Text Analysis II

Introduction to Computational Social Science

DEPARTMENT OF NETWORK AND DATA SCIENCE (DNDS)

BOJAN EVKOSKI

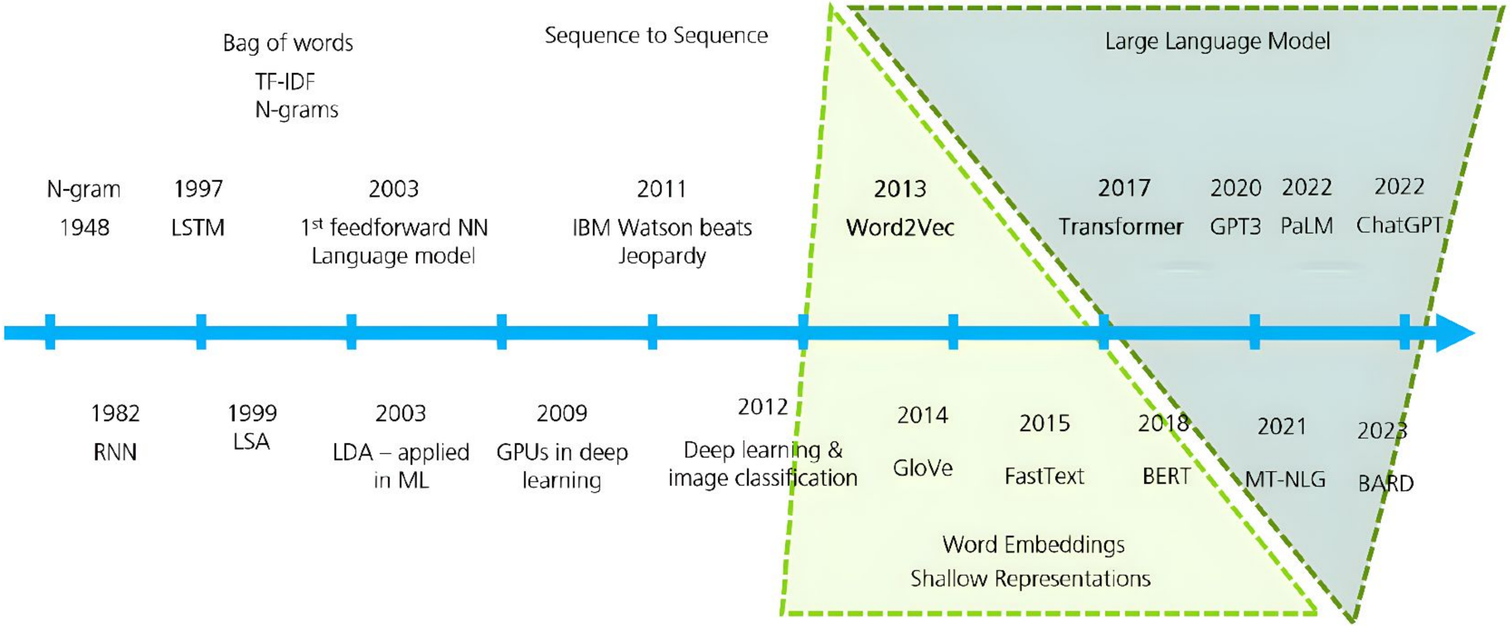
OCT 2025

Outline

- From word embeddings to Transformers
- Encoding and Decoding: Understanding Modern Models
- Fine-tuning and Adaptation
- Generative AI and Prompt Engineering

- Tutorial: Sentiment and Topic Modeling in Parliamentary Speech
- Discussion: The Promise and Pitfalls of NLP for Social Sciences

From word embeddings to Transformers



Text summarization, Classification – Spam, Sentiment Analysis

Cons: Semantic relations between words

Analogies, next word prediction, recommendation engines

Cons: Context in long sentences, transfer learning

Q&A, Chatbots, Language Translation, Image captioning, Natural Language inference

Cons: Bias, illogical outcomes

RNN: Recurrent neural network

GPU: Graphic processer unit

LSA: Latent semantic analysis

LSTM: Long short term memory

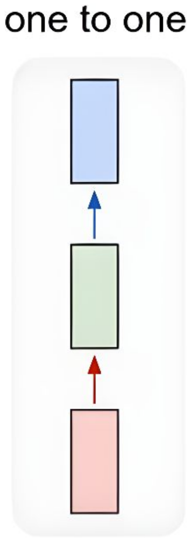
LDA: Latent drichlect Allocation

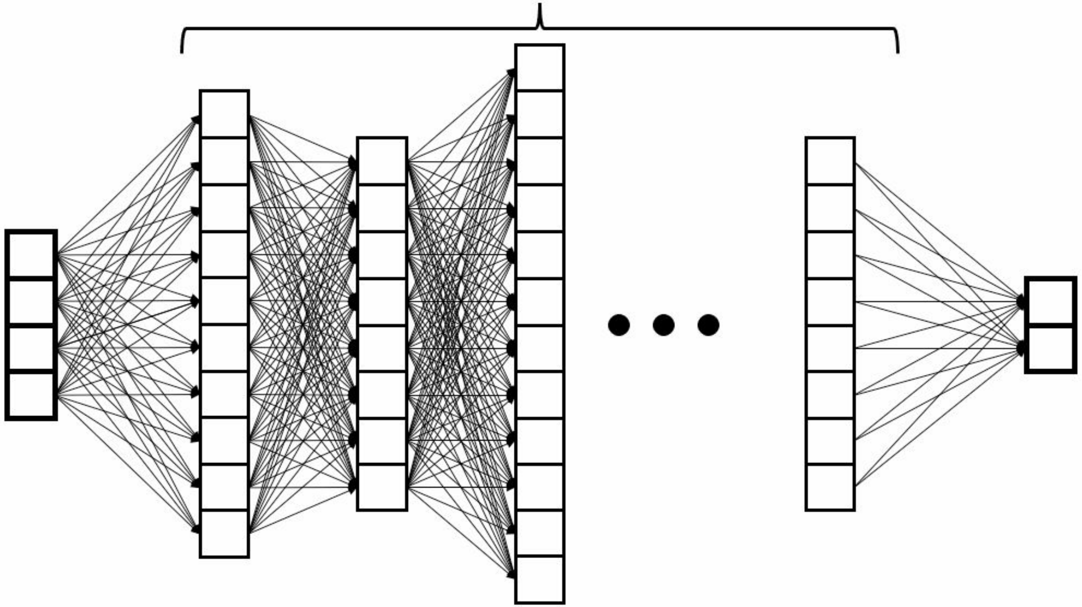
TF: term frequency, IDF: Inverse Document frequency

