Language Models

Rafael Mejía, Santiago Pineda, Juan López, Andrés Álvarez [1]

{rmejiaz, spinedaq, jslopezvi, aalvarezme}@unal.edu.co[1]
Universidad Nacional de Colombia

July 14, 2023

Overview

1 Introduction
- Definition of a language model
- Importance and applications
- Evolution

2 Basic Language Models
- Bigram Model
- Recurrent Neural Networks
- Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU)

3 Transformer Architecture
- Encoder Models
- Decoder Models
- Encoder - Decoder Models
- Fine Tuning

Introduction

What is a language model?

- At its core, a language model is a computational framework that aims to capture the underlying structure, patterns, and semantics of natural language.
- They serve as an intelligent system capable of predicting the likelihood of a sequence of words, given the context of previously observed words.
- Language models provide a way to estimate the probability distribution of words, enabling us to generate coherent and contextually appropriate sentences.

Importance and applications

- Language models act as a fundamental building block in many natural language processing (NLP) tasks, allowing machines to understand and generate human-like text.
- By modeling the patterns and dependencies within language, these models empower us to automate language-related tasks, enhance communication, and unlock new possibilities in various fields.
- Language models have found applications in diverse domains, including machine translation, question-answering systems, sentiment analysis, content generation, personalized recommendations, and much more.

Evolution

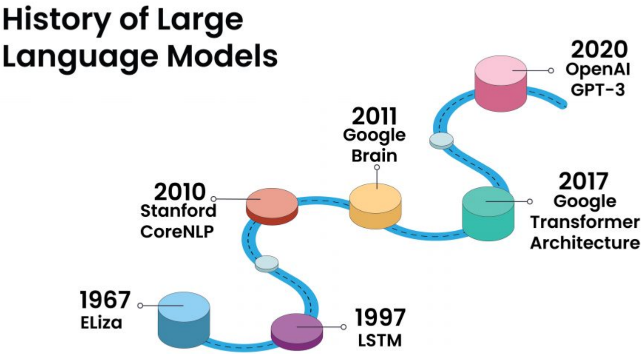
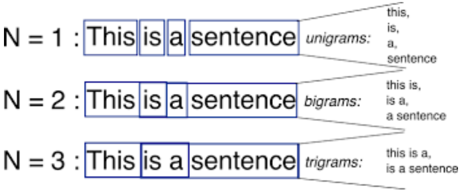


Figure: Evolution of language models

Basic Language Models

Bigram Model

- The Bigram Model is a simple language model that focuses on capturing dependencies between consecutive pairs of tokens in a given sequence.
- It assumes that the probability of a word depends only on the preceding word. In other words, it estimates the probability of a word based on the occurrence of the previous word.
- Key Assumption:
  $P(token_i|token_i - 1) \approx P(token_i|token_i - 1, token_i - 2, ..., token_1)$



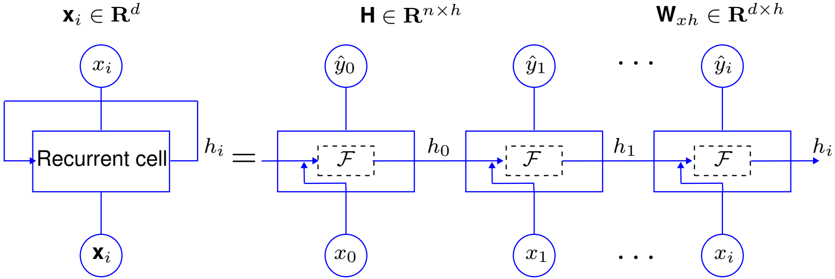
Calculating probabilities

- To estimate the probability of a word given the previous word, we can calculate the frequency of their co-occurrence in a large corpus.
- Example:
  Let's consider the sentence: "The cat is sitting on the mat." Using the Bigram Model, we calculate the probability of the word "sitting" given the previous word "is" as the frequency of the bigram "is sitting" divided by the frequency of the word "is".

Problems

- The Bigram Model assumes that the probability of a word depends only on its immediate predecessor, neglecting the influence of words beyond the previous one.
- It fails to capture long-range dependencies and contextual information, leading to limitations in generating coherent and contextually accurate sentences.

