Attention-based Architecture in General Terms (deep dive again later)

General Use Cases and Applications

Transformers in Information Retrieval Tasks

Transformer Neural Networks and Attention-based Models

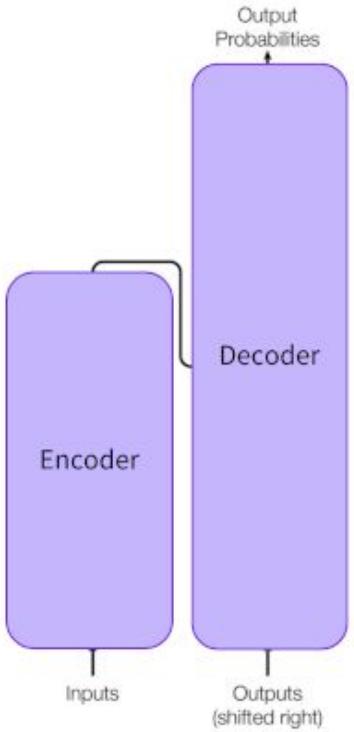
An evolution of NLP models:

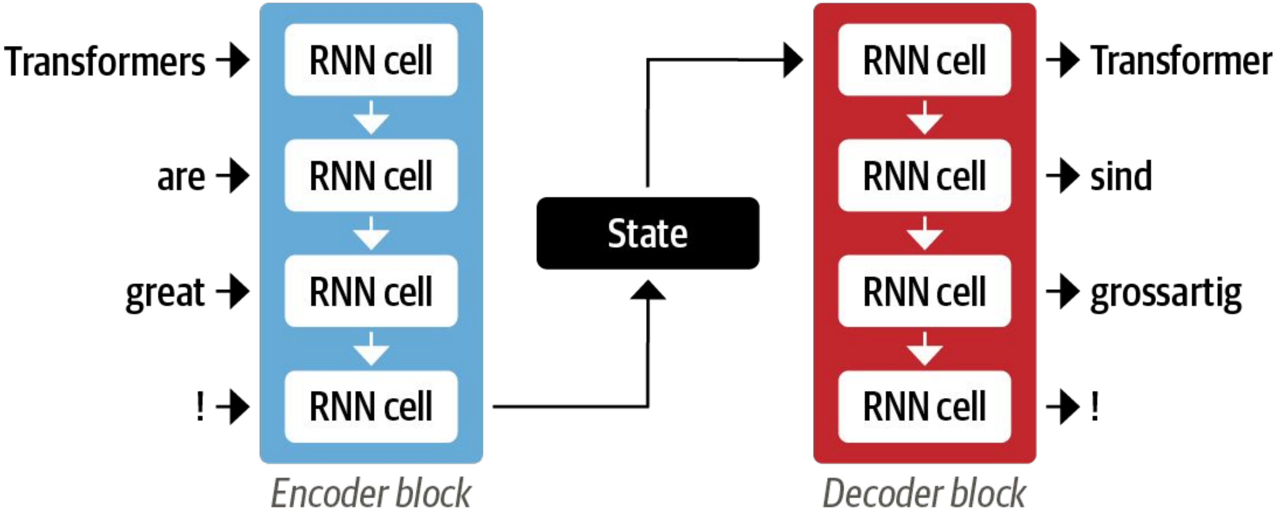
- Statistical NLP models
- Distributional semantics and word2vec
- Sequence-based neural models - RNNs, LSTMs, GRUs
- Attention-based models

Key Transformer Concepts

- The encoder-decoder framework
- Attention mechanism
- Transfer learning

Encoder-Decoder Framework (Transformers)

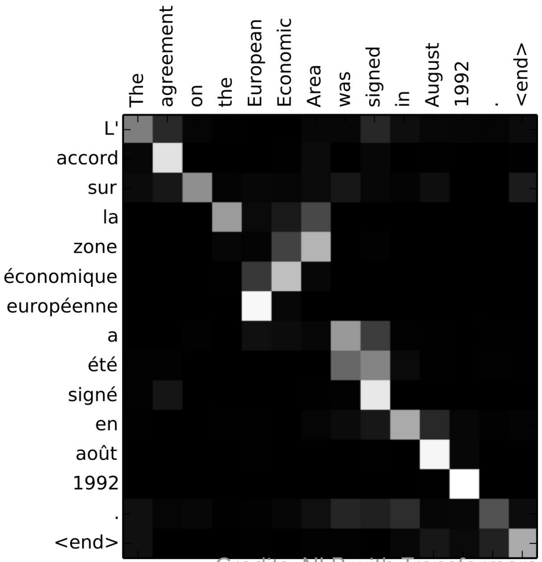


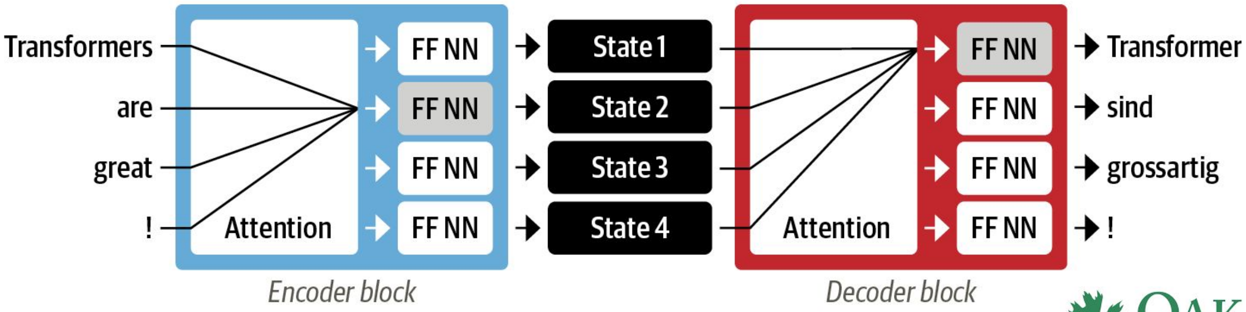


Credits: NLP with Transformers, https://github.com/nlp-with-transformers/notebooks/blob/main/03_transformer-anatomy.ipynb