For Watson, see: https://www.ibm.com/history/watson-jeopardy

GPT: Generative pre-trained transformer

Recurrent Neural Networks and LSTMs





Recurrent Neural Networks and LSTMs

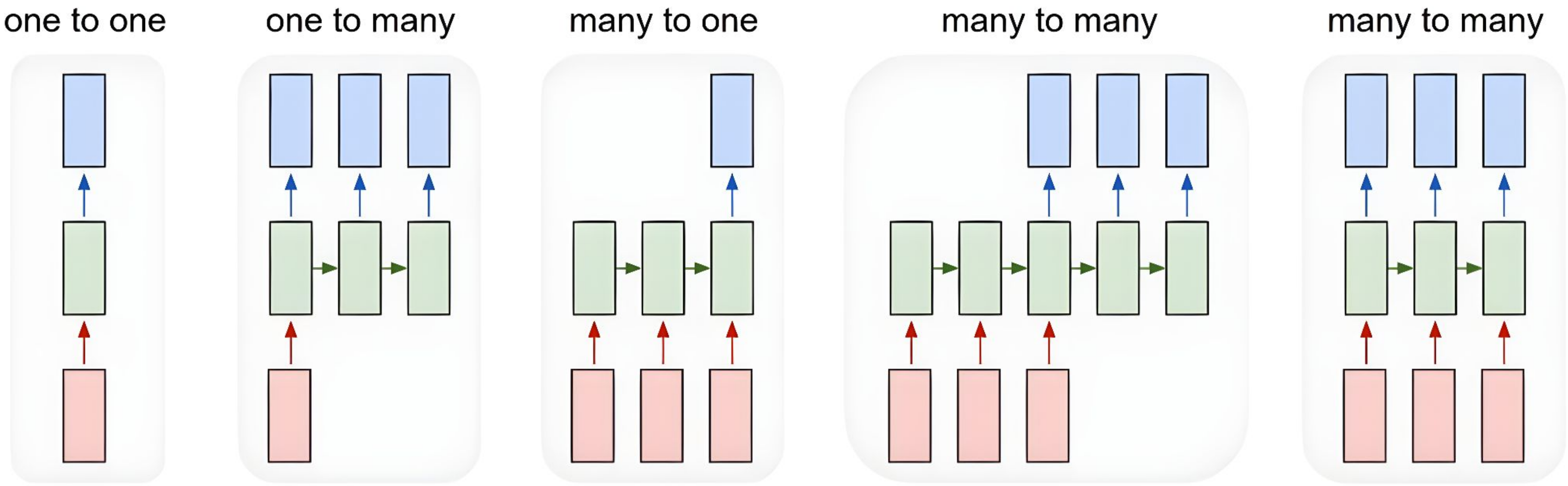


Image Captioning

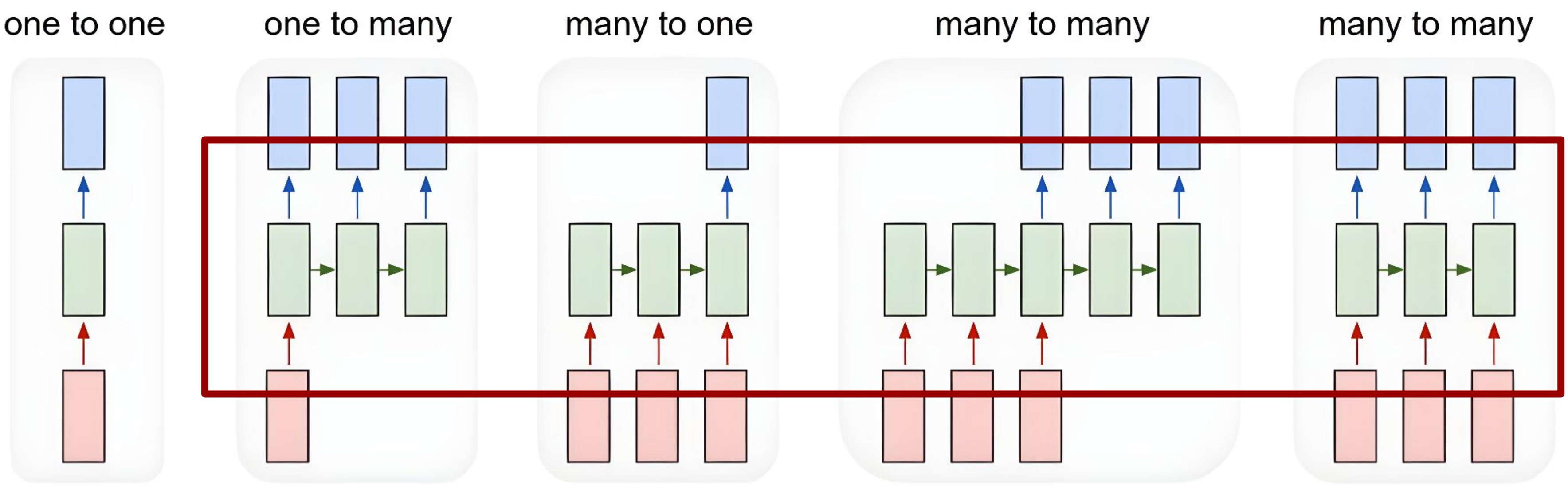
Sentiment

Machine Translation

Named Entity Recognition

Source: https://goodboychan.github.io/python/deep_learning/tensorflow-keras/2020/12/06/01-RNN-Many-to-one.html

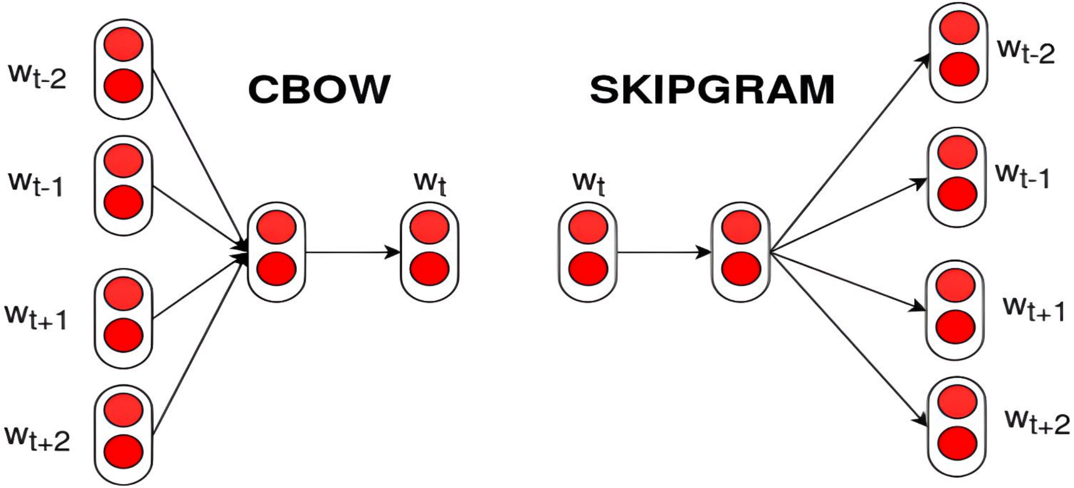
Recurrent Neural Networks and LSTMs



Issues: slow to train; memory loss

Source: https://goodboychan.github.io/python/deep_learning/tensorflow-keras/2020/12/06/01-RNN-Many-to-one.html