Recurrent Neural Networks (RNN)



$$\hat{y}_i = \mathcal{F}(x_i, h_{i-1}) \tag{1}$$

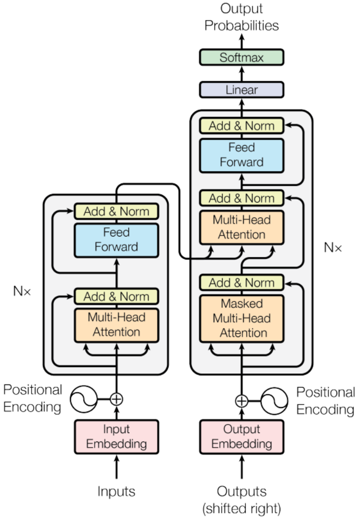
Problems

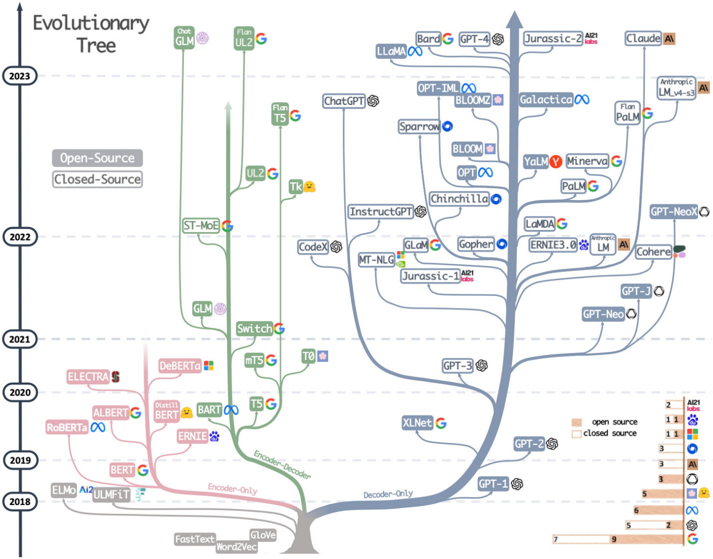
- Distant positions in the sequence can be disregarded
- Parallelizing the work is challenging because it processes variables sequentially

Transformer Architecture

Transformer Architecture

- Introduced by Vaswani et al. in 2017 for a language translation task.

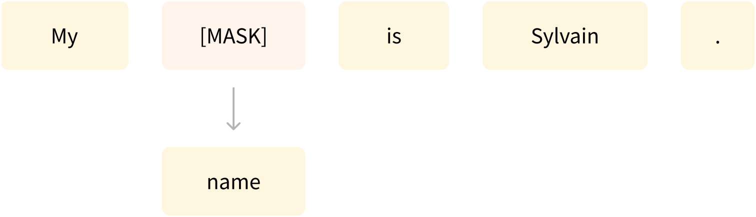



Evolutionary Tree
2023
2022
2021
2020
2019
2018
Open-Source
Closed-Source
Chat GLM
Flan UL2
Flan T5
UL2
Tk
ST-MoE
GLM
Switch
mT5
T0
T5
BART
DeBERTa
ELECTRA
ALBERT
Distill BERT
RoBERTa
ERNIE
BERT
ELMo
AI2
ULMFiT
FastText
GloVe
Word2Vec
Encoder-Only
Encoder-Decoder
Decoder-Only
Bard
GPT-4
LLaMA
Jurassic-2
AI21 labs
Claude
ChatGPT
OPT-IML
BLOOMZ
Galactica
Anthropic LM_v4-s3
Flan PaLM
Sparrow
BLOOM
YaLM
Minerva
OPT
PaLM
Chinchilla
InstructGPT
GPT-NeoX
LaMDA
CodeX
GLaM
Gopher
ERNIE3.0
Anthropic LM
Cohere
MT-NLG
Jurassic-1
AI21 labs
GPT-J
GPT-Neo
GPT-3
XLNet
GPT-2
GPT-1
open source
closed source
2
AI21 labs
1 1
1 1
3
3
3
5
6
5 2
7 9

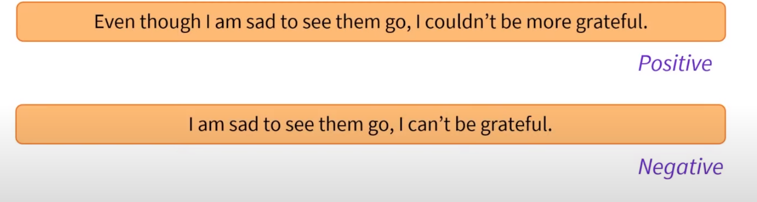
Encoder Models

- The encoder outputs a contextualized numerical representation for each word in the input.
- They are good at extracting meaningful information
- Common tasks: sequence classification, question answering, masked language modeling, named entity recognition, etc.
- Examples: BERT, RoBERTa, ALBERT.