Attention Mechanism

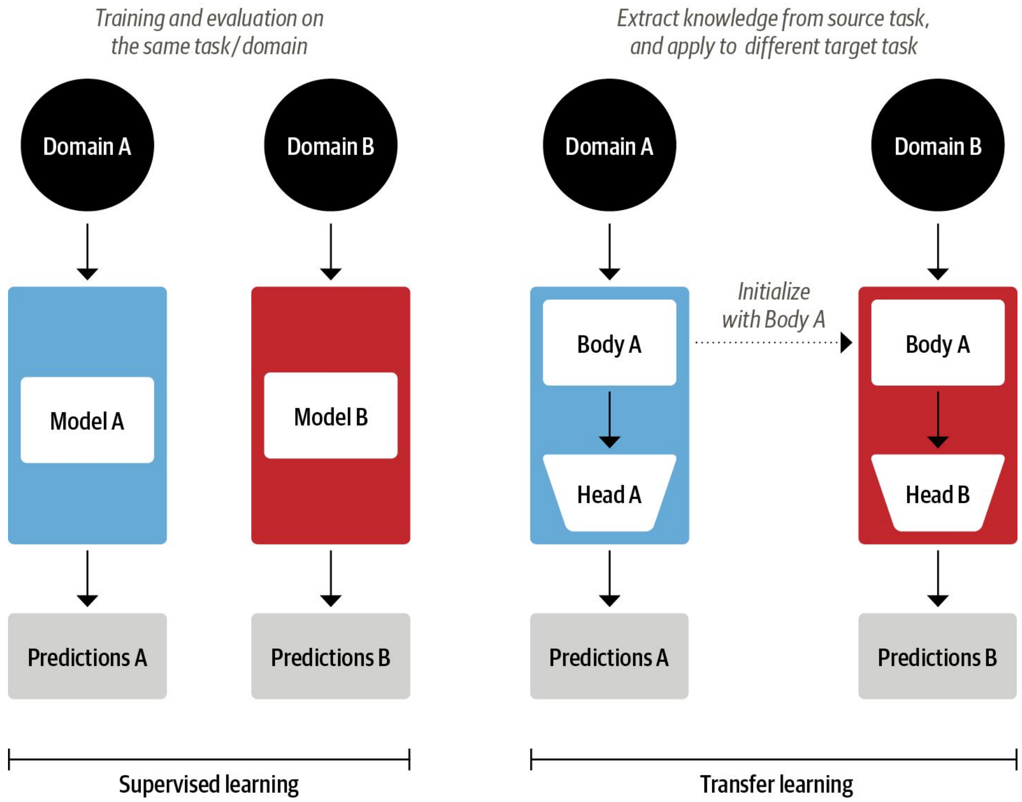
- An evolution of a word-embedding and shared-state concepts
- Effective way to encode complex relationships between the tokens in a sequence.





Credits: NLP with Transformers, https://github.com/nlp-with-transformers/notebooks/blob/main/03_transformer-anatomy.ipynb

Transfer Learning



Case study:

- Generic BERT
- Fine-tuning for clinical informatics scenario (suicide risk determination and prevention)

Credits: NLP with Transformers, https://github.com/nlp-with-transformers/notebooks/blob/main/03_transformer-anatomy.ipynb

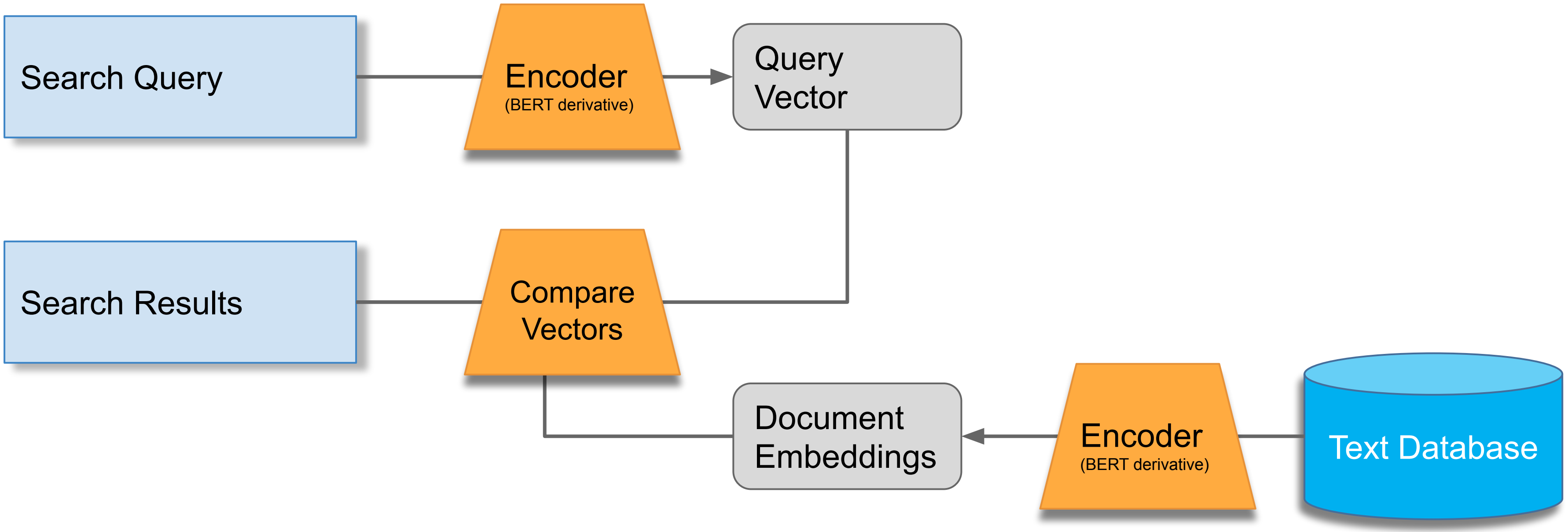
Relevant Information Retrieval Tasks

- Assistance in search
- Cross-lingual retrieval
- Auto-summarization
- Question Answering

Code examples

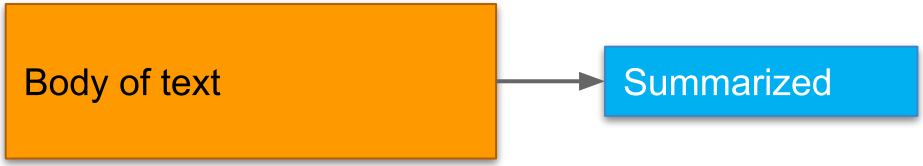
- Notebook 1 - Basic Transformer Examples
  - Auto-summarization - basic transformer
  - Question answering
- Notebook 2 - SBERT-based IR examples:
  - basic sentence encoding
  - semantic textual similarity
  - semantic search
  - paraphrase mining
  - retrieval and re-ranking
  - cross-lingual retrieval
- Notebook 3 - Optimization and Efficiencies