Word Embeddings

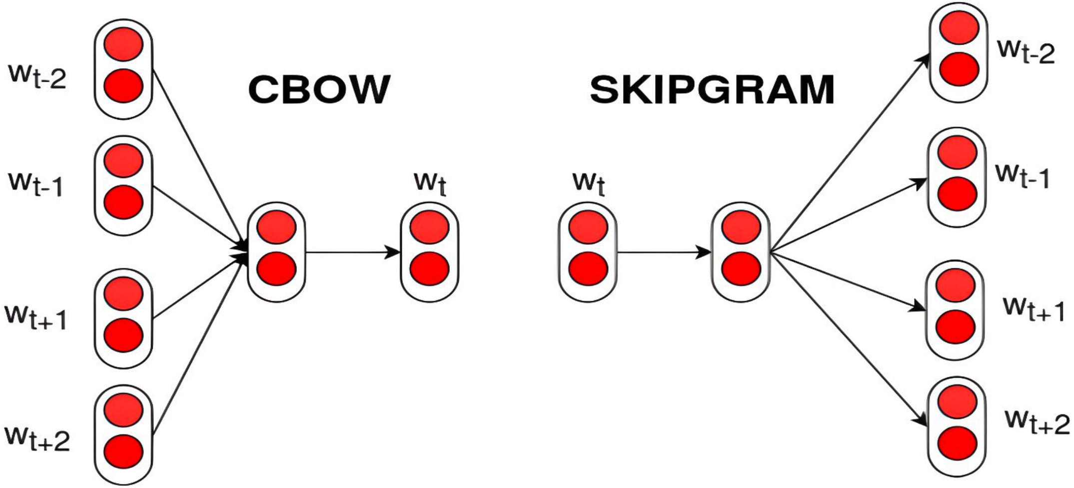


$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0} logp(w_t|w_{t+j})$$

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0} logp(w_{t+j}|w_t)$$

Source: https://commons.wikimedia.org/wiki/File:CBOW_eta_Skipgram.png

Word Embeddings



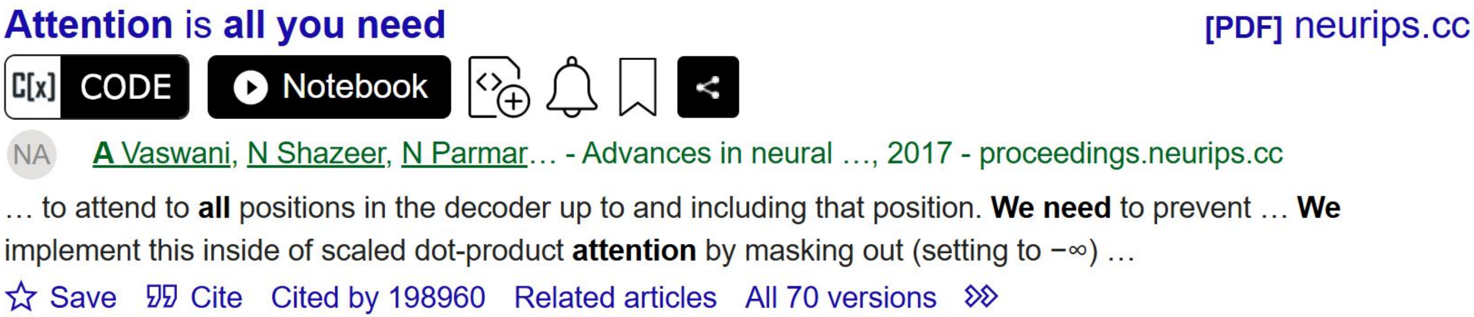
$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0} logp(w_t|w_{t+j})$$

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0} logp(w_{t+j}|w_t)$$

Issues: <u>aggregates contexts</u>; single embedding for a unique word (e.g., Slack)

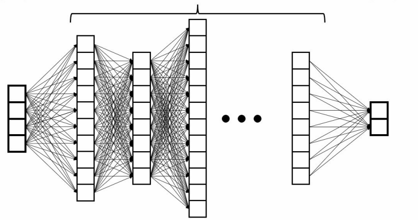
Source: https://commons.wikimedia.org/wiki/File:CBOW_eta_Skipgram.png

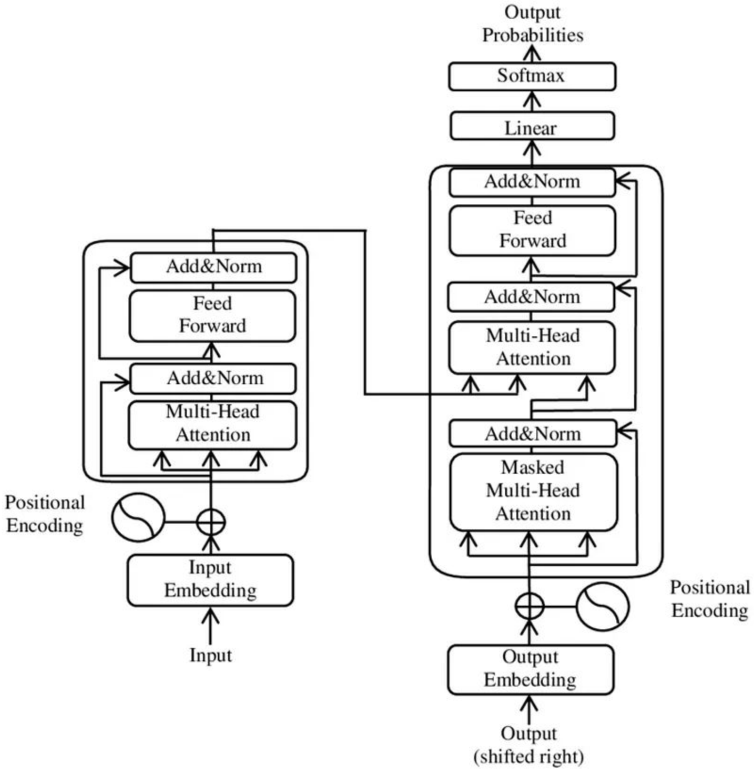
Transformers (Attention Mechanism)



Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. "Attention is all you need." Advances in neural information processing systems 30 (2017).