Masked Language Modeling



Sequence classification - Sentiment Analysis



Even though I am sad to see them go, I couldn't be more grateful.

Positive

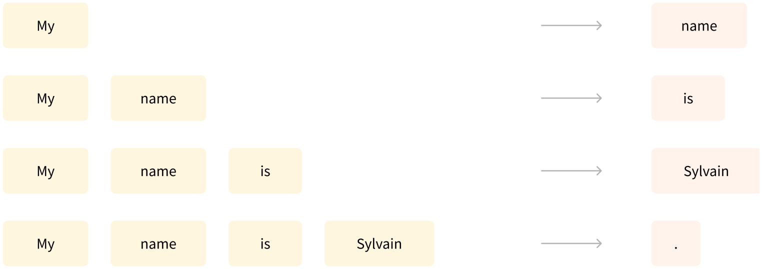
I am sad to see them go, I can't be grateful.

Negative

Decoder Models

- The main difference with the encoder is that they use masked self-attention, meaning a given token can only attend at past tokens.
- Well suited for causal tasks like sequence generation.
- Often called auto-regressive models.
- Examples: CTRL, GPT, Transformer XL.

Causal Language Modeling



My → name

My name → is

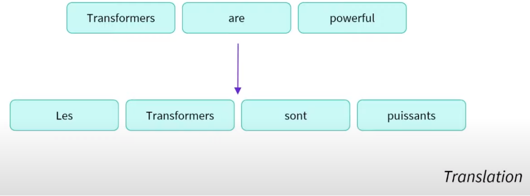
My name is → Sylvain

My name is Sylvain → .

Encoder - Decoder Models

- As seen before, the encoder generates a numerical representation of the input, which is passed to the decoder via cross-attention.
- At each stage, the attetnion layers of the encoder can attend to all the input sequence, whereas the attention layers of the decoder can only access the words positioned before a given word in the input.
- The encoder takes care of understanding the input sequence.
- The decoder takes care of generating a sequence according to the understanding of the encoder.
- Sequence to sequence tasks; many-to-many: translation, summarization, question answering, etc.
- Examples: BART, mBART, Marian, T5.

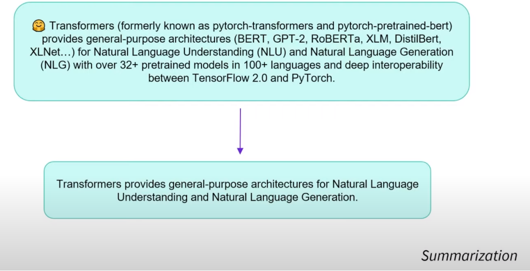
Examples



Transformers are powerful

Les Transformers sont puissants

Translation

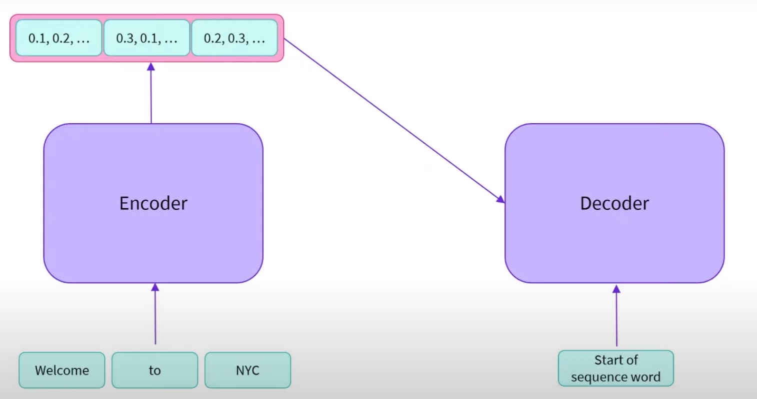


🤗 Transformers (formerly known as pytorch-transformers and pytorch-pretrained-bert) provides general-purpose architectures (BERT, GPT-2, RoBERTa, XLM, DistilBert, XLNet…) for Natural Language Understanding (NLU) and Natural Language Generation (NLG) with over 32+ pretrained models in 100+ languages and deep interoperability between TensorFlow 2.0 and PyTorch.