Assistance in Search



Auto-summarization

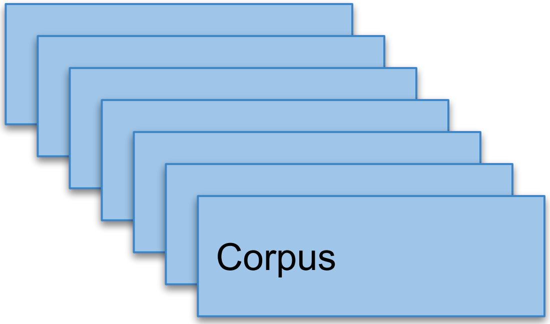
- Convert a body of text into accurately summarized body of text that is significantly smaller in size.
- Extractive and abstractive summarization

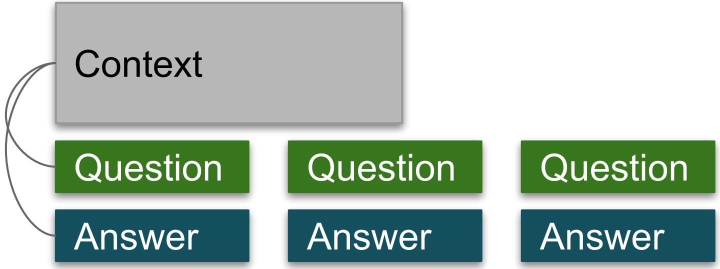


Role of transformers: train associations between the larger expressions and related short terms and expressions.

Question Answering (QA) / Machine Reading Comprehension (MRC)

With an aid of the transformer network, answer questions about the passage of text.





Use of transformer: Trains a transformer to establish a relationship between the passages of text and the questions and answers.

Transformers for Information Retrieval (IR)

An overview of the uses of transformer-based deep neural architectures in information retrieval (IR)

Semantic Textual Similarity

Semantic Search

Paraphrase Mining

Translated Sentence Mining

Cross-encoders

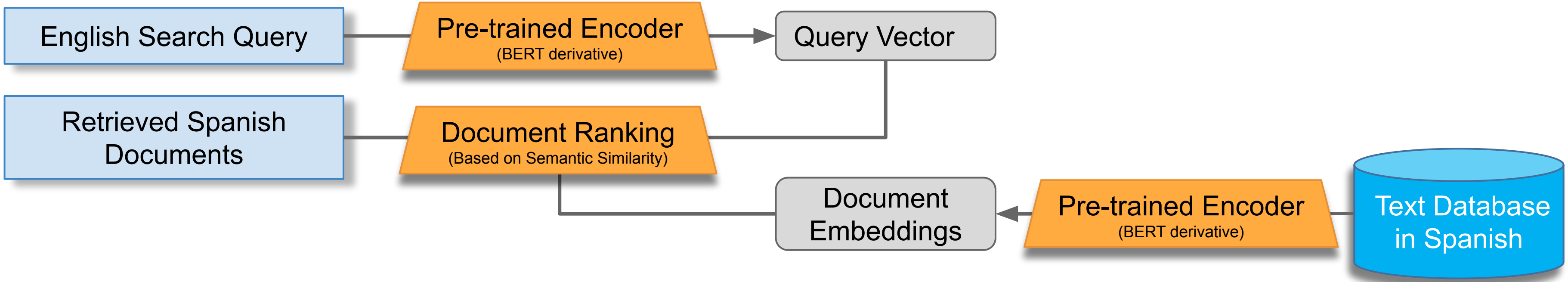
Retrieval and re-ranking

Cross-lingual Retrieval - an Active R&D Area

Rank and retrieve documents by relevance to a query where the document and query can be in a different language

Transformers are considered the state-of-the-art for this task

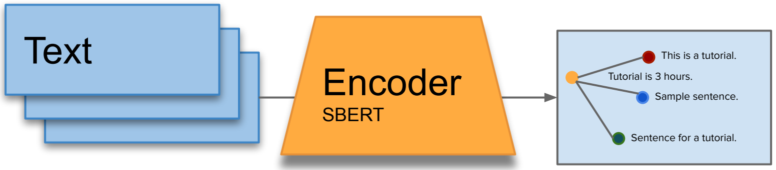
Sample uses: Document retrieval based on multilingual semantic similarity



Yinfei Yang, Daniel Cer, Amin Ahmad, Mandy Guo, Jax Law, Noah Constant, Gustavo Hernandez Abrego, Steve Yuan, Chris Tar, Yun-hsuan Sung, Brian Strope, and Ray Kurzweil. 2020. Multilingual Universal Sentence Encoder for Semantic Retrieval. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 87–94, Online. Association for Computational Linguistics.

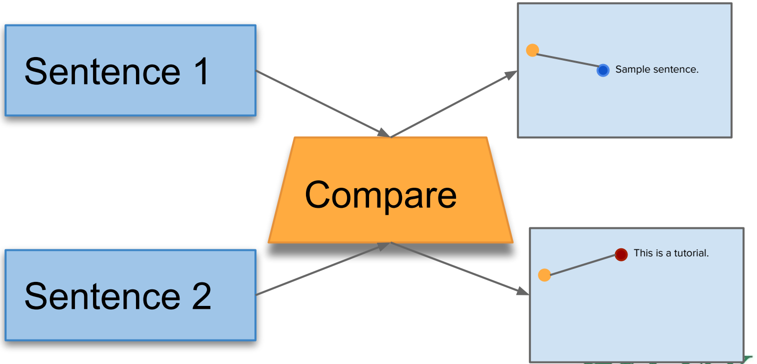
Semantic Textual Similarity

The goal of semantic textual similarity is to measure how similar two pieces of texts are. This is typically done by assigning some score (e.g. 1-5 or 1-10) that ranks the similarity between these two pieces of text.



Sentence Embeddings

Use of transformer (SBERT): Use pre-trained or develop/fine-tune your own model to compute sentence embeddings. Then, compare the embeddings of the sentences from two sets of text (e.g. use cosine similarity).



Semantic Search

- Semantic search seeks to improve search accuracy by understanding the content of the search query, unlike the traditional search engines which finds content based on lexical matching.
- Another advantage of a semantic search is that it can also find synonyms.
- Symmetric and asymmetric semantic search - query vs. content
- See later “Retrieval and Re-ranking”

Use of transformer: Use dense-vector embeddings for the representation of the content and the queries used in search.

Paraphrase Mining

The goal of paraphrase mining is to find paraphrases in a large corpus of sentences.