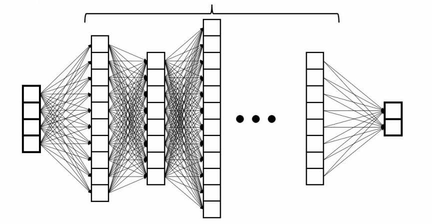
Transformers (Attention Mechanism)



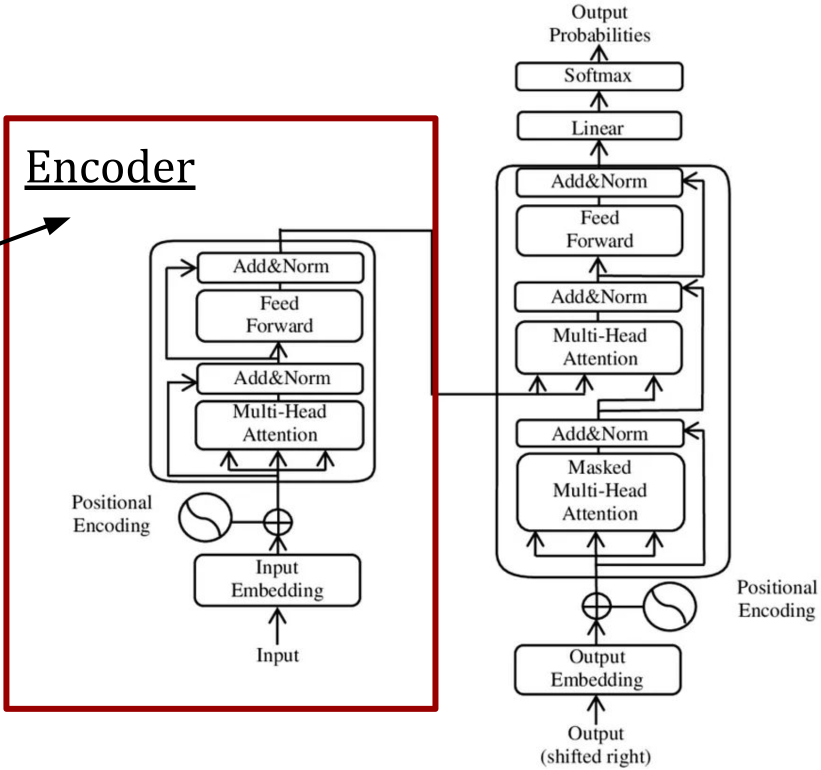


Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. "Attention is all you need." Advances in neural information processing systems 30 (2017).

Transformers (Attention Mechanism)

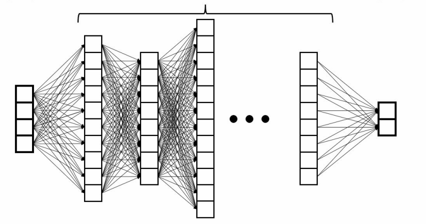


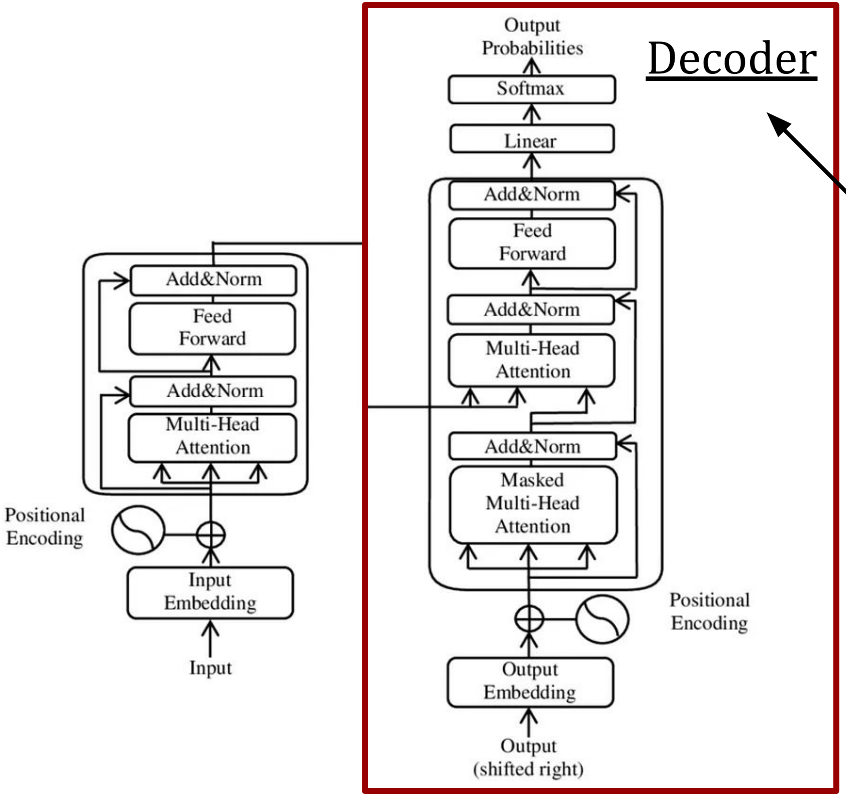
An encoder transforms input data into a compressed representation while maintaining its meaning.



Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. "Attention is all you need." Advances in neural information processing systems 30 (2017).

Transformers (Attention Mechanism)



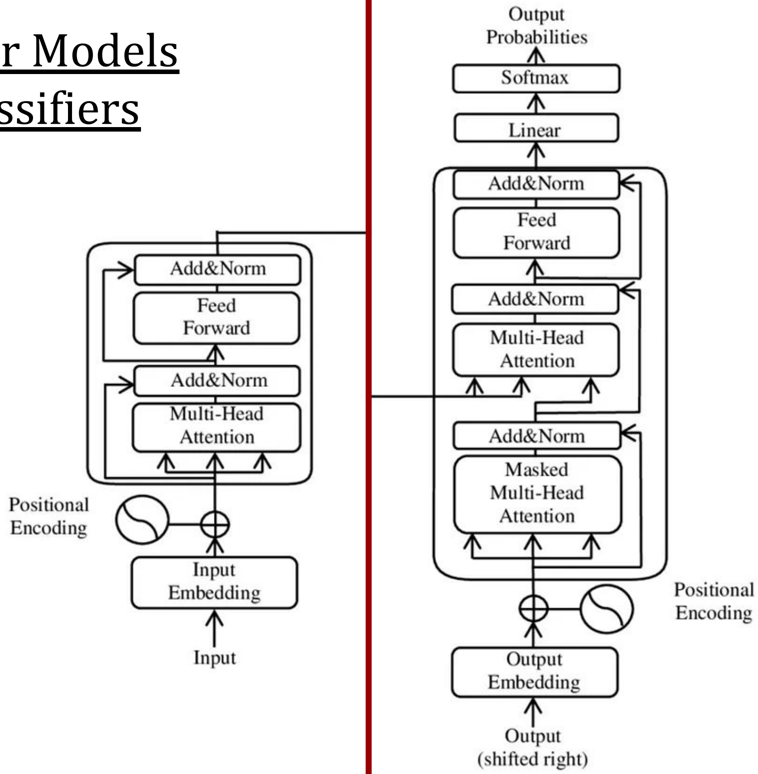


A decoder uses the representations of an encoder to generate new or reconstruct the desired output.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. "Attention is all you need." Advances in neural information processing systems 30 (2017).