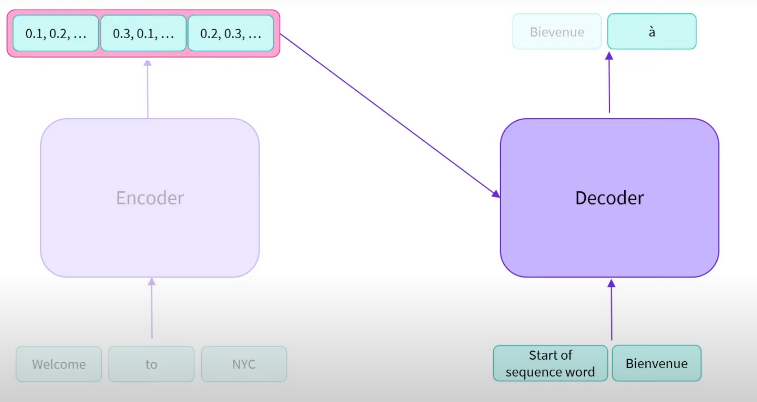
Transformers provides general-purpose architectures for Natural Language Understanding and Natural Language Generation.

Summarization

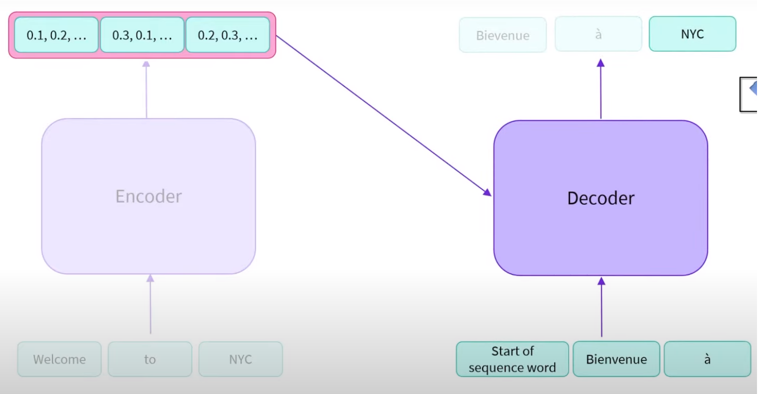
Translation example



Translation example

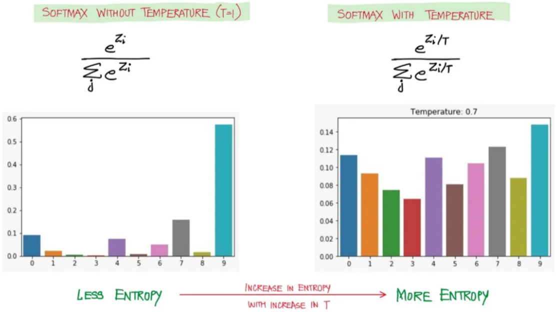


Translation example



Where does the randomness come from?

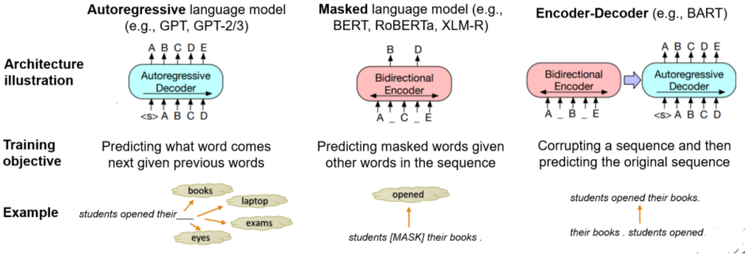
- Language models use an hyperparameter called "temperature", which basically increases or decreases the confidence the model has in its most likely prediction.
- This is done by simply scaling the logits before applying softmax.
- A higher temperature value makes the output more diverse and "creative" whereas a lower temperature make the output more focused and deterministic.



Interactive Example

Pre-Trained Models

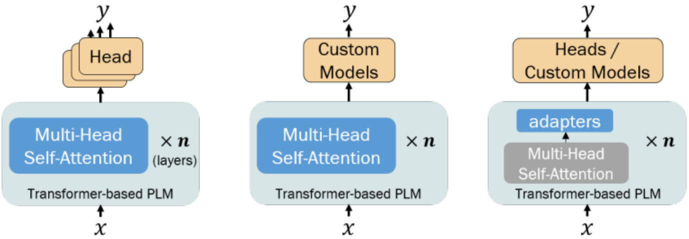
- Self-supervised learning allow LLMs to be trained on very large amounts of data.