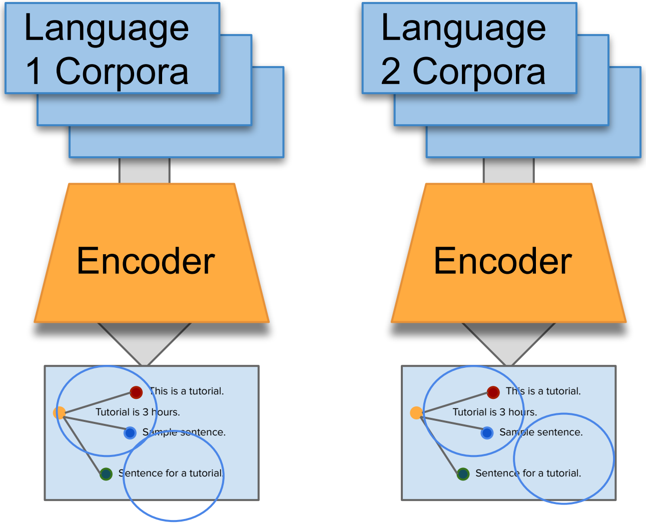
Use of transformer: Create a summarized version of corpus and mine it. Use, for example, SBERT to develop sentence-level embeddings and compare them for similarity.

Translated Sentence Mining (Bitext Mining)

Find parallel (translated) sentence pairs in monolingual corpora.

1. Encode all sentences in their respective corpora.
2. Find the closest pairs using neighbour algorithms (e.g. k-nearest neighbors)
3. Score the closest sentences
4. Rank and return the likely translated pairs.

Use of Transformers: Sentence encoding.



Language 1 sentence 1 - Language 2 sentence 1
Language 1 sentence 2 - Language 2 sentence 2
Language 1 sentence 3 - Language 2 sentence 3
....

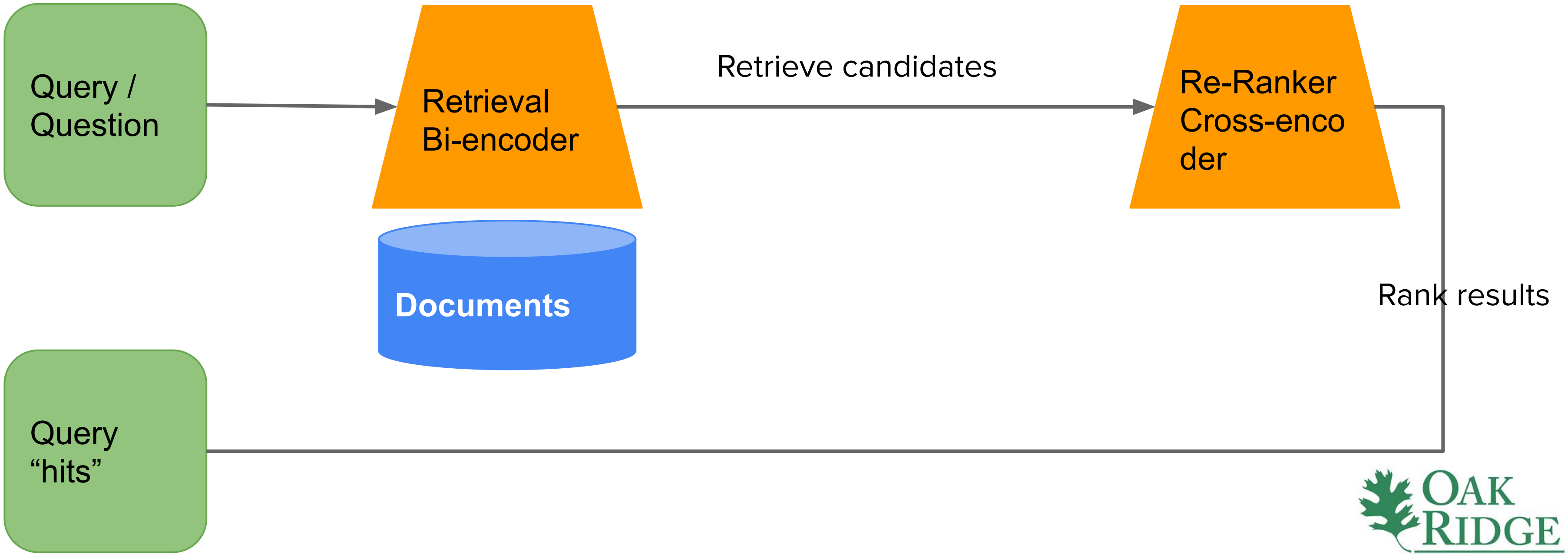
Oak Ridge National Laboratory

Cross-encoders

- In contrast to bi-encoders, useful for sentence **similarity ranking** whereas bi-encoders are useful for two sentence comparisons
- Do not produce embeddings
- Useful as a component in an information retrieval tasks such as retrieval and re-ranking

Retrieval and Re-ranking

- Bi-encoder is used to find/retrieve all closest matches
- Cross-encoder ranks the results of a retrieval