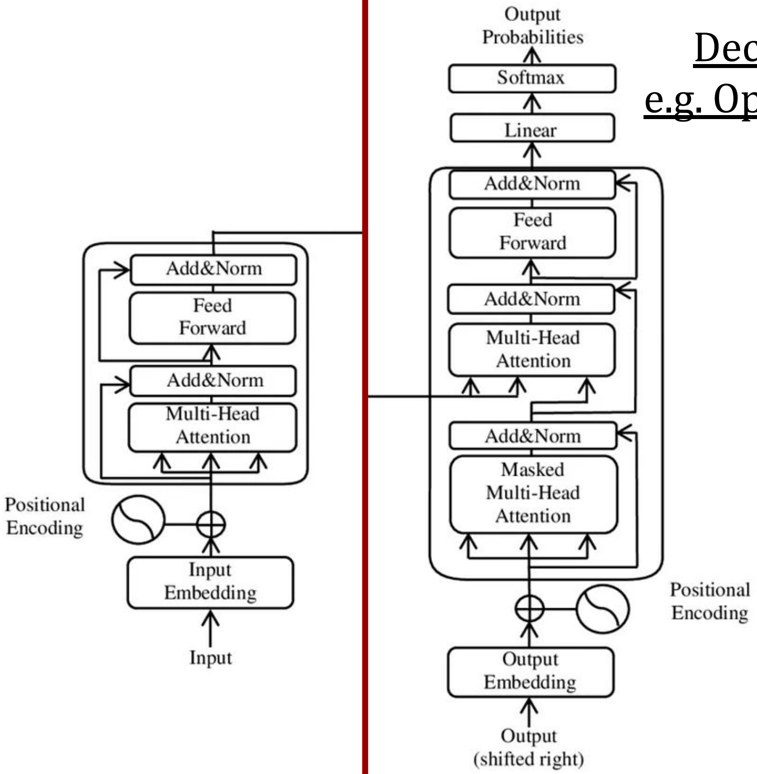
Transformers (Attention Mechanism)

Encoder Models
e.g., classifiers



Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. "Attention is all you need." Advances in neural information processing systems 30 (2017).

Transformers (Attention Mechanism)

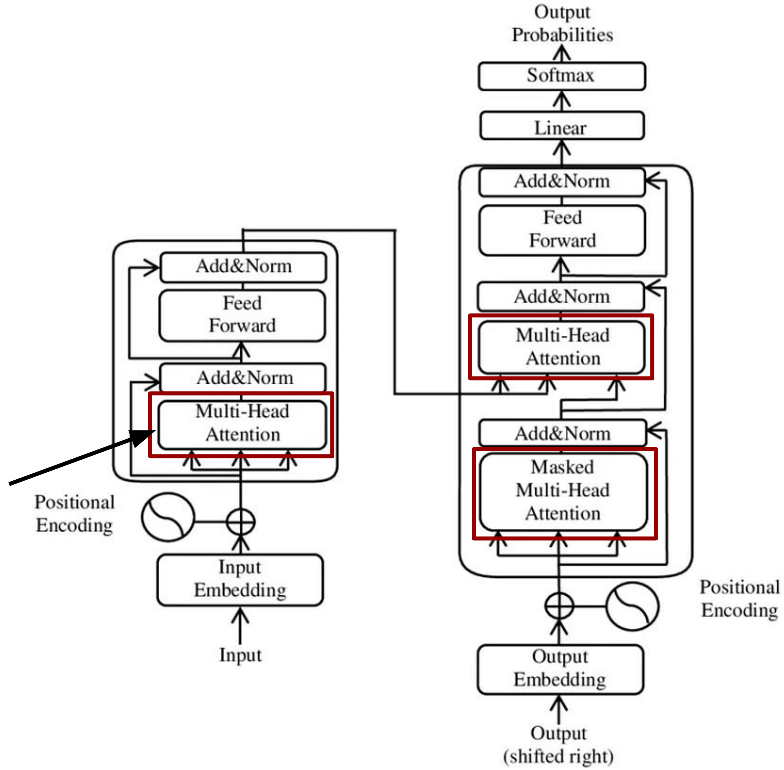


<u>Decoder Models</u>
<u>e.g. Open AI's GPTs (ChatGPT)</u>

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. "Attention is all you need." Advances in neural information processing systems 30 (2017).

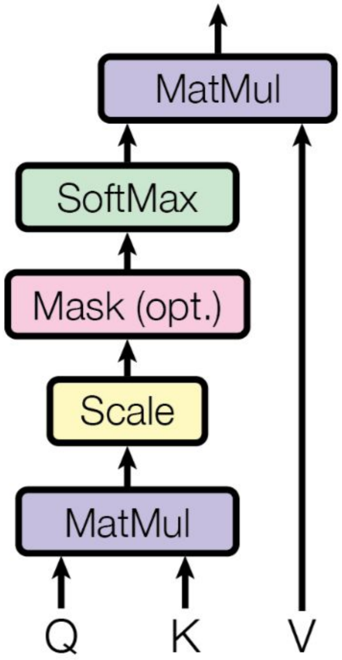
Transformers (Attention Mechanism)

<u>Attention Layer</u>



Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. "Attention is all you need." Advances in neural information processing systems 30 (2017).

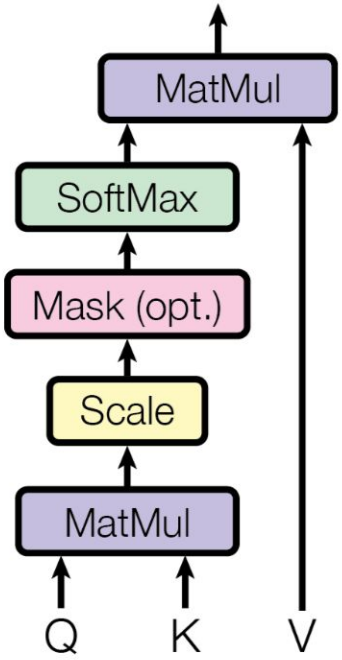
Attention Layer



$$\mathrm{Attention}(Q, K, V) = \mathrm{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

Vaswani, A. "Attention is all you need." Advances in Neural Information Processing Systems (2017).