	Autoregressive language model (e.g., GPT, GPT-2/3)	**Masked** language model (e.g., BERT, RoBERTa, XLM-R)	**Encoder-Decoder** (e.g., BART)
Architecture illustration			
Training objective	Predicting what word comes next given previous words	Predicting masked words given other words in the sequence	Corrupting a sequence and then predicting the original sequence
Example			

Fine Tuning Techniques

- Contextual Embeddings: "freeze" the model and use its output as sophisticated context-sensitive word embedding for a subsequent architecture.
- Fine Tune the Pre-trained language model: fine-tune some or all the layers of the PLM and then add one or two simple output layers known as prediction heads.
- Fine Tune the Pre-trained Language Model in Customized Models: Some tasks require significant additional architecture on top of a pre-trained model. With suicient training data and computational power, researchers may choose to train both a substantial task-speciic architecture and also fine-tune the language model.

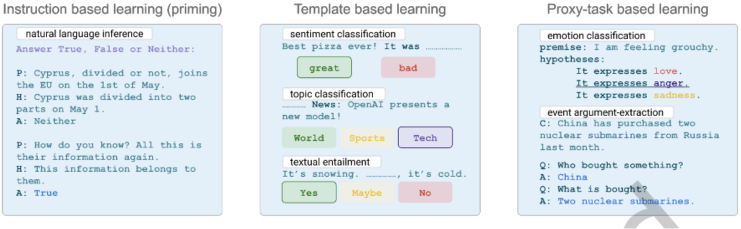


Few-Shot, One-Shot, Few-Shot Learning

- Few-Shot: The model is fine-tuned in a small set of samples.
- One-Shot: The model only seed one sample from each new class.
- Zero-Shot: No examples are given to the model and it must do the inference with its internal knowledge.

Prompt-Based Learning

- It consists in adding natural language at the input of a LLM to "encourage" it to perform specific tasks. It has the main advantage that there is no change in the parameters of the model, which reduces computational requirements.



Instruction based learning (priming)

natural language inference

```
Answer True, False or Neither:

P: Cyprus, divided or not, joins
the EU on the 1st of May.
H: Cyprus was divided into two
parts on May 1.
A: Neither

P: How do you know? All this is
their information again.
H: This information belongs to
them.
A: True
```

Template based learning

sentiment classification

```
Best pizza ever! It was ..........
```

great | bad

topic classification

```
.......... News: OpenAI presents a
new model!
```

World | Sports | Tech

textual entailment

```
It's snowing. .........., it's cold.
```

Yes | Maybe | No

Proxy-task based learning

emotion classification

```
premise: I am feeling grouchy.
hypotheses:
        It expresses love.
        It expresses anger.
        It expresses sadness.
```

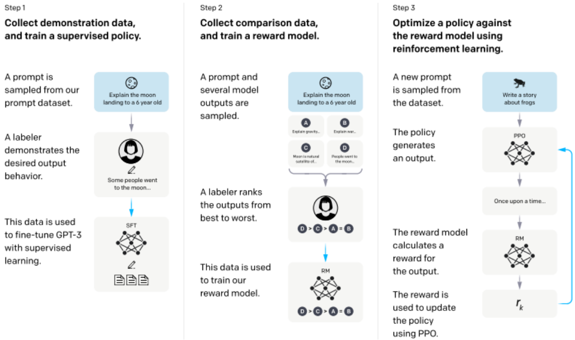
event argument-extraction

```
C: China has purchased two
nuclear submarines from Russia
last month.

Q: Who bought something?
A: China
Q: What is bought?
A: Two nuclear submarines.
```

Reinforcement Learning from Human Feedback

- Large language models are not inherently good are following instructions or users intent.
- They can, for example generate toxic or untruthful answers
- To solve this, it is possible to fine-tune language models using reinforcement learning with human feedback, to align the models with the users, OpenAI did with InstructGPT and presumably with ChatGPT.



Step 1

Collect demonstration data, and train a supervised policy.

A prompt is sampled from our prompt dataset.

A labeler demonstrates the desired output behavior.

This data is used to fine-tune GPT-3 with supervised learning.

Step 2

Collect comparison data, and train a reward model.

A prompt and several model outputs are sampled.

A labeler ranks the outputs from best to worst.

This data is used to train our reward model.

Step 3

Optimize a policy against the reward model using reinforcement learning.

A new prompt is sampled from the dataset.

The policy generates an output.

The reward model calculates a reward for the output.

The reward is used to update the policy using PPO.

Explain the moon landing to a 6 year old
Some people went to the moon...
SFT
Explain the moon landing to a 6 year old
A
Explain gravity...
B
Explain war...
C
Moon is natural satellite of...
D
People went to the moon...
D > C > A = B
RM
D > C > A = B
Write a story about frogs
PPO
Once upon a time...
RM
r_k

Thank you!