Demo / Code Examples

Break

Transformer Architecture

A Deep Dive

Background

Architecture Overview

Encoder-Decoder Architecture

Embedding Layer

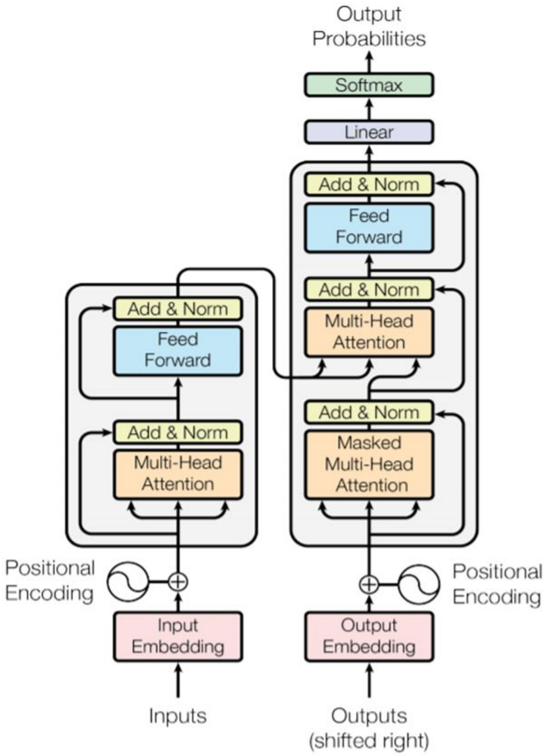
Positional Encoding

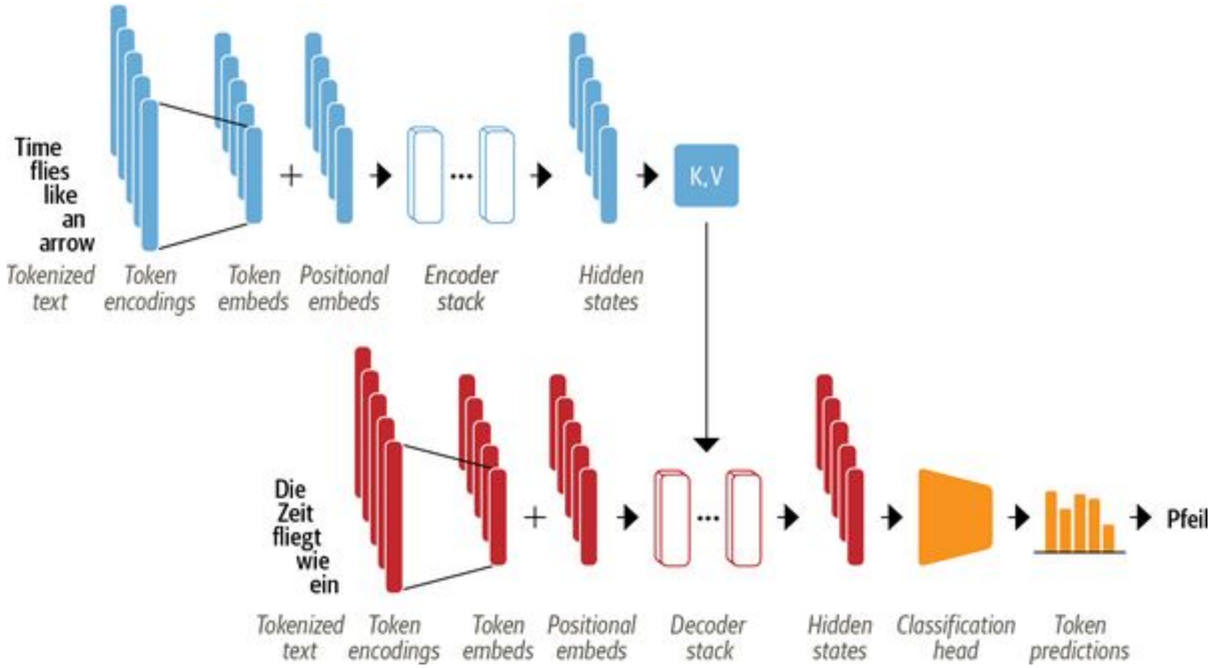
Attention Heads

Motivation for Transformers

- An evolution in distributional semantics-based NLP
- Performance on NLP tasks
- Parallelization
- The benefits of Deep Learning

Transformer Architecture





Ashish Vaswani et. al "Attention is all you need". Proceedings of the 31st International Conference on Neural Information Processing Systems.

Second figure credits, see Ref. 1

Transformer Variants

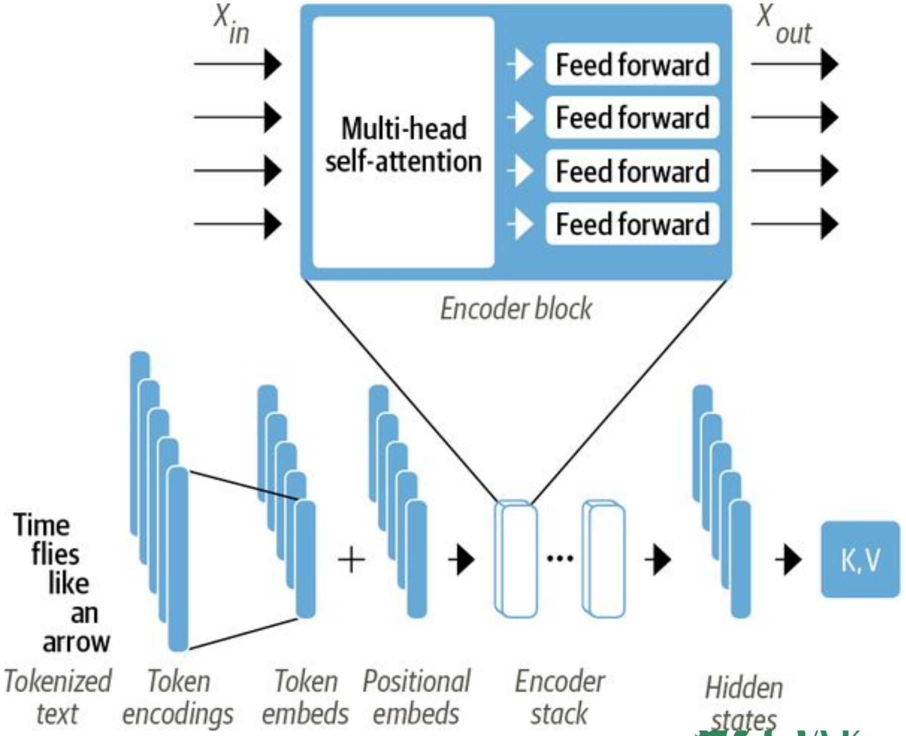
Encoder branch (BERT family, incl. ELEKTRA) - great for classification and other NLP tasks

Decoder branch (GPT family) - great for predicting tokens in the sequence (generation)

Encoder-Decoder branch (T5/BART/M2M100) - great for language and text-to-text tasks

The Encoder

- Consists of many encoder layers stacked next to each other
- Sublayers:
  - A multi-head self-attention layer
  - A fully connected feed-forward layer that is applied to each input embedding

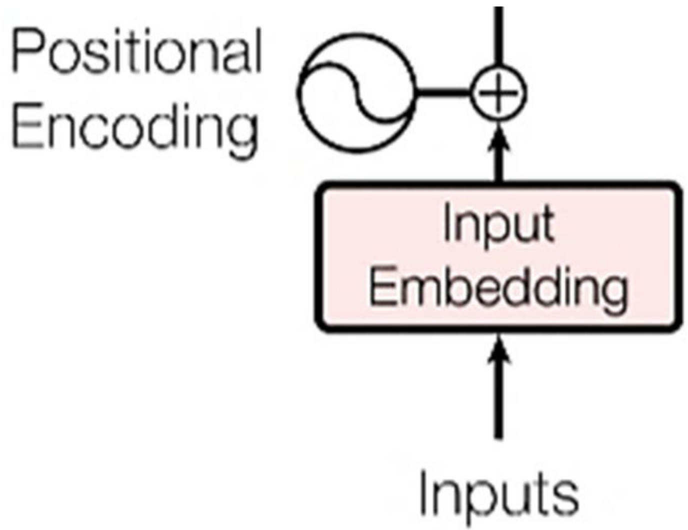


Credits: NLP with Transformers, https://github.com/nlp-with-transformers/notebooks/blob/main/03_transformer-anatomy.ipynb