Attention Layer

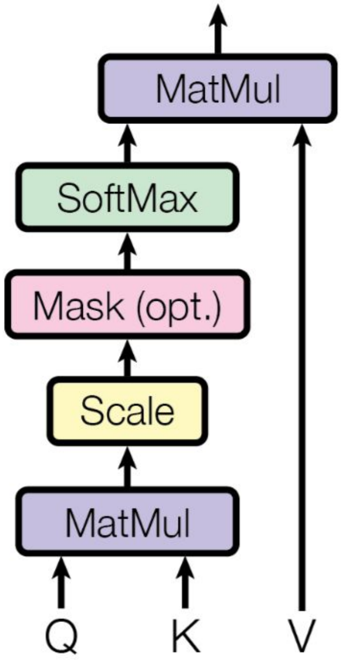


$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

QUERY
Word/Token: "These are the queues I should pay attention to."

Vaswani, A. "Attention is all you need." Advances in Neural Information Processing Systems (2017).

Attention Layer



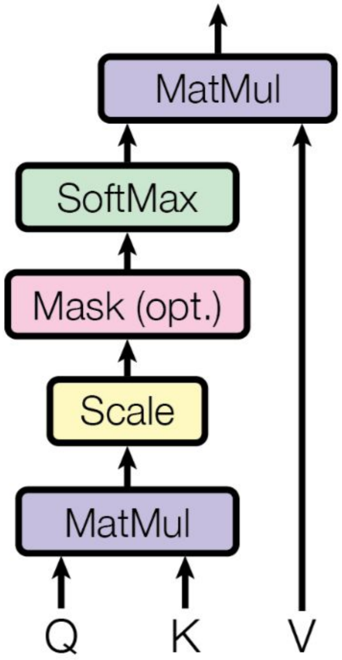
KEY
Word/Token: “These are the reasons why they should pay attention to me.”

$$\mathrm{Attention}(Q, K, V) = \mathrm{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

QUERY
Word/Token: “These are the queues I should pay attention to.”

Vaswani, A. "Attention is all you need." Advances in Neural Information Processing Systems (2017).

Attention Layer



KEY
Word/Token: “These are the reasons why they should pay attention to me.”

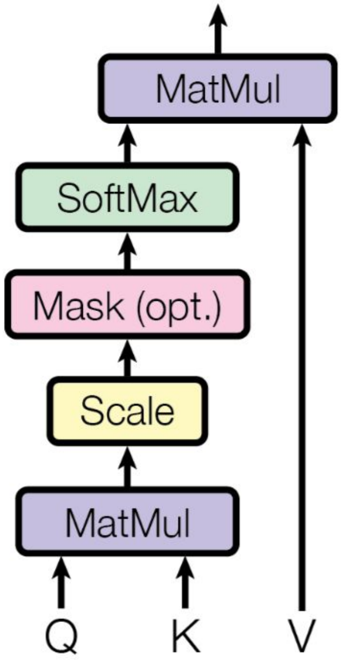
VALUE
Word/Token: “These are the values I bring.”

$$\mathrm{Attention}(Q, K, V) = \mathrm{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

QUERY
Word/Token: “These are the queues I should pay attention to.”

Vaswani, A. "Attention is all you need." Advances in Neural Information Processing Systems (2017).

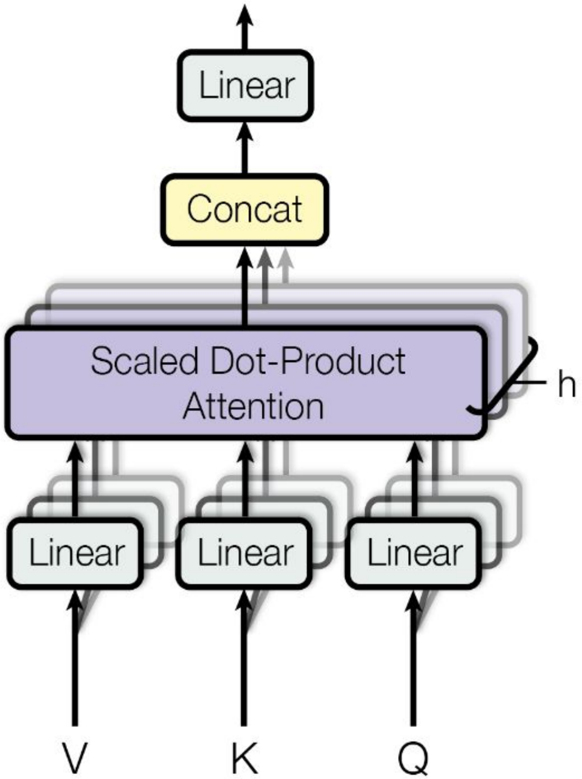
Attention Layer



Result: fixed representation matrices (same as word2vec).

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

However, the non-linear interactions of these matrices present **countless contexts trained in parallel**, not unique to words.



Vaswani, A. "Attention is all you need." Advances in Neural Information Processing Systems (2017).

Pre-training Models: Self-supervision

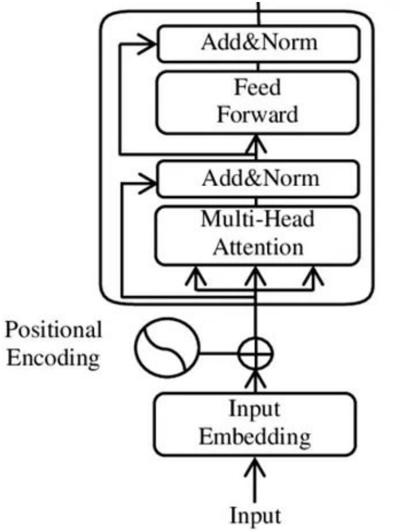
Random mask	A quick [MASK] fox jumps over the [MASK] dog.
Prediction	A quick brown fox jumps over the lazy dog.

Why are Transformers That Good?

