Transformers
• The **RNN** and **LSTM** neural models were designed to process language and perform tasks like classification, summarization, translation, and sentiment detection
  
  • RNN: Recurrent Neural Network
  • LSTM: Long Short Term Memory

• In both models, layers get the next input word and have access to some previous words, allowing it to use the word’s left context

• They used word embeddings where each word was encoded as a vector of 100-300 real numbers representing its meaning
Transformers extend this to allow the network to process a word input knowing the words in both its left and right context.

This provides a more powerful context model.

Transformers add additional features, like attention, which identifies the important words in this context.

And break the problem into two parts:
  • An encoder (e.g., Bert)
  • A decoder (e.g., GPT)
Transformer model

Encoder (e.g., BERT)  Decoder (e.g., GPT)
Transformers, GPT-2, and BERT

1. A transformer uses an **encoder stack** to model input, and uses **decoder stack** to model output (using input information from encoder side)

2. If we do not have input, we just want to model the “next word”, we can get rid of the encoder side of a transformer and output “next word” one by one. This gives us **GPT**

3. If we are only interested in training a language model for the input for some other tasks, then we do not need the decoder of the transformer, that gives us **BERT**
Training a Transformer

• Transformers typically use semi-supervised learning with
  • Unsupervised pretraining over a very large dataset of general text
  • Followed by supervised **fine-tuning** over a focused data set of inputs and outputs for a particular task

• Tasks for pretraining and fine-tuning commonly include:
  • language modeling
  • next-sentence prediction (aka completion)
  • question answering
  • reading comprehension
  • sentiment analysis
  • paraphrasing
Pretrained models

• Since training a model requires huge datasets of text and significant computation, researchers often use common pretrained models.

• Examples (circa December 2021) include:
  • Google’s BERT model
  • Huggingface’s various Transformer models
  • OpenAI’s and GPT-3 models
Huggingface Models
OpenAI Application Examples

- **Chat**: Open-ended conversation with an AI assistant.
- **Grammar correction**: Corrects sentences into standard English.
- **Natural language to OpenAI API**: Create code to call the OpenAI API using natural language.
- **English to French**: Translates English text into French.
- **SQL translate**: Translates natural language to SQL queries.
- **Classification**: Classifies items into categories via example.
- **Movie to Emoji**: Converts movie titles into emoji.
- **Translate programming languages**: Translates code from one programming language to another.
- **Q&A**: Answers questions based on existing knowledge.
- **Summarize for a 2nd grader**: Translates difficult text into simpler concepts.
- **Text to command**: Translates text into programmatic commands.
- **Natural language to Stripe API**: Creates code to call the Stripe API using natural language.
- **Parse unstructured data**: Creates tables from long form text.
- **Python to natural language**: Explains a piece of Python code in human understandable terms.
- **Calculate Time Complexity**: Finds the time complexity of a function.
- **Advanced tweet classifier**: Classifies tweets into categories.
GPT-2, BERT
GPT released June 2018
GPT-2 released Nov. 2019 with 1.5B parameters
GPT-3 released in 2020 with 175B parameters

117M parameters
345M
762M
1542M