Positional Encoding

- Embeddings do not carry any information about the relative order of the tokens in the sentence
- Requires a way to encode information about a token’s position in a sentence
- “I **google** for information. Thanks **Google**.”
- Variables:
    - *pos:* Position of the token *within* the input sequence
    - •*i*: Position of the *embedding dimension* within the vector representation of the token

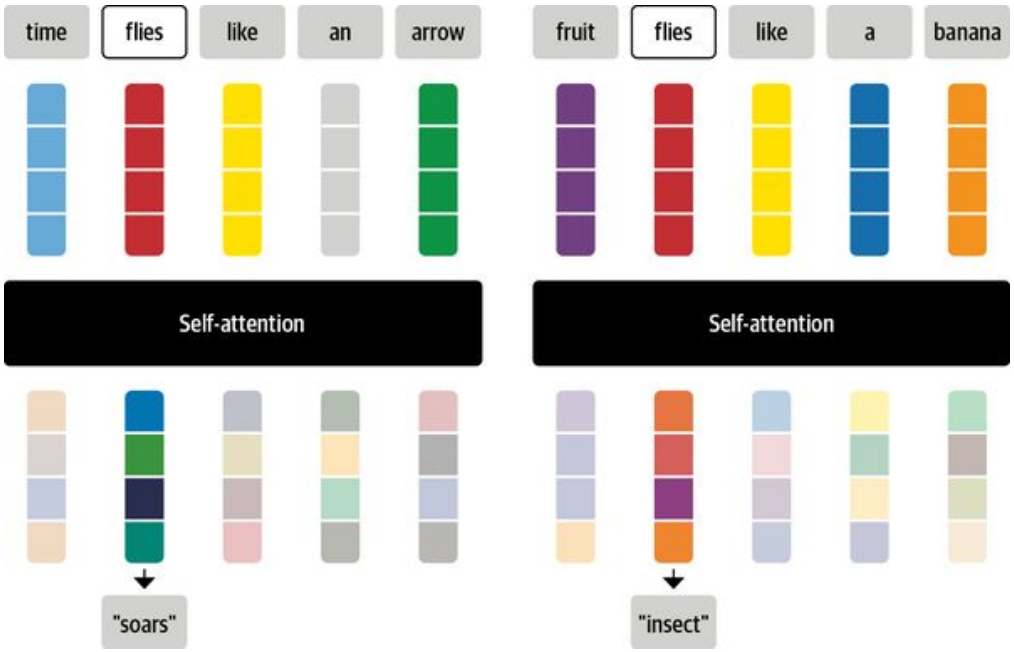
$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{\text{model}}})$$
$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{\text{model}}})$$



Oak Ridge National Laboratory

Attention and Self-Attention

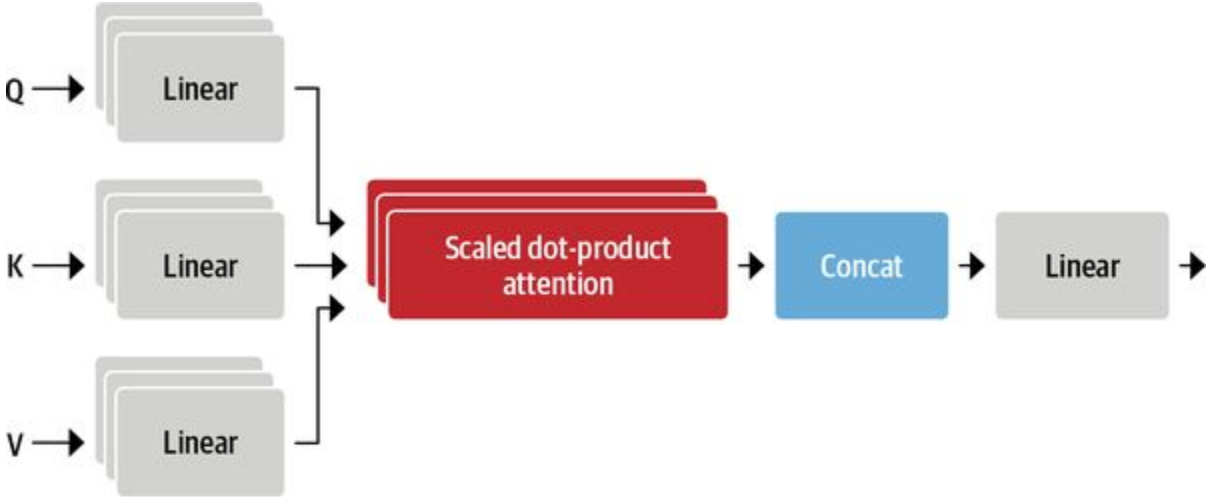
- Contextualized attention - it contextualizes the embeddings by encoding the context of a token in a sequence with respect to other tokens
- Flies -> arrow, time (“soars”)
- Flies -> fruit, banana (“insect”)



Credits: NLP with Transformers, https://github.com/nlp-with-transformers/notebooks/blob/main/03_transformer-anatomy.ipynb