- Attention Mechanism: Allows to model sequences (but RNNs can do that too!)
- Transformers can be massively parallelized. Ideal for GPUs!
- No memory loss as the RNNs. They take the entire text at the same time.
- And a few other tricks that we cannot cover…

Fine-tuning and Adaptation of Encoders

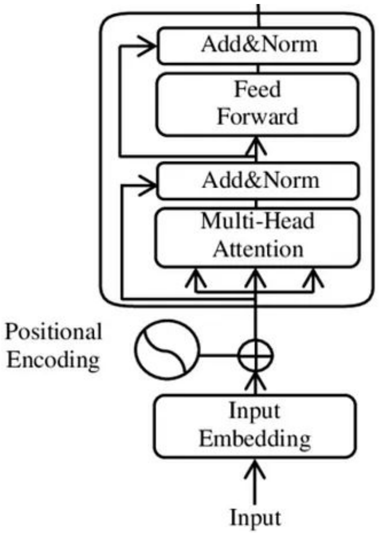
Encoder
Pre-trained with massive amounts of text using self-supervision



Fine-tuning and Adaptation of Encoders

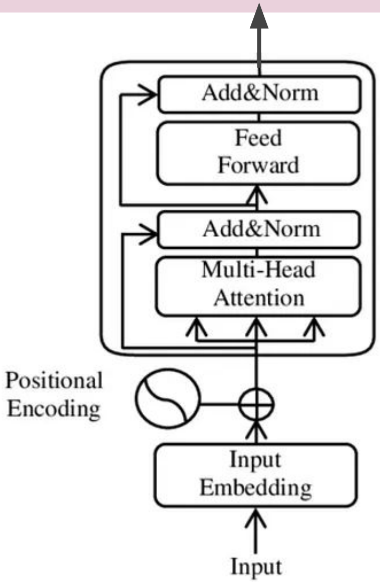
Encoder

Pre-trained with massive amounts of text using self-supervision



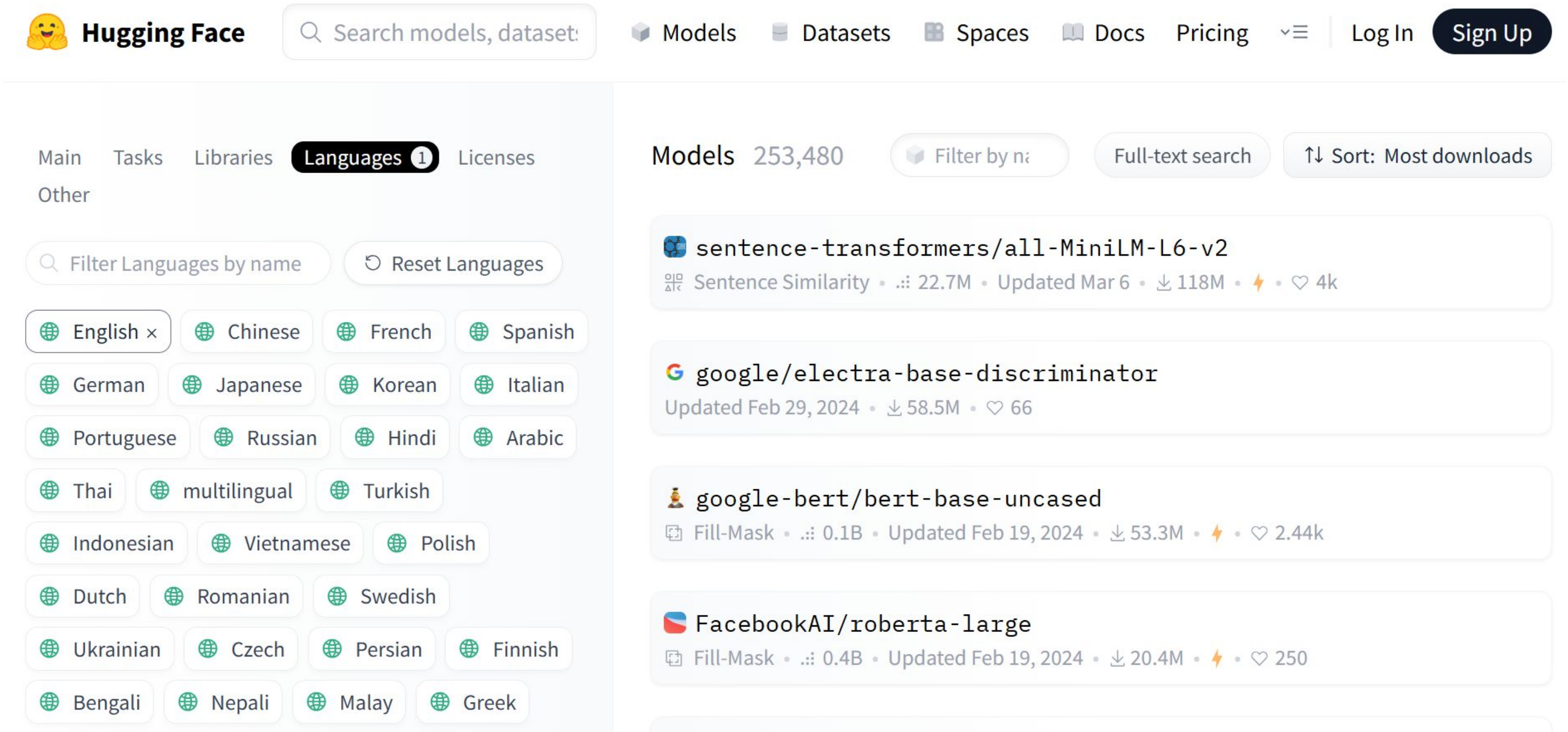
Fine-tuning

Classification Layer
(predicting manually labeled text)



Lightweight Adjustment of already language-aware models

Hugging Face: Transformers Repository



Hugging Face: Applying and Fine-tuning a Model

```python
import torch
from transformers import AutoModelForSequenceClassification
from transformers import AutoTokenizer

# load tokenizer
tokenizer = AutoTokenizer.from_pretrained("luerhard/PopBERT")

# load model
model = AutoModelForSequenceClassification.from_pretrained("luerhard/PopBERT")

# define text to be predicted
text = (
    "Das ist Klassenkampf von oben, das ist Klassenkampf im Interesse von "
    "Vermögenden und Besitzenden gegen die Mehrheit der Steuerzahlerinnen und "
    "Steuerzahler auf dieser Erde."
)

# encode text with tokenizer
encodings = tokenizer(text, return_tensors="pt")

# predict
with torch.inference_mode():
    out = model(**encodings)

# get probabilties
probs = torch.nn.functional.sigmoid(out.logits)
```

Hugging Face

luerhard / PopBERT

Erhard, Lukas, Sara Hanke, Uwe Remer, Agnieszka Falenska, and Raphael Heiko Heiberger. "PopBERT. Detecting populism and its host ideologies in the German Bundestag." Political Analysis 33, no. 1 (2025): 1-17.