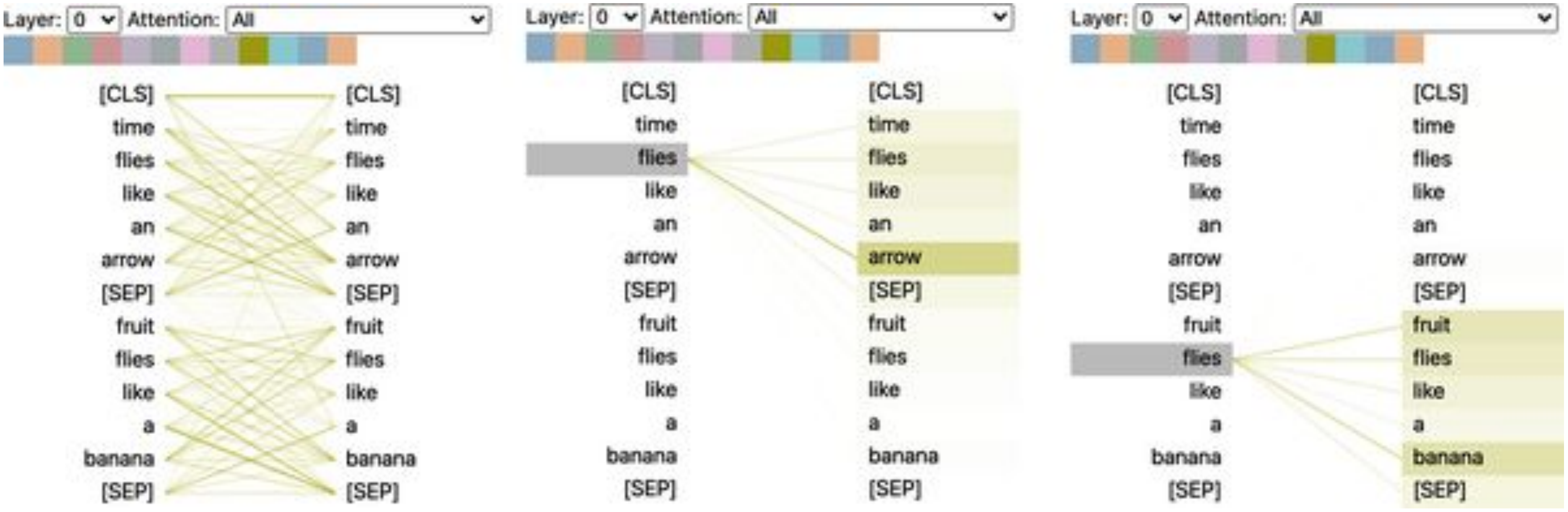
Multi-head attention

Allows the self-attention layer to focus on different semantic aspects of the sequence. Encodes complex relationships in a sequence.



Credits: NLP with Transformers, https://github.com/nlp-with-transformers/notebooks/blob/main/03_transformer-anatomy.ipynb

Attention Visualized



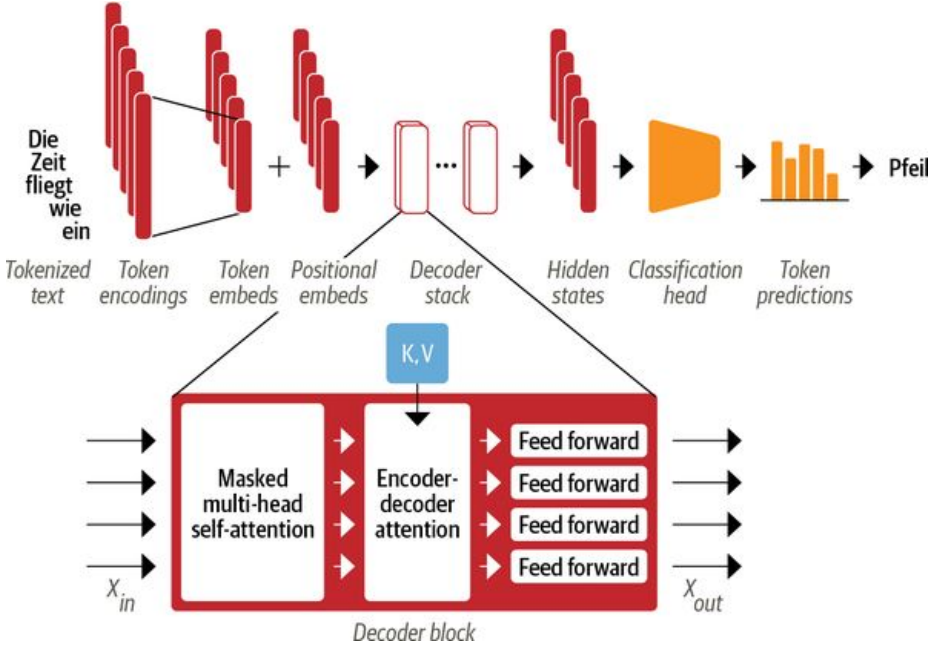
Note the attention given to the tokens (arrow) in the same sentence but also to the tokens in the other sentence (fruit,banana).

Credits: NLP with Transformers, https://github.com/nlp-with-transformers/notebooks/blob/main/03_transformer-anatomy.ipynb

Decoder Stack

Two attention sub-layers:

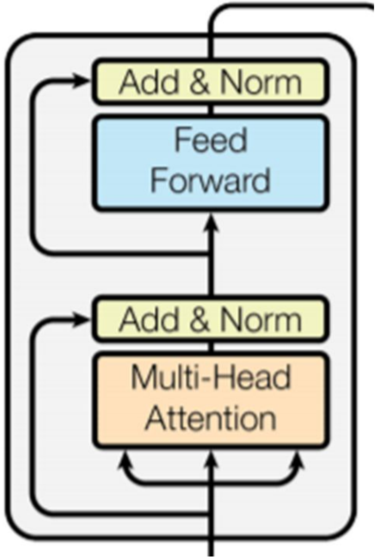
- Masked multi-head self-attention layer
- Encoder-decoder attention layer
- Classification and token prediction



Credits: NLP with Transformers, https://github.com/nlp-with-transformers/notebooks/blob/main/03_transformer-anatomy.ipynb

Bi-directional Encoder Representations from Transformers (BERT)

- BERT only has an encoder stack, giving the language model state encoding as the output
- Serves as an effective solution for 11+ NLP and IR-relevant tasks
- Effective language modelling/representation and classification related tasks
- Masked Language Model (MLM) and Next Sentence Prediction (NSP)



Performance and Efficiency

Optimizing Transformers' Performance and Efficiency

Knowledge Distillation

(Weight) Quantization

(Weight) Pruning

Performance Concerns

- Model performance
- Latency
- Memory

Efficiency Optimization Techniques

- Knowledge Distillation
- Quantization
- Weight pruning

Knowledge Distillation

The main idea:

Take label classification probabilities from a bigger, “teacher” model and use them with a smaller, “student” model to learn from.

The idea behind ***DistilBERT***

Knowledge Distillation Process

- Transfer “soft labels” from the teacher to student
- Train the student to mimic the probabilities that teacher assigns to the class members; there is likely some kind of relationship that student could learn from

$$D_{KL}(p,q) = \sum_i p_i(x) \log \frac{p_i(x)}{q_i(x)}$$

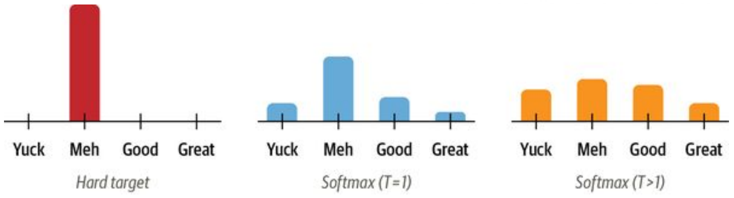
Kullback-Leibler (KL) Divergence

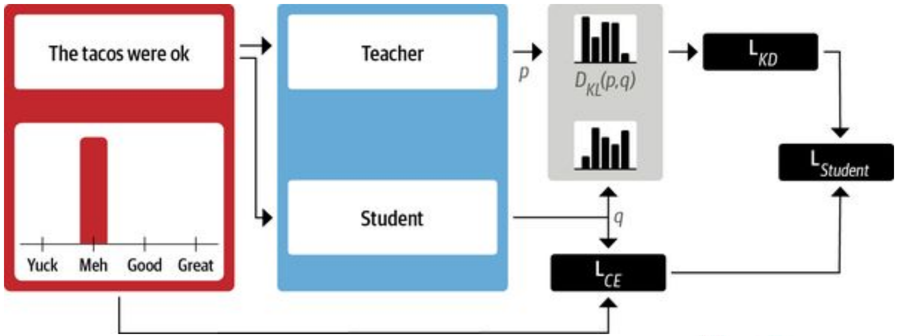
KD loss $L_{KD} = T^2 D_{KL}$

$$L_{\text{student}} = \alpha L_{CE} + (1-\alpha) L_{KD}$$

softmax $\frac{\exp(z_i(x))}{\sum_j \exp(z_i(x))}$

Softmax scaled with temperature T $p_i(x) = \frac{\exp(z_i(x)/T)}{\sum_j \exp(z_i(x)/T)}$





KD in Pre-training: $L_{\text{DistilBERT}} = \alpha L_{mlm} + \beta L_{KD} + \gamma L_{cos}$

Credits: NLP with Transformers, https://github.com/nlp-with-transformers/notebooks/blob/main/03_transformer-anatomy.ipynb

(Weight) Quantization

The main idea:

Make the model smaller by reducing the precision/width of the variables used to hold the weights (e.g. from 32-bit floating point (FP32) to 8-bit integers (INT8))

Quantization Process - generic

1. Observe the distribution of activation and weight values
2. Find the discretized representation
3. Re-map the activation and weight values to the new representation

Types of Quantization

- **Dynamic** - adaptations performed only during inference. Weights and activations are converted to INT8 (quantized) ahead of inference time, on the fly.

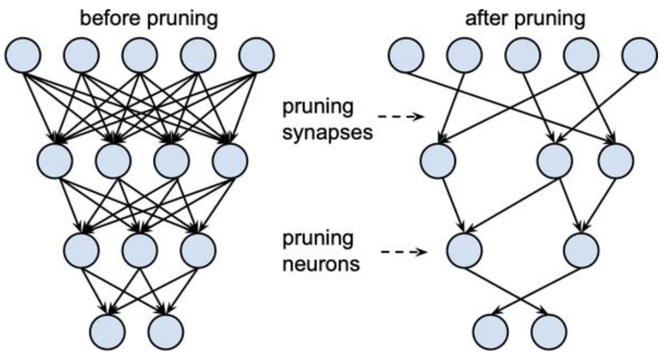
- **Static** - precomputes a quantization on a sample data, before the inference phase. It skips the FP32 to INT8 conversion, but a) it is dependent on a good sample data, and b) there can be a discrepancy between the training and inference data.

- **Quantization-aware training** - “fake” the quantization of FP32 values. FP32 are rounded to mimic the effects of static quantization.

(Weight) Pruning

The main idea:

Shrink the number of parameters (weights) in the network that do not have a significant effect on the functioning of the network.



Credits: NLP with Transformers, https://github.com/nlp-with-transformers/notebooks/blob/main/03_transformer-anatomy.ipynb

(Weight) Pruning

Magnitude pruning - prunes the weights, iteratively, according to their magnitude. It can be computationally intensive because it needs to train the model to convergence.

Movement pruning - gradually remove weights during fine-tuning such that the model becomes progressively sparser. Intuition: the weights that “move” the most from zero during fine tuning are the ones that have the most importance.

Demo / Code Examples

Oak Ridge National Laboratory

Improvements to Robustness

Adversarial Considerations

Training for Robustness

Adversarial AI and Robustness Considerations

- Data poisoning
  - Insert adversarial artifacts into the training data
  - Corrupt the classifier and force the misclassification at the inference time

- Adversarial examples

Is BERT Really Robust? A Strong Baseline for Natural Language Attack on Text Classification and Entailment

Di Jin,[1*] Zhijing Jin,[2*] Joey Tianyi Zhou,[3] Peter Szolovits[1]
[1]Computer Science & Artificial Intelligence Laboratory, MIT
[2]University of Hong Kong
[3]A*STAR, Singapore
jindi15@mit.edu, zhijing.jin@connect.hku.hk, zhouty@ihpc.a-star.edu.sg, psz@mit.edu

Nomen est Omen - The Role of Signatures in Ascribing Email Author Identity with Transformer Neural Networks

Publisher: IEEE Cite This PDF

Sudarshan Srinivasan ; Edmon Begoli ; Maria Mahbub ; Kathryn Knight All Authors

Techniques for Improving Robustness

- Control over Data Supply Chain
- Training with Adversarial Examples
- Monitoring and Anomaly Detection

Discussion and Q&A

30 Min

References

1. Tunstall, von Werra, Wolf, *Natural Language Processing with Transformers* - revised edition, O'Reilly, March 2021
2. Thakur, Nandan, Nils Reimers, Johannes Daxenberger, and Iryna Gurevych. "Augmented **SBERT**: Data augmentation method for improving bi-encoders for pairwise sentence scoring tasks." *arXiv preprint arXiv:2010.08240* (2020).

Oak Ridge National Laboratory

Acknowledgments

Leandro von Werra, for his permission to use a material from the book “Natural Language Processing with Transformers, revised edition” and the accompanying github.

This presentation was co-authored by UT-Battelle, LLC under Contract No. DE-AC05-00OR22725 with the U.S. Department of Energy, and under a joint programs (MVP CHAMPION and VICTOR), between the U.S. Department of Energy (DOE), and the U.S. Department of Veterans Affairs (VA).

OAK RIDGE National Laboratory