Hugging Face: Applying and Fine-tuning a Model

```python
import torch
from transformers import AutoModelForSequenceClassification
from transformers import AutoTokenizer

# load tokenizer
tokenizer = AutoTokenizer.from_pretrained("luerhard/PopBERT")

# load model
model = AutoModelForSequenceClassification.from_pretrained("luerhard/PopBERT")

# define text to be predicted
text = (
    "Das ist Klassenkampf von oben, das ist Klassenkampf im Interesse von "
    "Vermögenden und Besitzenden gegen die Mehrheit der Steuerzahlerinnen und "
    "Steuerzahler auf dieser Erde."
)

# encode text with tokenizer
encodings = tokenizer(text, return_tensors="pt")

# predict
with torch.inference_mode():
    out = model(**encodings)

# get probabilties
probs = torch.nn.functional.sigmoid(out.logits)
```

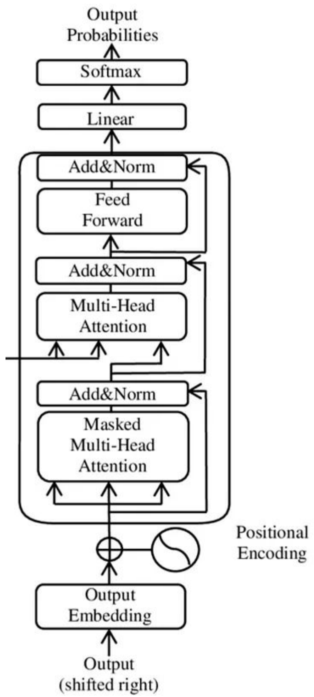
Hugging Face

luerhard / PopBERT

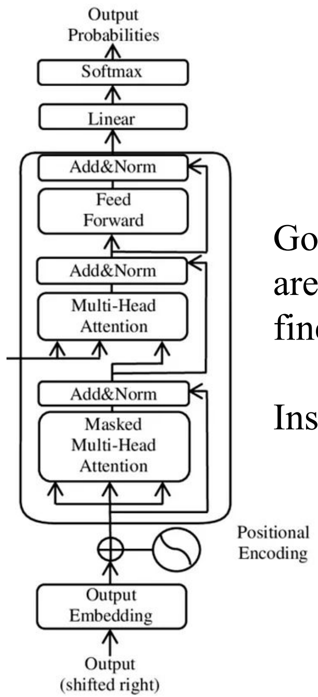
Check fine_tuning.py

Erhard, Lukas, Sara Hanke, Uwe Remer, Agnieszka Falenska, and Raphael Heiko Heiberger. "PopBERT. Detecting populism and its host ideologies in the German Bundestag." Political Analysis 33, no. 1 (2025): 1-17.

Generative AI and Prompt Engineering



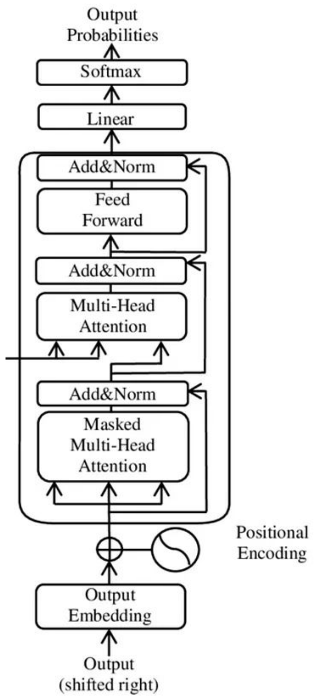
Decoders/Generators
Text Generation / Question Answering



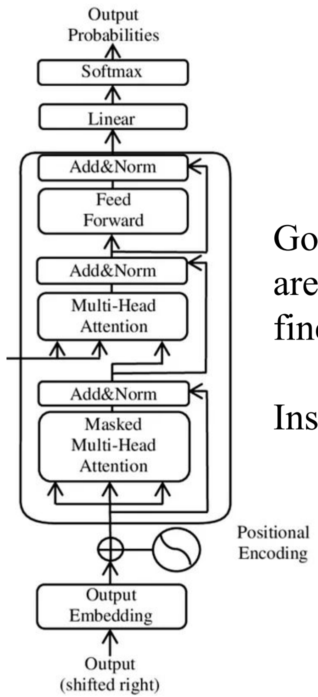
Good all-purpose generative models are too large and modular to be fine-tuned by a small group.

Instead…

Generative AI and Prompt Engineering



Decoders/Generators
Text Generation / Question Answering



Good all-purpose generative models are too large and modular to be fine-tuned by a small group.

Instead…

Prompt Engineering

Improving how we communicate with the model by providing more context.

Prompt Engineering for Text Annotation

- Goal: Improve consistency, validity, and interpretability of LLM-based annotations (e.g., sentiment, stance, ideology, populism, etc.)
- Steps:
  1. Specify task and schema clearly (**zero-shot vs. few-shot**)
  2. Control the **output format**
  3. Test **step by step reasoning** vs. **chain-of-thought suppression** (**temperature** = 0?)
  4. Test Prompt Robustness (rephrase prompt, check quality against human annotators)
  5. Post-process, combine models/prompts (majority voting), statistically validate

Caution: small textual changes can affect construct validity and replicability.

Prompt Engineering for Text Annotation (**API** calls, Python)

```python
import os
from litellm import completion

# --- Setup environment ---
os.environ["OPENAI_API_KEY"] = "sk-your-api-key-here"  # <-- replace with your real key

# --- Input text ---
text = "The corrupt elite has betrayed the people!"

# --- Call ChatGPT via LiteLLM ---
response = completion(
    model="gpt-4o-mini",
    messages=[{"role": "user", "content": f"Label the following text as 'populism' or 'not populism':\n\n{text}"}])

# --- Output result ---
print(response["choices"][0]["message"]["content"])
```

Prompt Engineering for Text Annotation (**API** calls, R)

```r
library(ellmer)

chat <- chat_openai(model = "gpt-4.1")

text <- "The corrupt elite has betrayed the people!"

prompt <- paste0(
  "You are a classifier. Label the following text as **\"populism\"** or
**\"not populism\"\"** (just output the label, nothing else):\n\n", text
)

schema <- type_object(
  label = type_string()
)

res_struct <- chat$chat_structured(prompt, type = schema)

res_struct$label
```

Outline

- From word embeddings to Transformers
- Encoding and Decoding: Understanding Modern Models
- Fine-tuning and Adaptation
- Generative AI and Prompt Engineering

- Tutorial: Sentiment and Topic Modeling in Parliamentary Speech
- Discussion: The Promise and Pitfalls of NLP for Social Sciences

Tutorial

Link to notebook: https://github.com/boevkoski/intro_to_css_text_analysis_II

Discussion

*Grimmer, Justin, and Brandon M. Stewart. "Text as data: The promise and pitfalls of automatic content analysis methods for political texts." Political analysis 21, no. 3 (**2013**): 267-297.*

Discussion

*Grimmer, Justin, and Brandon M. Stewart. "Text as data: The promise and pitfalls of automatic content analysis methods for political texts." Political analysis 21, no. 3 (**2013**): 267-297.*

Table 1 Four principles of quantitative text analysis

(1) All quantitative models of language are wrong—but some are useful.
(2) Quantitative methods for text amplify resources and augment humans.
(3) There is no globally best method for automated text analysis.
(4) Validate, Validate, Validate.